A Survey of Mobile Malware in the Wild

Adrienne Porter Felt, Matthew Finifter, Erika Chin, Steven Hanna, and David Wagner
University of California, Berkeley
{apf,finifter,emc,sch,daw}@cs.berkeley.edu

ABSTRACT

Mobile malware is rapidly becoming a serious threat. In this paper, we survey the current state of mobile malware in the wild. We analyze the incentives behind 46 pieces of iOS, Android, and Symbian malware that spread in the wild from 2009 to 2011. We also use this data set to evaluate the effectiveness of techniques for preventing and identifying mobile malware. After observing that 4 pieces of malware use root exploits to mount sophisticated attacks on Android phones, we also examine the incentives that cause non-malicious smartphone tinkerers to publish root exploits and survey the availability of root exploits.

Categories and Subject Descriptors

D.4.6 [**Operating Systems**]: Security and Protection; K.4.1 [**Public Policy Issues**]: Abuse and Crime Involving Computers

General Terms

Security

Keywords

Mobile devices, smartphones, malware

1. INTRODUCTION

People use smartphones for many of the same purposes as desktop computers: web browsing, social networking, online banking, and more. Smartphones also provide features that are unique to mobile phones, like SMS messaging, constantly-updated location data, and ubiquitous access. As a result of their popularity and functionality, smartphones are a burgeoning target for malicious activities.

Researchers have been studying mobile phone security for several years [32, 36]. At first, mobile malware was proof-of-concept. Over time, however, mobile malware has become a real threat. We survey the state of modern mobile malware in the wild to illuminate the current threat model and suggest future directions. Our survey encompasses all known iOS, Symbian, and Android malware that spread between January 2009 and June 2011. We collected information about 46 pieces of malware in this time period: 4 for iOS, 24 for Symbian, and 18 for Android.

In order to understand the motives of real mobile malware, we classify the malware in our data set by behavior. We find that the most common malicious activities are collecting user information (61%) and sending premium-rate SMS messages (52%), in addition to malware that was written for novelty or amusement, credential theft, SMS spam, search engine optimization fraud, and ransom. We describe the incentives that promote each type of malicious behavior and present defenses that disincentivize some of the behaviors. In particular, malware that abuses SMS messages could be prevented with small changes to the Android and Symbian platforms. We also discuss incentives that we believe will motivate future mobile malware.

iOS, Symbian, and Android use application permissions or review processes to prevent the spread of malware. We consider whether these mechanisms are effective defenses against the malware in our data set. First, we investigate whether a simple classifier could use the permission requests of Android malware to identify malicious applications; we find that the permission to send SMS messages is the most promising feature. Next, we determine that Apple's review process has successfully avoided approving malware, but Symbian's review-and-sign process failed to prevent almost a third of the Symbian malware.

Smartphone OS security mechanisms can be circumvented using root exploits. Unfortunately, smartphone owners are currently motivated to seek and publish root exploits that can be leveraged by attackers. 4 pieces of malware in our data set use root exploits which were originally published for the purpose of smartphone customization. We discuss the incentives that encourage the publication of root exploits by non-malicious smartphone owners. Our survey of root exploits for 6 Android phones demonstrates that root exploits are publicly available more than 74% of the time, rendering phones vulnerable to sophisticated malware.

Contributions. In this paper, we:

- Survey the behavior of current mobile malware, present defenses, and discuss the future of mobile malware.
- Evaluate whether existing defense mechanisms are effective at identifying current mobile malware.
- Examine the incentives that encourage the publication of root exploits and survey the availability of exploits.

2. THREAT MODEL

We present three types of threats posed by third-party smartphone applications and discuss the security measures that are intended to detect and prevent them.

2.1 Threat Types

The mobile threat model includes three types of threats: malware, grayware, and personal spyware. We distinguish between the three based on their delivery method, legality, and notice to the user. This paper focuses specifically on malware; personal spyware and grayware use different attack vectors, have different motivations, and require different defense mechanisms.

Malware. *Malware* gains access to a device for the purpose of stealing data, damaging the device, or annoying the user, etc. The attacker defrauds the user into installing the malicious application or gains unauthorized remote access by taking advantage of a device vulnerability. Malware provides no legal notice to the affected user. This threat includes Trojans, worms, botnets, and viruses. Malware is illegal in many countries, including the United States, and the distribution of it may be punishable by jail time.

Personal Spyware. Spyware collects personal information such as location or text message history over a period of time. With *personal spyware*, the attacker has physical access to the device and installs the software without the user's knowledge. Personal spyware sends the victim's information to the person who installed the application onto the victim's device, rather than to the author of the application. For example, a person might install personal spyware onto a spouse's phone. It is legal to sell personal spyware in the U.S. because it does not defraud the purchaser (i.e., the attacker). Personal spyware is honest about its purpose to the person who purchases and installs the application. However, it may be illegal to install personal spyware on another person's smartphone without his or her authorization.

Grayware. Some legitimate applications collect user data for the purpose of marketing or user profiling. *Grayware* spies on users, but the companies that distribute grayware do not aim to harm users. Pieces of grayware provide real functionality and value to the users. The companies that distribute grayware may disclose their collection habits in their privacy policies, with varying degrees of clarity. Grayware sits at the edge of legality; its behavior may be legal or illegal depending on the jurisdiction of the complaint and the wording of its privacy policy. Unlike malware or personal spyware, illegal grayware is punished with corporate fines rather than personal sentences. Even when the activity of grayware is legal, users may object to the data collection if they discover it. Application markets may choose to remove or allow grayware when detected on a case-by-case basis.

2.2 Security Measures

Smartphone operating system vendors use curated markets and/or application permissions to protect users. We focus on iOS, Android, and Symbian 9.x.[1]

[1]We limit our consideration of Symbian malware to Symbian 9.x. Earlier versions of the Symbian OS do not have all of the security measures discussed in this paper.

Markets. Smartphone users are encouraged to download and purchase applications from centralized application markets. Apple, Google, and Nokia promote the use of centralized markets with decreasing strictness.

Apple iOS devices allow the user to install applications only from the Apple App Store [4], and applications in the App Store are reviewed by Apple for security. If iOS users want to install applications from other sources, then they must *jailbreak* their devices, which involves exploiting a vulnerability in iOS to gain superuser access. This process carries the risk of rendering the phone inoperable, and it voids the phone's warranty. Apple's review process is intended to prevent malware from being distributed through the App Store. Their review standards also disallow personal spyware, but an attacker with physical access to the victim's device could jailbreak the phone without the victim's knowledge to install personal spyware. The App Store is known to have included grayware [31, 24]; in some cases, the grayware has been removed from the App Store.

Android also provides users with an official application store, the Android Market [2]. Most Android phones allow users to also install applications from unofficial markets, although the user is warned that installing non-Market applications may expose the user to malware. Google does not review applications prior to listing them in the Android Market, although they may review some applications later. Personal spyware (e.g., GPS Spy Plus) and grayware are listed in the Android Market [25]. The Android security team has removed malware from the Android Market following user complaints, and they are able to remotely uninstall known malware from users' devices [17].

Nokia runs Ovi [5], which is currently the official Symbian application market. Like the Apple App Store, all applications are reviewed prior to being listed in Ovi. Symbian does not prevent or discourage users from installing applications from other sources. Several popular alternative markets are available, and they lack review processes. However, Symbian offers an application signing service that incorporates security reviewing. Only Symbian Signed applications are allowed to access dangerous privileges. All Symbian Signed applications must undergo an automated security review. Applications that use the most dangerous privileges are additionally reviewed by humans, and some number of other Symbian Signed applications are also human-reviewed. As a consequence of the signing process, many applications in third-party Symbian markets have undergone review.

