Evaluating Bluetooth as a Medium for Botnet Command and Control

Kapil Singh, Samrit Sangal, Nehil Jain, Patrick Traynor and Wenke Lee

School of Computer Science, Georgia Institute of Technology
{ksingh, samrit, nehjain, traynor, wenke}@cc.gatech.edu

Abstract. Malware targeting mobile phones is being studied with increasing interest by the research community. While such attention has previously focused on viruses and worms, many of which use near-field communications in order to propagate, none have investigated whether more complex malware such as botnets can effectively operate in this environment. In this paper, we investigate the challenges of constructing and maintaining mobile phone-based botnets communicating nearly exclusively via Bluetooth. Through extensive large-scale simulation based on publicly available Bluetooth traces, we demonstrate that such a malicious infrastructure is possible in many areas due to the largely repetitive nature of human daily routines. In particular, we demonstrate that command and control messages can propagate to approximately 2/3 of infected nodes within 24 hours of being issued by the botmaster. We then explore how traditional defense mechanisms can be modified to take advantage of the same information to more effectively mitigate such systems. In so doing, we demonstrate that mobile phone-based botnets are a realistic threat and that defensive strategies should be modified to consider them.

1 Introduction

Mobile phones are being increasingly tasked with sophisticated duties. From trading on financial markets and mobile banking to carrying medical records [29], these devices are beginning to be trusted with some of our most sensitive information. Unfortunately, because the majority of mobile phones lack even basic security mechanisms (e.g., memory protection), they are becoming increasingly attractive targets for malware writers. The widespread usage of such devices and the sensitivity of cellular networks to even small amounts of malicious traffic make this actuality a significant threat.

A wide range of malware targeting mobile phones has already been documented. Whether arriving via MMS [22], a downloaded executable [35] or over a Bluetooth link [14], both viruses and worms are being extensively explored in this environment. Botnets, however, have not yet been studied in depth in this setting. Representing one of the most significant threats to the Internet, mobile botnets could use compromised phones to execute regularly updated mission requests (e.g., Denial of Service, premium number dialing, password/credential theft, etc). Like their Internet-based counterparts, mobile botnets can only achieve such flexibility given the presence of a robust *command and control* (C&C) infrastructure. Unlike traditional botnets however, we argue that

such an infrastructure can be successfully maintained outside of the purview of cellular providers, making detection and mitigation challenging given current strategies.

In this paper, we evaluate the potential for mobile phone-based botnets to communicate and coordinate predominantly via Bluetooth. Unlike previous work that investigates whether malware can spread over such links, we instead investigate *whether a command and control infrastructure can be maintained in an environment with almost entirely transient links.* We develop an understanding of the long term interaction of infected devices through the use of two large-scale datasets and, through simulation, demonstrate that the repetitive nature of the daily routines of human beings allows messages to be propagated to over 66% of infected nodes within a day. We then compare the impact of varying parameters including device popularity and polling interval to allow a botmaster to tradeoff the speed of propagation with their ability to remain hidden from a network provider. Finally, we conduct large-scale simulations to attempt to better model the dynamics of such botnets in a realistic setting - public transit.

In so doing, we make the following contributions:

- **Develop the first characterization of Bluetooth-based C&C for mobile devices:** Using publicly available data on mobile device interaction, we develop the first characterization of command and control operations in this setting. In particular, we show that mobile botnets are possible, but that instruction propagation latency can be significant. In exchange for such latency, botmasters are able to notably reduce the amount of traffic observable by the provider.
- **Create a new C&C architecture based on node popularity:** We develop a framework in which bots selectively communicate with the botmaster based on their popularity. In particular, only a small subset of bots with the highest degree ever speak directly to the botmaster. This mechanism helps to improve the speed of propagation without exposing all infected nodes to a network provider.
- **Develop countermeasures leveraging communication patterns:** From the information learned above, we develop patching and mitigation strategies that significantly reduce a mobile botnet's ability to defend against our countermeasures and remain hidden.

Note that traditional botnet C&C infrastructures are likely to be easily detected in these networks as providers are more likely to have a more complete global view.[1] This means that bots are unlikely to be successful in these networks unless they adopt a strategy similar to the one presented in this work.

