Unsolvable Solutions

Client: CSIR DSSR

Functional Requirements

Eavesdropping Protection in Conclave

Members:	*Student Number:*
Edwin Fullard	12048675
Jaco Bezuidenhoudt	11013878
Jandre Coetzee	10693077
Maret Stoffberg	11071762
Ryno Pierce	12003922

Contents

1 Introduction

1.1 Project Background

The Android Operating System officially took over the smartphone market in 2010 and it is suspected that about 700 000 Android devices are used in South Africa. It is mostly the corporate or more upper class communities that have access to these smartphone devices. It is also these individuals who sit in the big corporate meetings where extremely sensitive data can be discussed. For this reason, if these individuals should have eavesdropping malware on their smartphone, it could cause sensitive data to be easily leaked out.

1.2 Project Vision

This project consist of two unique parts: the eavesdropping malware and the protection against it.

1.3 Project Scope

This EPIC (Eavesdropping Protection in Conclave) project consist of a server, an Android application and a gateway device. The android device is held over the gateway node and, using NFC, a request to enter the meeting is send to the server via the gateway device. The server then respond with permission or denial. If permission is granted, the android device will proceed into protection mode and the meeting log is updated. After the meeting has been held, the device will then be held over the gateway node again to deactivate the protection mode. A user may query the log of a meeting.
The eavesdropping malware use a server and an application on the android device. Ideally the application will be hidden behind another application, but for the sake of this project it will be a visible application. A user will send an eavesdropping request from the server to a specified android device. If this device have the malware installed, it will start streaming the recording to the user. The user may then send a request to stop the recording and store the recording.

2 Functional Requirements

2.1 Server

Scope: The server is used with the gateway, it sends and receives data to and from the gateway. On the server an administrative user must also be able to add, remove, edit and view the users, administrative users included. Any user must be able to log in and log out of the system on the server. When logged in the user may create, update, delete and request summary from a meeting. The user may also view and edit his own profile.

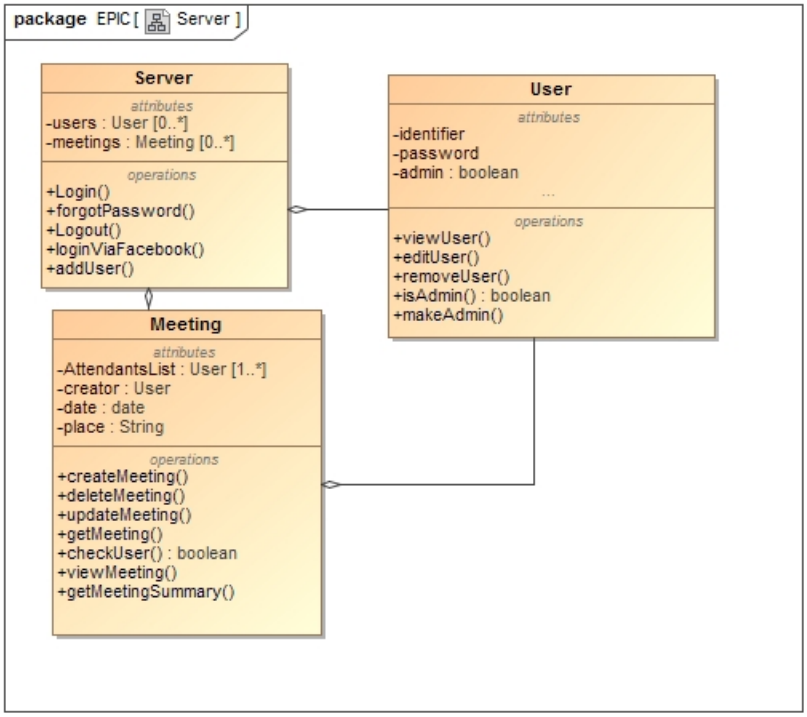

Figure 1: A Class Diagram of the Server

2.1.1 Server

- **Login**
  Priority - Critical
  The user log into the system using a registered identifier and password, or the user may also log in via facebook.