Permissions. Smartphone operating systems may also protect users by requiring user consent before an application can access sensitive information or dangerous capabilities. User-approved permissions can alert users to the activities of grayware or malware, although malware may seek to circumvent permission systems. Permissions do not prevent the installation of personal spyware because the attacker can grant all necessary permissions during installation.

Android informs users of an application's desired permissions during installation. The Android permission system is extensive; user-approved install-time permissions control access to the phone's number, list of contacts, camera, Bluetooth, etc. The iOS permission system is much less comprehensive than the Android permission system, likely because Apple primarily relies on the App Store review process for security. iOS requires user approval only to access location and send notifications, and permission is requested at runtime rather than during installation. Symbian prompts users to grant permissions during the installation of applications that have not been Symbian Signed. Symbian's user-granted permissions permit access to only a few privileges, like connecting to the network or a Bluetooth device; the remainder are off-limits. Symbian Signed applications automatically receive all requested privileges without user approval.

3. MOBILE MALWARE DATA SET

This paper surveys known iOS, Symbian, and Android malware, using information collected from public anti-virus databases. We identified 46 pieces of mobile malware from January 2009 to June 2011. In this section, we describe our survey methodology and the data set's characteristics.

3.1 Collection Methodology

To find information about known mobile malware, we combed the public databases of anti-virus companies (e.g., Symantec, F-Secure, Fortiguard, Lookout, and Panda Security) and news releases to find mentions of malware. The existence of each piece of malware is confirmed by at least two anti-virus vendors, and we compared malware reports to identify cases where researchers had used different names for the same piece of malware. We also coalesced variants of the same family of malware into one listing for the family.

Our goal was to collect statistics about malware in the wild so that we could (1) understand their underlying motivations and (2) evaluate how well current malware defense mechanisms protect users. To this end, we omitted personal spyware and grayware from consideration. We also omitted researchers' proofs of concept because they do not represent the current state of real malware.

3.2 Data Set

We identified 46 pieces of iOS, Symbian 9.x, and Android malware that were detected in the wild between January 2009 and June 2011. This data is available at `http://www.cs.berkeley.edu/~daw/malware.html` (summary in Appendix A). In this data set, 4 target iOS, 24 target Symbian 9.x, and 18 target Android. Notably, almost all of the Android malware is from 2011, and all of the iOS malware appeared within a single month in 2009. Figure 1 shows a timeline of when each piece of malware became known.

4. INCENTIVES

We discuss current and future incentives for writing mobile malware. To understand current incentives, we present the motivations that we believe are behind the malware in our data set. Our predictions for future mobile malware are based on trends in desktop malware [15], combined with characteristics of mobile platforms.

4.1 Current Incentives

We categorize known iOS, Symbian, and Android malware by the motivations indicated by their behavior. Table 1 shows an overview of the behavioral classification. These counts represent a lower bound because many pieces of malware support the ability to load additional code from the Internet, and it is possible that anti-virus reports were incomplete. Also, not all malware performs all of its supported operations on every infection; 15 of the 46 pieces of malware change their behavior based on commands received from command-and-control servers.

Exfiltrates user information	28
Premium calls or SMS	24
Sends SMS advertisement spam	8
Novelty and amusement	6
Exfiltrates user credentials	4
Search engine optimization	1
Ransom	1

Table 1: We classify 46 pieces of malware by behavior. Some samples exhibit more than one behavior, and every piece of malware exhibits at least one.

4.1.1 *Novelty and Amusement*

Some malware causes mischief or damage in a way that appears to be intended to amuse the author. For example, Ikee.A [48] changed the wallpaper of infected iPhone devices, and Smspacem [58] sent anti-religion text messages from Android phones. 6 of the 46 pieces of malware fall into this category and no other. Early desktop malware was similarly motivated by humor, bragging rights, or purposeless destruction. As mobile platforms mature, we expect that amusement-driven mobile malware will decrease in number and become overshadowed by profit-driven malware.

4.1.2 *Selling User Information*

Mobile operating system APIs provide applications with large amounts of information about users. Applications can query mobile APIs for the user's location, list of contacts, browser and download history, list of installed applications, and IMEI (the unique device identifier). 28 pieces of recent mobile malware have collected and exfiltrated one or more of these items. Although we cannot know for sure why malware collects this information, we hypothesize that this data is being sold by malware distributors for financial gain.

Advertising or marketing companies might be willing to purchase users' locations, browsing histories, and lists of installed applications to improve behavioral profiling and product targeting. However, advertising libraries in legitimate applications already routinely collect user location, and web-based advertisements already track browsing habits. Legitimate applications with advertising libraries can expect to earn between \$1.90 and \$9.50 per user per month, which includes both the value of collecting user location data and displaying advertisements [44]. This provides an upper bound on one measure of the value of user profiling data.

Consumer IMEIs have value to black market phone vendors. When a phone is reported stolen, its IMEI is blacklisted, which prevents it from connecting to cellular networks. This is supposed to render stolen phones useless. In practice, thieves can alter phone IMEIs to replace blacklisted IMEIs with valid IMEIs [35]. This motivates a market for valid consumer IMEIs. However, many legitimate applications and advertising libraries already collect IMEIs [25]. This large supply of IMEIs likely means that the IMEI-driven revenue per user infection is quite low.

Malware distributors also might sell users' contact lists. A user's contact list includes contacts' names, phone numbers, and e-mail addresses. This contact information could be sold to scammers, spammers, or phishers. As of 2008, e-mail addresses were worth between \$0.33/MB and \$40/MB on the black market [41].

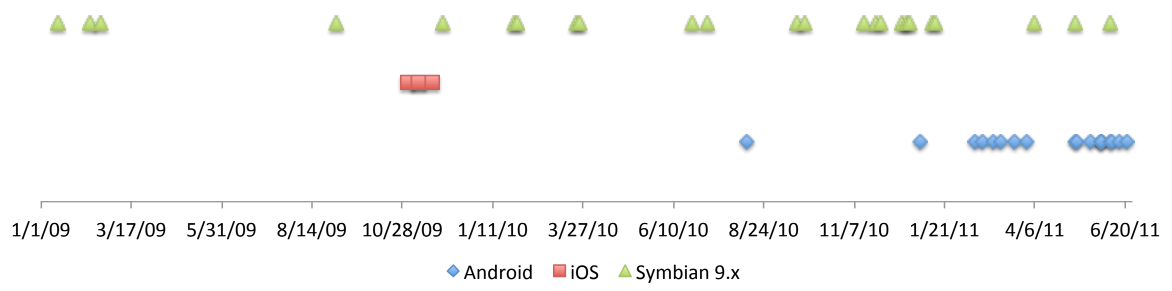

Figure 1: A timeline of when the 46 pieces of malware in our data set were detected by malware researchers.

In addition to the platform API, mobile phones also store application-specific data. Malware could search through available documents to glean more facts about users. This is difficult on iOS, where application data is tightly segregated. However, Android and Symbian applications can read some of each others' data if given the correct permissions. We have not yet observed inter-application data theft in the wild, likely because it is difficult to find information of interest in other applications' data stores.

Defense. To address IMEI theft, mobile operating systems could provide applications with an alternate, globally-unique device ID. As far as we are aware, the only legitimate reason for an application to request an IMEI is to uniquely identify a user in a database, which can be accomplished with a substitute ID. Unfortunately, the same tactic cannot be applied to other types of data; non-malicious applications often require direct access to location, contact lists, etc. However, inter-application data theft in Android and Symbian could be prevented by further restricting the rights of Android and Symbian applications to read other applications' data.

