Introduction

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. Project vision

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

1. 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.

Functional requirements

1. Server
   1. 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 include. 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.
   2. User:
      1. login: The user log into the system using a registered identifier and password, or the user may also log in via facebook
      2. logout: The user log out.
      3. makeAdmin: Change the specified user to an administrative user. This service may only be done by another administrative user.
      4. isadmin
      5. addUser: The specified user is added to the system, but only an administrative user may add a user to the system. An administrative user may add another administrative user.
      6. removeUser: The specified user is removed from the system, but only an administrative user may remove a user from the system.
      7. viewUser: View all the specified users data, but only an administrative user may view another user.
   3. Meeting:
      1. createMeeting: 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.
      2. viewMeeting: 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.
      3. updateMeeting: 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.
      4. deleteMeeting: Delete a meeting. only an administrative user, or the user that created the meeting may delete the meeting
      5. getMeetingSummary : 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. Application
   1. 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.
   2. ////pre android kitkat of meer
   3. Use cases
      1. login: The user login using his unique identifier and password. This service ensures that if the user use another device, his previous data is still intact.
      2. logout: The user log out of the device
      3. SendID : Send identifier to the Node via NFC.
      4. isValid : This service waits for respond from the Node for permission to continue or halt. If the Node’s respond is to continue, storesnapshot is called and the protection is enabled.
      5. enableProtection: The devices WiFi and GSM communication are turned off and the device go into silent mode.
      6. Disableprotection: The prevous state of the phone is restored.
      7. storeSnapshot: A snapshot of the phones current networking state is stored.
      8. loadSnapshot: The stored state is loaded.
      9. getNFCvalue: if the NFC device is scanned it gets a value
3. Gateway and Nodes
   1. 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. Node
      1. receiveID: Receive the userID from the Android devices via NFC
      2. checkID: This service verify if the identifier may access the meeting on the Gateway.
      3. Setup: initialize everything
      4. Loop: run in a loop and call the receiveID service until an identifier is returned, it then call the checkID function.
   3. Gateway
      1. The gateway receives the identifier from the node and verify the meeting access with the server. It also updates the meeting via the server.
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.

* 1. Malware server:
     1. Scope: With the server, the user will be able to target a specific android operating system device to start recording and streaming the data back to the server.
     2. Use cases
        1. sendStartRequest: 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.
        2. sendStopRequest: 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.
        3. storeRecording: The live stream is stored every 30 seconds, until stopStoreRecoriding is called
        4. stopStoreRecording: Stop storing the live stream.
  2. Malware application
     1. Scope: This application is hidden behind another application, but for the purpose of the project, the malware will be a visible application.
     2. Use cases
        1. handleStartRequest: This service receive a request to start recording. It then starts recording and send the stream back.
        2. handleStopRequest: This service receive a request to stop the streaming, and therefor stopStream is called.
        3. sendStream: the recording stream is send back until the stopStream service is called
        4. stopStream: stop recording.
        5. connectToServer: connect to the server that send the Requests