Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής
Πρόγραμμα Μεταπτυχιακών Σπουδών
«Προηγμένα Συστήματα Πληροφορικής»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Ανάλυση Ευπαθειών λογισμικού σε λειτουργικό σύστημα Android** **Code Vulnerability Assessment for Android**
Ονοματεπώνυμο Φοιτητή	**Αναστάσιος Τσαλαβούτας**
Πατρώνυμο	**Κωνσταντίνος**
Αριθμός Μητρώου	**ΜΠΣΠ/ 13112**
Επιβλέπων	**Κωνσταντίνος Πατσάκης, Επίκουρος Καθηγητής**

Ημερομηνία Παράδοσης **Οκτώβριος 2016**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Κωνσταντίνος Πατσάκης Επίκουρος Καθηγητής	Ευθύμιος Αλέπης Επίκουρος Καθηγητής	Γεώργιος Τσιχριντζής Καθηγητής

Περίληψη

Η εργασία μελετάει τις διάφορες αδυναμίες ασφαλείας που μπορούν να περιέχονται σε διάφορες εφαρμογές για το λειτουργικό σύστημα android. Ακόμα περιγράφει με ποιο τόπο τις εντοπίζει, πως μπορεί να διορθωθούν κάποιες ευπάθειες και τι θα έπρεπε να είχε κάνει ο προγραμματιστής που έγραψε την εφαρμογή ώστε να μην είχε δημιουργηθεί το κενό ασφαλείας.

Abstract

This project detected various security holes on any application in android operation system. Also the project analyze the way how to close these holes and in some issues, suggests what the programmer have to do to avoid these issues.

Ευχαριστίες

Ευχαριστώ όλους τους καθηγητές του μεταπτυχιακού που με βοηθήσαν στην ολοκλήρωση του μεταπτυχιακού προγράμματος «Προηγμένα συστήματα πληροφορικής», και ιδιαίτερα τον κ. Κ. Πατσάκη οποίος επέβλεψε την εκπόνηση αυτής της διατριβής.

Επίσης ευχαριστώ την σύζυγο μου και συμφοιτήτρια μου Κωνσταντίνα Αναγνώστου για τη συμπαράσταση που μου έδειξε.

Ακόμα ευχαριστώ τους γονείς μου για την στήριξη τους.

«Οκτώβριος 2016»

Αναστάσιος Τσαλαβούτας

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

Κεφάλαιο 1°

Εισαγωγή

Στην παρούσα εργασία θα εξετάσουμε τις διάφορες ευπάθειες που μπορεί να παρουσιάσει μια εφαρμογή, η οποία "τρέχει" σε λειτουργικό σύστημα android.

Η μέθοδος που χρησιμοποιείται για να διαπιστωθεί αν μια εφαρμογή εμφανίζει ευπάθειες είναι η στατική ανάλυση κώδικα (code static analysis).

Με την χρήση κάποιων εργαλείων και με την μέθοδο reverse engineering θα παραχθεί ο πηγαίος κώδικας (source code), κατά προσέγγιση, και θα εξετάσει με βάση κάποιους κανόνες για το τι μπορεί να περιέχει και αν έχει κάποιο ρίσκο από άποψη ασφάλειας.

Για να μπορέσουμε να αναλύσουμε ένα αρχεία της android εφαρμογής, έχει γραφτεί μια εφαρμογή για περιβάλλον Windows και Linux και κάνει τα παρακάτω βήματα:

- Αποσυμπιέζει το παραγόμενο (apk) αρχείο.
- Παράγει τα αρχεία που περιέχουν τον πηγαίο κώδικα
- Εξετάζει για το αν αληθεύουν κάποιοι κανόνες
- Παράγει ένα report με τα ευρήματα.
- Προτείνει πως μπορεί κάποιος να διορθώσει τις συγκεκριμένες ευπάθειες.

Είναι μια αντικειμενοστραφής (object oriented) εφαρμογή για Windows και linux. Αποτελείται από ένα script που εκτελείτε από ένα Terminal και μετά το τέλος της εκτέλεσης ανοίγει έναν Web browser και εμφανίζει μια report σελίδα με τα διάφορα ευρήματα που ανιχνευτήκαν κατά την διαδικασία της ανάλυσης.

Στο τέλος τις διαδικασίας εμφανίζει ένα report γραμμένο σε html με την χρήση του Bootstrap.

Βασικοί ορισμοί

Λειτουργικό σύστημα android[10]

Είναι ένα λειτουργικό σύστημα για κινητές συσκευές, το οποίο έχει αναπτυχθεί από την google (http://www.google.com). Βασίζετε στον πυρήνα (kernel) του Linux και έχει σχεδιαστεί κυρίως για smartphones και tablets. Το περιβάλλον διεπαφής με τον χρήστη (user interface), βασίζεται σε απ' ευθείας χειρισμό με διάφορες χειρονομίες και αγγίγματα της οθόνης και των διάφορων αντικειμένων και προγραμμάτων.

Τα διάφορα applications που είναι διαθέσιμα για το android διατίθενται από το google play store (υπάρχουν και αλλά επίσημα ή μη stores), και είναι γραμμένα είτε σε java είτε με κάποιο framework με την χρήση html, css, JavaScript.

Ασφάλεια Πληροφοριών[11]

Ασφάλεια Πληροφοριών είναι ο συνδυασμός της Εμπιστευτικότητας, της Ακεραιότητας και της Διαθεσιμότητας τους.

Εμπιστευτικότητα[11]

Εμπιστευτικότητα είναι η ιδιότητα της Πληροφορίας να είναι προσπελάσιμη μόνο

από οντότητες που είναι εξουσιοδοτημένες προς τούτο.

Ακεραιότητα[11]

Ακεραιότητα είναι η ιδιότητα της Πληροφορίας να τροποποιείται μόνο από εξουσιοδοτημένες προς τούτο οντότητες.

Διαθεσιμότητα[11]

Τέλος, Διαθεσιμότητα είναι η ιδιότητα της Πληροφορίας να είναι διαθέσιμη στις εξουσιοδοτημένες προς τούτο οντότητες μέσα σε λογικό χρονικό διάστημα από την υποβολή της σχετικής αίτησης διάθεσης.

Στατική ανάλυση κώδικα[12]

Στην πληροφορική, στατική ανάλυση, στατική ανάλυση κώδικα ή στατική ανάλυση προγράμματος (Αγγλ.: Static analysis, static code analysis, static program analysis), είναι η ανάλυση ενός προγράμματος που γίνεται χωρίς να εκτελεστεί το πρόγραμμα (η ανάλυση προγράμματος που γίνεται με εκτέλεσή του λέγεται δυναμική ανάλυση). Στις περισσότερες περιπτώσεις η ανάλυση γίνεται σε κάποιο επίπεδο πηγαίου κώδικα και πολλές φορές σε αντικειμενικό κώδικα. Ο όρος συνήθως αναφέρεται στην ανάλυση που γίνεται με κάποιο αυτόματο εργαλείο, ενώ η ανάλυση προγράμματος από άνθρωπο λέγεται κατανόηση προγράμματος.

Κακόβουλο λογισμικό (malware)[13]

Το «κακόβουλο λογισμικό» (malicious Software / malware Software) αποτελεί μείζον πρόβλημα για την ασφάλεια των Πληροφοριακών Συστημάτων. Το λογισμικό χαρακτηρίζεται ως κακόβουλο όταν βάσει των προθέσεων του προγραμματιστή το λογισμικό που προκύπτει διαθέτει τις απαιτούμενες εντολές προκειμένου να βλάψει ένα υπολογιστικό σύστημα. Το κακόβουλο λογισμικό μπορεί να χωριστεί σε δύο κατηγορίες.
Σε αυτό που χρειάζεται ένα πρόγραμμα «ξενιστή» και σε αυτό που δεν χρειάζεται «ξενιστή» και μπορεί να εκτελεστεί από μόνο του όπως κάθε άλλο πρόγραμμα.

Reverse engineering (of software)[15][16]

Reverse engineering είναι η διαδικασία ανάλυσης ενός συστήματος λογισμικού, ολοκλήρου ή μέρος αυτού, με σκοπό την εξαγωγή πληροφοριών σχετικά με τον σχεδιασμό και την υλοποίηση του.

Εργαλεία που χρησιμοποιηθήκαν για την εφαρμογή

Για την εφαρμογή που υλοποιεί την ανάλυση, χρησιμοποιήθηκε η script γλώσσα Python[17]. Επίσης χρησιμοποιήθηκαν και κάποια components όπως:

1. Το open source JavaScript framework bootstrap[3]
2. Το sb admin 2 free template για bootstrap[4].
3. Ο jd-gui[5] και jd-cmd java decompiler [6].
4. Το εργαλείο ανοιχτού κώδικα (open source) dex2jar[7] για την μετατροπή των αρχείων .dex σε αρχεία java class.
5. Εναλλακτικά ο Procyon java decompiler[8].
6. Mobile-Security-Framework (MobSF)[9]
7. AXMLPrinter[18] , μετατρέπει το manifest.xml σε κανονικό xml format αρχείο.