4.1.3 Stealing User Credentials

People use smartphones for shopping, banking, e-mail, and other activities that require passwords and payment information. Banks rely on cell phones for two-factor authentication. Users may also save authentication and payment credentials in text documents on their phones (for example, to use the phone as a mobile password manager). This makes cell phones a target for credential theft. As of 2008, bank account credentials, credit card numbers, and e-mail account passwords were worth $10 to $1,000, $.10 to $25, and $4 to $30, respectively, on the black market [41]. Credentials could be used directly by malware authors for greater financial gain, but financial fraud can be difficult to perpetrate and requires specialization [41].

Three pieces of malware in our data set target user credentials by intercepting SMS messages to capture bank account credentials. Notably, Spitmo [59] and ZeusMitmo [11] work in conjunction with desktop malware to mount sophisticated attacks against two-factor authentication. The desktop component phishes the user and tricks the user into also installing the smartphone malware. This indicates that criminals are beginning to use mobile malware to circumvent the added security of two-factor authentication. A fourth piece of malware launches a phishing attack on the phone. Phishing attacks can be more convincing on phones than in a desktop browser [46, 13, 29], and we expect to see more phishing attacks from mobile malware in the future. User credentials could also be captured by keylogging or scanning documents, although we have not yet seen evidence of these attack vectors in the wild.

Defense. Phishing is a difficult, unsolved problem, and two-factor authentication is proving insufficient to defend against the most sophisticated attacks. However, credential theft via document searching in Android or Symbian could be prevented by strengthening application isolation mechanisms. We believe that mobile credential theft will increase in the future and is a promising area of security research.

4.1.4 Premium-Rate Calls And SMS

Legitimate premium-rate phone calls and SMS messages deliver valuable content, such as stock quotes, technical support, or adult services. The cost of a premium-rate call or SMS is charged to the sender's phone bill. Premium-rate calls can cost several dollars per minute, and premium-rate SMS messages can cost several dollars per message. Premium-rate calls were abused by desktop malware for financial gain in the 1990s and early 2000s, when computers were connected to dial-up modems. Premium-rate SMS messages are stealthier than premium-rate calls because calls tie up the phone line. In Android and Symbian, malware can completely hide premium-rate SMS messages from the user. Premium-rate SMS attacks could feasibly go unnoticed until the user's next phone bill.

24 of the 46 pieces of recent mobile malware send premium-rate SMS messages. For example, an application purporting to be a Russian adult video player sent premium-rate SMS to an adult service [39]. Another piece of malware, Geinimi, was set up to send premium SMS messages to numbers specified by remote commands [57]. 2 of the 46 malicious applications place premium-rate phone calls. Each of these pieces of malware targets either Android or Symbian devices.

Defense. This popular and lucrative incentive for mobile malware could be completely removed by requiring user confirmation for premium-rate SMS messages. iOS already requires user confirmation for all SMS messages, but Android and Symbian do not; this difference could be partially responsible for the comparative lack of iOS malware. Phones can identify outgoing premium-rate SMS messages using the prefix or length of the number. We propose that the OS prompt the user to approve each premium-rate message.

We expect the impact on Android and Symbian applications would be low, as this is already a requirement in iOS. To evaluate the impact of requiring user approval for sending premium-rate SMS messages, we consider 956 non-malicious Android 2.2 applications.[2] 39 of the 956 applications request the ability to send SMS messages. We determined through manual review of the applications that 6 are de-

[2]Our data set consists of the 756 most popular free applications, 100 most popular paid applications, and 100 most recently uploaded applications from the Android Market.

signed to send premium-rate SMS messages, and only if the user manually enters a premium-rate number into the application. Only 0.6% of applications would ever generate premium-rate prompts, and each would do so infrequently. None of the applications would break if the user were required to approve premium-rate SMS messages.

4.1.5 SMS Spam

SMS spam is used for commercial advertising and spreading phishing links. Commercial spammers are incentivized to use malware to send SMS spam because sending SMS spam is illegal in most countries. Sending spam from a compromised machine reduces the risk to the spammer because it obscures the provenance of the spam. As we observed in Section 4.1.4, it is possible to stealthily send SMS spam on Android and Symbian devices. Users might not notice the outgoing SMS messages until their monthly phone bills arrive. Even then, users with unlimited SMS messaging plans may never notice the extra SMS messages. Furthermore, the use of SMS may lend more authenticity to spam than e-mail because phone contacts are often more intimately acquainted than e-mail contacts. 8 of the malicious Symbian and Android applications send SMS spam.

Defense. In iOS, an application that wants to send SMS messages has two options: (1) provide the built-in SMS messaging application with the appropriate parameters, or (2) embed an OS-provided UI element that displays the contents of the SMS message. Either way, the operating system ensures that the user has seen and approved the SMS message before it is sent. All SMS spam in Android and Symbian could be prevented by similarly not allowing applications to directly send SMS messages. Instead, applications should leverage the built-in SMS messaging application when possible. Applications that truly need to send SMS messages without an OS intermediary could be accommodated with a special platform setting that gives the application this right, after displaying a warning to the user (similar to how alternate input methods are approved in Android). The prompts for premium-rate SMS messages proposed in Section 4.1.4 should still be shown when the aforementioned special setting is toggled, although they would not be necessary for messages sent through the built-in SMS messenger.

We consider how this change would affect Android applications that request the ability to directly send SMS messages. 39 of the 956 applications in our set of non-malicious applications request this permission. Our manual review found that 23 of those applications could easily be rewritten to use the existing built-in Android SMS messaging application, without loss of functionality. Furthermore, 8 more applications already use the built-in SMS messenger despite requesting the `SEND_SMS` permission; the permission could be removed from the applications without any negative effect. The remaining 8 applications legitimately need to send SMS messages without going through an intermediary. 7 of these 8 applications are meant to be alternatives to the built-in SMS messenger, and 1 silently sends SMS messages with location coordinates when a phone is missing. It would be reasonable for a user to be expected to toggle a special setting to enable this behavior. Since only 0.8% of the 956 applications would ask the user to toggle the setting, users would not be bombarded by the requirement.

4.1.6 Search Engine Optimization

Many web sites rely on search engines for traffic, which makes web site owners desire high visibility in search engine results. Search engines rank web sites according to how relevant each web site is to a given search term. An engine's perception of relevance is influenced by the rate at which users click on the web sites returned for a search term. A web site will rise in the results for a search term if many people search for that term and then click on that web site.

Malware can be employed to improve a web site's ranking in search engine results. This type of malware sends web requests to the search engine for the target search term. The malware then fraudulently "clicks" on the search result that corresponds to the target web site. As a result, the web site's rank for that search term will increase. The value of fraudulent search engine optimization depends on how well the target site can capitalize on its increased visibility, but search engine optimization is a large and lucrative market. One recent Android Trojan, ADRD/HongTouTou, performs search engine optimization. ADRD was built to boost the Baidu search result ranking of a Chinese web site [54]. Desktop malware has also been known to fraudulently perform search engine optimization.

Defense. A potential solution is for mobile operating systems to add metadata to each web request that identifies the application that initiated the request. This would help web sites blacklist requests from applications that are known to perform click fraud. The identifying information could be added, for example, as a header. It would be similar to a user-agent string that applications could not change. However, this might have privacy implications for users.

