Anushree Sitaram Das
ad1707@rit.edu
Foundation of Artificial Intelligence
Project Report

Security Evaluation Of Android Smartphone

Summary

As android smartphones are the most widely used gadgets, android security has become a major concern. An expert system to evaluate an android device's security with the help of security metrics can be developed to address these concerns. The goal of this project is to develop an android application which can evaluate an android smartphone's security using rule-based expert system. The metrics used for implementing the solution are derived in the paper submitted in the Project one: Hierarchical expert system for security evaluation and its implementation on an Android smartphone. This document mentions the project specifications, prototype implementation and user's guide for running the prototype. As a result a working android application is developed for evaluating an android smartphone's security using rule-based expert system and give recommendations which will help users to improve security on their device.

Requirements

According to the problem statement, an android application needs to be built with the following functionalities:

1. Read parameters from the android smartphone automatically.
2. Evaluate the device's security based on those parameters using expert rules.
3. Display the final evaluation of the device on a well-defined scale.
4. Give recommendations to users to improve security based on the evaluation.

The application's performance (like speed and memory consumption) should be efficient and the application must be reliable and secure. For a quicker execution, the number of expert rules should be less but enough to give reliable evaluation. The rules are developed according to the hierarchical security metrics format i.e. the output of one rule is the input to another rule. The software should be well-tested and have simple GUI for ease of usability. The user should be aware how the conclusions were produced.

Specification

To build a rule-based expert system, the knowledge can be expressed in the form of rules. A rule consists of two parts: the IF part, called the antecedent (premise or condition) and the THEN part called the consequent (conclusion or action). A rule-based expert system is made up of five parts: knowledge base, database, inference engine, explanation facilities and user interface. The of rules for knowledge base can be developed using the security metrics shown in the following table:

Table 1

Metric	Category	Value
API Level	Software Security	Degree of membership to latest and not latest version
Security Patch Date	Software Security	Degree of membership to latest and not latest date
Root Access Status	Hardware Security	True or false
Device Lock Set Status	Hardware Security	True or false
Number of Dangerous App Permissions	Application Security	Number of applications

To incorporate fuzzy logic into our expert system to evaluate android security, the variables API Level and Security Patch Date are related to Software Security Risk based on degree of membership to latest or not latest API Level and Security Patch Date. The fuzzy sets for variables API Level, Security Patch Date and Software Security Risk are depicted in the following graphs:

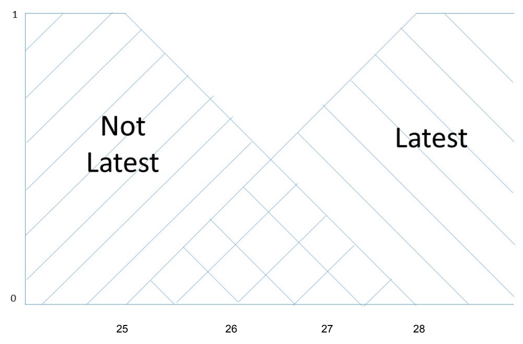


Figure 1. API Level Fuzzy set

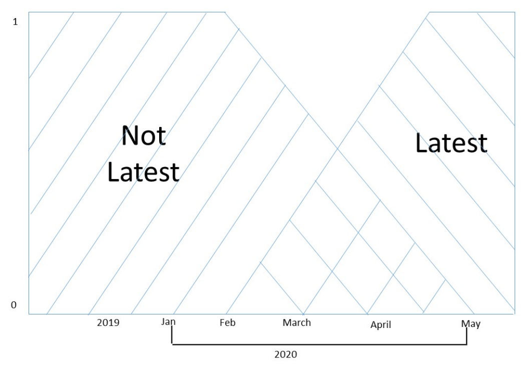


Figure 2. Security Patch Date Fuzzy set

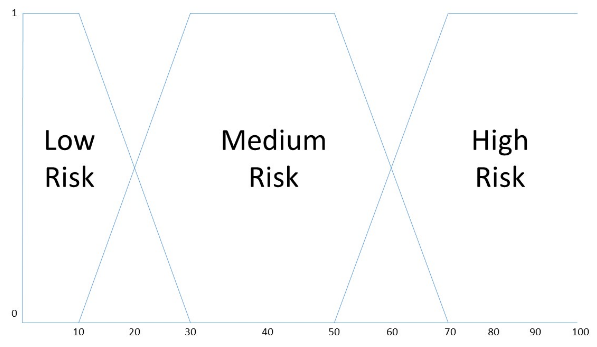


Figure 3. Software Risk Fuzzy set

Finally, weighted average is calculated from Software Security Risk, Hardware Security Risk and Application Security Risk to get final Android Security Risk. The user interface is the means of communication between a user seeking a solution to the problem and an expert system which will be provided by android application developed in Android Studio[7].

Description of the domain problem

System shell used to fill the expert system with knowledge in this project is CLIPS[1]. CLIPS[1] is a type of computer language designed for writing applications called expert systems. C Language Integrated Production System (CLIPS[1]) was developed at NASA's Johnson Space Center from 1985 to 1996, a rule-based programming language useful for creating expert systems and other programs where a heuristic solution is easier to implement and maintain than an algorithmic solution. It was written in C for portability.

It uses Lisp-like s-expressions, which is easy to learn, which is beneficial for this project, since, this project has to be implemented in short amount of time and requires less number of rules. CLIPS[1] is an expert system tool with a complete environment for developing expert systems which includes features such as an integrated editor and a debugging tool.

CLIPS[1] shell performs inferences or reasoning components of a rule-based expert system. The CLIPS[1] shell provides the basic elements of an expert system:

1. fact-list, and instance-list: Global memory for data
2. knowledge-base: Contains all the rules, the rule-base
3. inference engine: Controls overall execution of rules

A program written in CLIPS[1] consists of rules, facts, and objects. The inference engine decides which rules should be executed and when. A rule-based expert system written in CLIPS[1] is a data-driven program where the facts, and objects if desired, are the data that stimulate execution via the inference engine.

In this project, Android Studio[7] is used to build the android application GUI and run the expert system. Since CLIPS[1] is a C language library and Android Studio[7] allows building project using only Java or Kotlin, CLIPS JNI[2] is used for integrating CLIPS[1] with a Java GUI.

Feasibility Study

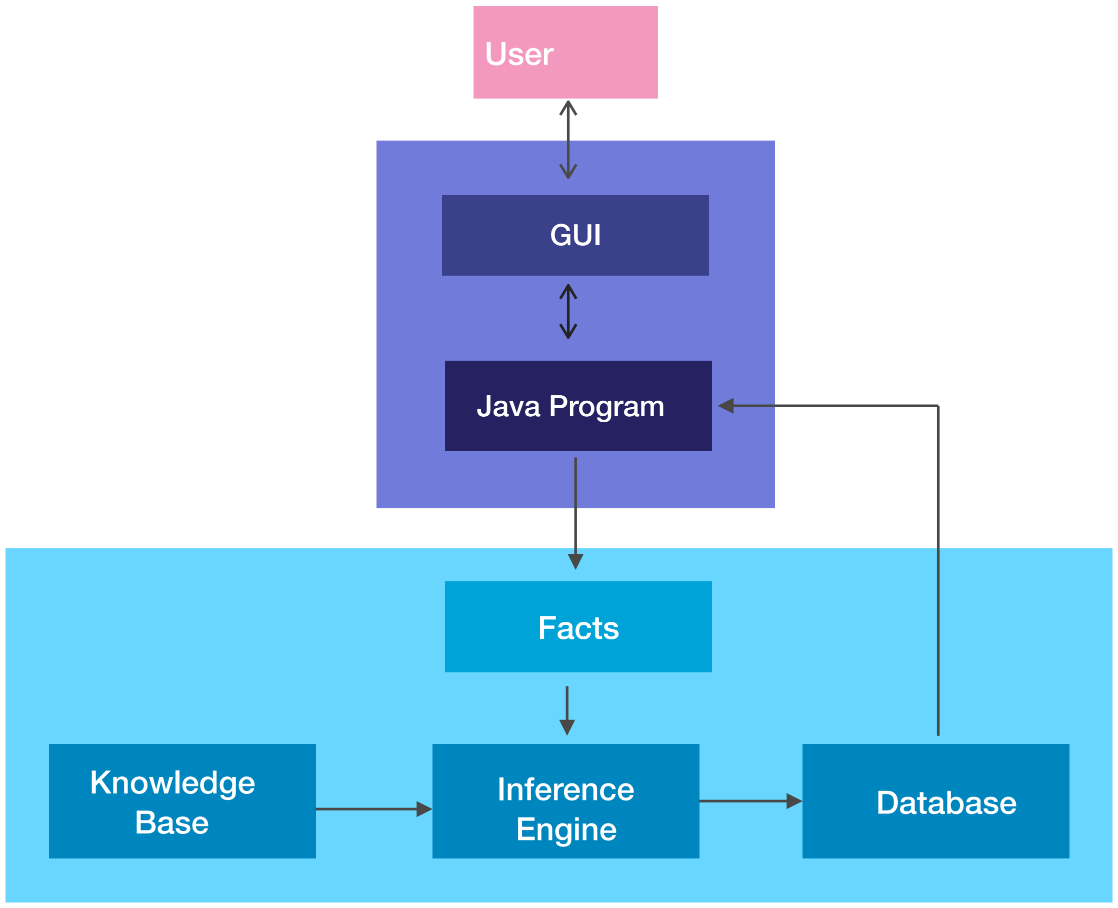
A shell is nothing but an expert system without knowledge base. A shell provides the developers with knowledge acquisition, inference engine, user interface, and explanation facility. There are various expert system shells available like CLIPS[1], Jess[3], Drools ,OpenRules, JXBRE, JEOPS, Roolie, Termware, JRuleEngine[4], Zilonis, But not all work with an android platform. We will consider three of them for this project: Jess[3], CLIPS[1] and JRuleEngine[4].

Jess[3] is a powerful scripting language that gives you access to all of Java's APIs, but it is not open source. It works on android but only when application is developed using Eclipse. Jess[3] is driven by a lisp-style scripting language built in Java which utilizes the Rete algorithm. CLIPS[1] is an open source rule-based programming language useful for creating expert systems written in C. Being written in C makes it highly portable. CLIPS[1] has a special library which can be used for integrating CLIPS[1] with a Java. Libraries like CLIPS4Android[5] and DROID-CLIPS[6] available specifically for android development which is an extension of CLIPS JNI[2]. Rules written in CLIPS[1] uses Lisp-like syntax. It utilizes Rete algorithm as reasoning method. JRuleEngine[4] is an an open source java rule engine. It works on Android. It uses condition-action pattern as rule syntax. This library is a forward-chaining rule engine. This expert system tools lacks documentation especially on how to port to Android Studio[7].

In conclusion, CLIPS[1] expert system tool is the best match for this project since it is easy to port on android platform. This project requires a short and simple rule based expert system which can be implemented with efficiency using CLIPS[1].

Implementation

A rule-based expert system is made up of five parts: knowledge base, database, inference engine, explanation facilities and user interface. CLIPS[1] provide knowledge base, database and inference engine parts of the rule-based expert system. Android application provides the explanation facilities and user interface which access the CLIPS[1] expert system. The flowchart of the application is as follows:

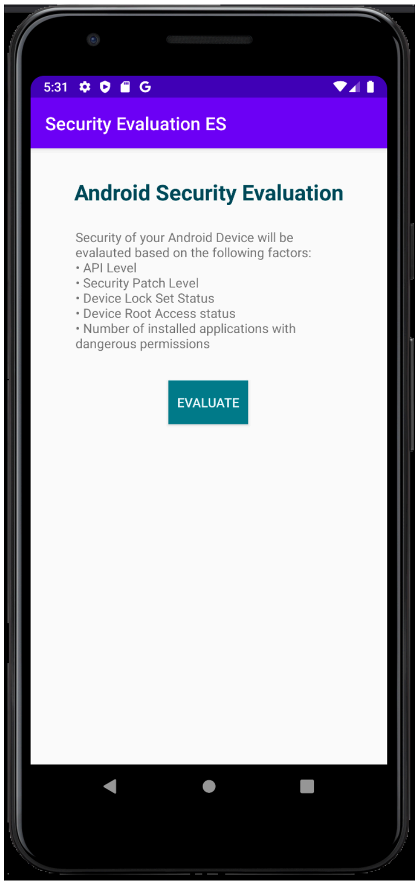


The knowledge base contains all the expert rules which is pre saved in .clp file by using CLIPS IDE. This file can be loaded again in the CLIPS[1] environment using load() command. The database stores all the facts. Facts can inserted using assert() command. The inference engine is in-built in CLIPS[1]. The user interface is designed in android using XML. Software used to develop the Android application is Android Studio[7]. The CLIPS[1] is integrated in the application using CLIPS4Android[5] library which is based on CLIPS JNI[2]. The testing is performed using Android Emulator provided by the Android Studio[7]. The working of the application as follows:

1. The user launches the application.
2. The GUI showed in figure 4 appears.
3. When the user presses the Evaluate button, the button listener is executed.
4. In the button listener, first all the values for variables (like API level and Security Patch Level) is collected.

5. A new CLIPS[1] environment is created and expert rules are loaded in the environment.
6. The values of variables (API level, Security Patch Level, Device Lock Set Status, Root Access Status and Number of Applications with dangerous permissions) collected is added to the CLIPS[1] environment as facts.

Figure 4. Initial GUI after app launch.


5:31

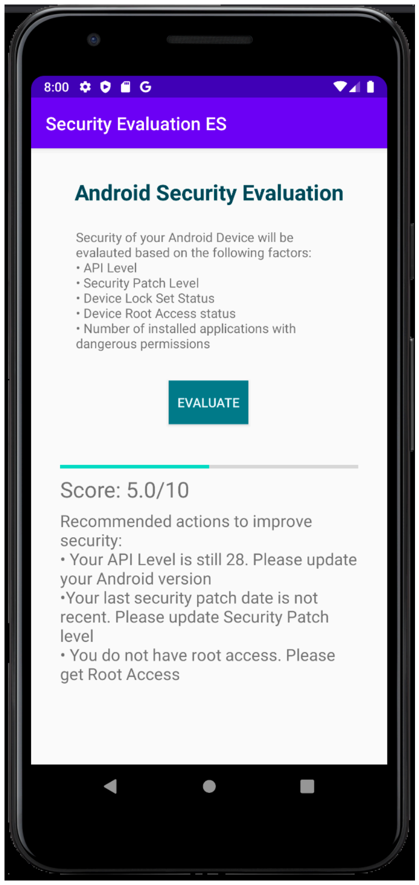
Security Evaluation ES

Android Security Evaluation

Security of your Android Device will be evalauted based on the following factors:
• API Level
• Security Patch Level
• Device Lock Set Status
• Device Root Access status
• Number of installed applications with dangerous permissions

EVALUATE

Figure 5. Screen after evaluation process


8:00

Security Evaluation ES

Android Security Evaluation

Security of your Android Device will be evalauted based on the following factors:
• API Level
• Security Patch Level
• Device Lock Set Status
• Device Root Access status
• Number of installed applications with dangerous permissions

EVALUATE

Score: 5.0/10

Recommended actions to improve security:
• Your API Level is still 28. Please update your Android version
•Your last security patch date is not recent. Please update Security Patch level
• You do not have root access. Please get Root Access

7. Then the inference engine in CLIPS[1] is run.
8. The respective rules from knowledge base will be fired and new facts will be generated in the database. These facts will hold all the values of risk rate factors like Android Security Risk, Software Security Risk, Hardware Security Risk and Application Security Risk.
9. The java program extracts relevant facts from the CLIPS[1] environment.
10. The final android security evaluation is displayed on the UI to the user along with recommendations to improve security. The figure 5 shows the final screen after evaluation.

Some of the rules in the knowledge base are:

1. IF LockSetStatus true AND RootAccessStatus true
   THEN HardwareSecurityRisk 0.0
2. IF LockSetStatus false AND RootAccessStatus true
   THEN HardwareSecurityRisk 50.0
3. IF LockSetStatus true AND RootAccessStatus false
   THEN HardwareSecurityRisk 50.0
4. IF LockSetStatus false AND RootAccessStatus false
   THEN HardwareSecurityRisk 100.0

The above rules show how Hardware Security Risk is calculated.
Similar expert rules can be formed to develop security evaluation expert system for other platforms like Windows machine, iPhone, etc. This application can be run on any android smartphone with API level 21 or more.

Testing

The testing consists of generating a score and advisory for different devices and measuring efficiency for various functionalities for a given test. The testing was performed on 2 Android Virtual Device generated in Android Studio[7] using AVD manager and one android device of my own.

Device 1 - Pixel 3a

Specifications:
API Level: 28
Security Patch Level: Aug 05, 2019
Device Lock: Set
Root Access: No
Number of apps with dangerous permissions: 0
Result: 4.6/10

Device 2 - Nexus 5

Specifications:
API Level: 25
Security Patch Level: June 5, 2017
Device Lock: Not Set
Root Access: No
Number of apps with dangerous permissions: 2
Result: 2.8/10

Device 3 - Redmi Note 7

Specifications:

API Level: 27

Security Patch Level: May 4, 2020

Device Lock: Set

Root Access: No

Number of apps with dangerous permissions: 5

Result: 5.2/10

Conclusion:

The application runs and produces results as per the requirements. The application is fast in terms of speed, taking around 1-2 seconds to evaluate an android smartphone.

User's Guide

To run this application, a user need an android device or an android emulator. There are various android emulators available. But the AVD manager that comes with Android Studio[7] allows the user to create multiple virtual android devices with different specifications.

Steps to install and run application on android smartphone:

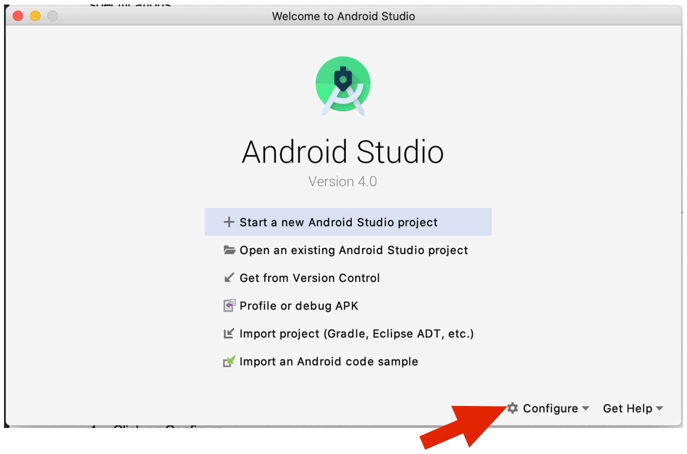
1. Copy the apk provided to your android smartphone.
2. Make sure your device allows installing applications from unknown source.
3. Install the application on your device from the apk file.
4. Launch application.
5. Click Evaluation button.
6. The security score will be displayed on the progress bar.
7. The recommendations will appear right below the score.

Steps to install AVD manager on your laptop or pc:

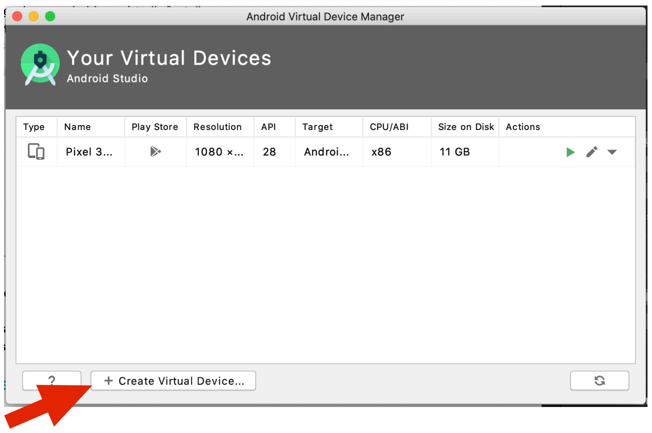
1. Download Android Studio[7] from https://developer.android.com/studio
2. Install Android Studio[7] by following the step mentioned on this website: https://developer.android.com/studio/install

Steps to create new android virtual device:

1. Start Android Studio[7].
2. Click on Configure.


Welcome to Android Studio
Android Studio
Version 4.0
Start a new Android Studio project
Open an existing Android Studio project
Get from Version Control
Profile or debug APK
Import project (Gradle, Eclipse ADT, etc.)
Import an Android code sample
Configure
Get Help

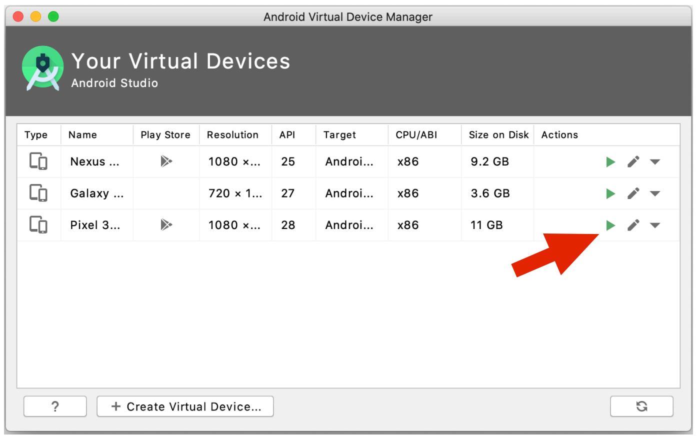
3. Select AVD Manager from the menu.
4. Click on Create Virtual Device.



Type	Name	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	Pixel 3...		1080 ×...	28	Androi...	x86	11 GB	

5. Create an android smartphone of your choosing.

Steps to install and run application on your virtual android device:
1. Open AVD Mananger.
2. Click on “Launch this AVD in emulator” button.



Type	Name	Play Store	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	Nexus ...		1080 ×...	25	Androi...	x86	9.2 GB	
	Galaxy ...		720 × 1...	27	Androi...	x86	3.6 GB	
	Pixel 3...		1080 ×...	28	Androi...	x86	11 GB	

3. Select and drag the provided apk to the virtual android device.
4. The application will get installed automatically.
5. Launch application.
6. Click Evaluation button.
7. The security score will be displayed on the progress bar.
8. The recommendations will appear right below the score.

References

1. CLIPS http://www.clipsrules.net/
2. CLIPS JNI http://www.clipsrules.net/CLIPSJNI.html
3. Jess https://jess.sandia.gov/
4. JRuleEngine http://jruleengine.sourceforge.net/
5. CLIPS4Android https://github.com/gomezgoiri/CLIPS4Android
6. DROID-CLIPS https://github.com/DrItanium/DROID-CLIPS
7. Android Studio https://developer.android.com/studio