Παραδοτέα της εργασίας

Τα παραδοτέα της εργασίας είναι:

1. Το έντυπο κείμενο της πτυχιακής εργασίας, το οποίο περιλαμβάνει την παρουσίαση και τεκμηρίωση της εφαρμογής.
2. Την ιδιά την εφαρμογή (πηγαίος κώδικας).

Δομή της εργασίας

Η εργασία πραγματεύεται την στατική ανάλυση σε ένα αρχείο .apk με σκοπό τον εντοπισμό διάφορων ευπαθειών. Αυτό επιτυγχάνεται με την χρήση μια εφαρμογής.
Στα επόμενα κεφάλαια θα ακολουθήσει η τεκμηρίωση της εφαρμογής αυτής.

Κεφάλαιο 2ο

Εφαρμογή για στατική ανάλυση κώδικα (Code Static Analysis)

Για να μπορέσουμε να αναλύσουμε τον κώδικα σε ένα αρχείο .apk αναπτύχθηκε μια εφαρμογή για λειτουργικό σύστημα windows και Linux και με αυτήν μπορούμε να πάρουμε μια ανάλυση για πιθανά λάθη ασφαλείας για το αρχείο που μας ενδιαφέρει.

Δομή εφαρμογής

Η εφαρμογή αποτελείται από μια βασική κλάση, μια κλάση με σταθερές και επτά βοηθητικές κλάσης που υλοποιούν διάφορα κομμάτια της εφαρμογής.

Βασική κλάση apk_analysis.py

Περιέχει όλη την λογική της εφαρμογής καθώς και τις κλήσεις προς άλλες κλάσης.

```
#define main method
def main():
    url = "file://" + reportPath + "/index.html"
    extractSources()
    #create report folder
    htmlBuilder.MaKeReportDir()
    MakeAnalysis(sourceDir)
    OpenWebBrowser(url)
```

Πηγαίος Κώδικας 1: η βασική μέθοδος της Εφαρμογής.

Κλάση AnalisysDescriptions.py

Περιέχει όλες τις σταθερές για τα κλειδιά του dictionary και για τα λεκτικά. Τα λεκτικά έχουν βασιστεί πάνω στα λεκτικά του Mobile-Security-Framework (MobSF).[9]

```
class  AnalisysDescriptions(object):

    #dect keys

    gpsKey = "gps"
    randomKey = "random"
    sqlliteKey = "sqllite"
    modeworldreadKey = "modeworldread"
    modeworldwriteKey = "modeworldwrite"
    privateKey = "private"
    logKey = "log"
    cryptoKey = "crypto"
    ipcKey = "ipc"
    httpclientKey = "httpclient"
    fileioKey = "fileio"
    infactKey = "inf_act"
    infserKey = "inf_ser"
    infbroKey = "inf_bro"
    jsenableKey = "jsenable"
    WebViewdebugKey = "WebViewdebug"
    sslerrorKey = "sslerror"
    sslKey = "ssl"
    sendataKey = "sendata"
    dexdebugKey = "dex_debug"
    dex2Key = "dex2"
    emulatorKey = "emulator"
    dexkeyKey = "dexkey"
    dexrootKey = "dexroot"
    dextamperKey = "dex_tamper"
    dexcertKey = "dex_cert"
    dsslpinKey = "d_ssl_pin"
    rootkey = "root"
```

Πηγαίος Κώδικας 2: Στιγμιότυπο από τις σταθερές.

Κλάση CodeAnalysis.py

Αυτή η κλάση υλοποιεί όλη την ανάλυση του κώδικα που έχουμε πάρει από το apk. Και με βάση κάποιους κανόνες, τους οποίους έχουμε αναπαράγει και προσαρμόσει από το Mobile-Security-Framework (MobSF)[9], επιστρέφει τα κατάλληλα μηνύματα.

Ακόμα περιέχει μεθόδους για να κάνει μετατροπή από dex αρχείο σε jar αρχείο και αργότερα να χρησιμοποιήσει αυτό το jar αρχείο για να το κάνει decompile.

Κλάση FileUtils.py

Περιέχει μεθόδους που εκτελούνε διάφορες λειτουργίες που αφορούν αρχεία. Οι λειτουργίες αυτές είναι:

- Δημιουργεί τον φάκελο για το decompile file (method:)
- Μετονομάζει ένα αρχείο (method:)
- Γραφεί στο log file τυχόν λάθη που μπορεί να εμφανιστούν.

Κλάση HtmlCreator.py

Δημιουργεί την πρώτη και την κυρίως σελίδα στο παραγόμενο report.

Κλάση infos.py

Παράγει τα κομμάτια του report που περιέχουν κάποιες πληροφορίες και υπολογίζει το Md5 και το sha hash του αρχείου apk. Αυτά είναι:

- File info
- App info

Κλάση ManifestExtractor.py

Περιέχει μεθόδους για την εξαγωγή, μετατροπή σε xml και ανάλυση του manifest. Επίσης εξάγει διάφορες πληροφορίες όπως ανάλυση του manifest, permissions της android εφαρμογής, main activity κ.α. Ακόμα κάνει extract το manifest και το μετατρέπει σε κανονικό xml format. Τα μηνύματα έχουν βασιστεί πάνω στα λεκτικά και τα μηνύματα του Mobile-Security-Framework (MobSF).[9]

Κλάση myutils.py

Περιέχει μια μέθοδο για να εκτελεί τα εξωτερικά προγράμματα στο terminal.

Script Archive.py

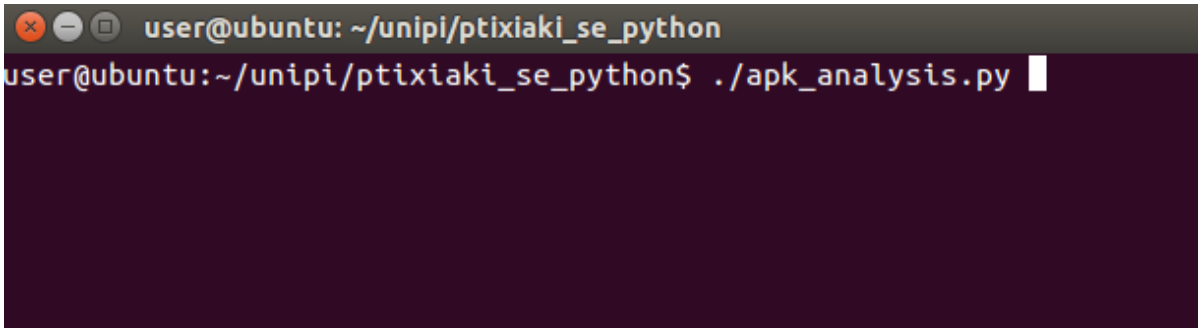
Είναι ένα εκτελέσιμο script που αναλαμβάνει να συμπιέσει σε ένα αρχείο zip τα αρχεία που έχουν προκύψει από την ανάλυση. Στο τέλος όταν το αρχείο zip είχε ολοκληρωθεί σβήνει το περιεχόμενο του φακέλου source.

config.py

Περιέχει όλα τα path που χρησιμοποιούνται στην εφαρμογή. Όταν θέλουμε να "τρέξουμε" την εφαρμογή σε κάποιο άλλο σύστημα, ενημερώνουμε τις σταθερές με τα σωστά path.

Αρχικό script apk_analysis.py

Είναι το βασικό αρχείο της εφαρμογής. Αυτό καλεί τις κατάλληλες μεθόδους για να υλοποιηθεί όλη η ανάλυση και το reporting και στο τέλος «ανοίγει» τον default web browser και φορτώνει την πρώτη σελίδα του report.



Εικόνα 1: βασική φόρμα εφαρμογής.

Εκκίνηση διαδικασίας ανάλυσης

Η διαδικασία ξεκινά εκτελώντας το script apik_analysis.py. Για να εκτελέσουμε το script σε ένα terminal γράφουμε ./apk_analysis.py ή python apk_analysis.py. Το script θα εκτελέσει τις παρακάτω διαδικασίες:

- Θα κάνει μετονομασία στα ή στο αρχείο akp ώστε το όνομα του να μην περιέχει κενούς χαρακτήρες
- Θα δημιουργήσει τους φακέλους μέσα στον φάκελο source, που θα γίνει το decompile.
- θα τρέξει το dex2jar για να παραγάγει το jar αρχείο.
- Θα κάνει decompile το jar αρχείο και θα παραγάγει τα source files
- Και τέλος θα αναλύσει τα παραγόμενα source files.
- Θα δημιουργήσει τον φάκελο που θα φιλοξενήσει το report
- Θα δημιουργήσει τις html σελίδες του report

```
#define method for extracting source files
def extractSources():…

def MakeAnalysis(sdir):…

def OpenWebBrowser(url):
    #for open in new tab
    new = 2
    webbrowser.open(url, new=new)
#define main method
def main():
    url = "file://" + reportPath + "/index.html"
    extractSources()
    #create report folder
    htmlBuilder.MaKeReportDir()
    MakeAnalysis(sourceDir)
    OpenWebBrowser(url)


main()
```

Πηγαίος Κώδικας 3: Η μέθοδος που εκτελεί όλες τις φάσεις της ανάλυσης.

