Unsolvable Solutions
Client: Francois Mouton at the CSIR DSSR

Functional Requirements

Eavesdrop

Github link: `https://github.com/Unsolvable-Solutions/Project-EPIC`

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 smart phone 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 smart phone 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 could have eavesdropping malicious software(malware) on their smart phone, it could cause sensitive data to be easily leaked out.

1.2 Project Vision

The aim of the malware is to eavesdrop on a person via their own smart phone. This is done by live streaming the conversation from the infected smartphone to a remote server which plays back the stream and saves a local copy of the recording on the server.

1.3 Project 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. The mobile device will start to stream live to the server which in turn plays back the stream and creates a local recording.

2 Functional Requirements

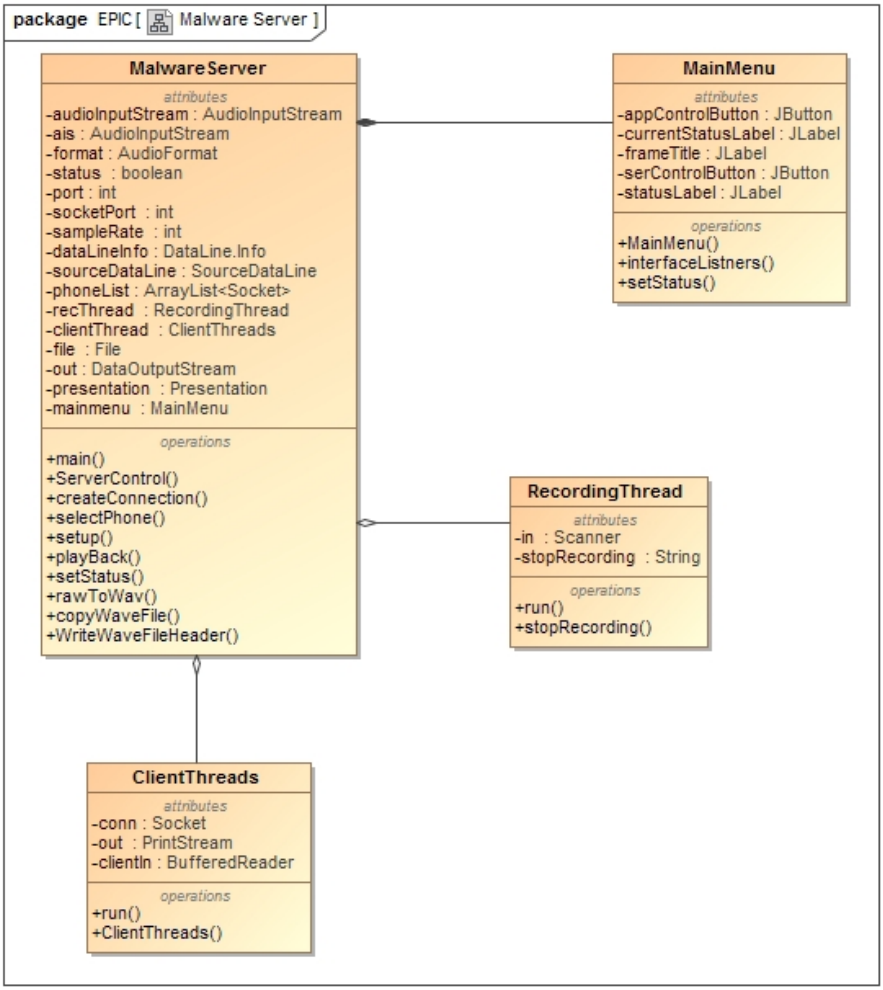

Figure 1: A Class Diagram of the Malware

2.1 Malware Server

2.1.1 Scope

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.

