Motivations of Recent Android Malware

Eric Chien
Technical Director, Security Response

Introduction

Every year for the last decade, the security industry has predicted a flood of mobile malware; however, only a trickle of mobile malware has emerged. The most widespread threats were SymbOS.Cabir and SymbOS.Commwarrior in 2004 and 2005. For most they represented a nuisance and affected a very small fraction of the phone population.

Three factors are needed before an increase of mobile malware will occur: an open platform, a ubiquitous platform, and attacker motivation—which is usually monetary. The first has been fulfilled most recently with the advent of Android. It is probably also the most likely open platform to achieve the second condition of being ubiquitous. Given that Android is now the most prolific smartphone operating system (43% of worldwide smartphone market in the second quarter of 2011 according to Gartner), the continued rise in market share seems all but inevitable, at the very least due to the adoption of smartphones in general over regular phones.

The most uncertain condition is the third, an ability to monetize the platform via malware. This paper discusses some of the monetization schemes seen in a recent spate of Android malware and also schemes we’re likely to see in the future. Only if these monetization schemes succeed do we expect attackers to continue to invest in the creation of Android malware.
Current Motivations

**Premium Rate Number Billing**

In this scenario, attackers set up and register a premium-rate number. Typically, these are “short codes”, which are shorter than usual phone numbers. Each country and carrier regulates short codes differently, but usually an oversight body issues the short codes for a fee. In the United States for example, a dedicated short code may cost $1500 USD to set up and then $1000 per month. A shared short code where the message must be preceded by a keyword can be obtained for as low as $50 per month.

When calling or sending an SMS to a short code, the caller is billed a premium rate above the normal cost of an SMS or phone call. The revenue is then shared by the attacker, carrier, and the SMS aggregator. The attacker receives 30-70% of the premium rate charge depending on the carrier, amount charged per message, and number of messages received. Most carriers allow a premium rate of up to $10.00 per message, but some carriers will allow charges in excess of $50.00 per message. If the attacker uses an SMS aggregator, the attacker will pay an additional fee. SMS aggregators provide short code services such that clients share the same short code, but are able to bill and differentiate services by ensuring users place a specific keyword related to their service within the SMS. This allows multiple services to essentially split the cost of a single short code number.

Android applications can request permissions to send SMS messages at installation. These SMS messages can be sent without the user confirmation. Sending an SMS to a premium short code causes the phone owner to incur a charge on their phone bill and the attacker to generate revenue. An application can easily send multiple messages, inflating charges. However, short codes are usually carrier and country-specific. This means multiple short codes are needed or threats may only target specific regions.

Premium-call phone numbers are also available, but may be restricted from automatically beingdialed on some devices. In addition, the dialer will be present on-screen and possibly noticed by the user.

**Android.FakePlayer** is a threat that sends multiple messages to two short-code, premium-rate numbers. The threat sends two messages at the premium rate of approximately $3.50 to the first number and another at $6 to the second, resulting in a $13 charge each time the application is executed (for those on participating networks in the Ukraine or Kazakhstan).
Implementing code to send SMS messages to premium short codes is quite simple, since the Android SDK provides a simple API. First, the application must request the SEND_SMS permission.

**Figure 2**

**Requesting the SEND_SMS permission**

```xml
<activity>
    </intent-filter>
</application>
<uses-permission
    android:name="android.permission.SEND_SMS"
>
</uses-permission>
</manifest>
```

Then, just call the sendTextMessage API with the premium phone number’s short code. In the Android.FakePlayer example below, a text message is sent to the phone numbers 3354 and 3353.

**Figure 3**

**Android.FakePlayer sending a text message**

```
| 001f: invoke-virtual/range {v0, v1, v2, v3, v4,
Landroid/telephony/SmsManager:.sendTextMe |
| 0022: const-string v1, "3354" // string 0002 |
| 0024: const/4 v2, #int 0 // #0 |
| 0025: const/4 v4, #int 0 // #0 |
| 0026: const/4 v5, #int 0 // #0 |
| 0027: invoke-virtual/range {v0, v1, v2, v3, v4, |
```

The SMS messages are sent in the background without prompting the user. Once the message is sent, the user is billed and may not even realize they were sent. Messages will still be saved in the sent box, but can be deleted by the malicious program.

**Spyware**

Multiple Android applications exist that allow someone to track and monitor a user of a mobile phone. For example, these applications may record and export all SMS messages, emails, call logs, GPS locations, or turn on the microphone. Typically, these applications require an attacker to purchase the application from the vendor and then gain physical access to the phone. While these applications may not generate revenue for the attacker, they generate revenue for the vendor of the spyware application. Examples include Android.Tapsnake and SpywareFlexispy. Such applications can sell for $400 and some of them are available on the Android Market.