Μετατροπή αρχείου dex σε αρχείο jar

Για να μπορέσει η εφαρμογή να εξάγει τον πηγαίο κώδικα θα χρησιμοποιήσουμε το πρόγραμμα dex2jar[7] .

Με μια κλήση σε γραμμή εντολών (command line - terminal) εκτελείται το dex2jar για κάθε ένα από τα αρχεία που έχουμε μέσα στον φάκελο apks. Όταν τελειώσει η διαδικασία θα υπάρχει μέσα στο φάκελο source και σε φάκελο με το όνομα που έχει το κάθε αρχείο apk ένα αρχείο jar.

```
def dex2jar(self, tooldir,apkfile,jarfile):

    mutils.runCommands(tooldir+'/dex2jar/d2j-dex2jar.sh '+ apkfile +
' -o ' + jarfile +' -f ')
```

Πηγαίος Κώδικας 4: Η μέθοδος που εκτελεί το πρόγραμμα dex2jar.

Decompile του αρχείου .jar.

Για να αποκτήσουμε πρόσβαση στον πηγαίο κώδικα τις κάθε εφαρμογής θα πρέπει να εφαρμοστεί μέθοδος reverse engineering. Μια από αυτές τις διαδικασίες είναι και το decompile του αρχείου jar. Τα αποτελέσματα αυτής της μεθόδου θα μας δώσουν τον πηγαίο κώδικα κατά μεγάλη προσέγγιση.

Η εφαρμογή έχει μια μέθοδο που εκτελεί την εντολή για οποίον java decompiler διαλέξουμε. Και τα αποτελέσματα είναι να δημιουργηθεί ένας φάκελος με τα sources για κάθε μια εφαρμογή που έχουμε μέσα στον φάκελο apks.

```
def decompile(self,tooldir, sourcedir, jarfile):

    mutils.runCommands(tooldir + '/jd_cmd/jd-cli -od ' +sourcedir +
'/source ' +  jarfile )
```

Πηγαίος Κώδικας 5: Η μέθοδος που εκτελεί το πρόγραμμα Java decompiler.

Ανάλυση κώδικα και εύρεση ευπαθών

Η ανάλυση βασίζεται σε διάφορους κανόνες και πληροφορίες που χρησιμοποίει και το Mobile-Security-Framework (MobSF)[9]

```
def Sic(self, sourceAppDir):

    reDict ={}
    murls = {}
    codenat=""
    #get manifest xml formated file
    manifestFile = sourceAppDir +"/manifest.xml"

    #boolean for some properties
    crypto = False
    obfus = False
    reflect = False
    dynamic = False
    native = False

    #define string variables for add java source files to dictionary

    ipcStr=""
    gpsStr = ""
    randomStr = ""
    sqlliteStr = ""
    modeworldreadStr = ""
    modeworldwriteStr = ""
    privateStr = ""
    logStr = ""
    cryptoStr = ""
...
```

Πηγαίος Κώδικας 6: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

```
for dirName, subDir, files in os.walk(sourceAppDir+"/source"):
    for file in files:
        if file.endswith(".java"):
            f=open(dirName+'/'+file, 'r')
            fileContent = f.read()
            f.close()

            #regular expression rules

            if (re.findall(r'\bjava.util.Random\b', fileContent)):
                randomStr += ", "+ '<a href="'+dirName+'/'+ file +'"
target="_blank">' + os.path.basename(file) +'</a>'

            if (re.findall(r'\bMODE_WORLD_READABLE\b', fileContent) or
re.findall(r'\bContext.MODE_WORLD_READABLE\b', fileContent)):
                modeworldreadStr +=", "+ '<a href="'+dirName+'/'+ file
+'" target="_blank">' + os.path.basename(file) +'</a>'

            if (re.findall(r'\bMODE_WORLD_WRITABLE\b', fileContent) or
re.findall(r'\bContext.MODE_WORLD_WRITABLE\b', fileContent)):
                modeworldwriteStr +=", "+ '<a href="'+dirName+'/'+ file
+'" target="_blank">' + os.path.basename(file) +'</a>'
...
```

Πηγαίος Κώδικας 7: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

```
# #content rules
if ((( ("rawQuery(" in fileContent) or ("query("in fileContent) or
("SQLiteDatabase" in fileContent)  or ("execSQL(" in fileContent))) and
(("android.database.sqlite" in fileContent))):

    sqlliteStr +=", "+ '<a href="' +dirName+'/'+ file + '"
target="_blank">' + os.path.basename(file) + '</a>'

if(("setJavaScriptEnabled(true)") in fileContent  and
(".addJavascriptInterface(") in fileContent):
    jsenableStr +=", "+ '<a href="' +dirName+'/'+ file + '"
target="_blank">' + os.path.basename(file) + '</a>'

if((".setWebContentsDebuggingEnabled(true)") in fileContent and
("WebView") in fileContent):
    WebViewdebugStr +=", "+  '<a href="' +dirName+'/'+ file + '"
target="_blank">' + os.path.basename(file) + '</a>'

if(("onReceivedSslError(WebView") in fileContent and (".proceed();") in
fileContent):
    sslerrorStr +=", "+  '<a href="' +dirName+'/'+ file + '"
target="_blank">' + os.path.basename(file) + '</a>'
...
```

Πηγαίος Κώδικας 8: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

```
# find urls

searchPatern =
re.compile(ur'((?:https?://|s?ftps?://|file://|javascript:|data:|www\
d{0,3}[.])[\w().=/;,#:@?&~*+!$%\'{}-]+)', re.UNICODE)
urllist = re.findall(searchPatern, fileContent.lower())

for url in urllist:
    if url not in murls:
        murls[url] = '<a href="' + dirName +'/'+ file + '"
target="_blank">' + os.path.basename(file) + '</a>'

#add values to dictionary

if(ipcStr !=""):
    reDict[AnalisysDescriptions.ipcKey] = ipcStr
if (randomStr != ""):
    reDict[AnalisysDescriptions.randomKey] = randomStr
if (modeworldreadStr != ""):
    reDict[AnalisysDescriptions.modeworldreadKey] = modeworldreadStr
if (modeworldwriteStr != ""):
    reDict[AnalisysDescriptions.modeworldwriteKey] =
modeworldwriteStr
if(privateStr != ""):
    reDict[AnalisysDescriptions.privateKey] = privateStr
if(logStr != ""):
    reDict[AnalisysDescriptions.logKey] = logStr
if(cryptoStr !=""):
    reDict[AnalisysDescriptions.cryptoKey] = cryptoStr
if(httpclientStr !=""):
    reDict[AnalisysDescriptions.httpclientKey] =httpclientStr
if(fileioStr !=""):
    reDict[AnalisysDescriptions.fileioKey] = fileioStr
if(infactStr != ""):
    reDict[AnalisysDescriptions.infactKey] = infactStr
```

Πηγαίος Κώδικας 9: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

```
if(infserStr != ""):
    reDict[AnalisysDescriptions.infserKey] = infserStr
if(infbroStr != ""):
    reDict[AnalisysDescriptions.infbroKey] = infbroStr
if(jsenableStr != ""):
    reDict[AnalisysDescriptions.jsenableKey] = jsenableStr
if(WebViewdebugStr != ""):
    reDict[AnalisysDescriptions.WebViewdebugKey] = WebViewdebugStr
if(sslerrorStr != ""):
    reDict[AnalisysDescriptions.sslerrorKey] = sslerrorStr
if(sslStr != ""):
    reDict[AnalisysDescriptions.sslKey] = sslStr
if(sendataStr != ""):
    reDict[AnalisysDescriptions.sendataKey] = sendataStr
if(dexdebugStr != ""):
    reDict[AnalisysDescriptions.dexdebugKey] = dexdebugStr
if(dex2Str != ""):
    reDict[AnalisysDescriptions.dex2Key] = dex2Str
if(emulatorStr != ""):
    reDict[AnalisysDescriptions.emulatorKey] = emulatorStr
if(dexStr != ""):
    reDict[AnalisysDescriptions.dexKey] = dexStr
if(dexrootStr != ""):
    reDict[AnalisysDescriptions.dextamperKey] = dextamperStr
if(dextamperStr != ""):
    reDict[AnalisysDescriptions.dextamperKey] = dextamperStr
if(dexcertStr != ""):
reDict[AnalisysDescriptions.dexcertKey] = dexcertStr
```