4.1.7 Ransom

Malware can be a tool for blackmail. For example, the desktop Trojan Kenzero stole the user's browser history, published it publicly on the Internet alongside the person's name, and then demanded 1500 yen to take down the person's browser history [38]. There has not yet been any mobile malware that seriously threatens or publicly embarrasses the user for profit, but one piece of mobile malware has sought a ransom. A Dutch worm locked iPhone screens and demanded 5 euros to unlock the screens of infected phones [18]. At least two other pieces of mobile malware read the user's browser history and bookmarks, but they have not attempted to use that information for blackmail. Mobile phones do not offer any technical advantages over desktop computers to criminals seeking to ransom users. However, users' behavior might differ between mobile phones and desktops in a way that makes one platform a more valuable ransom target than the other.

4.2 Future Incentives

In this section, we discuss incentives that we predict will motivate future mobile malware. Mobile malware has not yet exhibited evidence of these motivations in the wild.

4.2.1 Advertising Click Fraud

Advertisers pay advertising networks when users view or click on advertisements. In turn, advertising networks pay the web sites that host advertisements. Networks may also be chained in a series, with each network relaying the advertisement and paying the next one in the series. Unscrupulous web sites and advertising networks defraud advertisers and non-malicious networks by using desktop malware to load and click on advertisements [43, 22]. If undetected, click fraud generates a few cents (or even dollars) per instance of fraud. The attacker will directly benefit from the fraud by receiving some portion of the fraudulent payment. An attacker might also launch a click fraud attack on advertising competitors. This depletes the competitors' advertisement budgets, resulting in more legitimate traffic to the attacker's ads. Furthermore, competitors may lower their advertising bids after seeing a lower return on investment, causing the cost of advertisements to go down [21].

Advertising click fraud is very similar to search engine optimization fraud (Section 4.1.6). Although we are not aware of any mobile malware in the wild that performs advertising click fraud, we expect to see it soon.

Defense. Like search engine optimization fraud, advertising click fraud could be mitigated by attaching information to HTTP(S) requests that identifies the requesting application.

4.2.2 Invasive Advertising

Many legitimate applications use advertisements to earn money while providing the application to users for free. However, malicious applications can take advertising a step further with invasive advertising practices. Rather than placing advertisements alongside legitimate application content, malicious adware will display advertisements when the user is interacting with other applications. This could significantly interfere with a user's experience with other applications.

There are two main reasons for an attacker to display advertisements with malware. First, the attacker may want to advertise goods or services that are illegal or of a nature that legitimate advertising companies prohibit (e.g., pornography, gambling, endangered species products [1, 3]). An attacker might do this to advertise his own products or to create a black market advertising network for affiliates' products. Second, the attacker may simply want to collect revenue from displaying legitimate advertisements. The attacker may be able to generate more revenue with invasive advertising practices by displaying advertisements to users more often or in such a way that users accidentally click on them. This is not considered click fraud because it capitalizes on users' legitimate (albeit accidental) clicks instead of automated clicks. However, these invasive advertising practices are against legitimate networks' terms of service.

We were unable to determine whether any pieces of malware in our data set display invasive advertisements. Nearly all of the malware in our data set makes HTTP requests, but anti-virus reports did not include enough information to determine the nature of those web requests. However, adware exists for desktop machines, and we expect to see it in mobile applications as well.

4.2.3 In-Application Billing Fraud

Android and iOS support *in-application billing*, which allows a user to purchase a virtual item from an application using the payment account associated with the Android Market or Apple App Store. Users can therefore buy items such as game credits and music from applications without directly providing the application with payment information.

We predict that in-application billing may become a target of fraud in the future. First, the implementations of in-application billing protocols could include bugs that allow malware to charge users for items without their approval. Second, malicious applications could use social engineering, clickjacking, or phishing attacks to trick users into accidentally or unknowingly approving in-application purchases. For example, iOS sometimes prompts users to enter their App Store passwords into windows that hover over applications, as part of the in-application billing process; these windows are a potential target for phishing attacks.

4.2.4 Governments

Governments may use mobile phones to monitor citizens and their activities. Unlike the majority of other incentives discussed in this paper, government spying is not motivated directly by financial gain. This type of monitoring could be performed on a large scale (e.g., China's Internet monitoring) or targeted at known dissidents or suspected criminals. It could incorporate GPS tracking, audio and video recording, monitoring of e-mail and SMS messages, and extracting lists of contacts. For example, in 2009 an ISP in the United Arab Emirates pushed an "update" to $145,000$ Blackberry users that drained phone batteries and forwarded copies of e-mails to a government server [60]. Similarly, China partnered with eBay to produce and distribute a customized version of Skype that censors and tracks users; standard Skype is not available for download in China [42].

The government threat model is significantly more powerful than the criminal threat model. Governments have the ability to gain the cooperation of network carriers and device manufacturers to manipulate firmware updates, control official markets, and distribute rootkits on all devices. Permissions, signing, and anti-virus software can be circumvented by a government that can corrupt the integrity of the operating system. Markets and review processes cannot be trusted to filter out government-sponsored spyware because governments can compel the responsible agencies to publish it. Government agents also can gain physical access to targets' phones to install monitoring software.

Unlike criminals, there is little motivation for a government to use arbitrary third-party applications to spread malware, unless the government were unable to gain the cooperation of the necessary corporate parties to launch a stronger attack. Governments are more likely to subvert smartphone operating systems or modify specific popular applications (with the original application no longer available).

4.2.5 E-Mail Spam

Desktop malware uses compromised hosts to send e-mail spam for advertising and phishing. There are three ways for malware to send spam from infected hosts:

1. Malware can send spam from the user's e-mail account, for example by abusing a logged-in browser session. However, it is hard for mobile malware to manipulate a user's e-mail account; mobile applications are isolated from each other, and most mobile browsers do not currently support plug-ins or extensions.

2. Malware can make SMTP connections to spam recipients' Mail eXchange (MX) servers if the network permits outgoing traffic on port 25 [8]. Some ISPs, including all cellular networks, block port 25.

3. Malware can launder spam through open proxies [8]. In many cases, this only requires an outgoing HTTP request. Mobile malware can do this, although mobile phones typically have less bandwidth than desktops.

Desktop machines are more attractive as spam clients, so we do not expect to see e-mail spam as a major motivating factor for mobile malware. However, it may occur if a mobile e-mail or social networking client is found to have a vulnerability that enables abuse of the user's account.

4.2.6 Distributed Denial of Service

To perpetrate distributed Denial of Service (DDoS) attacks, botnet owners command large groups of compromised machines to simultaneously send requests to servers. DDoS attacks can be launched for ransom, amusement, cyberwarfare, or as a paid service to others. Traditional DDoS attacks are difficult to stop because of their distributed nature, but one approach is for the server to block the IP addresses of visitors that behave anomalously. Consequently, each attacking machine is limited to a small number of fraudulent requests. This would not be an effective defense mechanism against mobile-based DDoS attacks because cellular networks assign new IP addresses as often as every few minutes [10]. If that rate of IP assignment is not fast enough, mobile malware can force the assignment of a new IP address from the cellular network by resetting the data connection [10]. In comparison, many desktop machines have static or infrequently-changing public IP addresses that cannot be forcibly reassigned.

Despite this advantage, mobile phones also present challenges for DDoS attackers. Mobile phones on cellular networks have significantly less bandwidth than non-mobile devices. Furthermore, an attacker would need to avoid draining the phone's battery too much, limiting a mobile device to a few HTTP requests every few minutes. We expect to eventually see some DDoS malware for mobile phones, but not until phones' bandwidth and batteries improve.