<table>
<thead>
<tr>
<th>Action</th>
<th>Permission Required</th>
<th>API</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept SMS messages</td>
<td>RECEIVE_SMS</td>
<td>BroadcastReceiver</td>
</tr>
<tr>
<td>Read SMS messages</td>
<td>READ_SMS</td>
<td>getContentResolver()</td>
</tr>
<tr>
<td>Record audio</td>
<td>RECORD_AUDIO</td>
<td>AudioRecord.startRecording()</td>
</tr>
<tr>
<td>Read call logs</td>
<td>READ_CONTACTS</td>
<td>CallLog.Calls</td>
</tr>
<tr>
<td>Obtain GPS Coordinates</td>
<td>ACCESS_FINE_LOCATION</td>
<td>LocationManager</td>
</tr>
<tr>
<td></td>
<td>ACCESS_COARSE_LOCATION</td>
<td></td>
</tr>
</tbody>
</table>

Some of these applications also tout advanced capabilities such as recording phone calls. However, to enable some of these features, the phone must be rooted (the default system security removed). Nevertheless, without rooting the device, data can still be obtained by requesting standard permissions. Table 1 shows the permission required and APIs used to perform such tasks in an Android application.
Android.Tapsnake is an example of spyware that pretends to just be a game of snake, actually including a fully functional copy of the game.

However in the background, the application is uploading the GPS coordinates of the device every 15 minutes. The attacker then uses another program to view the saved locations as displayed below.

Android.Tapsnake tracking the compromised device
Search Engine Poisoning

Some search engines recommend sites or change search engine rankings by monitoring user’s visit rates. These recommendations may be further customized when using a mobile version of the search site, monitoring visits explicitly by mobile users. Malicious applications can initiate multiple requests to these sites, poisoning the hit rates monitored by the search engines. Artificially raising their search rank allows attackers to increase visits by prospective customers or generate revenue through pay-per-view or pay-per-click advertisements shown on the site.

Android.Adr is an example of a threat that was poisoning Baidu mobile search results by generating artificial visits to a mobile news site, potentially increasing their rank on Baidu search results.

In addition, Android.Adr may have had a second strategy. Baidu affiliates can place a search box on their site and users that search through this box will be shown search results along with advertising. Baidu then pays the affiliate who brought them the search traffic a share of any revenue generated from clicks on the advertisements. The revenue share ranges from 30-50%. The more searches conducted through the affiliates search box, the higher the revenue share.

Figure 7
Search engine poisoning process

An attacker begins by creating a mobile website, such as a news portal

The attacker registers with an affiliate program and includes a search engine in the page

Previously compromised devices boost the site’s ranking by generating artificial visits

A user visits the site and uses the search engine

The search results that are returned include advertisements

Compromised devices repeatedly hit the search engine with fake search queries, boosting the affiliate’s rankings

If the user purchases goods through the advertisements, the attacker earns revenue through the affiliate program
The attackers behind Android.Aдрд first registered with the Baidu Traffic Union program. Then they placed an affiliate search box on their mobile news site, 聚焦网 (Focus Online) at 3g.surfocus.com. If a regular visitor came this site and conducted a search the attackers would legitimately share any ad click revenue from those searches. However, if the number of average daily searches conducted was low, the attacker’s revenue share would only be 30%.

What Android.Aдрд does is repeatedly contacts the following URL, mimicking searches from the search box on the Focus Online website:

http://wap.baidu.com/s?word=[ENCODED SEARCH STRING]&vit=uni&from=[ID]

While these searches do not generate any direct revenue (since revenue is generated by clicking on subsequent advertising) they do increase their number of average daily searches, increasing their revenue share. When a legitimate search is conducted through their site, instead of only sharing in 30% of the revenue, they now share in up to 50%. In addition, the “popularity” of their site may increase their site’s ranking, drawing more traffic directly to it, where they can monetize further through advertisements.

**Pay-Per-Click**

A variety of services, such as advertising networks, pay each time an affiliate refers a user to a particular website (pay-per-click). Using malicious applications, the attacker can generate artificial visits to these websites and receive a few cents per click.

While pay-per-click advertising payouts are relatively low and require high-volumes to generate reasonable revenue, other pay-per-click opportunities exist. Many carriers provide value-added services that are billed directly to the phone bill when utilized. (For example, ring tones and news alerts.) In China, mobile TV is a widespread value-added service and content providers can participate in revenue sharing schemes with the carrier on a pay-per-view basis. An attacker can create and then register a video channel with a carrier, generating revenue each time a user views their video or channel. Malicious applications can surreptitiously download such video content, generating revenue for the attackers.