Πηγαίος Κώδικας 10: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

```
if(dsslpinStr != ""):
    reDict[AnalisysDescriptions.dsslpinKey] = dsslpinStr
if(rootStr != ""):
    reDict[AnalisysDescriptions.rootkey] = rootStr
if(drootcheckStr != ""):
    reDict[AnalisysDescriptions.drootcheckKey] = drootcheckStr
if(obfStr != ""):
    reDict[AnalisysDescriptions.obfKey] = obfStr
if(execStr != ""):
    reDict[AnalisysDescriptions.execKey] = execStr
if(serversocketStr != ""):
    reDict[AnalisysDescriptions.serversocketKey] = serversocketStr
if(socketStr != ""):
    reDict[AnalisysDescriptions.socketKey] = socketStr
if(datagramsStr != ""):
    reDict[AnalisysDescriptions.datagramsKey] = datagramsStr
if(msgStr != ""):
    reDict[AnalisysDescriptions.msgKey] = msgStr
if(webviewaddjsStr != ""):
    reDict[AnalisysDescriptions.webviewaddjsKey] = webviewaddjsStr
if(webviewgetStr != ""):
    reDict[AnalisysDescriptions.webviewgetKey] = webviewgetStr
if(webviewpostStr != ""):
    reDict[AnalisysDescriptions.webviewpostKey] = webviewpostStr
if(httpconStr != ""):
    reDict[AnalisysDescriptions.httpconKey] = httpconStr
if(urlconStr != ""):
    reDict[AnalisysDescriptions.urlconKey] = urlconStr
if(jurlStr != ""):
    reDict[AnalisysDescriptions.jurlKey] = jurlStr
if(httpsurlStr != ""):
    reDict[AnalisysDescriptions.httpsurlKey] = httpsurlStr
if(nurlStr != ""):
    reDict[AnalisysDescriptions.nurlKey] = nurlStr
if(notifyStr != ""):
    reDict[AnalisysDescriptions.notifyKey] = notifyStr
if(cellinfoStr != ""):
    reDict[AnalisysDescriptions.cellinfoKey] = cellinfoStr
if(celllocStr != ""):
    reDict[AnalisysDescriptions.celllocKey] = celllocStr
if(subidStr != ""):
    reDict[AnalisysDescriptions.subidKey] = subidStr
```

Πηγαίος Κώδικας 11: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

```
if(devidStr != ""):
    reDict[AnalisysDescriptions.devidKey] = devidStr
if(softverStr != ""):
    reDict[AnalisysDescriptions.softverKey] = softverStr
if(simserialStr != ""):
    reDict[AnalisysDescriptions.simserialKey] = simserialStr
if(simopStr != ""):
    reDict[AnalisysDescriptions.simopKey] = simopStr
if(opnameStr != ""):
    reDict[AnalisysDescriptions.opnameKey] = opnameStr
if(contentqStr != ""):
    reDict[AnalisysDescriptions.contentqKey] = contentqStr
if(refmethodStr != ""):
    reDict[AnalisysDescriptions.refmethodKey] = refmethodStr
if(gsStr != ""):
    reDict[AnalisysDescriptions.gsKey] = gsStr
if(bencodeStr != ""):
    reDict[AnalisysDescriptions.bencodeKey] = bencodeStr
if(bdecodeStr != ""):
    reDict[AnalisysDescriptions.bdecodeKey] = bdecodeStr
if(dexStr != ""):
    reDict[AnalisysDescriptions.dexKey] = dexStr
if(mdigestStr != ""):
    reDict[AnalisysDescriptions.mdigestKey] = mdigestStr
if(datagrampStr != ""):
    reDict[AnalisysDescriptions.datagrampKey] = datagrampStr
if(dextstorageStr != ""):
    reDict[AnalisysDescriptions.dextstorageKey] = dextstorageStr

if(setSeedStr !=""):
    reDict[AnalisysDescriptions.setSeedKey] = setSeedStr
if(md5Str !=""):
    reDict[AnalisysDescriptions.md5Key] = md5Str
if(desStr !=""):
    reDict[AnalisysDescriptions.desKey] = desStr
if(AESECBStr !=""):
    reDict[AnalisysDescriptions.AESECBKey] = AESECBStr
if(sqlliteStr !=""):
    reDict[AnalisysDescriptions.sqlliteKey] = sqlliteStr
```

Πηγαίος Κώδικας 12: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

```
codenat = "<b>Native: </b>" + str(native) + "<br> \n" \
        "<b>Dynamic: </b>" + str(dynamic) + "<br>\n " \
        "<b>Reflection: </b>" + str(reflect) + "<br>\n" \
        "<b>Crypto: </b>" + str(crypto) + "<br>\n" \
        "<b>Obfuscation: </b>" + str(obfus) + "<br>"

return reDict, murls, codenat
```

Πηγαίος Κώδικας 13: η μέθοδος που υλοποιεί την ανάλυση κώδικα (στιγμιότυπο).

Η μέθοδος αυτή θα επιστέψει δυο dictionaries, ένα με τα τυχόν ευρήματα - ευπάθειες , ένα με τα URLs και μια μεταβλητή που είχε html κώδικα.

Όταν η εφαρμογή φτάσει στην διαδικασία της ανάλυσης του κώδικα εκτελούνται τα παρακάτω τμήματα:

Manifest

Βρίσκει το manifest.xml. ελέγξει αν υπάρχει και αν δεν το βρει θα το κάνει extract από το akp και μετά θα το μετατρέψει σε κανονικό xml format με την βοήθεια του προγράμματος AXMLPrinter[18].

```
  def getManifest(self, sourcefile, destfile, sourcefilepath,
tooldir):
    # copy apk file to zip file
    copyfile(sourcefile, destfile)

    # extract only AndroidManifest.xml
    archive = zipfile.ZipFile(destfile)

    for file in archive.namelist():
        if file == 'AndroidManifest.xml':
            archive.extract(file, sourcefilepath)

    # convert AndroidManifest.xml in normal xml format
    mutils.runCommands(
        'java -jar ' + tooldir + '/axmlprinter/AXMLPrinter2.jar ' +
sourcefilepath + '/AndroidManifest.xml > ' + sourcefilepath +
'/manifest.xml')
```

Πηγαίος Κώδικας 14: Στιγμιότυπο της μεθόδου που βρίσκει το manifest.xml

Και καθώς προχωράει η ανάλυση θα το ανοίξει και θα αναλύσει το manifest για το τι υπάρχει πχ. Main activity, permissions, providers, κ.τ.λ.

Ανάλυση κώδικα

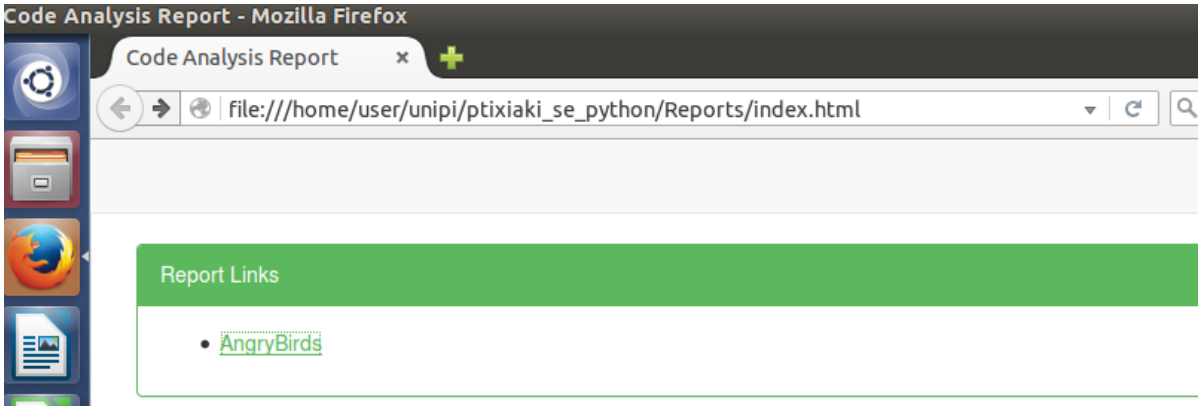
Όπως αναφέραμε και παραπάνω η ανάλυση του κώδικα γίνεται στην κλάση CodeAnalisys και την μέθοδο SiC.

Αυτή μας επιστρέφει την κυρίως ανάλυση του κώδικα με τις τυχόν ευπάθειες που μπορεί να περιέχει, τις πληροφορίες για την χρήση διαφόρων ενεργειών (actions) από το API του android και τα διαφορά urls που περιέχονται μέσα στον κώδικα.

Για κάθε ευπάθεια που βρίσκει η εφαρμογή, υπάρχει δίπλα από την ευπάθεια αυτή είτε κάποια πρόταση είτε κάποιο link εξωτερικού συνδέσμου για το πώς μπορεί να διορθωθεί. Επίσης σε κάποιες ευπάθειες υπάρχει ένα script στην στήλη fix issue script. Αν ο χρήστης εκτελέσει το script αναφέρετε στην στήλη αυτή. Η ευπάθεια θα διορθωθεί στον πηγαίο κώδικα (source code) της εφαρμογής που είχε γίνει η ανάλυση.

Αποτελέσματα

Τα αποτελέσματα της ανάλυσης εμφανίζονται στον default web του συστήματος.



Εικόνα 2: Ο web browser του συστήματος.

Στην αρχή φορτώνεται η αρχική σελίδα που περιέχει τα links σε κάθε report για όλα τα αρχεία που έχει αναλύσει.

Code Analysis

Issue	Severity	FileName	Solution	Fix issues script
The App uses an insecure Random Number Generator.	dangerous	StrobeLight.java aa.java cx.java	Use SecureRandom without Seeding. More Info): SecureRandom	python /home/user/unipi /ptixiaki_se_python /fixissues.py random run above script yo terminal for fixing issues.
The Object is World Writable. Any App can write to the Object	dangerous	test.java	Use other methods eg. such as ContentProvider, BroadcastReceiver, and Service. More Info): MODE_WORLD_WRITEABLE	python /home/user/unipi /ptixiaki_se_python /fixissues.py modeworldwrite run above script yo terminal for fixing issues.

Εικόνα 3: στιγμιότυπο τον αποτελεσμάτων της ανάλυσης

Μετά πατώντας στο link που μας ενδιαφέρει οδηγούμαστε στο αντίστοιχο report.

Για την δημιουργία των reports χρησιμοποιήθηκε το framework bootstrap[3] . Και παράγονται στατικές σελίδες html. Τα data εισάγονται μέσα στην σελίδα με τον παρακάτω τρόπο:

Η εφαρμογή μέσα ανοίγει το κατάλληλο html template και ψάχνει και βρίσκει τα διάφορα labels (ετικέτες) που υπάρχουν μέσα στην html. Οι ετικέτες αυτές είναι της μορφής __[Label Name]__ Πχ. __file_info__. Όταν τις βρει κάνει αντικατάσταση της ετικέτας με το περιεχόμενο που πρέπει να εμφανίσει.

Ευρήματα

Όταν τελειώσει η ανάλυση , στο report, και συγκεκριμένα στην ενότητα ανάλυση κώδικα (Code Analysis) εμφανίζονται διάφορες πληροφορίες από τα ευρήματα που είχε εντοπίσει η εφαρμογή. Τα ευρήματα χωρίζονται σε τέσσερις κατηγορίες

- Πληροφορία (info)
- Ασφαλής (secure)
- Προειδοποίηση (warning)
- Επικίνδυνο (dangerous)

Θα αναφερθούμε στα ευρήματα που χαρακτηρίζονται σαν επικίνδυνα και σαν προειδοποιητικά.

1. **Μήνυμα ευπάθειας**: η εφαρμογή χρησιμοποιεί μια μη ασφαλή γεννήτρια τυχαίων αριθμών (The App uses an insecure Random Number Generator).

Το θέμα που αντιμετώπιζε μια εφαρμογή που είχε χρησιμοποιήσει την βιβλιοθήκη java.util.random, είναι ότι η συγκεκριμένη βιβλιοθήκη δεν χρησιμοποιεί κρυπτογράφηση για την παραγωγή τυχαίων αριθμών (random numbers).
Η βιβλιοθήκη αυτή χρησιμοποιεί μια γεννήτρια ψευδο-τυχαίων αριθμών (generate pseudorandom numbers). Η γεννήτρια αυτή επειδή, είχε σαν αδυναμία να επαναλαμβάνει περιοδικά, κάποιους αριθμούς κατά την παραγωγή τους. Εξαιτίας αυτής της αδυναμίας είναι εύκολο να προβλεφθούν αυτοί οι αριθμοί. Οπότε η βιβλιοθήκη είναι μη ασφαλής.

	PRNG
Efficient (In terms of generating more numbers in less time)	More
Periodic (Repeats itself after some time)	Yes
Deterministic (Sequence can be reproduced at later time)	Yes
Useful	Simulation and Modelling Applications

Εικόνα 4: Χαρακτηριστικά και χρησιμότητα του αλγορίθμου δημιουργίας ψευδο-τυχαίων αριθμών (Pseudo-Random Number Generator)

Σαν λύση μπορεί να χρησιμοποιηθεί η βιβλιοθήκη java.security.SecureRandom που χρησιμοποιεί μια γεννήτρια ψευδο-τυχαίων αριθμών με κρυπτογράφηση (cryptographically strong pseudo random number generator (CSPRNG)), για την παραγωγή τυχαίων αριθμών και είναι ένας ασφαλής τρόπος υλοποίησης.[19][20]

Διορθωτική ενέργεια: όταν ο χρήστης εκτελέσει σε μια κονσόλα τερματικού(terminal) το script που αναφέρει η στήλη fix issues script, γίνεται αντικατάσταση το java.util.random με το java.security.SecureRandom και η κλήση new random() με το new SecureRandom(). [21]

2. **Μήνυμα ευπάθειας**: Το αντικείμενο χρησιμοποιεί World Writable mode. Οποιαδήποτε εφαρμογή μπορεί να γράψει στο αντικείμενο (The Object is World Writable. Any App can write to the Object).

Το πρόβλημα που δημιουργείται όταν χρησιμοποιούμε το mode World Writable είναι ότι μπορούν και άλλες εφαρμογές να χρησιμοποιήσουν τα αρχεία που δημιούργει η εφαρμογή μας. Αυτό γιατί εξ ορισμού τα αρχεία που δημιουργούνται στον εσωτερικό αποθηκευτικό χώρο (internal storage) είναι προσπελάσιμα μόνο από την εφαρμογή που τα δημιούργησε.

Ενώ όταν χρησιμοποιούμε το MODE_WORLD_WRITEABLE δίνουμε την δυνατότητα σε άλλες εφαρμογές να γράψουν στα αρχεία αυτά .[22]

Διορθωτική ενέργεια: όταν ο χρήστης εκτελέσει σε μια κονσόλα τερματικού(terminal) το script που αναφέρει η στήλη fix issues script, γίνεται αντικατάσταση το MODE_WORLD_WRITEABLE με το MODE_PRIVATE. [23]

3. **Μήνυμα ευπάθειας**: Το αντικείμενο χρησιμοποιεί World Readable mode. Οποιαδήποτε εφαρμογή μπορεί να «διαβάσει» το αντικείμενο (The Object is World Readable. Any App can read from the Object).

Το θέμα που δημιουργείται με την χρήση του MODE_WORLD_READABLE είναι το εξής: εξ ορισμού τα αρχεία που δημιουργούνται στον εσωτερικό αποθηκευτικό χώρο (internal storage) είναι προσπελάσιμα μόνο από την εφαρμογή που τα δημιούργησε.

Όταν χρησιμοποιούμε το MODE_WORLD_READABLE δίνουμε την δυνατότητα σε άλλες εφαρμογές να διαβάζουν τα αρχεία αυτά.[22]

Διορθωτική ενέργεια: όταν ο χρήστης εκτελέσει σε μια κονσόλα τερματικού(terminal) το script που αναφέρει η στήλη fix issues script, γίνεται αντικατάσταση το MODE_WORLD_ READABLE με το MODE_PRIVATE [24]

4. **Μήνυμα ευπάθειας**: Είναι ενεργοποιημένο η απομακρυσμένη αποσφαλμάτοση για το WebView (Remote WebView debugging is enabled).

το WebView είναι ένα view που επιτρέπει να ενσωματωθούν web pages (html, css , JavaScript) στην εφαρμογή μας . Επειδή το WebView εκτελεί τον κώδικα της σελίδας, υπάρχει κίνδυνος να εκτελέσει και κακόβουλο κωδικό σαν cross-site-scripting (JavaScript injection).

Ακόμα έχοντας το setWebContentsDebuggingEnabled(true) είναι ενεργοποιημένο το debug mode και μπορεί να τρέξει οποιοδήποτε κώδικα JavaScript.[22] [25]

Διορθωτική ενέργεια: όταν ο χρήστης εκτελέσει σε μια κονσόλα τερματικού(terminal) το script που αναφέρει η στήλη fix issues script, γίνεται αντικατάσταση το .setWebContentsDebuggingEnabled(true) με το .setWebContentsDebuggingEnabled(false).[26]

5. **Μήνυμα ευπάθειας**: Η εφαρμογή χρησιμοποιεί τον «αδύναμο» αλγόριθμο MD5 (Use weak algorithms - MD5).

Ο αλγόριθμος MD5 θεωρείτε μη ασφαλής. Το 2004 ανακαλύφτηκαν πολλά ελαττώματα για τον md5. Πιο συγκεκριμένα δημιουργήθηκε ένα ζεύγος αρχείων που μοιραζόντουσαν το ίδιο md5 hash. Σαν αποτέλεσμα ο συγκεκριμένος αλγόριθμος είναι μη ασφαλής.[27]

Διορθωτική ενέργεια: όταν ο χρήστης εκτελέσει σε μια κονσόλα τερματικού(terminal) το script που αναφέρει η στήλη fix issues script, γίνεται αντικατάσταση το MD5 με το sha512. [28]

6. **Μήνυμα ευπάθειας**: Η εφαρμογή χρησιμοποιεί τον αλγόριθμο κρυπτογράφησης DES (use DES).