4.2.7 NFC and Credit Cards

Mobile phones are beginning to incorporate Near Field Communication (NFC), which allows short, paired transactions with other NFC-enabled devices in close proximity. NFC can be used for commerce (i.e., accepting credit card transactions), social networking (e.g., sharing contact information), device configuration (e.g., automatically configuring WiFi), and more. It is predicted that mobile payments using NFC will reach $670 billion by 2015 [16].

Previous research has studied NFC and its potential impact on device security [34, 45]. Already, bugs have been found in an Android implementation of NFC [50], resulting in denial of service and battery degradation. We predict that NFC will become a popular target for malware due to the ease with which financial transactions can occur using NFC. Malware that is capable of using NFC has the potential ability to surreptitiously read and interact with NFC-enabled devices placed close to the phone (e.g., in a pocket), such as credit cards or personal identification cards.

5. MALWARE DETECTION

Apple, Google, and Nokia use application permissions and review (as part of market curation and signing) to protect users from malware. We evaluate the effectiveness of these mechanisms against the malware in our data set.

Number of Dangerous permissions	Number of non-malicious applications		Number of malicious applications
0	75	(8%)	-
1	154	(16%)	1
2	182	(19%)	1
3	152	(16%)	-
4	140	(15%)	2
5	82	(9%)	1
6	65	(7%)	-
7	28	(3%)	2
8	19	(2%)	1
9	21	(2%)	1
10	10	(1%)	1
11	6	(0.6%)	1
12	7	(0.7%)	-
13	4	(0.4%)	-
14	4	(0.4%)	-
15	2	(0.2%)	-
16	1	(0.1%)	-
17	1	(0.1%)	-
18	-		-
19	-		-
20	1	(0.1%)	-
21	-		-
22	-		-
23	1	(0.1%)	-
24	-		-
25	-		-
26	1	(0.1%)	-

Table 2: The number of "Dangerous" Android permissions requested by 11 pieces of malware and 956 non-malicious applications [28].

5.1 Permissions

Permissions theoretically could help users or reviewers identify malware, if the malicious applications' permission requests differ from normal, non-malicious applications' permission requests. Our goal is to determine whether permissions are a useful tool for identifying malware in practice.

This section evaluates whether the Android malware in our data set exhibits anomalous patterns of permission requests. Could a simple classifier use the permission requests of malware to differentiate malicious from non-malicious applications? We chose to study Android malware because Android has the most extensive permission system and therefore represents the best platform for permission-based classification. To perform the study, we collected the permission requests made by each piece of Android malware. This data was not always available from anti-virus researchers, but we were able to identify the permission requirements for 11 of the 18 Android malware applications.

Number of Permissions. Table 2 displays the number of permissions requested by the 11 pieces of malware in our data set, alongside the permission requests of non-malicious applications. On average, the Android malware in our data set is more privileged than non-malicious applications: malicious applications request an average of 6.18 Dangerous[3] permissions, and non-malicious applications request an av-

[3]Android has four categories of permissions; users see permissions from the "Dangerous" category during installation.

erage of 3.46 Dangerous permissions. However, as Table 2 shows, the distribution of permissions for malware falls entirely within the distribution for non-malicious applications. 2% of non-malicious applications request more permissions than any piece of malware in our set [28], and two pieces of malware ask for fewer permissions than the average for non-malicious applications. This shows that the number of permissions alone is not sufficient to identify malware, but it could be used as part of a set of classification features.

Common Permissions. We consider whether any of the permissions that are common within the set of malware are infrequent among non-malicious applications. If so, those permissions could be used as features to classify malware. The most promising characteristic of the malware is the use of SMS permissions. 8 of the 11 malicious applications (73%) request the ability to send SMS messages without user confirmation, while 4% of non-malicious applications ask for that permission [28]. This indicates that the SMS sending permission might be an effective classification feature. We also found that the `READ_PHONE_STATE` permission (which controls access to the IMEI) may be a promising feature for classifying malware, if used in combination with other features. In particular, 8 of 11 (73%) of malicious applications in our sample ask for this permission, while only 33% of legitimate applications request this permission [28].

None of the other common malware permissions appears promising for identifying malware because of a high false positive rate. Besides the SMS sending permission and `READ_PHONE_STATE`, all of the permissions that are common within the malware are also requested frequently by non-malicious applications. 9 malicious applications ask for Internet access, 4 ask for the ability to read data on the SD card, and 4 want to read the user's location. Despite their association with abuse by malware, these permissions are also very popular with non-malicious applications: 87% of legitimate applications use the Internet and about one-third of applications ask for each of the other permissions [28].

Sets of Permissions. Enck et al. propose to identify malware with sets of permissions [26]. Their security rules classify applications based on *sets* of permissions rather than individual permissions to reduce the number of false positives. For example, nearly one-third of applications request access to user location, but far fewer request access to user location *and* the ability to launch as soon as the phone boots.

Their set-based security rules identify 4 of the 11 pieces of malware. In our data set, 2 malicious applications violate their rules pertaining to SMS messaging, 1 malicious application violates their location rules, and 1 malicious application violates their rule about recording audio. These rules would have caught 36% of this set of malware, and they report a 3.8% false positive rate [26]. They achieve a low false positive rate at the expense of a low recall rate. This has approximately the same false positive rate as using the single SMS sending permission for classification, but with half the recall. (The SMS sending permission captures a superset of the applications captured by their rules.) However, the recall of the SMS rule would dramatically decrease if the rule were put into place or smartphones began to require user confirmation for premium-rate SMS messages. As a consequence, more sophisticated rules and classification features, such as future work on permission sets, would be required.

5.2 Application Review

Apple reviews all applications that are in the App Store. Symbian automatically tests all Symbian Signed applications; some are also human-reviewed. We consider whether these processes prevented the malware in our data set.

iOS. Although Apple has accidentally listed (and then removed) grayware in the Apple App Store, we are unaware of any malware that has been listed in the Apple App Store. All 4 pieces of Apple malware spread through the same SSH backdoor in jailbroken iOS devices, and none were listed in the App Store. This may indicate that Apple's review process is successful at identifying or discouraging malware, although an alternate explanation is that malware authors are not targeting iOS because of the SMS restrictions.

Symbian. At least 5 of the 24 pieces of Symbian malware in our data set were Symbian Signed. This implies that they passed through at least the automated review, although we were unable to determine which were human-reviewed. Additionally, 2 malicious applications (ZeusMitmo [11] and Spitmo [59]) evaded the signing process by working with desktop malware to phish for users' IMEIs prior to infection. After tricking users into revealing their IMEIs, the malware authors added the IMEIs as "test phones" to their developer certificates. Each malware author obtained multiple developer certificates to circumvent the restriction of 1000 phones per certificate. As a result, the malware would have appeared Symbian Signed to the victims. At least 2 pieces of Symbian malware lacked certificates and, consequently, were limited to the permissions that users were able and willing to grant during installation. We were unable to determine how many of the 15 remaining pieces of malware were Symbian Signed. However, at least 29% of the Symbian malware passed through or evaded the Symbian signing process, which indicates that it is not an effective deterrent.

6. ROOT EXPLOITS

Root exploits (also known as "jailbreaks") are used by both malware authors and smartphone owners. Malware authors want to circumvent security mechanisms, and smartphone owners want to customize their phones. As a result, the widespread public availability of root exploits found by smartphone owners inadvertently enables malware authors to write sophisticated malware.

In this section, we discuss the ecosystem surrounding root exploits of mobile handsets. First, we describe the incentive structure that encourages people to find and publicize root exploits. Next, we present data regarding the availability of root exploits over time for 6 Android handsets. Finally, we discuss a strategy for fixing the incentive structure so that it no longer promotes the availability of root exploits.

6.1 Incentives

Root exploits for mobile phones are coveted by two different groups of people: malware authors and smartphone users who want to modify their phones. Malware authors can use root exploits to gain extra privileges and perform any operation on the phone. For example, an attacker can use a root exploit on an Android phone to gain access to parts of the API that are supposed to be protected by the permission system. On the other hand, smartphone owners covet root exploits to create and install *homebrew* (i.e., customized) versions of operating systems. Consumers desire

Phone	Phone Release Date	Days *without* known root exploit	Days *with* known root exploit	Percent of time with known root exploit
EVO 4G	June 4, 2010	83	304	79%
Epic 4G	August 31, 2010	9	290	97%
Atrix 4G	February 22, 2011	3	121	98%
Thunderbolt	March 17, 2011	18	83	82%
T-Mobile G2X	April 15, 2011	0	72	100%
Droid Charge	May 14, 2011	11	32	74%

Table 3: We report the number and percentage of days between a handset's release date and June 26, 2011 in which there was a publicly available root exploit published by the Android homebrew community.

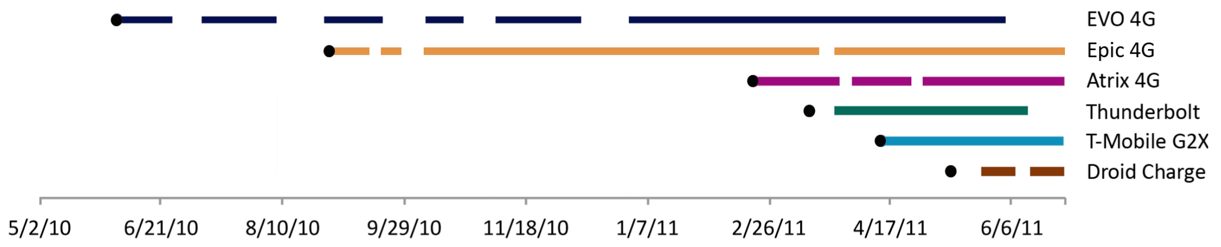

Figure 2: A timeline displaying the dates that known root exploits were available for 6 popular Android phones. Circles mark the release dates of the phones.

homebrew handsets because the majority of smartphones ship with at least one of the following limitations:

- Users can install only those applications that are distributed through the official application store (e.g., iOS only permits applications from the App Store).
- Users cannot perform complete system backups.
- Carriers forbid or restrict tethering so that users are required to pay an additional fee to share the phone's Internet connection with a computer.
- Carriers pre-install certain applications on phones and then disable their removal, such as Sprint's NASCAR Android application.
- Users cannot install custom versions of the operating system that contain additional features or improvements. For example, one custom version of Android provides OpenVPN support [20].

A root exploit allows a user to evade these restrictions because it provides the user with complete control of the phone's software stack.

Currently, the incentives of malware authors and smartphone owners are aligned with respect to root exploits. Malware authors benefit from the community of non-malicious individuals who are interested in finding root exploits for mobile phones. The homebrew community typically finds root exploits as quickly as possible when new phones or software upgrades are released, in order to maintain full access to the software stack. Malware authors do not need to develop their own root exploits because they can wait for smartphone tinkerers to do so for them.

Malware from our data set is already taking advantage of root exploits that are found by smartphone users. Four pieces of Android malware (DroidDream, Zhash, Droid Kung Fu, and Basebridge) used root exploits that were published by the homebrew community to attack phones [33, 56, 37, 23]. At least two of them included multiple root exploits to ensure success on different devices.

6.2 Root Exploit Availability

We performed a measurement experiment to determine the availability of root exploits. If the homebrew community has published a root exploit for a given device and operating system version, then that device is at risk of attack with a root exploit. To quantify the scope of this threat, we chose to study the availability of root exploits for 6 of PCWorld's "Top 10 Android Phones" [7]. We chose the top 5 phones in the list plus the phone with the earliest release date, the EVO 4G. We focus on Android because existing Android malware uses root exploits, but we are not aware of Symbian or iOS malware that does so.

Methodology. For each phone, we gathered data on (1) the release date and firmware updates for the phone, and (2) the dates when the homebrew community reported that a root exploit was available for the current firmware version of the phone. By comparing these two dates, we can see how long a root exploit was available for a phone relative to the lifespan of that phone and firmware version.

We collected data on firmware updates by looking for news stories that report the release dates of new phones and OTA (over-the-air) firmware updates. We gathered the release dates of root exploits by looking through the forums on the `xda-developers` web site [6], a popular web site dedicated to Android and Windows mobile developers. Root exploits are often published on `xda-developers` as a set of detailed instructions on how to root a firmware version of a device. This represents a lower bound of the time that a root exploit is available because we may not have found the earliest instructions to root the device.

Results. The summary data from this experiment is provided in Table 3. The availability of root exploits over time is also depicted graphically in Figure 2. Every device had a root exploit publicly available for at least 74% of the device's lifetime. The latest firmware versions lacked a published root exploit for only the first 5.2 days, on average. Additionally, root exploits were available prior to or on the official release dates of 9 of the 24 total versions.

6.3 Discussion

As our study shows, the homebrew community is quick to develop root exploits for new handsets and software versions. Although our study focuses on Android, other platforms also face this problem; for example, the latest iOS version was released on May 4, 2011, and jailbreaking tutorials were available by May 6, 2011 [9]. Currently, legitimate users are incentivized to develop root exploits in an effort to gain control of their devices and to maintain the increased functionality that they obtain with root access. We therefore believe that the current "locked" model used by phone vendors is detrimental to the security of end users because it aligns the incentives of attackers and smartphone users. For Android, the percentage of days that root exploits are available is so high that it might be more effective for malware authors to use these root exploits in lieu of tricking users into accepting dangerous permissions.

An alternative model that does not incentivize homebrew root exploits exists. Some device manufacturers have begun to ship phones with unlocked bootloaders [14]. This allows users to customize their phones without relying on an exploit, but it requires physical access to the phone so that remote attackers cannot abuse it. Under this model, attackers must find exploits without the help of the homebrew community. This prevents the efforts of the homebrew community from affecting the safety of all phone users.

Disincentivizing root exploits does not solve all problems associated with device customization. Users with jailbroken phones face the risk of vulnerabilities introduced by customized operating systems. All 4 pieces of iOS malware targeted a backdoor in incorrectly configured jailbroken phones [47], and one piece of Android malware took advantage of a security error introduced by certain custom Android versions [55]. However, unlike the threat posed by root exploits, this type of malware is limited to phones that have been voluntarily customized by their owners. According to estimates, 15% to 20% of Android phones are rooted [51], and 6% of iPhones are jailbroken [65].

7. RELATED WORK

Mobile Malware. In 2004, Guo et al. predicted that mobile malware would be used for attacks against telecom networks, call centers, spam, identity theft, and wiretapping [32]. Others have surveyed and discussed mobile malware seen in 2005 through 2008 [49, 53, 61, 52, 40]. We present an updated discussion of the feasibility of mobile-based attacks, given modern smartphone capabilities. We also evaluate the incentives that underlie the different types of attacks. In 2009, Enck et al. presented potential incentives for mobile malware [26]. We follow their work with a more in-depth consideration of mobile malware incentives and a survey that validates their predictions that premium SMS and information-gathering malware would become prevalent. In more recent work, Becher et al. and Vidas et al. discuss potential mobile attack vectors, such as the web browser and physical access [12, 64]. They focus on describing attack mechanisms, whereas we survey malware.