The remainder of this paper is organized as follows: Section 2 provides an overview related work in the area of mobile malware; Section 3 discusses how mobile phones can and are likely to be infected and explores a number of possible mechanisms for command and control; Section 4 explains our data and simulator that we use to model Bluetooth-based botnets; Section 5 simulates such networks using well-known public data sets; Section 6 models a large-scale botnet in a public-transportation setting; Section 7 discusses how the above observations can be leveraged to combat such botnets; Section 8 offers concluding remarks.

---

[1] This is also true because associating messages with their origin given that each device authenticates to the network.

2 Related Work

Botnets [10, 26] represent the major source of malicious activity on the Internet – they send spam [27], perform DDoS attacks [16] and host phishing web sites [7]. Significant attempts have been made by the research community to both categorize [6, 10] and mitigate [17–19, 27] such threats. Unfortunately, understanding such threats in the context of cellular networks is still very limited [33, 34].

The transformation of mobile devices from simple voice terminals into highly-capable, general purpose computing platforms makes the possibility of attacks originating from within the network a reality. The tremendous increase of cell phone adoption and the lack of widely implemented security mechanisms makes such platforms attractive targets to botmasters. Research has previously shown cellular infrastructure as a potential target of botnet attacks [33]; however, such botnets are composed entirely of compromised machines across the Internet. Previous work has not considered whether or not such a malicious overlay can be created and maintained exclusively on mobile phones.

Bluetooth-based malware has been extensively explored. Su et. al highlighted the presence of a diverse set of known security vulnerabilities in the Bluetooth protocol's implementation [31]. They argued that the presence of such vulnerabilities coupled with the complexity of the Bluetooth specification and its large codebase will likely lead to more complex attacks using the Bluetooth channel. Worms including Cabir [8], Mabir [9] and CommWarrior [21] have already successfully exploited this channel. Previous efforts to model the *propagation* behavior of Bluetooth-based malware have focused entirely on the analysis of Bluetooth worms. Yan et. al studied the effect of mobility on worm propagation by restricting the devices in an area with sides of length 150 meters [37, 39]. The same authors later provided a comprehensive analytical model for such Bluetooth worm propagation [38]. Other studies have investigated the effect of population characteristics and device behavior on the outbreak dynamics of Bluetooth worms [23, 28]. Such characteristics have been previously exploited for peer-to-peer (P2P) content distribution [20] and for studying human social behavior [12].

Mobile phone-based botnets using Bluetooth to propagate control messages bear a striking resemblance to Internet-based P2P botnets [5, 13]. In particular, even if defenders identify a subset of the bots in a botnet, communication among the remaining bots will not be disrupted. Second, in contrast to other centralized approaches (such as IRC [6]), there is no fixed endpoint from which the botmaster must transmit commands. For Bluetooth botnets, the botmaster can send messages from any Bluetooth-enabled device, and he can frequently change these source devices to evade detection. Bluetooth communication has other additional, attractive properties that benefits botnet creators. Unlike the Internet-based P2P channels, Bluetooth by principle is proximity based: this gives the defenders limited scope to observe the communication between two bot devices. Additionally, P2P botnets suffer from the problem of losing bots whenever those bots change their dynamic IP addresses. Bluetooth channels are resilient to such changes.

3 Bluetooth-based Botnets

Our goal is to evaluate the suitability of Bluetooth as a command and control channel. In particular, we focus on the challenges facing a botmaster trying to coordinate a large number of infected devices over a transient, near-field communication channel that cannot be easily identified or blocked by cellular providers. In this section, we discuss how mobile phones can be compromised, detail our threat model and assumptions, and discuss the C&C architecture of mobile phone-based botnets.