ο αλγόριθμος κρυπτογράφησης DES, the Data Encryption Standard, δεν μπορεί να θεωρηθεί ασφαλής. Επειδή χρησιμοποίει μικρό κλειδί κρυπτογράφησης 56-bit. Έτσι είναι ευάλωτος σε επιθέσεις τύπου brute-force search. [29]

Διορθωτική ενέργεια: όταν ο χρήστης εκτελέσει σε μια κονσόλα τερματικού(terminal) το script που αναφέρει η στήλη fix issues script, γίνεται αντικατάσταση το DES με το AES.

7. **Μήνυμα ευπάθειας**: Μη ασφαλής υλοποίηση του SSL. Η εφαρμογή εμπιστεύεται όλα τα πιστοποιητικά SSL ή δέχεται ένα αυτο-υπογεγραμμένο (self-sign) πιστοποιητικό. (Insecure Implementation of SSL. Trusting all the certificates or accepting self signed certificates is a critical Security Hole).

Το πρόβλημα είναι ότι η εφαρμογή θεωρεί όλα τα ssl πιστοποιητικά ότι είναι έμπιστα και ασφαλή. Στην πραγματικότητα όλα τα λειτουργικά θεωρούν ασφαλή μόνο οσα πιστοποιητικά έχουν εκδοθεί από κάποια αρχή έκδοσης πιστοποιητικού.

Το πρωτόκολλο SSL (Secure Sockets Layer), γνωστό και ως Transport Layer Security (TLS), είναι ένα κοινό πρωτόκολλο κρυπτογραφημένης επικοινωνίας μεταξύ πελάτη και εξυπηρετητή (client - server). Είναι πιθανόν αν μια εφαρμογή χρησιμοποιεί το SSL χωρίς να έχει εγκατασταθεί και παραμετροποιηθεί σωστά, μια κακόβουλη οντότητα να της κλέψει δεδομένα μέσο του δικτύου.

Ένα τυπικό σενάριο χρήσης του SSL είναι το εξής: Σε έναν server έχει εγκατασταθεί ένα πιστοποιητικό SSL. Το πιστοποιητικό περιέχει το δημόσιο κλειδί (public key) και είναι συνδυασμένο με ένα προσωπικό κλειδί (private key). Σαν μέρος της χειραψίας (handshake) μεταξύ του client και του server , ο server αποδεικνύει ότι έχει το προσωπικό κλειδί μέσο ενός υπογεγραμμένου πιστοποιητικού με κρυπτογραφία δημοσίου κλειδιού.

Έτσι με αυτόν τον τρόπο ο καθένας δημιουργεί το δικό του πιστοποιητικό και το δικό του προσωπικό κλειδί. Μια απλή χειραψία (handshake) δεν αποδεικνύει τίποτα άλλο από το ότι ο server γνωρίζει το προσωπικό κλειδί το οποίο είναι συνδυασμένο με το δημόσιο κλειδί του πιστοποιητικού. Ένας τρόπος για να λυθεί αυτό το πρόβλημα είναι ο πελάτης (client) να εγκαταστήσει ένα ή περισσότερα πιστοποιητικά που να είναι έμπιστα. Αν κάποιο πιστοποιητικό δεν περιλαμβάνετε σε αυτά που έχουν εγκατασταθεί, ο server δεν μπορεί να είναι έμπιστος.

Τα πιο συνηθισμένα προβλήματα εξακρίβωσης πιστοποιητικών είναι:

- Η αρχή έκδοσης του πιστοποιητικού (Certification Authority, CA) είναι άγνωστη.
- Το πιστοποιητικό του server δεν είναι υπογεγραμμένο από κάποια αρχή έκδοσης (Certification Authority, CA)
- Λείπει από την παραμετροποίηση του server ο μεσολαβητής με την αρχή έκδοσης (Certification Authority, CA)

Άγνωστη αρχή έκδοσης πιστοποιητικού έκδοσης (Certification Authority, CA)

Σε αυτή την περίπτωση η κλάση SSLHandshakeException εμφανίζει ένα μήνυμα εξαίρεσης (exception message), επειδή η αρχή έκδοσης (Certification Authority, CA) του πιστοποιητικού δεν είναι ακόμα έμπιστη από το σύστημα. Αυτό μπορεί να σημάνει ότι η αρχή έκδοσης δεν είναι κάποια γνωστή – δημόσια αρχή έκδοσης αλλά να είχε εκδοθεί το πιστοποιητικό από κάποιον προσωπικό εκδότη πχ κάποια επιχείρηση ή εκπαιδευτικό ίδρυμα.

Μπορούμε να «εκπαιδεύσουμε» το HttpsURLConnection να εμπιστευτεί κάποια συγκεκριμένα πιστοποιητικά από την άγνωστη αρχή έκδοσης.
Για να δημιουργήσουμε έναν TrustManager, θα πρέπει να χρησιμοποιήσουμε την κλάση InputStream και να δημιουργήσει ένα KeyStore. Με αυτό το KeyStore θα δημιουργηθεί ο TrustManager.

Ένας TrustManager είναι μια κλάση που χρησιμοποιεί το σύστημα για να ελέγξει τα πιστοποιητικά από τον server, και να θεωρήσει έμπιστες τις αρχές έκδοσης.

Αυτό-Υπογεγραμμένο πιστοποιητικό

Άλλη μια περίπτωση που η κλάση SSLHandshakeException εμφανίζει ένα μήνυμα εξαίρεσης (exception message) είναι αν το πιστοποιητικό είναι αυτό-υπογεγραμμένο (self-sign) και το σύστημα δεν το θώρει έμπιστο.
Σαν λύση, όπως αναφέρθηκε και παραπάνω, μπορούμε μεσώ του TrustManager να δηλώσουμε στο σύστημα ότι το συγκεκριμένο πιστοποιητικό είναι έμπιστο.

Λείπει από τον server ο μεσολαβητής με την αρχή έκδοσης (Certification Authority, CA)

Η τρίτη περίπτωση που η κλάση SSLHandshakeException εμφανίζει ένα μήνυμα εξαίρεσης (exception message) είναι αν δεν υπάρχει ο μεσολαβητής με την αρχή έκδοσης στον server. Οι περισσότερες δημοσιές αρχές έκδοσης, δεν υπογράφουν στα πιστοποιητικά ευθέως αλλά χρησιμοποιούν μια αναφορά γνωστή ως root CA. Σαν root CA μπορεί να αποθηκευτεί χωρίς σύνδεση (offline) για να μειωθεί η πιθανότατα διαρροής. Κάποια λειτουργικά συστήματα σαν το Android, εμπιστεύονται μόνο το root CA. Αυτό δημιουργεί μια έλλειψη εμπιστευτικότητας μεταξύ του server και της αρχής που θα επιβεβαιώσει αν το πιστοποιητικό είναι σωστό.
Αυτό το πρόβλημα μπορεί να λυθεί αν ο server δεν στείλει μόνο το πιστοποιητικό κατά την SSL χειραψία (handshake) , αλλά στείλει μια «αλυσίδα» από πιστοποιητικά. [41][42]

Για να είναι ασφαλής η εφαρμογή θα πρέπει κατά την υλοποίηση της να δέχεται μόνο πιστοποιητικά που να έχουν εκδοθεί από κάποια αρχή έκδοσης πιστοποιητικού (Certification Authority, CA) [30]

Όσον αφορά τα αυτό-υπογεγραμμένα όσο και προσεκτικά και με ασφαλή τρόπο να τα έχουμε δημιουργήσει, πάντα θα θεωρούνται μη ασφαλή μια και όλοι οι browsers και τα λειτουργικά συστήματα όσο και οι χρήστες θα τα θεωρούν μη έμπιστα. [31]

8. **Μήνυμα ευπάθειας**: Μη ασφαλή υλοποίηση του WebView. Το WebView αγνοεί τα λάθη από το SSL (insecure WebView Implementation. WebView ignores SSL Certificate Errors).

Το WebView είναι ένα view για να εμφανίζει ιστοσελίδες (web pages)[32]

Αυτή η κλάση χρησιμοποιείτε όταν θέλουμε να φταίξουμε τον δικό μας web browser ή όταν θέλουμε να εμφανίσουμε τα περιεχόμενα μιας σελίδας μέσα στο Activity μας.[33]

Το WebView αγνοεί τα λάθη που πιθανόν να προκύπτουν από ssl πιστοποιητικά που δεν είναι έμπιστα ή έχουν λήξει , έτσι συνεχίζει να εκτελείται η εφαρμογή κανονικά χωρίς να παρέχει κάποια ενημέρωση στον χρήστη. Το ότι αγνοεί τα λάθη είναι ένα κενό ασφαλείας γιατί με αυτόν τον τρόπο υπάρχει περίπτωση να εκθέσει δεδομένα λόγο τις μη ασφαλούς επικοινωνίας με κάποιο server.