- **Logout**

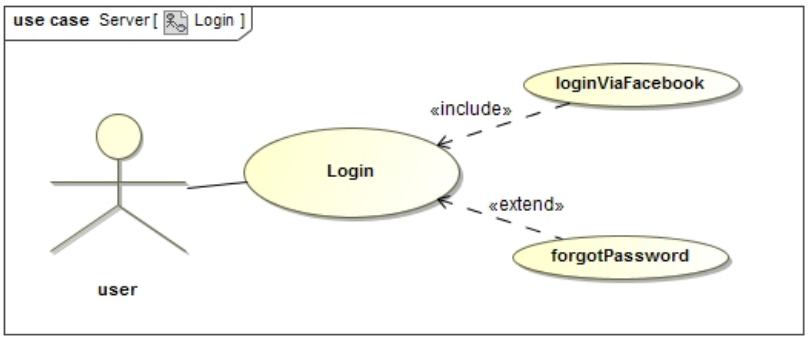

Figure 2: A Use Case Diagram of the Login services

Priority - Critical
This service allows the user to log out if he is currently logged in.

- **addUser**
  Priority - Critical
  The specified user is added to the system, but this service is only applicable to administrative users.

2.1.2 User

- **isAdmin**
  Priority - Important
  This service respond if the specified user is an administrative user.

- **makeAdmin**
  Priority - Important
  Change the specified user to an administrative user. This service may only be done by another administrative user.

- **removeUser**
  Priority - Important
  The specified user is removed from the system, but this service is only applicable to administrative users.

- **viewUser**
  Priority - Important
  View all the specified users data, but this service is only applicable to administrative users.

2.1.3 Meeting

- **createMeeting**
  Priority - Critical
  This service create a meeting. On creation the list of users attending the meeting must be added as well as the time and place of the meeting.

- **viewMeeting**
  Priority - Important
  This service allows a user to view all the participants and the time and place of a meeting. An administrative user may view any meeting, but a common user may only view a meeting that they are attending.

- **updateMeeting**
  Priority - Important
  This service allows the creator of the meeting or any administrative user to add or remove users to a meeting, as well as change the date, time or place.

- **deleteMeeting**
  Priority - Important
  This service allows the creator of the meeting or any administrative user to delete a meeting.

- **getMeetingSummary**
  Priority - Important
  This service give the user a list of all the users that attended the meeting as well as their entrance and exit times. This service is only active after the meeting has taken place.

2.2 Application

Scope: The Application is used when entering a meeting. The user will hold the android device over a gateway node. The application will send the identifier to the server via the Node and Gateway. The application will then, after confirmation from the Node, take a snapshot and enable the protection. After the meeting have occurred, the user will then hold the device over the Node again, the protection will be disabled and the phone will be stored to its previous state.

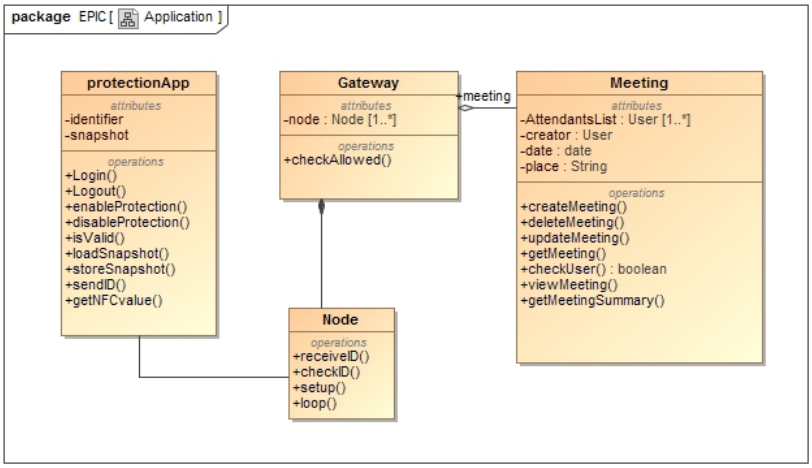

Figure 3: A Class Diagram of the protection Application, Meeting, Node and Gateway

2.2.1 protectionApp

- **onNewIntent**
  Priority - Critical
  This function is called when an Android system intent is triggered. It checks whether the intent was caused by NFC and then parses the NFC message that will check whether the user is allowed to enter the room. If the user is allowed, the store- and loadSnapshot functions are called respectively on whether the user is entering or leaving.

- **StoreSnapshot**
  Priority - Critical
  This function stores the state of all communication mechanisms and then turns them off to enter *safe mode*.

