Analyzing the SS8 Interceptor Application for the BlackBerry Handheld

Sheran A. Gunasekera

Zensay Labs
http://chirashi.zensay.com
E-mail: sheran@zensay.com

ABSTRACT

This paper provides information about the BlackBerry[1] spyware known as "Interceptor." The software was developed by Legal Interception company SS8[2] and was rolled out to subscribers of the UAE Telecommunications operator, Etisalat[3] as an update. Subscribers who had installed the software noticed increased power consumption on their handhelds and battery drainage that rendered handhelds usable for shorter periods of time than normal.

In this paper, we will analyze the source code of the Interceptor program, review the key points that the software tries to achieve and look at where the developers made mistakes. It is worthwhile noting that other parties have conducted initial research into the spyware as well.[4]

INTRODUCTION

On or around the 8th of July 2009, BlackBerry subscribers of the UAE based telecommunications operator received a WAP push message prompting them to download an upgrade. The text of the message indicated that the upgrade was essential to the continued service of their service. Upon installing this software, users began to notice increased battery drainage on their phones. The effective usable time on a single battery charged drastically dropped.

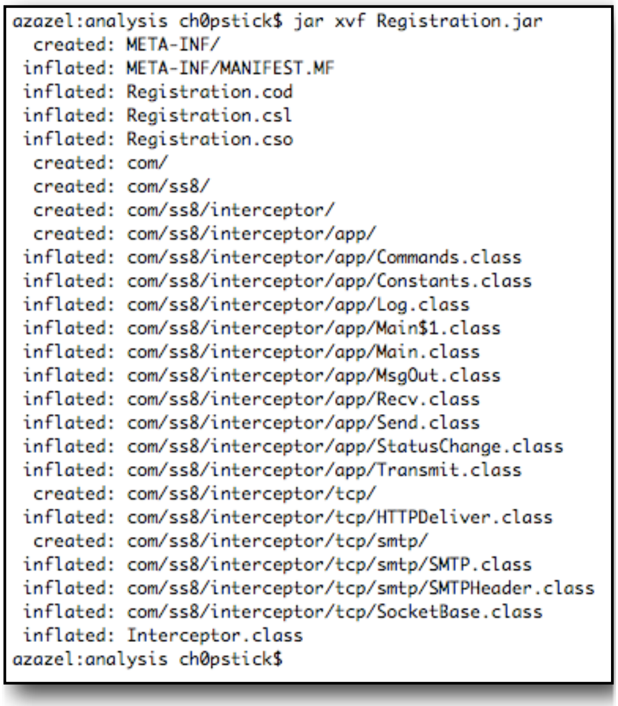
The program that subscribers downloaded was called "Registration." It was available in several different file formats. The standard BlackBerry COD file format and the more open, Java JAR format. The JAR format is simply a collection of Java class files and resources compressed in a ZIP format. Simply running a standard ZIP decompressor will reveal all files stored within the JAR archive. Even though a COD file can be decompiled, a JAR file is far easier to work with and can greatly improve analysis times. Thus all the analysis was conducted on the JAR file.

ARCHIVE CONTENTS

The Registration.jar file was first decompressed. On closer examination of the file structure and file naming conventions, it was immediately apparent that this application was more than a performance upgrade. The directory structure is depicted in Figure 1.

By the file naming conventions, the application is more of a message interceptor. The directory structure "com/ss8" also provides some insight. SS8 are providers of Lawful Interception products & services to telecommunications providers.