Η εφαρμογή θα μπορούσε να δέχεται μόνο τα σωστά και ασφαλή πιστοποιητικά ssl, και να απορρίπτει όσα εμφανίζουν λάθη χωρίς να κάνει κάποια προσπάθεια να επικοινωνήσει με τον server που δεν είχε στο σωστό πιστοποιητικό.

Αυτό μπορούμε να το πετύχουμε αν αντί για την μέθοδο proceed() χρησιμοποιήσουμε την μέθοδο cancel().

Παράδειγμα:

```
void onReceivedSslError (WebView view, SslErrorHandler handler, SslError error)
{
    handler.cancel();
}
```

9. **Μήνυμα ευπάθειας**: Μη ασφαλή υλοποίηση του WebView. Εκτελεί ελεγχόμενο κώδικα από τον χρήστη. (Insecure WebView Implementation. Execution of user controlled code in WebView is a critical Security Hole).

Η ευπάθεια αυτή αναφέρετε στο ότι το webview επιτρέπει να εκτελεστεί JavaScript κώδικας. Έτσι κάποιος μπορεί να εκτελέσει και κακόβουλο κώδικα και να εκθέσει σε κίνδυνο ευαίσθητα δεδομένα. [35] [36] [37]

Το WebView εκτελεί- μεταφράζει περιεχόμενο από τον ιστό (web), αυτό το περιεχόμενο περιέχει HTML και JavaScript. Μη σωστή χρήση του WebView μπορεί να προκαλέσει προβλήματα ασφαλείας όπως cross-site-scripting (JavaScript injection). Το Android περιλαμβάνει διάφορους μηχανισμούς για να μειώσει την εμφάνιση τέτοιων προβλημάτων.

Αν η εφαρμογή που φτιάχνουμε δεν χρησιμοποιεί απευθείας JavaScript μέσα στο WebView, δεν θα πρέπει να χρησιμοποιήσουμε την μέθοδο setJavaScriptEnabled(). Εξ ορισμού το WebView δεν εκτελεί κώδικα JavaScript επομένως οι επιθέσεις τύπου cross-site-scripting δεν μπορούν να συμβούν.

Αν θέλουμε να εκτελέσουμε κώδικα JavaScript, μπορούμε να χρησιμοποιήσουμε , με προσοχή, την μέθοδο addJavaScriptInterface(). Αν την χρησιμοποιήσουμε, εκτελεί κώδικα μόνο από αξιόπιστες σελίδες.
Αν κάποιος μη αξιόπιστος κωδικός JavaScript εκτελεστεί, μπορεί να καλέσει μεθόδους που περιχέονται στην εφαρμογή μας . Σε γενικές γραμμές είναι προτεινόμενο να εκτελούμε μόνο JavaScript κώδικα που είναι μέσα στην εφαρμογή μας.

Αν η εφαρμογή μας έχει πρόσβαση σε ευαίσθητα δεδομένα θα πρέπει να χρησιμοποιήσουμε την μέθοδο clearCache(), ώστε να σβηστούν όλα τα τοπικά αρχεία. [43]

Θα έπρεπε να διατηρήσουμε το setJavaScriptEnabled(false) που είναι και η default κατάσταση και όχι να αλλάξει σε setJavaScriptEnabled(true) ώστε να μην μπορεί να τρέξει JavaScript κώδικα.

10. **Μήνυμα ευπάθειας**: Στα αρχεία μπορεί να περιέχονται «ευαίσθητα» δεόμενα. (Files may contain hardcoded sensitive informations like usernames, passwords, keys, etc.).

Όταν τα δεδομένα είναι αποθηκευμένα χωρίς να είναι κρυπτογραφημένα, υπάρχει ο κίνδυνος να κλαπούν η να αλλοιωθούν.

Θα μπορούσαμε να χρησιμοποιήσουμε κρυπτογραφημένα passwords hardcoded, αλλά να μην έχουμε στο ίδιο αρχείο το κλειδί κρυπτογράφησης, γιατί έτσι δεν θα λύναμε το πρόβλημα επειδή θα έπρεπε να υπάρχει και το κλειδί κρυπτογράφησης μέσα στον κώδικα κύριος αν ο αλγόριθμος κρυπτογράφησης είναι συμμετρικός π.χ. AES.

Εναλλακτικά μπορούμε να κάνουμε τα παρακάτω:

- Να ρωτάμε τον χρήστη για το password: Σε πολλές περιπτώσεις ο χρήστης γνωρίζει το password και όχι η εφαρμογή. Είναι η καλύτερη λύση για όπου μπορεί να εφαρμοστεί.

- Να κρατάμε τα ευαίσθητα δεδομένα σε διαφορετικά αρχεία: Είναι ευκολότερο να φυλάμε ξεχωριστά αρχεία config που να περιχέουν ευαίσθητα δεδομένα, από το να έχουμε τα δεδομένα αυτά διάσπαρτα μέσα στον κώδικα. Σαν αρχεία μπορούν να έχουν διαφορετικά δικαιώματα ώστε να μην μπορούν να διαβαστούν από όλους.
- Να χρησιμοποιηθεί κρυπτογραφία: Μπορούμε να κρυπτογραφήσουμε τα δεδομένα που θεωρούμε ευαίσθητα με διάφορους κρυπτογραφικούς αλγορίθμους σαν AES, 3DES, RSA, οι οποίοι παρέχονται μέσα από διάφορες βιβλιοθήκες,
- Να χρησιμοποιήσουμε μια βάση δεδομένων για να αποθηκεύσουμε τα δεδομένα αυτά.
- Να χρησιμοποιήσουμε συναρτήσεις hash για τα password του χρήστη. Πχ. Sha256, sha512. Κυρίος αν πρέπει να φυλάγετε το password στο σύστημα του χρήστη. [34]

11. **Μήνυμα ευπάθειας**: Η εφαρμογή μπορεί να απαιτεί δικαιώματα root. (This App may request root (Super User) privileges).

Όταν μια εφαρμογή εκτελείται με δικαιώματα root τότε είχε πρόσβαση σε όλη την συσκευή και σε όλο το λειτουργικό σύστημα. Με αυτόν τον τρόπο το σύστημα μας είναι εκτεθειμένο σε κάθε είδους απειλές. Θα πρέπει να μη απαιτεί η εφαρμογή root δικαιώματα γιατί δεν είναι ασφαλής.

Μέσο του manifest στην περιοχή <permissions> ορίζουμε τι δικαιώματα θα ζητάει η εφαρμογή κατά την εγκατάσταση της σε κάποιο σύστημα. Ακόμα μέσα από κώδικα μπορεί να ξεκινήσει κάποια διεργασία με δικαιώματα root. [38][39]

12. **Μήνυμα ευπάθειας**: Η εφαρμογή διαβάζει / γραφεί σε εξωτερικό αποθηκευτικό χώρο. Οποιαδήποτε εφαρμογή μπορεί να διαβάσει τα δεδομένα που έχουν γραφεί σε εξωτερικό χώρο αποθήκευσης. (App can read/write to External Storage. Any App can read data written to External Storage).

Τα αρχεία που αποθηκεύονται στον εξωτερικό χώρο αποθήκευσης (external storage) σαν τις κάρτες sd, είναι προσπελάσιμα για εγγραφή ή ανάγνωση από όλες τις εφαρμογές ή άλλους χρήστες.

Αυτό γιατί μια κάρτα sd μπορεί να αφαιρεθεί από τον χρήστη και επίσης οποιαδήποτε εφαρμογή μπορεί να τροποποιήσει τα αποθηκεμένα αρχεία.

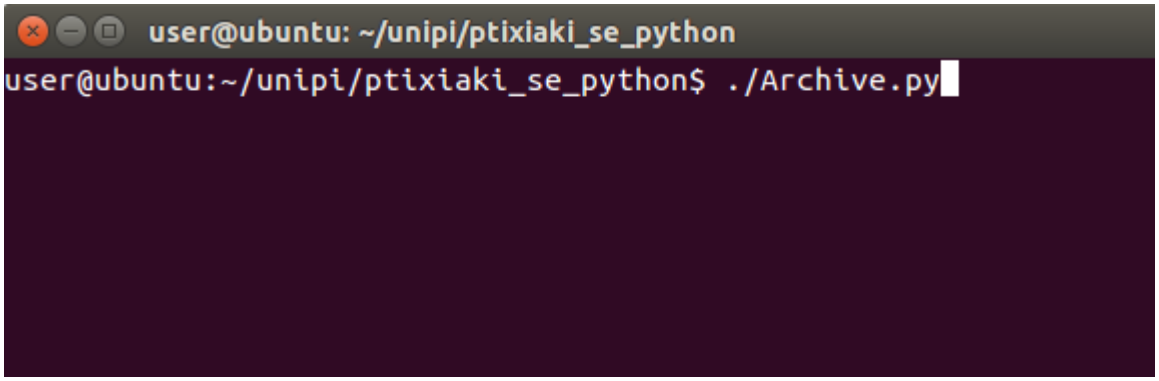
Αν θέλουμε να η εφαρμογή μας να χειρίζεται ευαίσθητα δεδομένα δεν θα πρέπει τα δεδομένα αυτά να αποθηκεύονται στον εξωτερικό χώρο αποθήκευσης.

Επίσης για δεδομένα που θεωρούνται ότι προέρχονται από μη έμπιστες πηγές θα πρέπει να μπορούμε να τα ελέγχουμε για την ακεραιότητα και την ασφάλεια τους.

Σύμφωνα με τον επίσημο οδήγο για τους προγραμματιστές που γράφουν εφαρμογές για το λειτουργικό σύστημα android, είναι προτεινόμενο να μην αποθηκεύονται εκτελέσιμα αρχεία ή αρχεία με κλάσεις στον εξωτερικό χώρο αποθήκευσης. Αν μια εφαρμογή χρησιμοποιεί εκτελέσιμα αρχεία από την κάρτα sd θα πρέπει αυτά τα αρχεία να είναι κρυπτογραφημένα και να γίνετε επαλήθευση πριν από κάθε χρήση. [40]

Αρχειοθέτηση

Ο χρήστης έχει την δυνατότητα να αρχειοθετήσει τα διάφορα sources που έχουν γίνει decompile. Αυτό το πετυχαίνει με το να εκτελέσει το script Archive.py.
Το script θα μετακινήσει τα περιεχόμενα του φακέλου source στον φάκελο sourcearchive.



Εικόνα 5: script αρχειοθέτησης

Κεφάλαιο 3°

Συμπεράσματα

Μετά από αναλύσεις σε διαφορά αρχεία apk , παρατηρήθηκε ότι οι πιο συνηθισμένες ευπάθειες είναι:

1. η χρήση μη ασφαλούς γεννήτριας τυχαίων αριθμών (insecure Random Number Generator).
2. Η μη ασφαλής υλοποίηση του WebView. Το WebView αγνοεί τα λάθη από το SSL (insecure WebView Implementation. WebView ignores SSL Certificate Errors).
3. Στα αρχεία μπορεί να περιέχονται «ευαίσθητα» δεόμενα. (Files may contain hardcoded sensitive informations like usernames, passwords, keys etc).
4. Η εφαρμογή διαβάζει / γραφεί σε εξωτερικό αποθηκευτικό χώρο. Οποιαδήποτε εφαρμογή μπορεί να διαβάσει τα δεδομένα που έχουν γραφεί σε εξωτερικό χώρο αποθήκευσης. (App can read/write to External Storage. Any App can read data written to External Storage).
5. Μη ασφαλής υλοποίηση του SSL. Η εφαρμογή εμπιστεύεται όλα τα πιστοποιητικά SSL ή δέχεται ένα αυτο-υπογεγραμμένο (self-sign) πιστοποιητικό. (Insecure Implementation of SSL. Trusting all the certificates or accepting self signed certificates is a critical Security Hole).

Όπως αναφέρεται και στην ενότητα «Ευρήματα», οι ευπάθειες αυτές μπορούν να διορθωθούν αν ο προγραμματιστής τις λάβει υπόψη του και ακολουθήσει την κατάλληλη μεθοδολογία ώστε η τελική του εφαρμογή να είναι ασφαλής και χωρίς ή με λιγότερα κενά ασφαλείας.

Βιβλιογραφικές Πηγές

1. Microsoft .NET Framework 4.5 (https://www.microsoft.com/en-us/download/details.aspx?id=30653)
2. Microsoft visual studio express (https://www.visualstudio.com/en-US/products/visual-studio-express-vs)
3. Bootstrap framework (http://getbootstrap.com/)
4. SB Admin free template for bootstrap (http://startbootstrap.com/template-overviews/sb-admin/)
5. JD-Gui (http://jd.benow.ca /)
6. jd-cmd (https://github.com/kwart/jd-cmd)
7. dex2jar (https://github.com/pxb1988/dex2jar), (https://sourceforge.net/projects/dex2jar/)
8. procyon java decompiler (https://bitbucket.org/mstrobel/procyon/downloads) , (https://bitbucket.org/mstrobel/procyon/wiki/Java%20Decompiler)
9. Mobile-Security-Framework (MobSF) (https://github.com/ajinabraham/Mobile-Security-Framework-MobSF)
10. Wikipedia , Android (operating system) (https://en.wikipedia.org/wiki/Android_(operating_system))
11. **Σωκράτης Κάτσικας.** *Ασφάλεια Υπολογιστών.* Πάτρα 2001 : Ελληνικό Ανοιχτό Πανεπιστήμιο. ISBN: 960–538–226–1.
12. Βικιπαίδεια (Wikipedia), (https://el.wikipedia.org/wiki/Στατική_ανάλυση)
13. Βικιπαίδεια (Wikipedia), (https://el.wikipedia.org/wiki/Κακόβουλο_λογισμικό)
14. http://stackoverflow.com/, How can I read the manifest of an Android apk file using C# .Net? , (http://stackoverflow.com/questions/18997163/how-can-i-read-the-manifest-of-an-android-apk-file-using-c-sharp-net)
15. Wikipedia, Reverse engineering - Reverse engineering of software, (https://en.wikipedia.org/wiki/Reverse_engineering)
16. **Teodoro Cipresso,** *SOFTWARE REVERSE ENGINEERING EDUCATION,* Software Reverse Engineering (SRE), Web Supplement to Master's Thesis at San José State University (http://reversingproject.info/)
17. Γλώσσα Python (https://www.python.org/)
18. AXMLPrinter (https://code.google.com/archive/p/android4me/downloads)
19. **INFOSEC INSTITUTE** , *Secure Random Number Generation in JAVA* (http://resources.infosecinstitute.com/random-number-generation-java/)
20. **Android Developer**, *Random* (https://developer.android.com/reference/java/util/Random.html)
21. **Android Developer**, *SecureRandom* (http://developer.android.com/reference/java/security/SecureRandom.html)
22. **Android Developer**, *Security Tips* (https://developer.android.com/training/articles/security-tips.html)
23. **Android Developer**, *Context -MODE_WORLD_WRITEABLE* (https://developer.android.com/reference/android/content/Context.html#MODE_WORLD_WRITEABLE)
24. **Android Developer**, *Context -MODE_WORLD_READABLE* (https://developer.android.com/reference/android/content/Context.html#MODE_WORLD_READABLE)
25. **INFOSEC INSTITUTE** , *Attacks on Android WebViews* (http://resources.infosecinstitute.com/android-hacking-security-part-7-attacks-android-webviews/)

26. **Android Developer**, *WebView - setWebContentsDebuggingEnabled* (https://developer.android.com/reference/android/webkit/WebView.html#setWebContentsDebuggingEnabled(boolean))
27. **Wikipedia**, *MD5* (https://en.wikipedia.org/wiki/MD5)
28. **Wikipedia**, *Secure Hash Algorithm* (https://en.wikipedia.org/wiki/Secure_Hash_Algorithm)
29. **Freewan**, *DES* (http://www.freeswan.org/freeswan_trees/freeswan-1.5/doc/DES.html)
30. **Wikipedia**, *certificate authority* (https://en.wikipedia.org/wiki/Certificate_authority)
31. **Digicert**,.*Security Certificate Errors* (https://www.digicert.com/ssl-support/certificate-not-trusted-error.htm)
32. **Android Developer**, *WebView* (https://developer.android.com/reference/android/webkit/WebView.html)
33. **Android Developer**, *SslErrorHandler* (https://developer.android.com/reference/android/webkit/SslErrorHandler.html)
34. **CERN Computer Security**, *How to keep secrets secret* (https://security.web.cern.ch/security/recommendations/en/password_alternatives.shtml)
35. **trustlook blog**, *Alert: Android WebView addJavascriptInterface Code execution Vulnerability* (http://blog.trustlook.com/2013/09/04/alert-android-webview-addjavascriptinterface-code-execution-vulnerability/)
36. **Android Developer**, *WebView- AddJavascriptInterface* (https://developer.android.com/reference/android/webkit/WebView.html#addJavascriptInterface(java.lang.Object, java.lang.String))
37. **Android Developer**, *WebSettings- setJavaScriptEnabled* (https://developer.android.com/reference/android/webkit/WebSettings.html#setJavaScriptEnabled(boolean))
38. **Android Developer**, *Security Tips - Permissions* (https://developer.android.com/training/articles/security-tips.html#Permissions)
39. **Stackoverflow**, *execute shell command from android* (http://stackoverflow.com/questions/20932102/execute-shell-command-from-android)
40. **Android Developer**, *Security Tips - StoringData* (https://developer.android.com/training/articles/security-tips.html#StoringData)
41. **Android Developer**, *Best Practices for Security & Privacy* (https://developer.android.com/training/best-security.html)
42. **Android Developer**, *Security with HTTPS and SSL* (https://developer.android.com/training/articles/security-ssl.html)
43. **Android Developer**, *Security Tips - WebView* (https://developer.android.com/training/articles/security-tips.html#WebView)