*Android.Bгser* is a threat that utilizes this monetization scheme. First, it changes the access point name (APN) settings to cmnet/mmсс.mонternет.com, which services a Chinese mobile TV network. Next, the threat downloads a configuration file that specifies which video to download. Downloading a video through this network allows billing to occur and the attackers to generate revenue. Depending on the video, the attacker may generate (in China) up to a $1 USD per view. As carriers in other countries expand such offerings, revenues may be larger with respective economies.
Pay-Per-Install

Pay-per-install schemes in the mobile marketplace often refer to a different model than pay-per-install schemes in the PC malware space. In the mobile marketplace, the term often refers to legitimate distribution marketplaces that host applications for download and charge the vendors based on the number of downloads and installs. Pay-per-install in the PC malware space refers to the reverse – the distributor pays an affiliate each time they are able to install an application on a user’s computer. This allows attackers to generate revenue by installing pay-per-install software on compromised computers. While many pay-per-install schemes exist for PC applications, only a few exist for mobile applications.

The affiliate is paid up to $1 USD for each mobile application installed. While we have not seen attackers use pay-per-install schemes to generate revenue, multiple threats have included the ability to install new applications. The purpose of this functionality could have been to install pay-per-install software. Android.Geinimi and Android.Rootcager are examples of threats that can download and install additional applications.

Adware

Many advertising networks pay content providers for each view and click when they display their ads, averaging around $1-2 USD per thousand impressions. A malicious application can simply display the advertisement by launching a browser or displaying in-application advertisements on the mobile device. For example, multiple attackers have simply repackaged or cloned popular, legitimate games and included a mobile advertisement library registered to themselves. The modified mobile application is then seeded on application marketplaces. Every time the application is used and ads are displayed, the attacker generates advertising revenue. The application itself works as originally designed and the user is unaware they are using an illegitimate version.

mTAN Stealing

When making a transaction or logging into an online bank account, some banks require additional credentials sent out-of-band to prevent man-in-the-middle attacks. In particular, the bank will send a random number, known as Transaction Authentication Number (mTAN), to a previously registered mobile phone number. For attackers to succeed, they need malware on the phone in order to receive this number. Android.Smssniffer is an example of a threat that we believe has been used for such activities.

Future Motivations

While the above schemes have been seen used by recent Android malware, future possibilities exist. For example, data selling is quite lucrative in the PC space. Stealing information such as login credentials and financial data is the primary motivation for malware in the PC space. Mobile devices provide an additional vector when devices are used as payment devices via protocols such as near-field communications (NFC) that allow someone to pay for goods using their mobile device. How malware may take advantage of mobile payment devices remains to be seen, as this payment method is still in its infancy.
Another example of monetizable data is stealing identifiers such as the IMEI—a unique number that identifies a particular device. While cloning mobile phones using data gathered from applications isn’t possible (for GSM an additional value must be obtained directly from the SIM card), IMEIs can be sold and reused on previously blocked phones, or counterfeit phones that may not have proper IMEIs. Many of the recent Android threats do export the IMEI (and IMSI); however, the purpose of exporting these values appears to be to uniquely identify the infected device, rather than reselling these identifiers.

Finally, a common risk on the PC is fake security products that trick a user into purchasing a full version of the software to remove nonexistent threats. This model of revenue generation could equally work on a mobile device. Interestingly, in China a similar scheme was reported where phones were preinstalled with Fei Liu, a download manager application, which was claimed to have caused system reliability issues and unconfirmed reports of improper billing. In addition, these same phones were reported to have a mobile security product installed from NetQin, which would only remove Fei Liu if the user paid an additional $2 USD. However, according to investigators, NetQin was colluding with Fei Liu and generating revenue on removal of a product they created.

Summary

Monetization is a key factor necessary for the rise of mobile malware. Currently, the mobile technology landscape provides some malicious monetization opportunities, but none at the revenue scale achievable in Windows. Most mobile monetization schemes have an extremely low revenue-per-infection ratio and thus, to become worthwhile require a large number of infections. This ratio could increase in the future when more devices store credentials backed by monetary funds. Further, for each attack we have seen on Android, none were repeated. It is possible that the attackers did not generate enough revenue, and thus did not repeat the effort. So while we will continue to see malicious Android applications, additional advances in the mobile technology space that allow greater monetization are likely required before malicious Android applications reach parity with Windows.
About Symantec
Symantec is a global leader in providing security, storage and systems management solutions to help businesses and consumers secure and manage their information. Headquartered in Mountain View, Calif., Symantec has operations in more than 40 countries. More information is available at www.symantec.com.

About the author
Eric Chien is a Technical Director for Symantec Security Response in Culver City, California.

For specific country offices and contact numbers, please visit our Web site. For product information in the U.S., call toll-free 1 (800) 745 6054.

Symantec Corporation
World Headquarters
350 Ellis Street
Mountain View, CA 94043 USA
+1 (650) 527-8000
www.symantec.com

Copyright © 2011 Symantec Corporation. All rights reserved. Symantec and the Symantec logo are trademarks or registered trademarks of Symantec Corporation or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners.