SMobile Global Threat Center

Study of BlackBerry Proof-of-Concept Malicious Applications

Mayank Aggarwal, GTC Research Engineer, maggarwal@smobilesystems.com
Troy Vennon, GTC Research Engineer, tvennon@smobilesystems.com

Jan. 06, 2010

TABLE OF CONTENTS

ABSTRACT 3
INTRODUCTION 4
HISTORY OF BLACKBERRY VULNERABILITIES AND MALWARE THREATS 8
CODE SIGNING 10
BLACKBERRY PROOF-OF-CONCEPT 11
CONCLUSION 21
REFERENCES 23

ABSTRACT

Recent reports indicate that Research In Motion's (RIM) BlackBerry handheld device currently holds the second largest market share amongst the world's Smartphone devices at 21% [1]. When compared against the Smartphone market, BlackBerry generally garners the reputation as being the most secure platform and is often chosen as the preferred mobile device amongst enterprises and government entities. Two of the major selling points being BlackBerry's integrated wireless messaging system over the cellular network and the backend BlackBerry Enterprise Server suite that allows IT personnel to centrally manage BlackBerry policies across an enterprise.

In its current form, RIM markets their BlackBerry devices to two distinctly separate user bases. Historically, BlackBerry devices were adopted by enterprises whose users required mobile access to enterprise data and resources. Such remote access is often managed by the BlackBerry Enterprise Server (BES), which allows IT and Security Managers to enforce security and access policies which, among other things, can serve to limit the types of applications and permissions that enterprise users can install. Recently, RIM began focusing marketing efforts towards every day consumers as well. BlackBerry achieves similar functionality for consumers by offering the BlackBerry Internet Server (BIS), which offers no security and access policies to protect the consumer. Instead of providing a means for specialized individuals to apply security best practices and controls to consumers, BlackBerry relies upon the security framework that is built-in to the platform to protect consumers from malicious code.

The research and proof-of-concept (POC) malicious applications that will be discussed in the remainder of this project are aimed at taking advantage of the weakened security posture of BlackBerry devices that operate under the BIS environment. The nature of the BlackBerry security framework requires user intervention and the acceptance of certain risks when installing 3rd party applications. The security framework is meant to enforce certain aspects of control over application development by forcing developers to register for a signing certificate. Application of the developer's signing certificate is fundamental in controlling malicious applications being introduced into the BlackBerry application market space. We will take a look at whether the security framework, coupled with the application signing certificate goes far enough to ensure that ordinary, everyday users are protected against malicious applications developed for BlackBerry devices. Specifically, this research attempts to determine whether it is possible to develop malicious applications, with varying degrees of anonymity, which take advantage of unsuspecting users accepting unnecessary permissions.

This research focuses primarily on application attack vectors and does not focus on the role of BlackBerry Enterprise Server and BlackBerry Internet Service solutions.

INTRODUCTION

We'll begin the discussion with a fictional narrative of an ordinary user who has come across an application that he determines to be useful for some purpose on his BlackBerry handset. Dave is an ordinary user with no specific technical, security training, but does know how to leverage technology to increase productivity and communication. Dave has purchased a personal BlackBerry Storm2 device from his carrier and is configured to access corporate email over IMAP and the BIS infrastructure.

In this instance, Dave works as an internal auditor for a reputable financial firm. Dave has been searching for a single application that would allow him to manage multiple IM accounts from a single application. In his searching, Dave came across an application named "Instant Messenger" that integrates Google Talk, Yahoo Instant Messenger, AOL IM, and Twitter into a single application and is said to facilitate easy migration from one client to another.

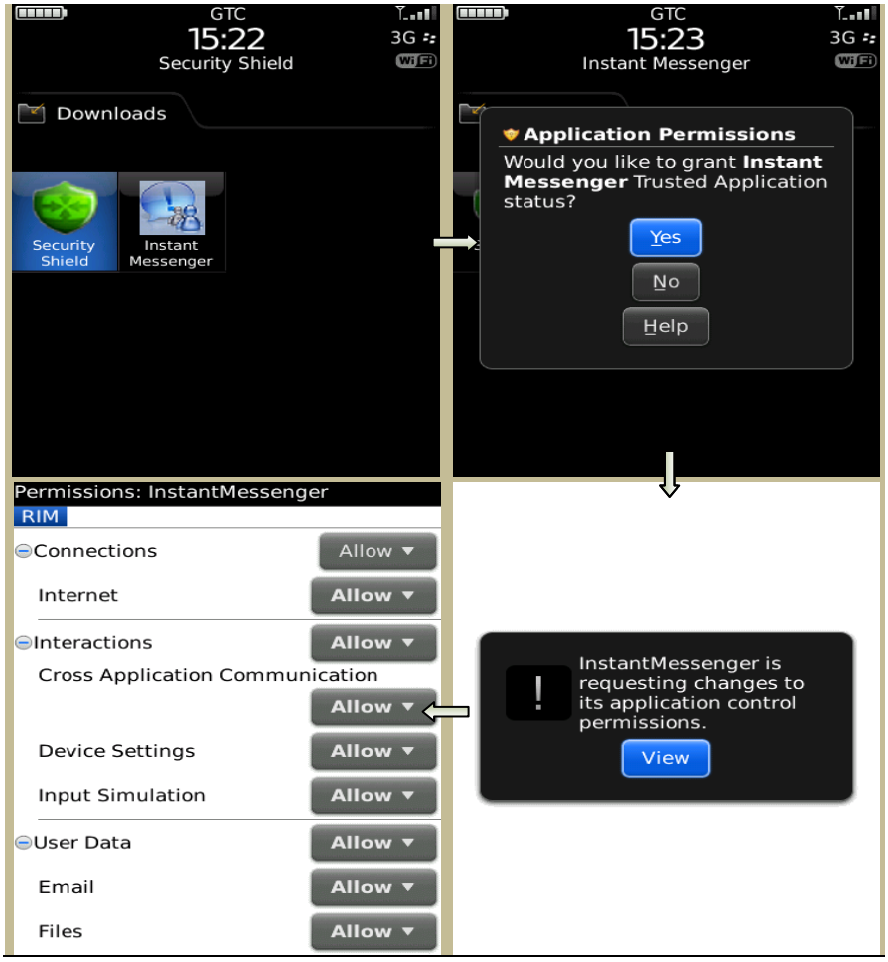

Figure 1: The screenshot of Dave's BlackBerry.

After successfully downloading the "Instant Messenger" application, Dave proceeds to the "Downloads" folder where he runs the application for the first time. Upon running the application, Dave is prompted to grant the new application "Trusted Application" status. Dave clicks "Yes" and is then asked to view the application control permissions that the application is attempting to

change. Dave is not a technical individual and chooses to allow the default permissions to be set on the newly installed application as they are displayed in Fig. 1.

Up to this point, the BlackBerry's internal security framework has done its job to alert Dave to the types of permissions he was granting the application. As was noted, knowing no better, Dave chose to allow the default permissions for the application to be set on the device. How many of us have done that when installing an application? At this point, the security framework takes no further responsibility for ensuring that the application performs as Dave believes it should and anything that the application does has been approved when it was originally installed

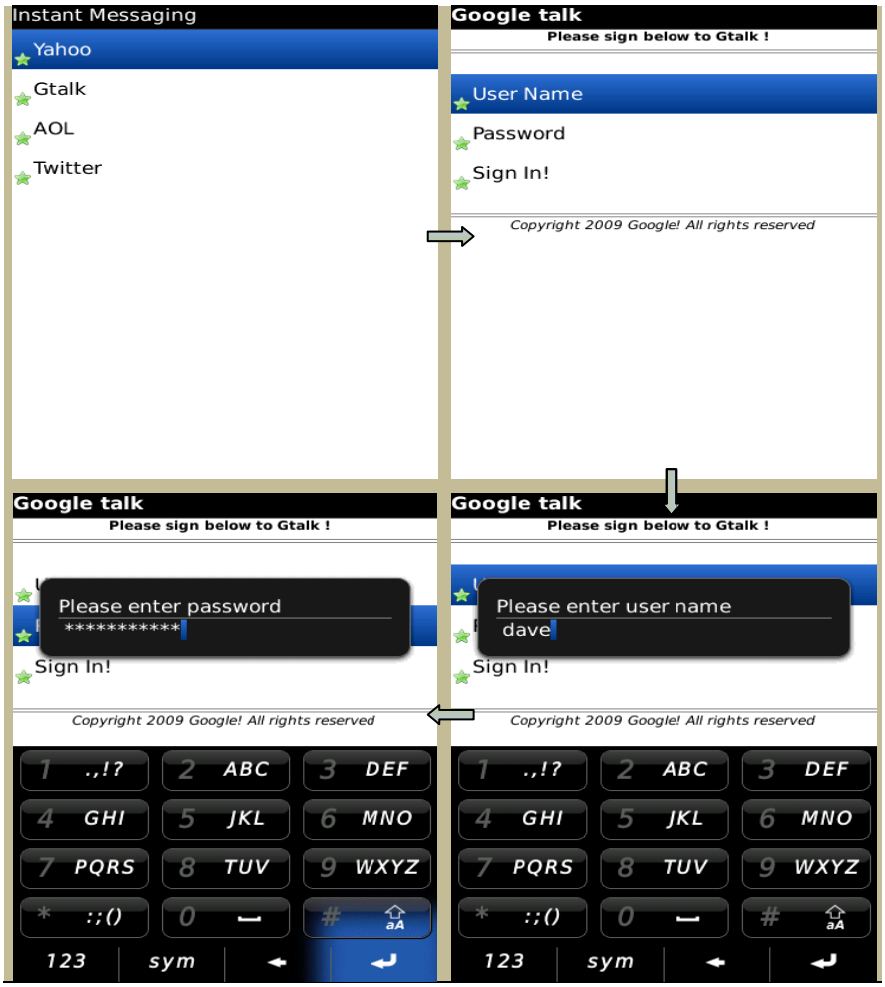

Figure 2: The screenshot of Dave's BlackBerry.

Using his newly installed "Instant Messenger" application, Dave decides to check out his friends that are online on Gtalk and is prompted to supply the username and password to his Google Talk account, as shown in Fig. 2.

Once Dave clicks "Sign In", he is informed that Gtalk is experiencing connectivity problems, as shown in Fig.3, and is asked to send a test email message to test the device's connectivity. By clicking "OK", the device appears to have tried to send the test email, but also experienced connectivity problems and was unable to send the message.

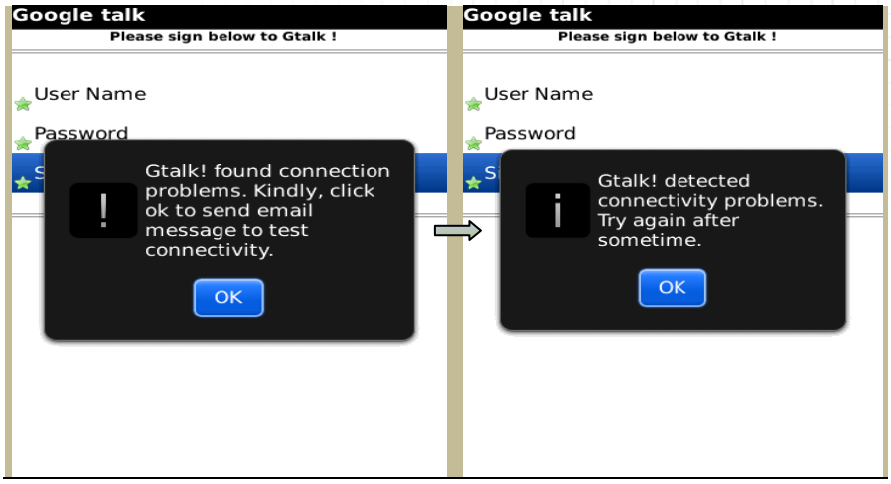

Figure 3: The screenshot of Dave's BlackBerry.

Most ordinary users would begin to think that either the application they downloaded simply didn't work or that there was a connectivity problem that really was not allowing the application to connect. Either way, one could pretty much guarantee that the user would then attempt to log in to the remaining three clients that the application says it supports.

If Dave were to wonder about the failed test email messages and decided to check his outbound mail messages, he would find messages that resemble Fig. 4. This screen capture represents the actual email message that was sent by the "Instant Messenger" application. The first screen capture shows the content of the email. Dave also notices an attachment along with the sent email. Being curious, he clicks on it and, as shown in the following screen capture, the BlackBerry fails to find the attachment and gives an error that says the file was not found.

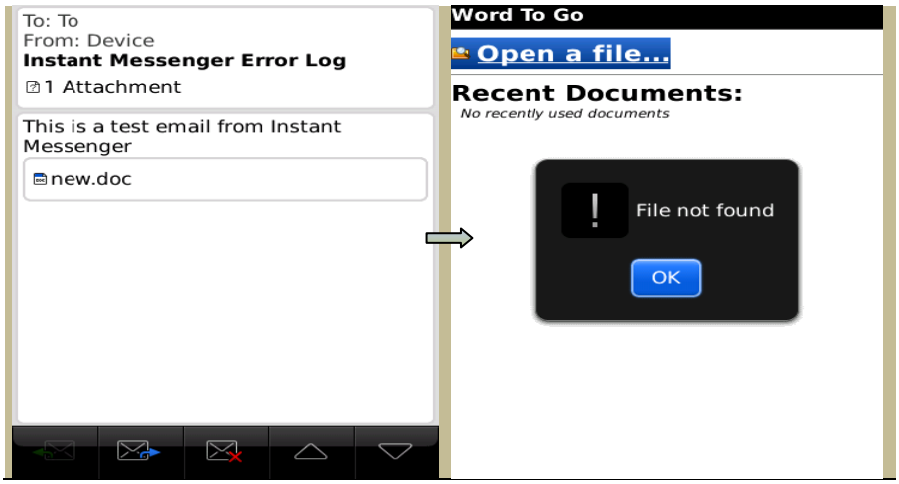

Figure 4: The screenshot of Dave's BlackBerry.

Dave has one of two choices. He can uninstall the application and try to find another one that meets his needs, or he can try to determine what the connectivity problem the application is encountering might be. For the sake of brevity, let's say that Dave just uninstalls the application and goes on his way.

Before proceeding any further, let's briefly review the behavior of this application on Dave's BlackBerry. Initially, the application asks the user to allow certain permissions. Then, when the user enters his Gtalk account credentials, it gives a connection error and asks the user to allow it to send a test email. However, after this step it gives a connection error problem again and fails to connect to Gtalk. The user decides to investigate the email sent by this application and finds an attachment accompanying the email message. When Dave tries to open the attachment, it appears that the attachment does not exist on the BlackBerry.

The "Instant Messenger" application that was used in this scenario is a proof-of-concept Trojan application that has performed differently than advertised and intercepted Dave's Gtalk account credentials and emailed them to an attacker's address that was hardcoded into the application. Figure 5 represents the screen capture of the message that was received in the attacker's email. The attacker receives an email with a subject "Instant Messenger Error log" which consists of an attachment that includes the username and password for Dave's Gtalk account.

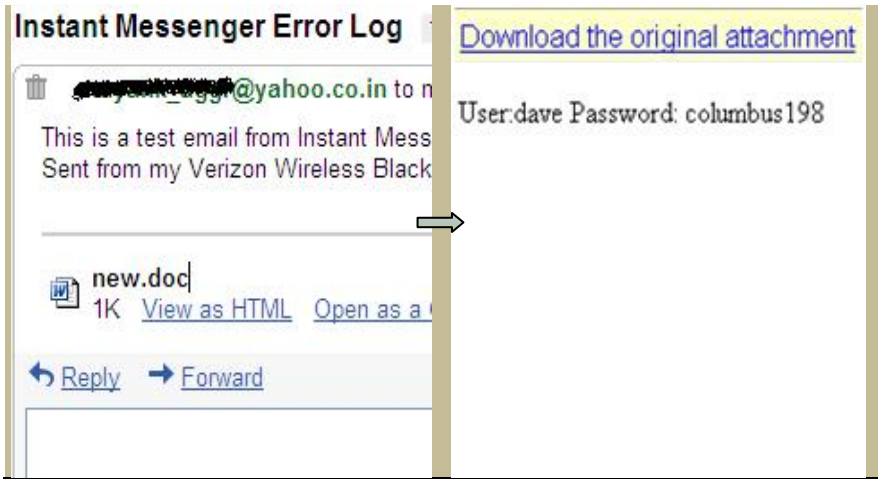

Figure 5: The screenshot of attacker's email.

Dave was not able to open the attachment because the attacker developed the application in such a way that it wrote the credentials into a .doc file during run time and attached it along with the email. This attachment was never saved or stored anywhere on the device so when Dave tried opening it, the BlackBerry issued "File not found error". When Dave chose to allow the default permissions to be set on the 3rd party application, he implicitly allowed the application to perform functions he was not aware that would be occurring.

HISTORY OF BLACKBERRY VULNERABILITIES AND MALWARE THREATS

BlackBerry devices are widely considered to be the most secure devices in the Smartphone market place. It has been shown that there are certain loopholes that can be exploited by an attacker to develop malicious applications. In this section, some of the more well-known spyware applications and vulnerabilities that have been discovered are discussed.

Trojan & Spyware

FlexiSpy: This application has gained a lot of attention in the last few months. Many news organizations have begun picking up on the mobile spyware trend and have begun reporting on them to inform their viewers. Once installed, FlexiSpy can be configured to remotely sniff user's call logs, text messages, emails and can be used as a microphone for an attacker to listen in on ambient room conversation. FlexiSpy is labeled spyware because it attempts to hide itself and runs in a stealth manner, so as not to alert the intended victim [2]. On Oct-30, 2009 the FlexiSpy PRO-X version for BlackBerry was released, which offers LIVE CALL INTERCETION, SPY CALL and STEALTH GPS TRACKING. It also enables the attacker to remotely start or stop the application.

MobileSpy: A new spyware application that silently monitors and logs GPS location, incoming and outgoing text messages and call information. After a set regular interval of time, it uploads all the logged activities to the attacker's online account. It hides itself on the device and runs in complete stealth mode that makes it difficult for a victim to identify and delete [3].

Etisalat: A controversial spyware application that was WAP pushed to BlackBerry subscribers of the Etisalat network in the United Arab Emirates (UAE) as an approved performance patch earlier this year. It was described as a fix to network problems. The unintended side effect of the spyware application was draining the device's battery, which forced independent investigations revealing the true nature of the "update". Etisalat had teamed with a U.S. based application developer to create a spyware tool that could be triggered by the service provider to begin logging and transmitting unsuspecting user's email correspondence back to the service provider.

RedBrowser: A J2ME based Trojan, which once installed and started, sends SMS messages to a premium number. This Trojan causes a financial loss to the victim as the victim is charged for each message transmitted to the premium number and the attacker receives the proceeds. There are a large number of variants in existence for this type of Trojan.

BBProxy: The hacking program developed by Jesse D'Aguanno that exploits the network connection between the BlackBerry device and a company's internal server by hijacking a connection to a network. Once installed on the device, it causes the BlackBerry to call back to the attacker's system in the background, opening a communication channel between the attacker and the company's internal network [4].

Vulnerabilities and other threats

Oct. 2nd, 2009: According to eweek.com, Research In Motion has plugged a security hole that left BlackBerry users open to phishing attacks. The malicious user can perform a phishing attack by sending a SMS or email consisting of a link to the fake web site [5].

Feb. 11th, 2009: Earlier this year, BlackBerry maker Research In Motion (RIM) addressed a vulnerability in its BlackBerry Application Web Loader, an ActiveX control that is typically started on a web page and downloads software through a USB cable connected to the phone. According to the advisory, "When a BlackBerry device user browses to a website that is designed to install the BlackBerry Application Web Loader ActiveX control on BlackBerry devices over a USB connection, and clicks 'Yes' to install and run the ActiveX control, the ActiveX control introduces the vulnerability [a buffer overflow] to the computer" [6].

July 16th, 2008: A "high" severity flaw was found in the BlackBerry Attachment Service, which allowed BlackBerry Enterprise Server (BES) to process PDF attachments for users to view on their BlackBerry devices. The flaw relates to how the service processes PDF files, which can be exploited via a maliciously crafted PDF. This implied that if someone loses their device and it's not locked, then it could be used to browse internally to that company's Web-based resources. This implies that corporate networks were at risk due to this flaw [7].

May 1st, 2006: A BlackBerry software vulnerability was detected that prevented users from opening email attachments. The software failed to open TIFF file attachments [8].

January 4th, 2006: Three vulnerabilities were discovered in the device that could allow an attacker to launch a Denial of Service attack on the BlackBerry device [9].

CODE SIGNING

RIM does not verify third-party applications that run on a BlackBerry device. However, RIM controls the use of BlackBerry device APIs by restricting third party applications access to sensitive and critical packages. RIM implements the restriction by enforcing code signing for all the third party applications that make use of sensitive packages and classes. If a user tries to run a third party application, which requires code signing but is not code signed then that application fails to load on the device.

RIM clearly states that it does not approve or endorse third party applications in any way but rather it is the developer's responsibility to ensure the proper implementation and use of the application [10]. RIM places some level of trust that the developer will supply true identity information when registering to receive a signing certificate. Below is an image of the signing certificate registration questionnaire.

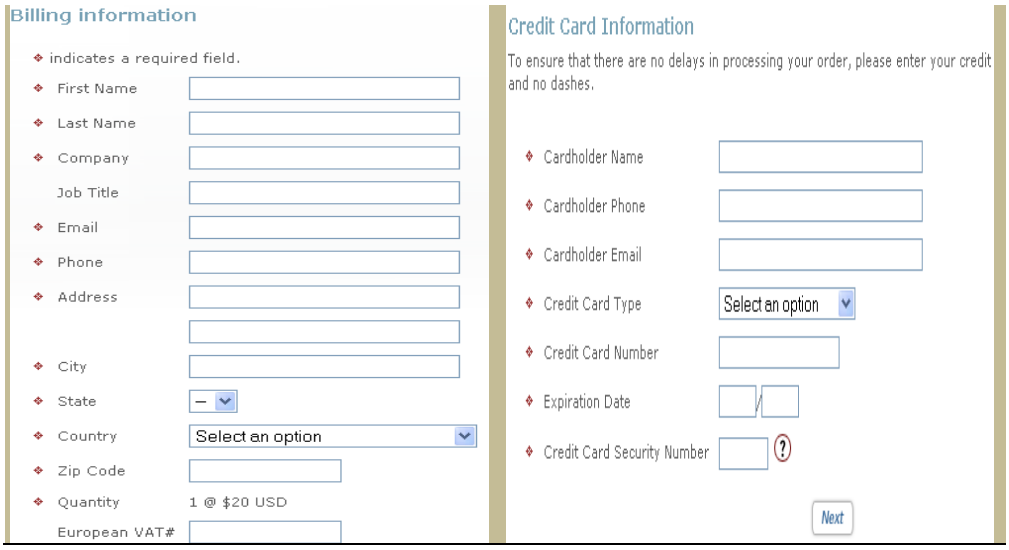

Figure 6: Screenshots of code signing form [11].

According to individuals well versed, it is fairly simple to obtain anonymous pre-paid credit cards [12]. These pre-paid cards can be bought with cash from a local store without supplying any form of identification. It could be implied that an attacker could also obtain an anonymous pre-paid credit card in order to register for an anonymous or fraudulent signing certificate directly from RIM. This registration and signing process is a fundamental component of the security framework of the BlackBerry device. If the attacker is able to obtain a fraudulent certificate, that same attacker could then develop malicious applications without risk of recourse.

SMobile would like to stress the importance of ensuring that third party vendors and application developers are properly protecting their signing certificates from unauthorized developers.

BLACKBERRY PROOF-OF-CONCEPT

The following section describes the (POC) applications developed by SMobile Global Threat Center to study and analyze the response of the BlackBerry security framework. These POC applications fall into two distinct categories, as mentioned in table 1. The research results are based upon a BlackBerry Curve 8310 from AT&T, running OS version 4.5.0.110. While developing the POC applications, emphasis was placed on the application's backend logic, not on the frontend content presented to the user. Therefore, some readers may find the frontend of the POC applications to be unexciting, but this frontend can be easily replaced by any business or game frontend.

Category	Purpose	Code Signing	Impact
Trojan	Delete all of the files from internal and external memory.	No	Loss of data.
Trojan	Send SMS messages to premium numbers.	No	Financial loss.
Trojan	Delete all the contact information from Phonebook.	No	Loss of personal information.
Spyware	Search for particular file types and email results as attachment.	Yes	Theft of confidential data.
Spyware	Email all contact information, Memo, To-do & Event list.	Yes	Theft of confidential data.

Table 1: List of Proof of Concepts for BlackBerry.

Trojan: The term Trojan Horse originally indicated a non-self-replicating program that was designed to allow an attacker to gain unauthorized access to a system or resource. Over time, the term has been shortened to simply "Trojan" and has also taken on the definition of a program that performs a malicious function other than that which it is advertised to perform. Often times, this includes stealing authentication credentials, facilitating theft of personal or financial data and/or allowing unauthorized access to a system or resource.

A. The first proof of concept application that will be discussed here is a Trojan Horse that is capable of deleting all of the files from the system and external memory (i.e. SD card) of the BlackBerry device. Development of this application did not require code signing. Therefore, it is impossible to determine the identity of the developer of the application. Fig. 7 represents the behavior of the BlackBerry device when the user runs the POC application. Fig. 7 consists of a sequence of events in a form of multiple screenshots arranged in clockwise direction, starting from a screenshot in the upper left hand corner.

   When the user clicks on the POC application for the first time, the device prompts the user to allow the "Jesus" application to gain "Trusted Application" status in the "Application Permissions" window. If the user clicks on "Yes" then the device sets this application as a trusted application. This particular "Application Permissions" window appears for all the third party applications as a component of the security framework when they are installed on a BlackBerry device. Clicking "No" at this request will cause the application to not be able to run, as the user is effectively telling the security framework that it does not trust the application that he is trying to install. If the user clicks "Yes", then clicks on the application again, he will enter the first instruction page of this particular application.

Reading through the application instructions, the user determines that he needs to select the menu key to choose an option as shown in the second image of the application sequence. When the user clicks on "View Commandments", the device prompts the user to set the "Application Permissions" that will apply to this "Jesus" application. In this particular instance, the "Application Permissions" window is asking user to allow or deny file connection to listen to messages, which is an interestingly vague request. If the user decides to select both of the options and clicks the "Allow" button, the BlackBerry will no longer prompt the user to allow this particular action.

However, what is not discussed when asked to allow these permissions is the fact that when the user is reading the commandment, the application is deleting all of the stored data from both the internal and the external memory. However, if the user had selected only the second option in the Application Permissions window, the BlackBerry would have displayed the "Application Permissions" window each time before deleting a new file. What has happened here is that the attacker developed the application is such a way that an uninformed user would not know what they are allowing the application to do.

Figure 7: BlackBerry's behavior on execution of POC.

Figure 8 and 9 represents the result of this POC. The first image represents the initial state of the device before running the Trojan, and the second column represents the state of device after running it. The screen captures illustrate that all the stored data from the test device has been deleted.

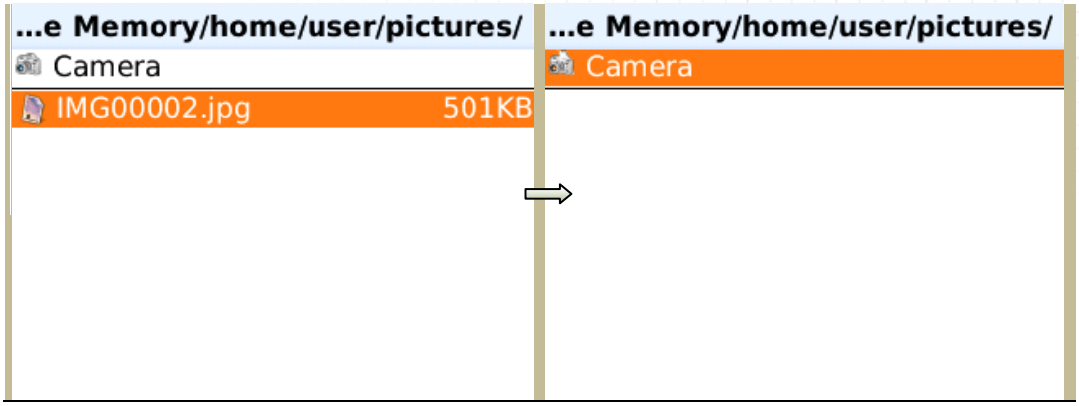

Figure 8: Initial state and final state of internal memory.

In today's fast-paced world, Smartphones have become an essential to personal and professional productivity. Most of today's Smartphone devices come with an extendable memory slot, allowing users to store large amounts of data. The inherent nature of BlackBerry devices is such that users would find it trivial to store and manipulate what might be considered sensitive corporate or personal data on their handsets. This POC demonstrates that it may be possible to trick a user into allowing a third party application to gain access to and delete this data, even though the security framework attempted to perform its function and alerted the user that an action that may be considered abnormal is occurring, albeit quite vaguely.

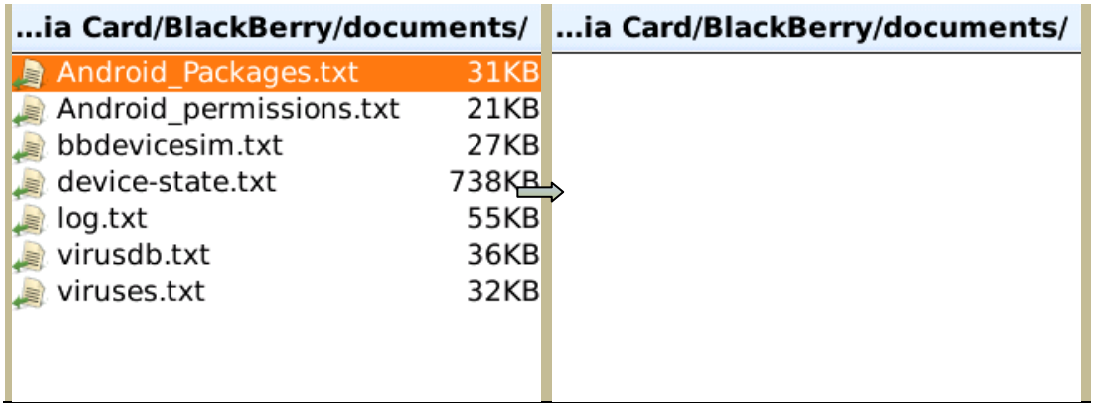

Figure 9: Initial state and final state of SD card.

B. The second POC application discussed here causes financial loss to the victim by sending SMS messages to a premium number. For testing purposes, the code of this POC application is modified to send SMS messages to an HTC Tilt (Windows) device that we control. This POC application does not require code signing and uses the same frontend application that was used by the previous POC application.

Fig. 10 depicts the sequence of events that takes place on the BlackBerry device upon starting the POC. The device prompts the user to grant the application "Trusted Application" status. Once set, the user selects the "Pictures" option from the menu, as instructed in the application's instructions. The device pops up the "Application Permissions" window to notify the user that the application has requested an SMS connection to the mentioned number as shown in the following screen capture.

So far, the BlackBerry's security framework does a good job of notifying the user that the application is attempting to call another application to perform a service. It just so happens that most people would consider this particular occurrence of sending an SMS to be an anomaly and would likely question what the application is trying to do. In this particular instance, it would be incumbent upon the developer to develop a frontend to the application in which it would make sense to send an SMS. For example, what if the developer were peddling some sort of game that gave the user the opportunity to send the scores to a site that tracks the scores. Instead of sending the scores to a useful site, it sends a series of SMS messages to a premium number in which the user would have no recourse to recover the expended funds.

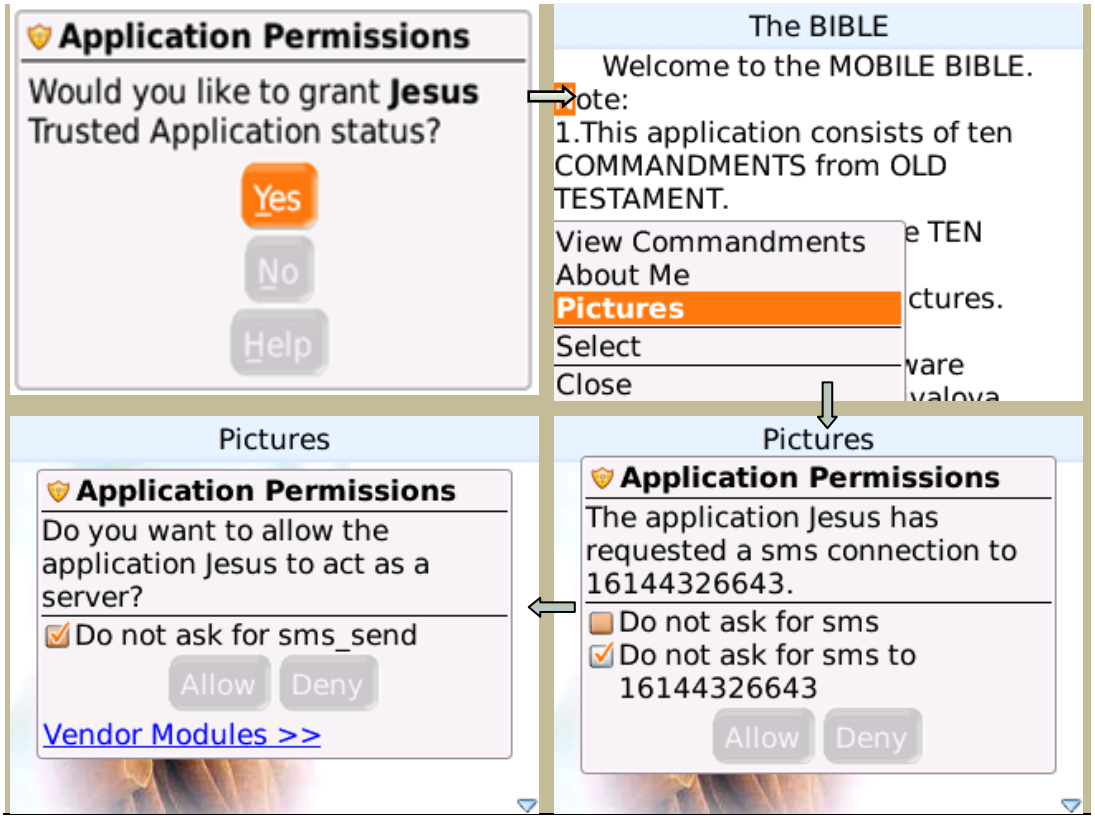

Figure 10: BlackBerry's behavior on execution of POC.

Fig. 11 illustrates that an SMS message was actually sent from the application to the phone number specified. Had this been a premium number, the user would have been billed for whatever the rate was. The first screen capture is from the "*SMobile Security Shield and Parental Control*" dashboard window that logs all of the sent and received messages from the target device. The next screen capture is from the device that has received the SMS message.

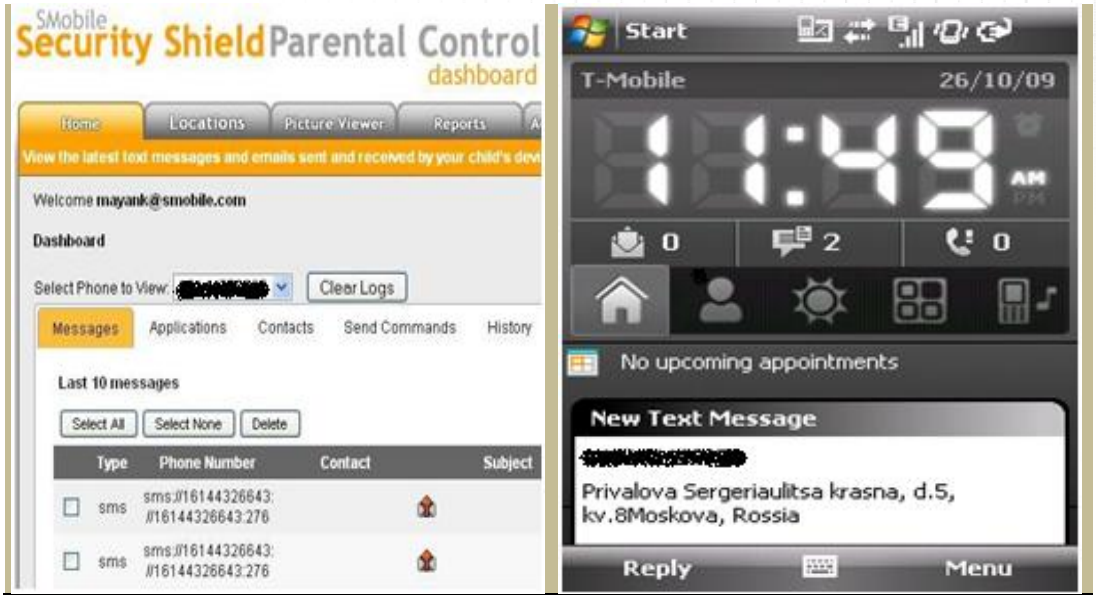

Figure 11: The screenshot of Security Shield logs and SMS message prompt on Test device.

C. The final POC Trojan is an SMS application that deletes all of the contact information from the BlackBerry handset. This application does not require code signing. Fig. 12 consists of the initial response of the device upon starting the application. As previously observed, the user is initially prompted to allow the application to be granted "Trusted Application" status on the handset. Once the user has granted the application the appropriate status, the user will then be presented with another "Application Permissions" window as shown in the next image. Here, we see that the BlackBerry security framework is notifying the user that the application wants to modify the user data. What we do not see is any detail that would alert the user to the fact that the modification in question is actually deleting contact data, as opposed to manipulating the data in some manner that makes sense for a normal SMS application. This pop up message appears only once, and if permission is granted, this pop up does not appear again. However, the user can change the permissions for the application anytime by editing them manually.

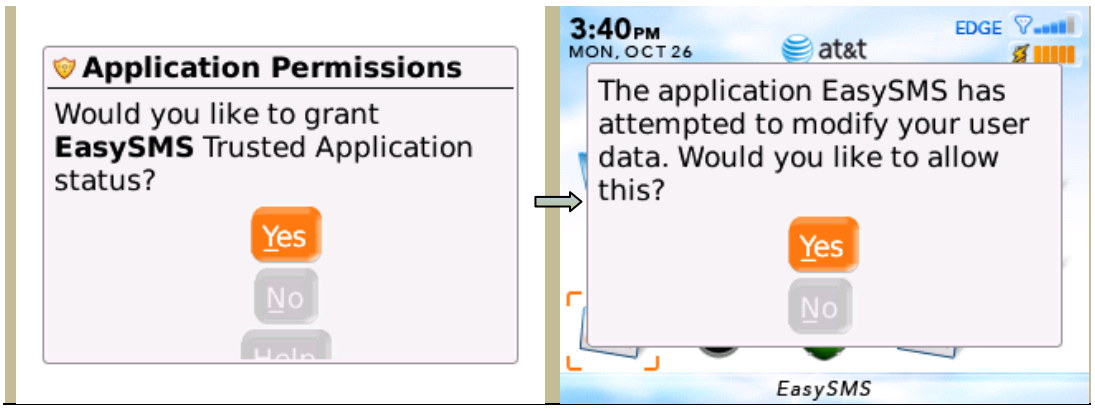

Figure 12: Initial response of BlackBerry device.

Figure 13 represents the initial state of the device phonebook and the state of phonebook after running the POC application. The screen capture of the device phonebook after running the

application shows no contact information on the device, which indicates that the POC was successful in deleting the contact information from the BlackBerry handset.

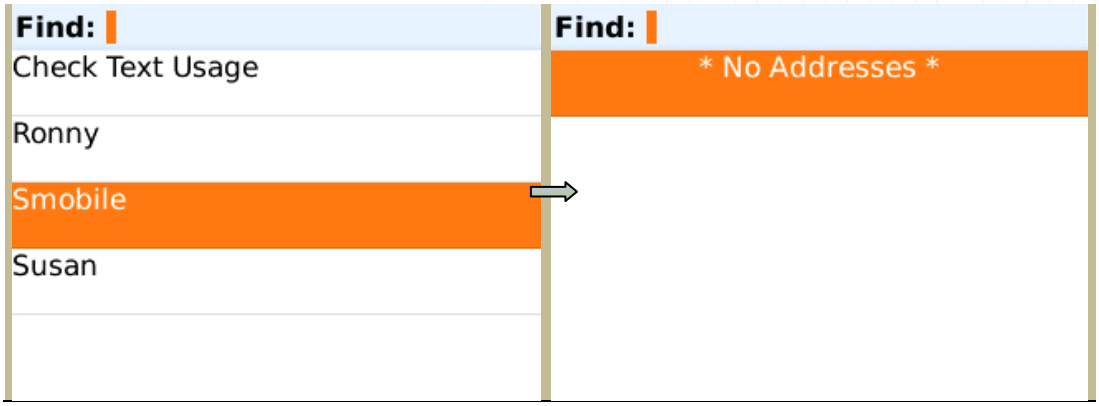

Figure 13: Screenshots of BlackBerry device initial & final state after running POC.

In this particular instance, as soon as the application is started, it tries to read and delete all of the contact entries in device's phone book. We notice that the BlackBerry security framework does its job, in as much as it alerts the user that the application is attempting to modify the user data. As was previously mentioned, the prompt that the user receives to allow the modification is rather vague and could easily be misconstrued as something a normal SMS application might need in order to function properly. At no time, does the security framework alert the user to the fact that the application is actually trying to delete important contact information from the handset. As soon as the user accepts the request to allow the application to modify the data, all of the contact data is successfully deleted. It comes as a surprise that this application did not require that the code be signed by the developer. It is also worth noting that this application can be extended or modified to also delete or modify stored Memos, Events and To-Do list entries.

Spyware: Historically, spyware is considered to be a malicious application that attempts to remain hidden from the victim while it collects information about the user. In mobile devices and Smartphone's, spyware has recently gained notoriety by allowing an attacker to intercept SMS, MMS and email messages, monitor phone calls and phone call logs, as well as track GPS locations. In some cases, it may even be possible to remotely activate the device's microphone and transmit the ambient conversation to the attacker. The key difference between a spyware application and others that perform similar functions is the fact that spyware attempts to remain hidden from the user.

A. The first POC application for spyware searches for a particular file format in the external memory and emails it as an attachment. The attacker can modify the malware application to search for a particular format. This spyware application scans the external memory for .doc and .jpeg file formats. Deployment of this type of application does require code signing. An interesting thing to note is that BlackBerry does not provide support for attachments for files located in the device memory.

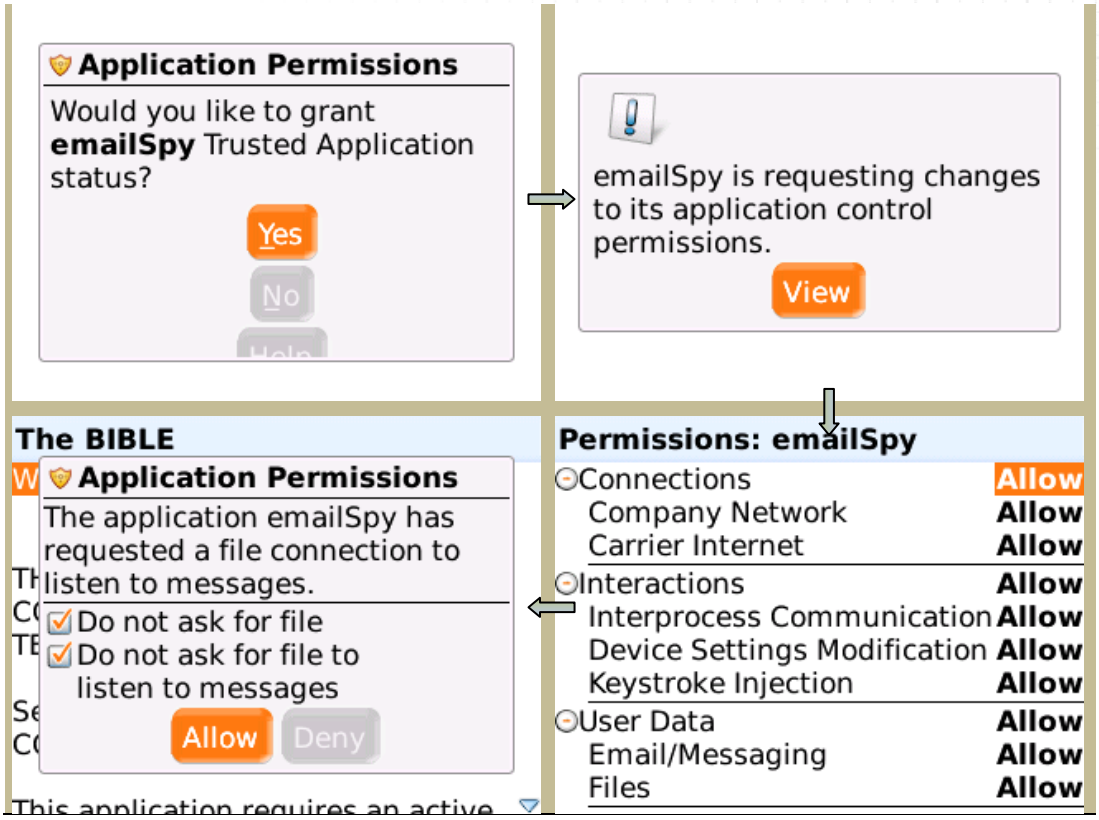

Figure 14: Initial response of BlackBerry on running the Spyware.

Figure 14 represents the initial response from the BlackBerry upon running the spyware application. Once the user tags the application as trusted, the spyware application pops up a window that requests changes to the "Application Control Permissions". The following screen capture represents the permissions window that the spyware application needs. The user has an option to manually deny each individual set of permissions or accept all of the permissions as default. The signed application serves as a blessing in disguise for the malicious application developer. The developer has an option of requesting the user to accept all the required permission changes at the beginning of the process.

Finally, the user enters the application and a new "Application Permissions" window pops up this time, notifying user that an application wants a file connection to listen to messages. This permission implies that the application wants to access the data stored on external memory. Figure 15 consists of screen captures of documents and pictures found on the external memory of the BlackBerry device.

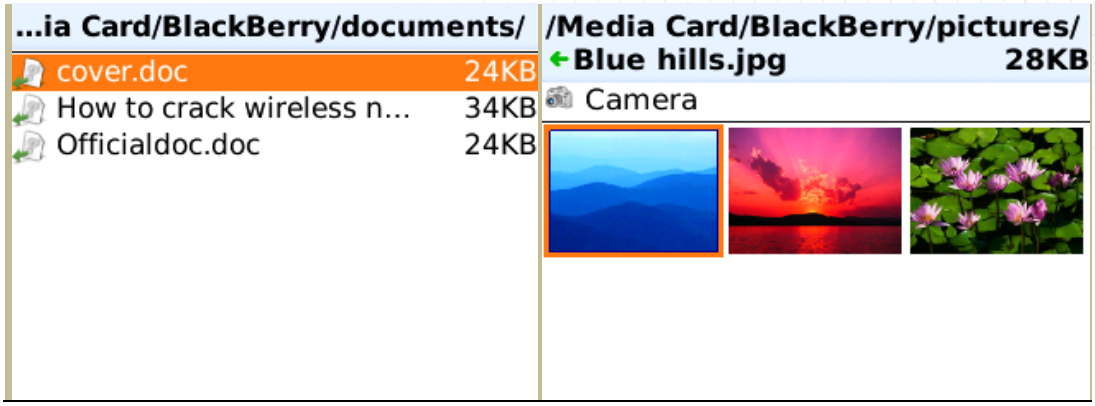

Figure 15: Screenshots of documents and pictures in the BlackBerry.

In this instance, as soon as the proper permissions are set (presumably by the attacker), the application starts a thread that begins searching the device for .doc and .jpeg file formats that exist in external memory. When a matching format is identified, the application starts another thread that attaches the file and emails it to a hardcoded email address.

Figure 16 represents the screen captures of the attacker's email inbox after the spyware application has identified and emailed the now stolen documents and images.

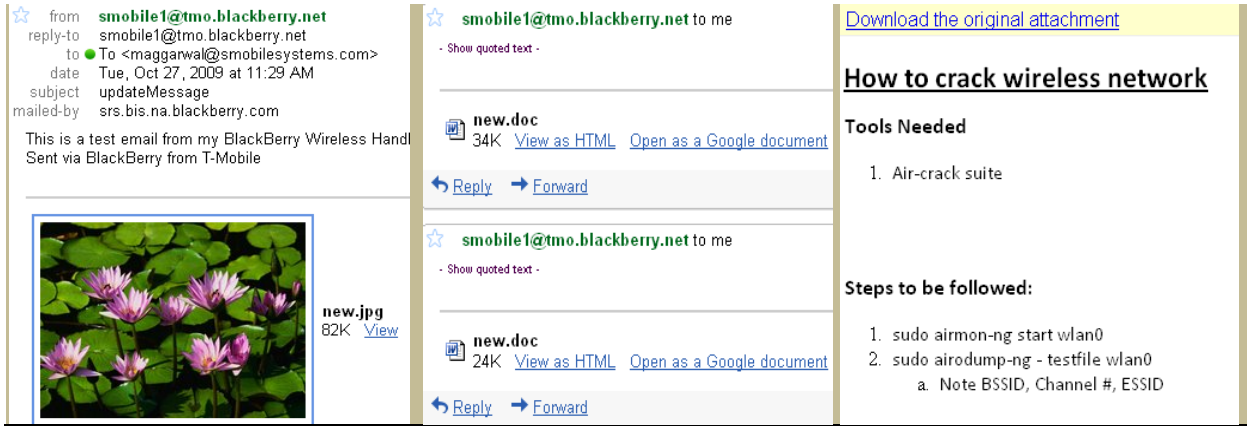

Figure 16: The screenshot of email received by an attacker.

Finally, the ultimate goal of every attacker is to hide the true intentions of their malware. The same is possible with this spyware application. Figure 17 represents the screen capture of the victim's device. The first image shows that certain emails were sent from the victim's device, and the next image illustrates that the victim has opened one of the sent emails. He notices an attachment, but receives a permission denied error when trying to view the contents of the attachment.

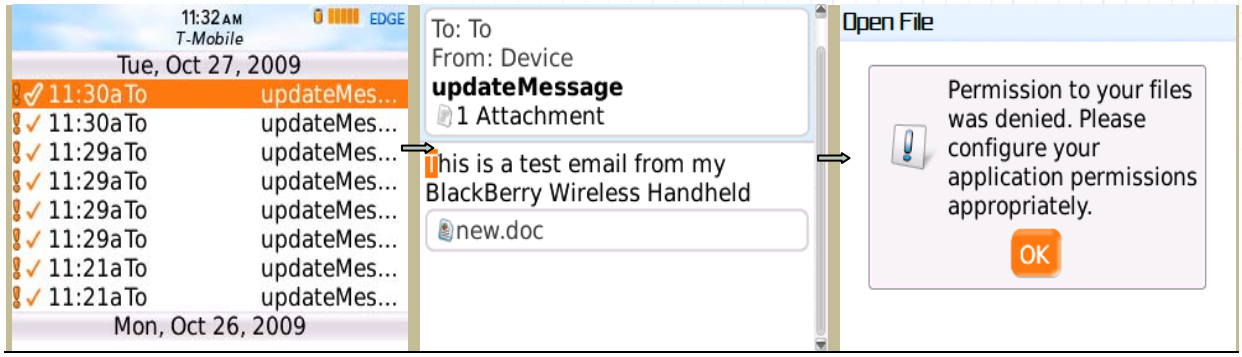

Figure 17: The screenshot of victim's BlackBerry.

B. The second POC spyware application reads all of the contact information, writes it to the message body, and then sends it as an email. This POC can easily be modified to include Memo, To-Do and Event lists.

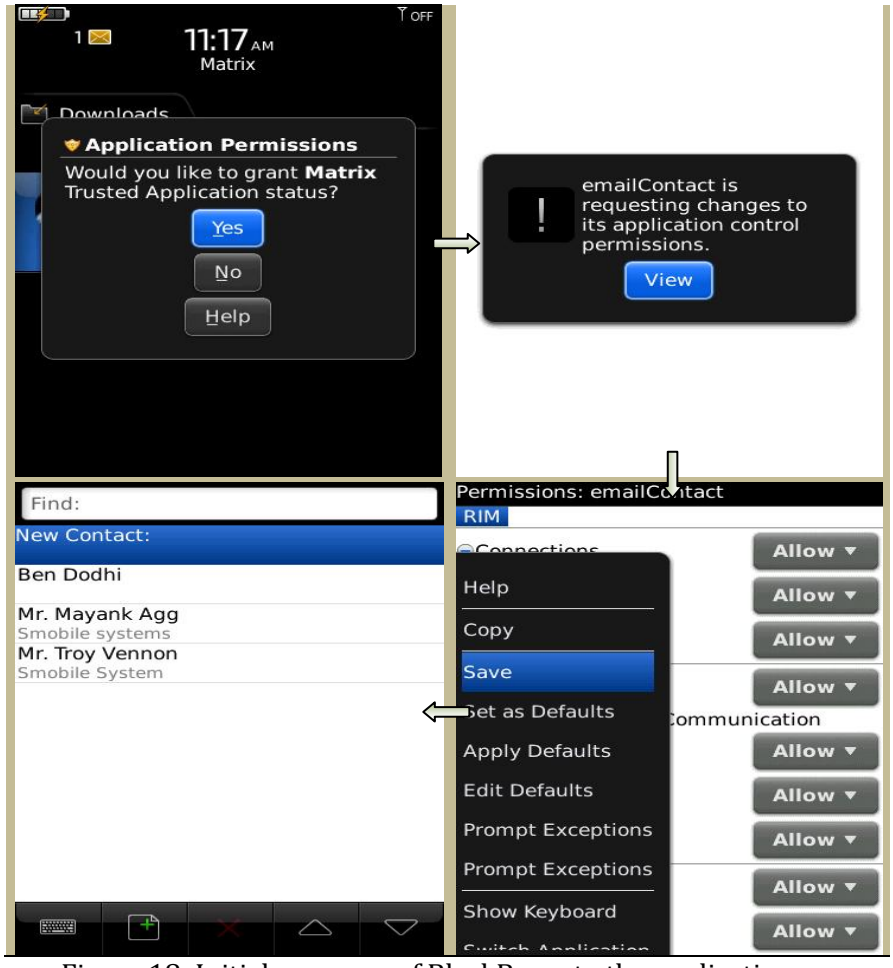

Figure 18: Initial response of BlackBerry to the application.

Figure 18 shows the initial response of the BlackBerry upon starting the application. The response is the same as was observed in the prior examples. The final screen capture (i.e. the image at lower left) is of the address book on the victim's device. As soon as the victim starts the POC application,

the application scans the victim's address book and sends the address book as an attachment to the email address that is hardcoded into the application.

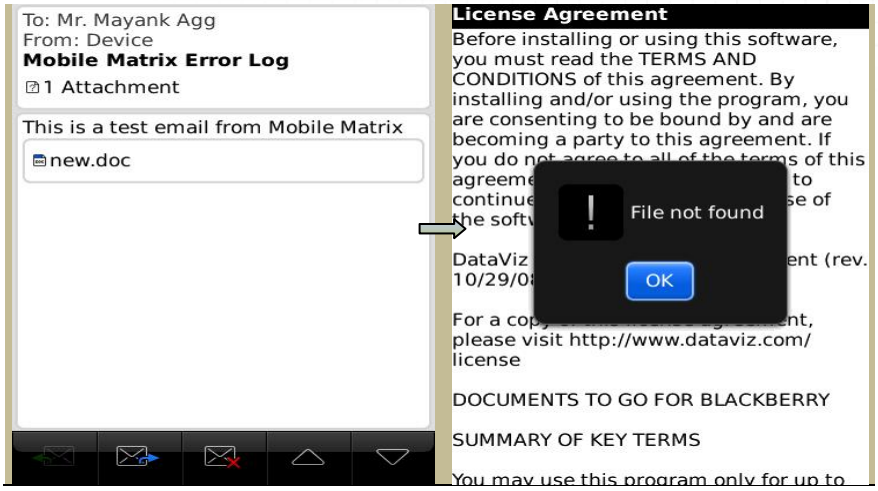

Figure 19: The screenshot of victim's BlackBerry.

Figure 19 represents the screen capture of the email sent by the POC application from the BlackBerry device. As we can see, the email consists of an attachment, but on the following screen capture, we see that the victim is not able to open the attachment to view its content. The attacker implements this to remain discrete and to elude suspicion of the true nature of the attachment.

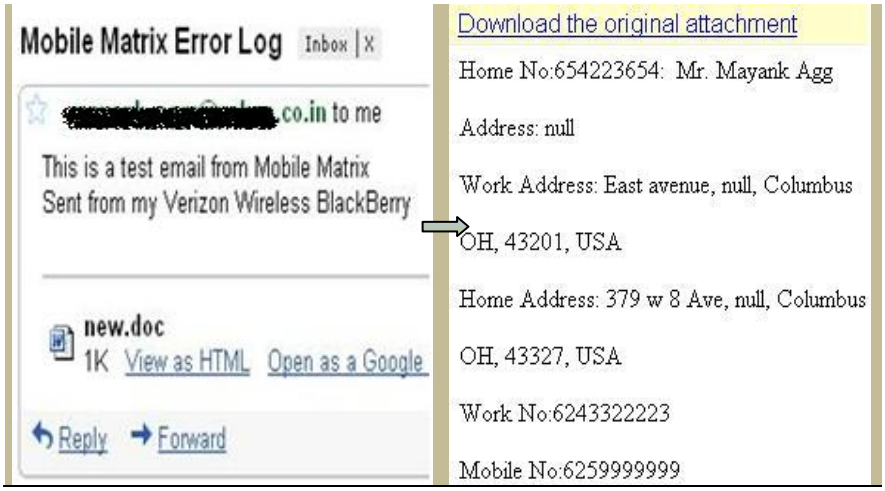

Figure 20: The screenshot of attacker's email.

In Fig. 20, we see the screen captures of the attacker's email inbox. The attacker has received an email from the victim's device and we see the victims address book entries are embedded in the attachment.

CONCLUSION

Through the course of developing and testing POC code that was designed to bypass security controls established by the BlackBerry security framework, SMobile was able to determine that there are certain instances of attacks that may be successful in bypassing the security posture of the device. From the information obtained by the various application test cases, the primary attack vector that could lead towards the loss of confidentiality, integrity and/or availability of sensitive data would be through the use of social engineering methods to trick users into downloading and installing malicious third party applications on the device. As was illustrated, the BlackBerry security framework appeared to function properly in alerting the user that the third party applications should be confirmed as trusted and whenever an application attempted to perform a service or function outside of its activity. However, it was noted that the notifications that prompted users that the applications were requesting permissions that might be considered abnormal were often vague and/or misleading. Enterprising attackers would not find it difficult to develop malicious applications around the context of the permission alerts that are being generated, as a few examples were given.

This research paper outlined malware applications that are capable of performing a number of malicious activities, including sending SMS messages that could lead to financial loss, deleting stored data, deleting contact information, surreptitiously searching for and emailing files of various formats, and stealing contact information. Other capabilities that have been identified as viable attacks against BlackBerry devices in the wild are sniffing calls, SMS, live call interception, and stealth GPS tracking. For these attacks to succeed, the malware application will need to be installed on the device.

Our research also indicates that the application signing process for developers is somewhat effective. Unsigned malicious applications proved somewhat effective, if not slightly limited by the access granted to specific API's. However, leveraging signed, malicious code provides plenty of options to an attacker to exploit the BlackBerry device, even though signing restrictions may go a long way towards deterring the casual developer from creating malicious applications. If an individual were intent upon creating malicious applications, we believe the signing process may be fairly trivial to defraud or anonymize, through the use of anonymous, pre-paid credit cards and fraudulent information.

As BlackBerry use and market share continues to grow amongst enterprise users and everyday consumers, SMobile stresses the importance of user education. The BlackBerry Enterprise Server solution provides fairly robust policy capabilities that can and should be managed by security professionals to ensure user and corporate data is secured on the device. However, normal consumers are all too often left to their own devices when asked to secure their information. It's these users that need to seek out and understand the risks associated with installing untrusted, third party applications, coupled with understanding the rights and permissions that are being granted to these applications upon install.

Along with the educational aspect of protecting sensitive data, it still remains imperative to leverage technology and tools to provide protections as well. Just as even the least technical individuals would no longer find it acceptable to conduct financial or sensitive transactions from a PC that is not protected by some sort of firewall and anti-virus software, users should consider the same protections for their Smartphone handsets. As we've seen with other Smartphone platforms, sensitive data residing on the device is not the only threat users are exposed to. Malware currently

exists that is designed specifically to take money out of the pocket of the user and pass it directly to the attacker in the form of premium SMS messages. Even users who believe they don't use their devices to manipulate sensitive data could still be at risk, financially. SMobile Anti-virus solution for BlackBerry provides the necessary detection and removal capabilities to thwart these and other types of malware attack that consumers faces today.

REFERENCES

1. Canalys- Independent technology focused analyst house, http://www.canalys.com/pr/2009/r2009112.htm
2. FlexiSpy- Commercial Mobile Spyware application, http://www.flexispy.com/spyphone-remote-listening-blackberry.htm
3. MobileSpy- Commercial Mobile and PC Spyware application, http://www.mobile-spy.com/spy-blackberry.html
4. BBProxy- Article write-up, http://www.wired.com/science/discoveries/news/2006/08/71548
5. BlackBerry Vulnerability news, http://www.eweek.com/c/a/Security/RIM-Plugs-BlackBerry-Security-Hole-165742/
6. BlackBerry Vulnerability news, http://www.scmagazineus.com/BlackBerry-security-hole-patched/article/127235/
7. BlackBerry Vulnerability news, http://www.itresource.com.au/2008/07/16/blackberry-security-flaw/
8. BlackBerry Vulnerability news, http://www.out-law.com/page-6509
9. BlackBerry Vulnerability news, http://www.theregister.co.uk/2006/01/04/blackberry_security_bugs/
10. BlackBerry- Code Signing Keys, http://na.blackberry.com/eng/developers/javaappdev/codekeys.jsp
11. BlackBerry- Code Signing Keys form, https://www.blackberry.com/SignedKeys/
12. http://www.theprivacyguy.com/2007/03/30/anonymous-prepaid-credit-cards/

About SMobile Systems

SMobile Systems, founded in 2002 and headquartered in Columbus, Ohio, is the world leader in providing comprehensive software security solutions for all major mobile device platforms, including BlackBerry, Windows Mobile, Symbian, Palm, iPhone and Android.

In response to the growing demand for mobile device security, SMobile has created a complete mobile security suite including Antivirus, Firewall, AntiSpam, Anti-Theft and Identity Protection, Secure Mobile Banking, and Parental and Enterprise Controls.

SMobile's mission is to enable end-to-end voice and data continuity on wireless networks by providing a range of specialized, leading edge security software and services, all built on a proprietary secure back plane infrastructure.