The "class" files contain compiled Java bytecode. They can be de-compiled to reveal the original source code. To decompile the files, the tool JAD The Java Decompiler[5] was used.



```
azazel:analysis ch0pstick$ jar xvf Registration.jar
  created: META-INF/
 inflated: META-INF/MANIFEST.MF
 inflated: Registration.cod
 inflated: Registration.csl
 inflated: Registration.cso
  created: com/
  created: com/ss8/
  created: com/ss8/interceptor/
  created: com/ss8/interceptor/app/
 inflated: com/ss8/interceptor/app/Commands.class
 inflated: com/ss8/interceptor/app/Constants.class
 inflated: com/ss8/interceptor/app/Log.class
 inflated: com/ss8/interceptor/app/Main$1.class
 inflated: com/ss8/interceptor/app/Main.class
 inflated: com/ss8/interceptor/app/MsgOut.class
 inflated: com/ss8/interceptor/app/Recv.class
 inflated: com/ss8/interceptor/app/Send.class
 inflated: com/ss8/interceptor/app/StatusChange.class
 inflated: com/ss8/interceptor/app/Transmit.class
  created: com/ss8/interceptor/tcp/
 inflated: com/ss8/interceptor/tcp/HTTPDeliver.class
  created: com/ss8/interceptor/tcp/smtp/
 inflated: com/ss8/interceptor/tcp/smtp/SMTP.class
 inflated: com/ss8/interceptor/tcp/smtp/SMTPHeader.class
 inflated: com/ss8/interceptor/tcp/SocketBase.class
 inflated: Interceptor.class
azazel:analysis ch0pstick$
```

Fig 1. Unpacking the JAR file

SOURCE CODE ANALYSIS

Generally, there are many ways of obfuscating or rendering Java source code unreadable. In this case, the application developers had not performed any type of source code obfuscation. This makes it far easier to read and analyze the code. Starting at the program entry-point, a visual depiction of the program flow can be summarized in Figure 2.

The core functionality of the program can be listed as follows:

(1) Checks to see if it is listed as visible in the BlackBerry installed applications
(2) If it is visible, it hides itself from view of the subscriber. This prevents a user from finding it and deleting it.
(3) It sets iterates over all the ServiceBooks[6] on the handheld and attaches itself to each of them, looking for received email messages and PIN messages.
(4) It intercepts and monitors the state of the handheld for network events that occur. It notifies the service-provider's server when these events occur.
(5) It listens for messages received from specific addresses either through email or BlackBerry PIN. These are control messages that can enable or disable the interception of the subscribers' messages.
(6) It reports back to the predefined service-provider server regularly.
(7) If enabled, the application will forward a copy of emails sent out by the subscriber to the service-provider server.