3.1 Infecting Devices

Mobile devices have rapidly transformed from limited embedded systems to highly capable general purpose computing platforms. While such devices have long enjoyed significant diversity in hardware and operating systems, the rising popularity of smartphones and the ability to sell applications to users is leading to the establishment of standardized mobile software platforms and operating systems, such as Microsoft's Windows Mobile, Google's Android and Apple's Mobile OS X. Unfortunately, many devices are only now beginning to implement basic security mechanisms including memory protection and separation of privilege. Accordingly, such systems are expected to be increasingly targeted by malware. Malware targeting mobile devices may come from any of a number of sources. Given that 10% of cellular users downloaded games to their mobile devices at least once a month in 2007 [24] and the wide availability of free ringtones, downloadable content and executables are one of the more likely origins. Like their desktop counterparts, mobile devices are also likely to be susceptible to a range of browser exploits including drive-by downloads [25]. Finally, the presence of multiple communications interfaces makes mobile devices susceptible to malware that propagates not only through the cellular network itself [15], but also potentially through WiFi and Bluetooth [8,9,21]. Accordingly, the breadth of infection vectors exceeds that of many traditional networked systems.

3.2 Threat Model and Reasoning

Given the above, we assume that bots have already been installed on a subset of mobile phones within a network but that the C&C infrastructure remains unestablished.

We expect that powerful defenses exist to detect and disrupt botnets. In our threat model, we assume that defenders have access to malicious binaries and are able to learn the bot's entire execution behavior through forensic analysis techniques. This allows defenders to identify the command list and the algorithms used by the bot to extract commands from Bluetooth messages. Simply stated: the threat model allows defenders to know everything that is known to the bot.

Botmasters logically aim to sustain their networks for as long as possible. Stealth is therefore one of the most critical characteristics of such a botnet. The short range and ad-hoc nature of communications via Bluetooth potentially provides such an opportunity: defenders need to be *within range* of the communicating infected devices *at the time of communication*, which might not be practical considering the changing network topology of the ad-hoc network. Given that providers are not able to observe the vast
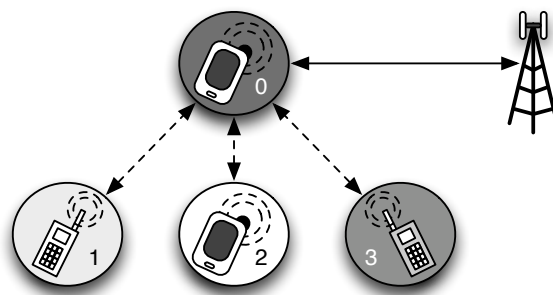
Fig. 1. Only nodes with the highest exposure to other bots (shown here by the darkness of the circle) contact the botmaster directly. All other nodes receive commands and updates when they are within Bluetooth range of an updated node.

majority of messages between infected devices, this means of communication appears attractive.

Reliance upon near-field communications also makes commanding bots more challenging. Specifically, a botmaster must rely on infected nodes being within range of each other on a regular basis in order to successfully propagate commands. We leverage the fact that a large portion of the population follows regular patterns of behavior. For instance, person A gets up every day at 7 am and goes to his office using the subway. In the process, he will encounter a large subset of the same people every day (who also take the train at roughly the same time everyday). He will also interact with a large subset of the same people everyday in the office. The highly regular nature of this routine provides structure in an unstructured environment. Specifically, while the botmaster may not know the exact topology of the network at any particular moment, he or she will know with some assurance the subset of devices with which a user is likely to interact with during a period of time.

3.3 Botnet Construction and Message Passing

We now explore the construction and operation of mobile phone-based botnets based on our threat model. While traditional strategies such as having all nodes use cellular data connections to reach Internet-based centralized or P2P designs are possible, large volumes of such communication are easily detectable by a provider. Accordingly, we aim to develop a mechanism that avoids such detection while overcoming a number of challenges. For instance, the botmaster must be able to learn the identities of all of the phones under his control so as to be able to accurately portray the size of the botnet when renting it. This task must be accomplished without these devices contacting the botmaster directly. Additionally, the botmaster needs to ensure that he can contact the largest possible number of infected phones within a short period of time. Unfortunately, a Bluetooth-only solution is unlikely to be sufficient in this context. In particular, an adversary would need to physically place himself near as many nodes as possible, which is more likely to be a burdensome task.

We instead propose a hybrid approach designed to maximize the speed of distribution while only minimally reducing stealth. In particular, we allow a botmaster to communicate with a *very small* number of infected nodes through cellular channels (e.g., SMS, cellular data). These nodes are selected via their relative frequency of contact with other infected devices. Specifically, as infected devices pass within range of each other, they record the identity of the other device. After reaching some threshold set by the botmaster, those nodes with a high degree of connectivity over time contact the botmaster and provide their contact log. In so doing, these devices not only provide the botmaster with knowledge of the devices under his control, but also inform him of which nodes are most likely to be able to help rapidly disseminate commands.

Figure 1 represents one such typical scenario, where the darkness of the circle around the phone reflects the popularity of that device. As we can determine from the figure, phone 0 is the most popular and therefore act the seed for communication with the botmaster. If phone 0 is disinfected, phone 3 would likely report back to the botmaster (potentially after the expiration of some long-term timeout value).

The botmaster also disseminates commands through this hierarchical structure. When a new task arises, the botmaster simply contacts the seed nodes in a particular area and provides an updated mission/payload to be distributed to other infected nodes. These nodes are most likely to be able to deliver such a payload to the largest possible number of infected nodes without requiring them to directly interact with the botmaster because of their high degree of connectivity over time.

Seed nodes logically present attractive targets to defenders. Those nodes reporting back to some centralized point are more likely to be singled out by the provider. Naïvely constructed, such a strategy would allow a provider to cripple the communications of these systems. If the very low volume of traffic is not sufficient for avoiding detection, such bots can further obfuscate their activities through a number of anonymizing techniques ranging from Tor [11] and Publius [36] to the use of a free temporary email address. Such communication could be further obscured through the use of the WiFi connection available to many smart phones. Communications from the botmaster can avoid detection by spoofing the source address of a communication from the Internet, including text messages claiming to be from within a target node's community of interest [33].

4   Experimental Setup

Given a proposed communications architecture and our threat model, we now seek to determine whether or not mobile phone-based botnets can effectively communicate using Bluetooth. In this section, we discuss a number of details related to our experimental design and testing. This infrastructure is used throughout the remainder of the paper.

4.1   Prototype Bot

Rather than running a simulation with nodes simply exchanging meaningless messages, we implemented a proof of concept mobile phone-based bot. Our prototype bot is coded in Java and deployed on the Sun Wireless Toolkit that emulates infected mobile devices.

Each bot instance acts as a peer in the bot network, listening for new commands and simultaneously sending commands to other discovered bots. At the initial infection stage, the bot registers itself with a Universally Unique Identifier (UUID) in the service register present in the mobile device, thus allowing it to be discovered by other bots. It then waits for new incoming connections. A two-way Bluetooth connection is established with other bots when they come within range. As part of our protocol for information exchange among bots, the bot is updated with the latest version of the command, which also includes the updated parameters of the command.

As a proof-of-concept, we implemented a botnet command that directs bots to send an SMS to a specified mobile number without being noticed by the sender. Such a command can launch a denial-of-service attack on a targeted area [33] should enough devices participate. Because most service providers also charge for incoming messages, such attacks can also incur substantial costs for the targeted victim. These initial experiments demonstrate that Bluetooth can successfully be used to pass commands between infected nodes with relative ease.

4.2 Experimental Goals

We use our proof-of-concept bot to evaluate various parameters that directly or indirectly impact command propagation in a mobile phone-based botnet. While some of these parameters, such as the device polling interval, can be controlled by the botmaster, other parameters are influenced by the inherent characteristics of near-field communications and the human movement patterns. We enumerate these parameters and evaluate their impact using a range of simulations that model a range of settings and scenarios.

Our simulations are categorized into two experimental sets. The first set of simulations are performed based on two trace logs, one each from MIT [12] and NUS [30], that are collected using real devices carried by individuals for their day-to-day activities. By means of these experiments, we evaluate various operational parameters of the proposed botnet C&C mechanism, which are useful in determining the correctness of our hypothesis.

In our second set of experiments, we use publicly-available information to create a large scale simulation model of New York City's subway system and thereafter, use this model to demonstrate command propagation in a typical scenario of public transportation. The goal of these simulations is to expand our evaluation beyond the boundaries imposed by the limited traces, and demonstrate the viability of a large-scale mobile phone-based botnet.