Underground Economies. In 2007, Franklin and Paxson collected IRC logs relating to the Internet black market [30]. They investigated the types of data traded and found that the price of a compromised desktop machine varies between $2 and $25, depending on the time of year. A 2008 Symantec white paper also explored the prices and availability of stolen data on the black market [41].

DDoS on Cellular Networks. In addition to the Internet attacks we discussed in Section 4.2.6, DDoS attacks can be launched against the cellular network. Phones in close physical proximity to each other could simultaneously place calls to flood call centers and GSM base stations [32]. The primary difficulty with this attack is that users are likely to notice spurious phone calls on their phones, so we do not expect this to become a popular attack vector. Traynor et al. also showed that $11,750$ mobile phones in the same area-code sized region could degrade service to other phones in the region by 93% by sending data messages to cellular network servers [62]. This type of attack would require a high infection rate in a localized area but could be accomplished with a rapid-spreading worm. Finally, malformed SMS messages can crash phones, causing a DDoS when the phones drop off the network and repeatedly try to reconnect at the same time [19]. This attack simply requires a hit list of mobile phone numbers and a bulk SMS operator, and it does not require installation of malware onto the phone.

Other Platforms. We do not discuss Windows Mobile, Windows Phone 7, Palm, or BlackBerry malware. Although defenses differ between platforms, the incentives for mobile malware are not platform-dependent. Similar malware exists for BlackBerry and Windows phones (for example, a BlackBerry variant of ZeusMitmo defeats two-factor authentication [63], and a Windows Mobile Trojan makes premium international calls [27]). We are not aware of any BlackBerry, Palm, or Windows phone malware that currently implements the future attacks discussed in Section 4.2.

8. CONCLUSION

Mobile malware is evolving into a complex ecosystem that will likely eventually rival the desktop malware landscape. In this paper, we survey the behavior of current mobile malware payloads. At present, mobile malware is motivated primarily by a desire to send premium-rate SMS messages and sell information. The former motivation can be defeated by requiring user confirmation for premium-rate SMS messages (as iOS does), but more research is required on the topic of defending against malware that steals user data and credentials. We also explore potential future directions of malware; in particular, we think that credential theft, credit card theft via NFC, and advertising click fraud are the most likely to be targeted by future malware authors.

As part of our survey, we examined the permissions of Android malware. Android malware commonly requests the ability to directly send SMS messages, which is uncommon among non-malicious applications. However, we were unable to identify any other permission-based patterns for malware classification. Permission-based classification will require future consideration as the set of known Android malware grows. We also observed that none of the malware in our data set was approved by the Apple App Store, which indicates that human review may be an effective preventative measure for malware. Symbian's automated review-and-sign process fared worse; nearly a third of the Symbian malware in our data set was approved by or evaded this process.

Currently, both malware authors and smartphone users are incentivized to find root exploits. The homebrew community publishes root exploits to help smartphone owners customize their phones. However, malware can use these same root exploits to circumvent smartphone security mechanisms; indeed, 4 pieces of malware in our data set do this. We consider the impact of the homebrew community and find that root exploits are available between 74% and 100% of phones' lifetimes. We recommend that phone manufacturers support smartphone customization so that the homebrew community does not need to seek root exploits.

Acknowledgements

We would like to thank Chris Grier for his insightful comments about the motivations for current desktop malware. This work is partially supported by a gift from Google and NSF grants CNS-1018924 and CCF-0424422. This material is also based upon work supported under NSF Graduate Research Fellowships. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the views of the NSF.

9. REFERENCES

[1] Adwords content guidelines. `http://adwords.google.com/support/aw/bin/static.py?hl=en&guide=28435&page=guide.cs`.

[2] Android Market. `http://www.android.com/market`.

[3] Google AdSense Program Policies. `https://www.google.com/adsense/support/bin/answer.py?answer=48182`.

[4] iPhone App Store. `http://www.apple.com/iphone/apps-for-iphone`.

[5] Ovi store. `http://store.ovi.com`.

[6] xda-developers. `http://www.xda-developers.com`.

[7] Top 10 Android Phones, 2011. `http://www.pcworld.com/reviews/collection/3286/top_10_android_phones.html`.

[8] A. Al-Bataineh and G. White. Detection and Prevention Methods of Botnet-generated Spam. In *MIT Spam Conference*, 2009.

[9] T. Asad. Jailbreak ios 4.3.3 untethered on iphone 4, 3gs, ipad, ipod touch with pwnagetool 4.3.3 [tutorial]. Redmond Pie, 2011. `http://www.redmondpie.com/jailbreak-ios-4.3.3-untethered-iphone-4-3gs-ipad-ipod-touch-4g-3g-using-pwnagetool-4.3.3-tutorial`.

[10] M. Balakrishnan, I. Mohomed, and V. Ramasubramanian. Where's That Phone? Geolocating IP Addresses on 3G Networks. In *IMC*, 2009.

[11] D. Barroso. ZeuS Mitmo: Man-in-the-mobile (III). `http://securityblog.s21sec.com/2010/09/zeus-mitmo-man-in-mobile-iii.html`.

[12] M. Becher, F. Freiling, J. Hoffmann, T. Holz, S. Uellenbeck, and C. Wolf. Mobile Security Catching Up? Revealing the Nuts and Bolts of the Security of Mobile Devices. In *IEEE Symposium on Security and Privacy*, 2011.

[13] M. Boodaei. Mobile Users Three Times More Vulnerable to Phishing Attacks. Trusteer Technical Report.

[14] C. Burns. HTC Unlocking Bootloaders Across the Board [OFFICIAL], 2011. `http://www.slashgear.com/htc-unlocking-bootloaders-across-the-board-official-26155031`.

[15] J. Caballero, C. Grier, C. Kreibich, and V. Paxson. Measuring pay-per-install: The commoditization of malware distribution. In *USENIX Security*, 2011.

[16] M. Calamia. Mobile payments to surge to $670 billion by 2015. `http://www.mobiledia.com/news/96900.html`, 2011.

[17] R. Cannings. An update on Android Market security. Google Mobile Blog. `http://googlemobile.blogspot.com/2011/03/update-on-android-market-security.html`.

[18] G. Clucley. Hacked iPhones held hostage for 5 Euros. *Naked Security*, 2009.

[19] C.Mulliner, N. Golde, and J. Seifert. SMS of Death: From Analyzing to Attacking Mobile Phones on a Large Scale. In *USENIX Security*, 2011.

[20] Cyanogen(mod). OpenVPN, 2011. `http://www.cyanogenmod.com/features/openvpn`.

[21] N. Daswani, C. Mysen, V. Rao, S. Weis, K. Gharachorloo, and S. Ghosemajumder. Online advertising fraud. *Crimeware: Understanding New Attacks and Defenses*, 2008.

[22] N. Daswani and M. Stoppelman. The anatomy of Clickbot. A. In *Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 11–11. USENIX Association, 2007.

[23] S. Doherty and P. Krysiuk. Android.Basebridge. Symantec, 2011. `http://www.symantec.com/security_response/writeup.jsp?docid=2011-060915-4938-99`.

[24] M. Egele, C. Kruegel, E. Kirda, and G. Vigna. PiOS: Detecting Privacy Leaks in iOS Applications. In *NDSS*, 2011.

[25] W. Enck, P. Gilbert, B. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. In *OSDI*, 2010.

[26] W. Enck, M. Ongtang, and P. McDaniel. On Lightweight Mobile Phone Application Certification. In *CCS*, 2009.

[27] F-Secure. Trojanised mobile phone game makes expensive phone calls. `http://www.f-secure.com/weblog/archives/00001930.html`, 2010.

[28] A. P. Felt, K. Greenwood, and D. Wagner. The Effectiveness of Application Permissions. In *USENIX WebApps*, 2011.

[29] A. P. Felt and D. Wagner. Phishing on Mobile Devices. In *W2SP*, 2011.

[30] J. Franklin and V. Paxson. An inquiry into the nature and causes of the wealth of Internet miscreants. In *CCS*, 2007.

[31] D. Goodin. Backdoor in top iPhone games stole user data, suit claims. *The Register*, 2009.

[32] C. Guo, H. J. Wang, and W. Zhu. Smart Phone Attacks and Defenses. In *ACM Workshop on Hot Topics in Networks*, 2004.

[33] J. Hamada. New Android Threat Gives Phone a Root Canal. Symantec, 2011. `http://www.symantec.com/connect/blogs/new-android-threat-gives-phone-root-canal`.

[34] E. Haselsteiner and K. Breitfuß. Security in near field communication. *Workshop on RFID Security*, 2006.

[35] iClarified. How to change your iPhone IMEI with ZiPhone (Windows). http://www.iClarified.com/entry/index.php?enid=657.

[36] J. Jamaluddin, N. Zotou, and P. Coulton. Mobile phone vulnerabilities: a new generation of malware. In *IEEE International Symposium on Consumer Electronics*, 2004.

[37] X. Jiang. Security Alert: New Sophisticated Android Malware DroidKungFu Found in Alternative Chinese App Markets. http://www.cs.ncsu.edu/faculty/jiang/DroidKungFu.html, 2011.

[38] C. Johnson. Kenzero virus blackmails those who illegally download anime porn. BBC. http://news.bbc.co.uk/2/hi/technology/8622665.stm.

[39] Juniper Global Threat Center. Fake player. http://globalthreatcenter.com/?p=1907.

[40] G. Lawton. Is it finally time to worry about mobile malware? *Computer*, May 2008.

[41] M. Fossi (Editor). Symantec Report on the Underground Economy. Symantec Corporation, 2008.

[42] J. Markoff. Surveillance of Skype Messages Found in China. *New York Times*, 2008.

[43] B. Miller, P. Pearce, C. Grier, C. Kreibich, and V. Paxson. What's Clicking What? Techniques and Innovations of Today's Clickbots. In *DIMVA*, 2011.

[44] Mobclix. Monthly value of an app user. http://blog.mobclix.com/index/PDF/january_infographic.pdf.

[45] C. Mulliner. Vulnerability Analysis and Attacks on NFC-enabled Mobile Phones. In *Proceedings of the 1st International Workshop on Sensor Security (IWSS) at ARES*, Fukuoka, Japan, 2009.

[46] Y. Niu, F. Hsu, and H. Chen. iPhish: Phishing Vulnerabilities on Consumer Electronics. In *UPSEC*, 2009.

[47] P. Porras and H. Saidi and V. Yegneswaran. An Analysis of the Ikee.B (Duh) iPhone Botnet. *SRI International*, 2009. http://mtc.sri.com/iPhone.

[48] Panda Security. Eeki.A. http://www.pandasecurity.com/homeusers/security-info/215107/Eeki.A, 2009.

[49] C. Peikari. PDA attacks, part 2: airborne viruses-evolution of the latest threats. *(IN) SECURE Magazine*, 2005.

[50] P. Roberts. Android NFC bug could be the first of many. http://threatpost.com/en_us/blogs/android-nfc-bug-could-be-first-many-062011, 2011.

[51] S. Rosenblatt. Avast to go mobile, get VPN. The Download Blog, 2011. http://download.cnet.com/8301-2007_4-20074377-12/avast-to-go-mobile-get-vpn.

[52] A. Schmidt, H. Schmidt, L. Batyuk, J. H. Clausen, S. A. Camtepe, and S. Albayrak. Smartphone Malware Evolution Regisited: Android Next Target? In *MALWARE*, 2009.

[53] A. Shevchenko. An overview of mobile device security. http://www.viruslist.com/en/analysis.

[54] T. Strazzere. Security Alert: HongTouTou, New Android Trojan, Found in China. *The Lookout Blog*, 2011.

[55] T. Strazzere. Security Alert: Malware Found Targeting Custom ROMs (jSMSHider). *The Lookout Blog*, 2011.

[56] T. Strazzere. Security Alert: zHash, A Binary that can Root Android Phones, Found in Chinese App Markets and Android Market. *The Lookout Blog*, 2011.

[57] Symantec. Android.geinimi. http://www.symantec.com/security_response/writeup.jsp?docid=2011-010111-5403-99.

[58] Symantec. Android threat set to trigger on the end of days, or the day's end. http://www.symantec.com/connect/blogs/android-threat-set-trigger-end-days-or-day-s-end, 2011.

[59] Symantec. Symbos.spitmo. http://www.symantec.com/security_response/writeup.jsp?docid=2011-040610-5334-99, 2011.

[60] B. Thompson. UAE Blackberry update was spyware. http://news.bbc.co.uk/2/hi/technology/8161190.stm.

[61] S. Toyssy and M. Helenius. About malicious software in smartphones. *Journal in Computer Virology*, 2006.

[62] P. Traynor, M. Lin, M. Ongtang, V. Rao, T. Jaeger, P. McDaniel, and T. La Porta. On Cellular Botnets: Measuring the Impact of Malicious Devices on a Cellular Network Core. In *CCS*, 2009.

[63] Trend Micro. BBOS_ZITMO.B. http://about-threats.trendmicro.com/Malware.aspx?language=us&name=BBOS_ZITMO.B, 2011.

[64] T. Vidas, D. Votipka, and N. Christin. All your droid are belong to us: A survey of current android attacks. In *WOOT*, 2011.

[65] J. Wortham. Unofficial Software Incurs Apple's Wrath. *The New York Times*, 2009.

APPENDIX

A. LIST OF MALWARE

Android. Fake Player, Geinimi, ADRD a.k.a. HongTouTou, PJApps, DroidDream a.k.a. Rootcager, Bgserv, Zhash a.k.a. Zeahache, Walk&Text a.k.a. Walkinwat, Adsms, Zsone a.k.a. Smstibook, Smspacem, Lightdd a.k.a. Droid Dream Light, DroidKungFu a.k.a. Legacy a.k.a. Gonfu, Basebridge, YZHC-SMS a.k.a. Uxipp, Plankton a.k.a. Tonclank, jSMSHider, and Ggtracker.

iOS. Dutch 5Euro Ransom, Ikee a.k.a. Eeki.A, Privacy.A, and Ikee.B a.k.a. Duh.

Symbian. Kinap a.k.a. Panika a.k.a. Appdisabler, Flocker, Exy a.k.a. Yxe, Feixiang, BadAssist, Album, CommDN, MerogoSMS a.k.a. Merogo, Enoriv a.k.a. SMSSend.2, Downsis, Nokmaplug a.k.a. NMPlugin a.k.a. Nmapplug, CReadMe, ZeusMitmo a.k.a. Zitmo a.k.a. Zbot, Sagasi a.k.a. SPIsSaga, Themeinstaller, GamePackage, Zhaomiao a.k.a. SZhaomiao, InSpirit, Instalarm a.k.a. Installer.A, Shurufa, Lopsoy, Spitmo, Sslcrypt, and SkinServer.