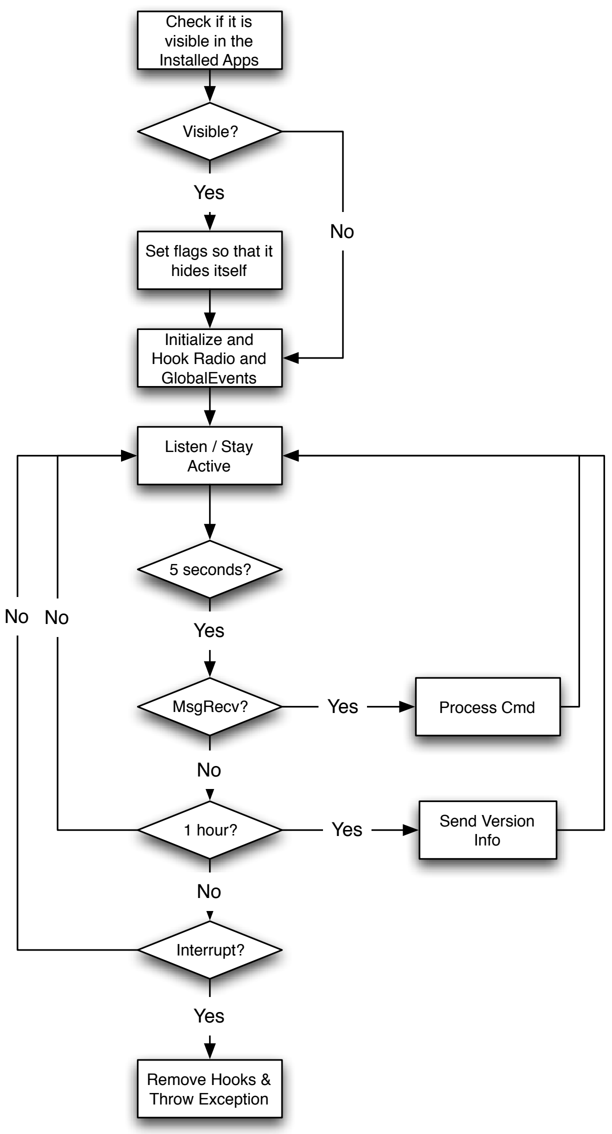

Check if it is visible in the Installed Apps
Visible?
Yes
No
Set flags so that it hides itself
Initialize and Hook Radio and GlobalEvents
Listen / Stay Active
5 seconds?
No
No
Yes
MsgRecv?
Yes
Process Cmd
No
1 hour?
Yes
Send Version Info
No
Interrupt?
Yes
Remove Hooks & Throw Exception

Fig 2. The overall program flow of the Interceptor Application

On analysis of the source code, it was evident that this program was no mature enough to be deployed. This is especially relevant if Etisalat planned to conduct full-scale legal interception on BlackBerry users.

Visibility

One key aspect of any successful legal interception framework is stealth. Generally, a service provider or government will want to conduct its activities covertly. For the sake of argument, lets take the example where a government would like to track down activities of a terrorist cell. To be effective, the activities of the terrorists should be monitored without their knowledge. If they know they are being monitored, they may alter their activities or deliberately behave in a manner to deceive the government. To avoid this, stealth is a key factor in the monitoring process.

The very nature of the BlackBerry infrastructure renders it an unsuitable medium for such legal interception. This is especially true when you consider electronic communications such as email or instant messaging. BlackBerry users can enjoy secure messaging by making use of the proprietary BlackBerry messenger[7] software that comes as part of their handheld. To circumvent these security features, service-providers would need to resort to installation of purpose-built applications onto the handheld device. Thus, to ensure that the application goes undetected and so as not to arouse suspicion, the SS8 Interceptor application checks its visibility state on each reboot of the handheld. If the application is visible, then a flag is set to hide it from the applications list.

The Interceptor application uses one of the BlackBerry's built in libraries known as CodeModuleGroup to set the FLAG_HIDDEN bit to true.

Interception

An ideal legal interception framework comprises of three domains, the Network Entities domain, the Interception domain and the Law Enforcement Collection and Administration domain. To allow for comprehensive interception, it is essential that all communications traffic for a specific Network Entity be collected. For example, if the Network Entity of email traffic was identified for interception, then it is essential that the Interception domain should be capable of intercepting both transmitted and received data.

The version of the Interceptor application that was analyzed was only capable of intercepting data that the handheld transmitted. This was obviously a shortcoming by the developer, unless it was specifically requested by either Etisalat or the law enforcement authorities of the UAE.

Control Channel

Another cornerstone of a legal interception framework is for the service-provider to enable or disable interception on a granular basis. Often, this is referred to as a control channel where the service-provider sends specific commands to the software. The software then takes specific action against each of the control commands.

The Interceptor application has a control channel built-in. The following commands are available in the version that was analyzed:

(1) **version**: sends the version details of the handheld and Interceptor application back to the central server.
(2) **bCkp**: sends similar information back to the central server, except in this case the information is tagged as a backup.
(3) **start**: initiates the program and starts intercepting outgoing messages from the handheld.
(4) **stop**: stops the interception of the outgoing messages on the handheld, but the control channel still remains active.

Command Structure

The commands can be sent through two methods, (1) through email and (2) through the proprietary BlackBerry PIN messaging system[8]. The command format consists of the following structure:

```
<Command>[Command name]</Command>
<s>[Serial No.]</s>
```

To differentiate the control commands from other legitimate communications, the following triggers are used:

For commands sent through email:
(1) The subject line will contain the word "cmd_mail" and commands are contained in the message body as plain-text
(2) Besides (1) above, if the subject line also contains the word "XXX" then the application treats the commands in the message body as encrypted. The encryption key is the unique handheld identifier or PIN[9]

For commands sent through PIN messaging:
(1) The originating PIN number should match the string "Customer Service"
(2) The originating PIN name should match the string "Customer Service"

For PIN message based control commands, there is no provision for plain-text. It is encrypted by default.