In order to run the simulation, one initial node is selected and subsequently all the nodes encountered by the selected node are enumerated in time stamp order. If the period of contact shown by the traces is equal to or more than 5 minutes (this is the least time granularity for the MIT data set), we assume that the other node successfully received the command. This is a conservative assumption given that botnet commands are generally negligible in size (on the order of tens of bytes) and Bluetooth can transfer data at a rate of approximately 1Mbps. In our experiments, we examine various factors and environmental variables that have an effect on the latency of the command propagation in the proposed Bluetooth-based botnet.
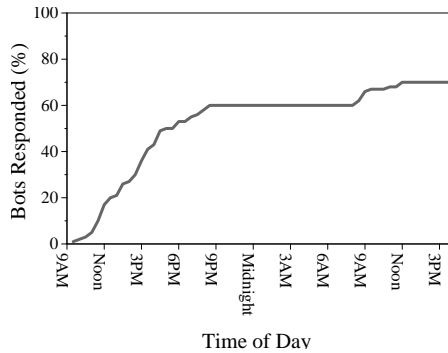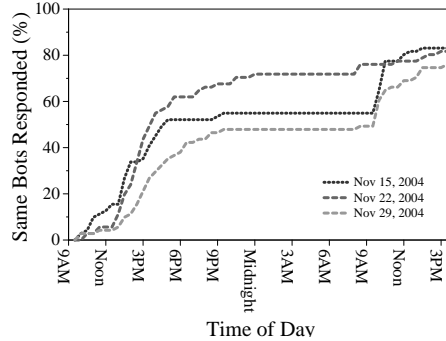
Fig. 2. Command Propagation Rate.

Fig. 3. Recurrent bot connections over different days.

We do not consider device heterogeneity in our experiments because we view infection and command propagation as two separate tasks. In particular, because devices can be infected through any number of different vectors (e.g., drive-by downloads, malicious executables, browser bugs, etc), phones of all varieties can be forced to run bot software. Our evaluation focuses not on how infection happens but how messages spread after infection has occurred.

5 Trace Based Simulations

We use the architecture and assumptions detailed in the previous two sections to develop simulations of mobile phone-based botnets. In this section, we use publicly available Bluetooth traces to perform a number of simulations. These experiments characterize the impact of various factors on effective botnet communication.

5.1 Description of Datasets

Reality Mining Dataset (MIT): The Reality Mining project is a collection of environmental data gathered by one hundred mobile phones over a course of six months. Polling by each of these phones was used to determine the presence and identity of other Bluetooth-capable devices in their proximity. This data was initially used to provide insight into the dynamics of the social behavior of both individuals and groups [12]. While one of the more extensive available datasets, the major limitation of this study is that the time between subsequent polling for devices is five minutes and as a result, our evaluations miss interaction of devices that were in contact for less time. The proposed message dissemination techniques are therefore likely to perform even better than the results we present. We attempt to overcome this limitation with a large-scale simulation in Section 6.

Bluetooth Dataset (NUS). We use a second collection of logs known as the the National University of Singapore (NUS) Bluetooth [30] dataset that contains traces

of Bluetooth sightings by 12 devices over a period of 7 months. Each device polled for other devices every 30 seconds and recorded device identifiers of all the Bluetooth enabled devices in its range, providing significantly finer granularity of interaction than the MIT dataset. Out of the 12 experimental devices, 3 were static devices placed near lecture halls on the NUS campus and the rest were given to the faculty and students. This dataset makes it possible for us to evaluate the effect of varying polling interval on the characteristics of the Bluetooth-based botnet, and also demonstrate how mobility of the commanding device can influence the command propagation rate in the botnet.

5.2 Simulation Results

Command Propagation Rate Figure 2 provides a single but representative view of command dissemination by the most popular node. Note that the command is rapidly dispersed during the morning hours, plateaus in the evening and the increases again slightly the next morning. Note that because of the regularity of regularity of human behavior, the increase on the second day is relatively low given that few different nodes are observed between any two given days. Our experiments demonstrate that messages are consistently delivered to greater than 2/3 of all infected nodes within 24 hours of the botmaster contacting the seed node.