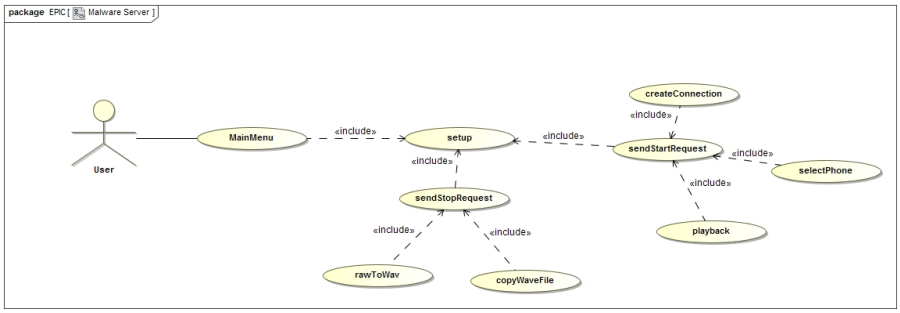

Figure 2: A Use Case Diagram of the Malware Server

2.1.2 Functionality

2.1.2.1 sendStartRequest

Priority: Critical

Service Contract: This service will send a request to a specific mobile device that has the malware application installed to start a live stream.

Pre-conditions:

- The malware application has created a connection with the server.
- The setup modules has been initialised.

Post-conditions:

- Receive the live stream.

2.1.2.2 sendStopRequest

Priority: Critical

Service Contract: This service sends a request to the malware application on a specified device to stop the live streaming.

Pre-conditions:

- The server must be busy recording a live steam.
- The setup modules has been initialised.

Post-conditions:

- A stop request was sent to the malware application on the remote mobile device.
- A local recording of the live stream was stored on the server.

2.1.2.3 playback

Priority: Critical

Service Contract: The recorded stream is played back to the user over the speakers.

Pre-conditions:

- The server must be busy recording a live steam.
- The setup modules has been initialised.

Post-conditions:

- A stop request was sent to the malware application on a specified device.
- The RecordingThread thread was closed.

2.1.2.4 setup

Priority:Critical

Service Contract: Initialise the audio recording modules. Creates a recording thread to handle the live stream and to create a local copy of the stream.

Pre-conditions:

- The malware application has created a connection with the server.

Post-conditions:

- A RecordingThread thread was created.
- Playback has been initialised.

2.1.2.5 selectPhone

Priority: Critical

Service Contract: Creates a list for selection of a device to stream and record from.

Pre-conditions:

- The malware application has created a connection with the server.
- A ClientThreads thread exists which handles the connections

Post-conditions:

- The setup modules has been initialised.
- A RecordingThread thread was created.

2.1.2.6 createConnection

Priority: Critical

Service Contract: Provides the means to create a list of devices to connect to and to create the connection to a mobile device running the malware application.

Pre-conditions:

- Ports 4545 and 8080 must be allowed in the firewall
- The server is not currently receiving a live stream.

Post-conditions:

- A ClientThreads thread was created.
- The connection to a mobile device running the malware application was successful.

2.1.2.7 rawToWav

Priority: Important

Service Contract: Converts the RAW audio file to a WAVE file.

Pre-conditions:

- A stop request was sent to the malware application on the remote mobile device.
- The RecordingThread thread was closed.

Post-conditions:

- The RAW audio file was removed

2.1.2.8 copyWaveFile

Priority: Important

Service Contract:Takes the WAVE file and adds the required header to the file to enable correct playback.

Pre-conditions:

- The RAW audio file was converted to a WAVE file
- The RAW audio file was removed

Post-conditions:

- The required header values was added to the wave file.

2.1.2.9 MainMenu

Priority: Nice to have

Service Contract: Creates a user friendly interface for using the server.

Pre-conditions:

- No previous interface exists.

Post-conditions:

- Interface listeners are active
- Main menu is visible.

2.2 Malware Application

2.2.1 Scope

The malware application on the mobile device will allow a lives stream to be sent to the malware server. The malware would be embedded inside an application that will most likely be used by the target.

2.2.2 Functionality

2.2.2.1 startStreaming

Priority: Critical

Service Contract: This service starts a live stream from the devices microphone and sends it to the server that requested the stream.

Pre-conditions:

- Allowed access to the devices microphone.
- A connection to the server has been established.

Post-conditions:

- A recording thread was created to handle the live stream.

2.2.2.2 stopRecording

Priority: Critical

Service Contract: This service stops the live streaming to the server.

Pre-conditions:

- A live stream is being sent to the server.

Post-conditions:

- Live stream to the server was closed.

2.2.2.3 connections

Priority: Critical

Service Contract: The connection function in the client thread handles the communication between the malware server and the malware application. It listens for the server on a specific ip address and port. When a server is found it sends a request to connect to the server.

Pre-conditions:

- The server must be available.
- Allowed access to the mobile devices network.

Post-conditions:

- A connection to the server was established.