PRIVACY CONCERNS

The version of the Interceptor application that was analyzed only intercepts outgoing email messages of accounts configured on the handheld. No evidence was found of interception of instant messages, BlackBerry instant messages, phone calls, SMS messages, bluetooth data, wireless data or GPS data.

When the Interceptor application is installed, it is in a dormant state by default, meaning it will not forward your sent messages to the central server.

OBSERVATIONS

There are several anomalies in the application that lead to the conclusion that it was either not the version intended for deployment; it was mistakenly rolled out or it was an early release that was being tested.

The reasons for such a conclusion can be argued as:

(1) No capability for intercepting incoming messages.
(2) No possibility of silently updating the application with newer releases.
(3) Lack of comprehensive interception capabilities. Only outgoing email messages.
(4) Several segments of unused source code and references that have been hardcoded into the application.

Further observations have been listed below.

Disabled Email Control Channel

The email based control channel to send commands to the application is disabled. On further analysis, why it was disabled became clear. When the service-provider sends an email message to activate the application, a copy of this control email would also be delivered to the recipient's email server. Thus the user would be alerted to possible suspicious activity.

Control Channel Messages

Control channel commands are momentarily visible when they are received. Thus a user who happens to be looking at his handheld screen would see a message appear for a fraction of a second and then instantly disappear. This behavior was observed on a BlackBerry handheld but was not apparent on the BlackBerry handheld simulator.

Hardcoded References

A standard program that is redistributed will usually have some sort of constants or configuration file. The Interceptor application did contain such a file, however the configuration parameters from the file were not used in the execution of the program. Instead, there were hardcoded references that were used. This is what lead to the conclusion that this version of the application was either a early testing version that was mistakenly deployed or it was a badly modified version of an original file.

Battery Drain

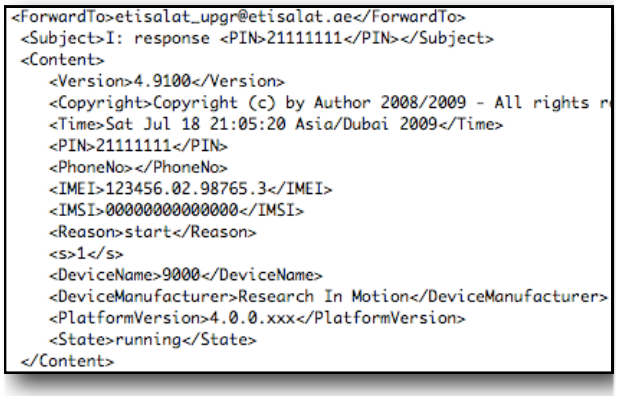
The application implements a watcher on all the handheld message folders. This watcher triggers other components whenever a message is received. Despite this, the application polls a function to check if a new message has been received. This constant polling uses processing cycles and thus increases power consumption. It is very likely that less powerful processors may overheat due to the increased processing activity. This is bad programming practice, especially for handheld devices. It was also the reason users were made suspicious of the program.

Heartbeat

Every hour, each handheld will report its status and version information to the central server. This happens regardless of whether the application is intercepting messages or not. A sample message sent to the central server is depicted in Figure 3. The image has been modified to retain formatting and hide device specific information.

Encryption

The Interceptor application makes use of encryption when sending intercepted messages or receiving control commands. It does this by encrypting outgoing messages using AES. The keys are hardcoded into the application. For incoming control commands, the messages are decrypted using the device PIN as the decryption key. The encryption type is still AES.

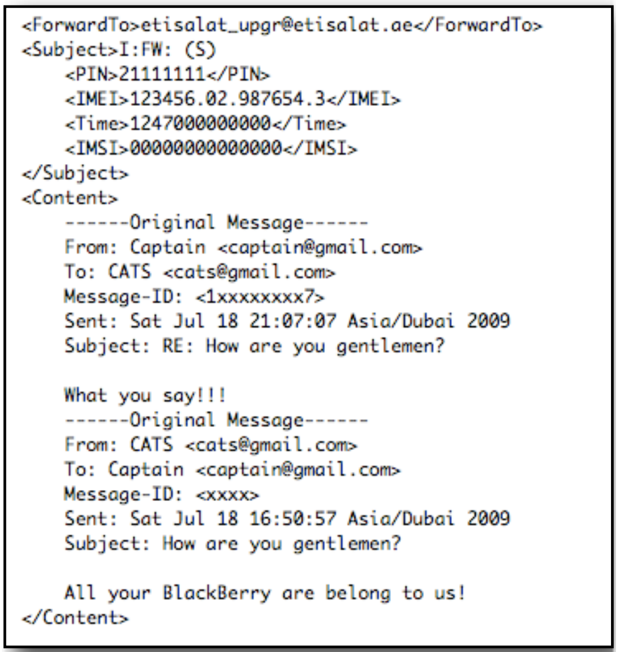


```
<ForwardTo>etisalat_upgr@etisalat.ae</ForwardTo>
 <Subject>I: response <PIN>21111111</PIN></Subject>
 <Content>
    <Version>4.9100</Version>
    <Copyright>Copyright (c) by Author 2008/2009 - All rights r
    <Time>Sat Jul 18 21:05:20 Asia/Dubai 2009</Time>
    <PIN>21111111</PIN>
    <PhoneNo></PhoneNo>
    <IMEI>123456.02.98765.3</IMEI>
    <IMSI>00000000000000</IMSI>
    <Reason>start</Reason>
    <s>1</s>
    <DeviceName>9000</DeviceName>
    <DeviceManufacturer>Research In Motion</DeviceManufacturer>
    <PlatformVersion>4.0.0.xxx</PlatformVersion>
    <State>running</State>
 </Content>
```

Fig 3. Version information sent to the central server

INTERCEPTED MESSAGE

An intercepted message is shown in Figure 4 below. Device specific information was removed and the image was modified to match formatting.



```
<ForwardTo>etisalat_upgr@etisalat.ae</ForwardTo>
<Subject>I:FW: (S)
    <PIN>21111111</PIN>
    <IMEI>123456.02.987654.3</IMEI>
    <Time>1247000000000</Time>
    <IMSI>00000000000000</IMSI>
</Subject>
<Content>
    ------Original Message------
    From: Captain <captain@gmail.com>
    To: CATS <cats@gmail.com>
    Message-ID: <1xxxxxxxx7>
    Sent: Sat Jul 18 21:07:07 Asia/Dubai 2009
    Subject: RE: How are you gentlemen?

    What you say!!!
    ------Original Message------
    From: CATS <cats@gmail.com>
    To: Captain <captain@gmail.com>
    Message-ID: <xxxx>
    Sent: Sat Jul 18 16:50:57 Asia/Dubai 2009
    Subject: How are you gentlemen?

    All your BlackBerry are belong to us!
</Content>
```

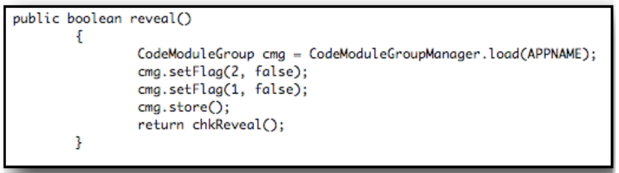
Fig. 4 Intercepted email message

REMOVING THE SPYWARE

Although easy, it is not a straightforward task to remove the spyware. The application hides itself and so is not visible when in the applications screen. There are several ways that this spyware can be removed or contained.

(1) Use a program to reveal the spyware. The Interceptor application is installed on the handheld and is named "Registration:Etisalat". To reveal the program the FLAG_HIDDEN bit must be set to false for the application. Java code to accomplish this is shown in Figure 5. Zensay

Labs have released a program to do this and also provide details of other hidden applications on the handheld. This program can be downloaded directly to the handheld from http://www.zensay.com/HiddenProgs.jad



```
public boolean reveal()
        {
                CodeModuleGroup cmg = CodeModuleGroupManager.load(APPNAME);
                cmg.setFlag(2, false);
                cmg.setFlag(1, false);
                cmg.store();
                return chkReveal();
        }
```

Fig 5. Java Source Code to reveal a hidden program

(2) Use the JavaLoader utility to remove the software externally. For this you will need to obtain the JavaLoader software from RIM. The JavaLoader file can be found on the RIM Java Development Environment[10] packages. Connect the handheld to a PC containing the JavaLoader software and execute the command: "javaloader -u -f Registration.cod"
    a) the *-u* option will tell Javaloader to connect to the device on the USB port.
    b) the *-f* option tells Javaloader to forcibly remove a module even if it is presently in use.
    c) *Registration.cod* is the name under which the spyware module is installed.