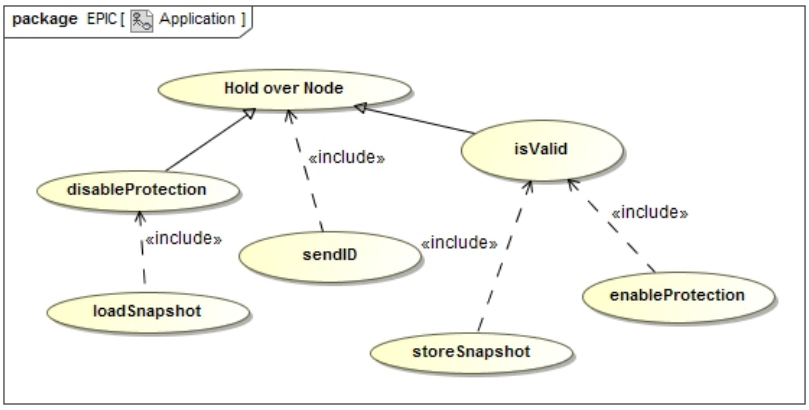

Figure 4: A Use Case Diagram protection Application

- **LoadSnapShot**
  Priority - Critical
  This function restores the previous state of the phones connections that were saved with the *StoreSnapshot* function.

- **processCommandApdu**
  Priority - Critical
  This function is an overwritten function. It is used to create the user authentication data and return it when the phone is being read from via NFC.

- **StoreEmpID**
  Priority - Critical
  This function stores a combination of fields entered by the user into a file that is used to with other functions to either retrieve the data if the application is closed and reopen, or send it via NFC to get use as authentication.

- **getDeviceId**
  Priority - Important
  Used to added an extra security layer to authentication by getting the unique device id.

- **unitTests**
  Priority - Critical
  This function uses dummy scenario's to test whether all of the functions work as they should.

- **setActivityBackgroundColor**
  Priority - Nice to have
  This function changes the background colour of the application to visually notify the user if he has been granted access to the meeting room.

- **convertStreamToString**
  Priority - Nice to have
  This function helps to convert an input stream of bytes into a *String type* that is used for file reading.

2.3 Gateway and Node

Scope: The Gateway consist of a Gateway and Nodes, connected in serial. The gateway node use NFC to communicate with the Android device. After the Android device is scanned the Node sends the identifier to the Gateway to see if the user may have access to the meeting and to update the meeting log.

2.3.1 Node

- **receiveID**
  Priority - Critical
  Receive the identifier from the Android devices via NFC .

- **checkID**
  Priority - Critical
  This service verify if the identifier may access the meeting on the Gateway.

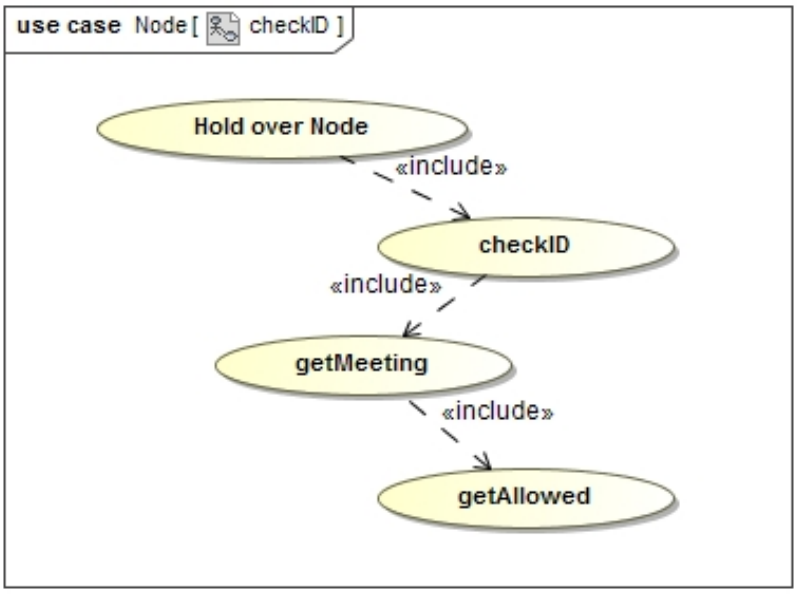

Figure 5: A Use Case Diagram of the CheckID service

- **setup**
  Priority - Critical
  Initialize everything

- **loop**
  Priority - Critical
  run in a loop and call the receiveID service until an identifier is returned, it then call the checkID function.

2.3.2 Gateway

The gateway receives the identifier from the node and verify the meeting access with the server. It also updates the meeting via the server.

- **getMeeting**
  Priority - Critical
  Return the current meeting on loaded on the gateway device.

- **getAllowed**
  Priority - Critical
  Return if the specified user is allowed in the meeting.

2.4 Malware

Scope: The malware consist of a webserver that connects with the mobile device via a web socket. A request is send from the server to the application on the mobile device to start recording. As the device records, the result is streamed directly to the server.

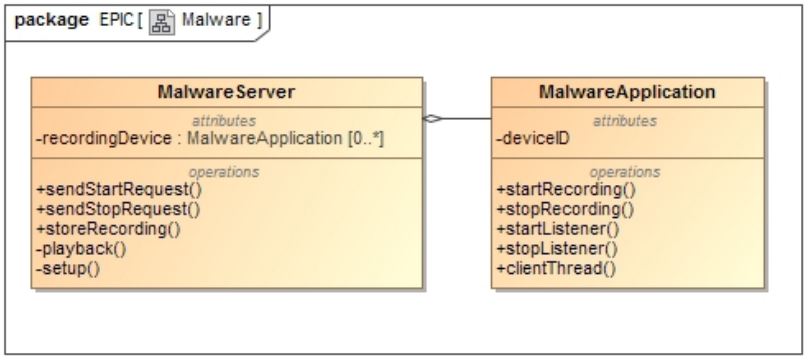

Figure 6: A Class Diagram of the Malware

2.4.1 Malware Server

With the server the user will be able to target a specific android device to start recording and streaming the data back to the server.

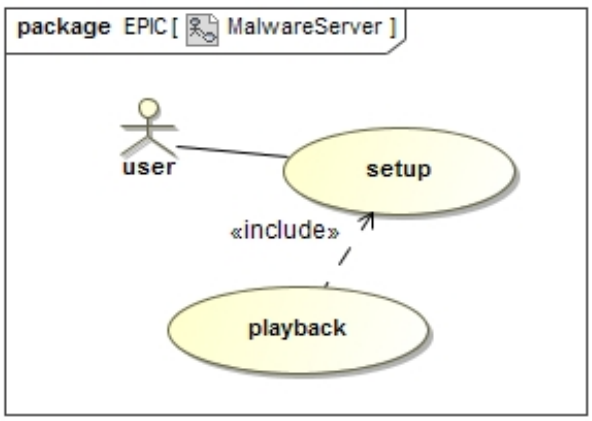

Figure 7: A Use Case Diagram of the Malware Server

- **sendStartRequest**
  Priority - Critical
  This service send a request to start a recording to a specified device. If the device has the malware application installed and is reachable, the application will start streaming the recording to the server. If the device cannot be reached the service will send an error.

- **sendStopRequest**
  Priority - Critical
  This service sends a request to the application on a specified device to stop the streaming and if it is currently storing the recording, stopStoreRecording is called.

- **playback**
  Priority - Critical
  The recorded stream is played back to the user over the speakers.

- **setup**
  Priority - Critical
  Initialize everything

2.4.2 Malware Application

This application is hidden behind another application, but for the purpose of the project, the malware will be a visible application.

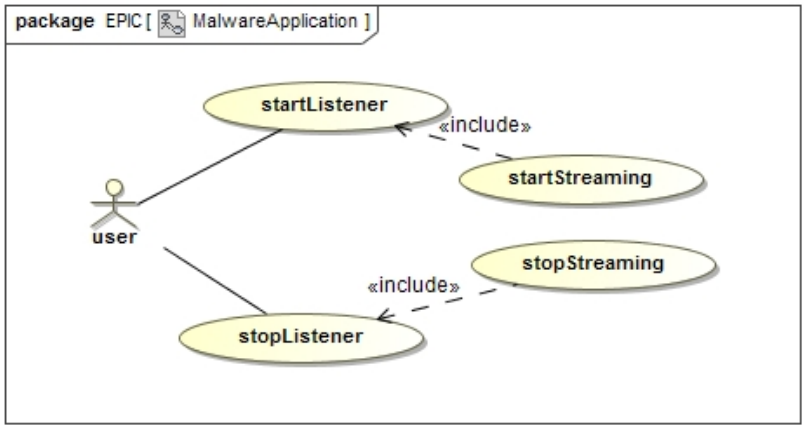

Figure 8: A Use Case Diagram of the Malware Application

- **startRecording**
  Priority - Critical
  This service send a request to start a recording to a specified device. If the device has the malware application installed and is reachable, the application will start streaming the recording to the server. If the device cannot be reached the service will send an error.

- **stopRecording**
  Priority - Critical
  This service stops the recording and stop streaming to the server.

- **startListener**
  Priority - Critical
  The device listen for a request from the server to start the recording.

- **stopListener**
  Priority - Critical
  The device listen for a request from the server to stop the recording.