(3) A full wipe and reinstallation of the handheld can also remove the spyware.

LESSONS TO BE LEARNED

Rolling out legal interception software is something that has to be done with good planning, preparation and management. A deployment of this nature should always work under the common assumption: "There will be no second chance." Thus, the software that will be pushed to subscribers has to be near perfect. At best the software should:

(1) Do no harm. The legal interception software should not be easily compromised by an attacker who knows about it. Essentially this is a backdoor providing near full access to people's handhelds. Therefore the software should be secure.

(2) Be thoroughly tested. Doing things like draining user's batteries is a sure way of alerting them to suspicious activity. The software should be tested on a limited set of handhelds for a few weeks to resolve any issues that may surface with extended testing. The entire system should be tested and retested several times. The software developers should be highly experienced in mobile development. Making trivial mistakes like polling so often that the battery drains is certainly a sign of bad programming and lack of testing.

A service-provider should always be prepared for the worst. In case things do not work out as planned, there needs to be a dedicated PR team who is ready to step up and deal with the public. User's should not be lied to or ignored, they will accept it better if they know the provider is well within legal rights to perform such interception. The final decision will be left to them.

FUTURE PROTECTION

Now that the software has been analyzed and its techniques have been identified, there are several practices that a user can adopt to be vigilant. Possibly the two most common practices would be to set the phone's default permissions and to constantly monitor all programs on the handheld.

Default Permissions

BlackBerry handhelds are often held as the icon of secure, enterprise communications devices. They are equipped with a range of security features. One such feature is the Default Permissions option. These permissions allow a user to turn off access to (1) Connections, (2) Interactions and (3) User Data by default when a new program is installed. These features can be accessed on the handheld by navigating to:

Options > Advanced Options > Applications > Menu > Edit Default Permissions

Truly paranoid users may want to set all to disable. This does mean that they will have to fine-tune the permissions whenever a new application is installed, but this does provide some peace of mind because an alert is displayed each time an application requests access to the Internet or personal data.

Monitoring Hidden Programs

While the arena for BlackBerry forensics and spyware is still somewhat limited, Zensay Labs is focusing on this area to provide some further information on the subject. With the release of the HiddenProgs tool, users can examine what other programs have hidden themselves and identify potentially suspicious programs. They can then continue to uninstall the suspicious software manually with the JavaLoader tool or they can contact Zensay Labs directly for assistance. With continued research, the HiddenProgs tool will evolve into a more comprehensive spyware detector and disabler.

REFERENCES

[1] BlackBerry : RIM BlackBerry http://www.blackberry.com

[2] SS8: Legal Interception Solutions, http://www.ss8.com

[3] Etisalat: Emirates Telecommunications Corporation, http://www.etisalat.ae

[4] Chris Eng, Veracode, BlackBerry Spyware Dissected, http://www.veracode.com/blog/2009/07/blackberry-spyware-dissected/

[5] JAD - The Java Decompiler, by Thomas Varaneckas, http://www.varaneckas.com/jad

[6] Service Books: http://na.blackberry.com/eng/deliverables/2584/about_service_books_32170_11.jsp and http://www.berryreview.com/2009/01/22/faq-explanation-of-each-blackberry-service-book-type/

[7] BlackBerry Messenger: http://na.blackberry.com/eng/devices/features/im/blackberry_messenger.jsp

[8] BlackBerry PIN Messaging System: http://forums.pinstack.com/showthread.php?s=60879514238665e850a4497790d9ab00&t=35

[9] BlackBerry PIN: See [7]

[10] BlackBery JDE: http://na.blackberry.com/eng/developers/javaappdev/javadevenv.jsp