We define *Probability of Delivery (PoD)* as the percentage of bots responding within a predefined time period. The desired time represents the maximum acceptable latency for a botmaster to successfully distribute a command to some portion of the nodes under his control. Given Figure 2, a realistic value for such a response time is approximately 24 hours; however, the nature of the command may offer the botmaster additional flexibility. For instance, the botmaster may not require that a spam campaign be coordinated, allowing nodes to being their job immediately after receiving the command. Alternatively, denial of service attacks are likely to require greater coordination, meaning that such attacks may need to be planned further in advance in order to be successful. When sent at optimal times (i.e, the morning), PoD values rise relatively quickly, with 40% of nodes having received commands within five hours (Figure 2). Mission and command launch time must therefore be carefully considered by the botmaster.

Note that our experiments assume the use of only a single seed node. A botmaster could potentially seed multiple nodes with commands in order to compensate for known geographic barriers or increase the speed of command dissemination. However, such a strategy must be carefully balanced against the botnet's need for stealth.

Long-Term Communications The results thus far demonstrate that Bluetooth-based mobile botnets are capable of passing commands on a single day. However, the ability to re-establish connections across long periods of time is necessary for such a network to be worth the effort to construct. For this analysis, we take the traces from November 1, 2004 as the initial list of infected bots. We then evaluate how many of these infected nodes come in contact with a command-carrying node over subsequent days. The same node is chosen as the initial node for iterations repeated over different days. Note that the experiments were conducted over the closed set of 100 nodes in the MIT data set; other datasets show similar behavior (Section 6).
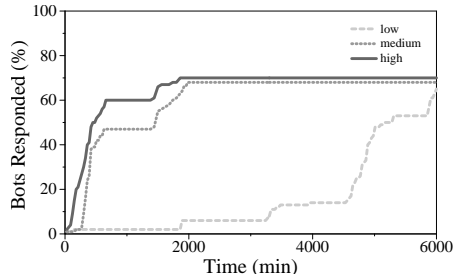
Fig. 4. Effect of the node popularity on command propagation.

	30	50	70
High Popularity Node	5hrs	8hrs	13hrs
Medium Popularity Node	5.5hrs	25hrs	31hrs
Low Popularity Node	80hrs	86hrs	102hrs

Fig. 5. Propagation Times for various nodes.

Figure 3 shows the percentage of infected nodes that repeatedly come in contact with a command-carrying node over different days. In other words, it represents the number of bot nodes that would successfully receive a botnet command. Our results show that bots tend to come in contact with a large subset of the same nodes repeatedly over different days. This number is on an average about 50% within the first 8 hours of a day's schedule and goes up to more than 80% after a day.

As seen in Figure 3, the pattern does not vary significantly across different days, which shows that a command issued at a particular node will follow a generally predictable spread for that node on any given day. While we observed a maximum variance of about 18% at any particular time, eventually this variance becomes negligible with more than 80% of the same infected nodes are consistently seen on any given day.

Device Popularity Figure 4 shows the command propagation rate for seed nodes of high, medium and low popularity. In order to determine the popularity of the nodes, we sort all nodes based on the number of contacts they have with other nodes. High and low popular nodes have the maximum and minimum contacts respectively; node with medium popularity is chosen as the median of the sorted data set. The patterns in this figure are representative of all days.

The popularity of a seed node intuitively has an effect on the time it takes for the command to propagate. However, this difference is negligible for some cases for nodes of medium and high popularity. Nodes with very low popularity take significantly longer to reach a desired PoD but eventually exhibit the same saturation. A likely explanation for this behavior is that such nodes eventually encounter more popular nodes, thereby increasing the rate of message dissemination. Accordingly, regardless of which node is selected, commands are eventually propagated. This results may not always hold true for larger datasets, which may have much more significant outliers than the MIT and NUS datasets; however, most nodes will encounter a large enough number of other nodes that message delivery can occur within a reasonable timeframe.

Figure 5 provides numerical results corresponding to the range of popular nodes. Note that while reaching 70% requires a significant amount of time in all but the most popular case, such widespread distribution may not be required. In particular, given a large number of nodes, a botmaster may only want to dedicate a subset of their bot-
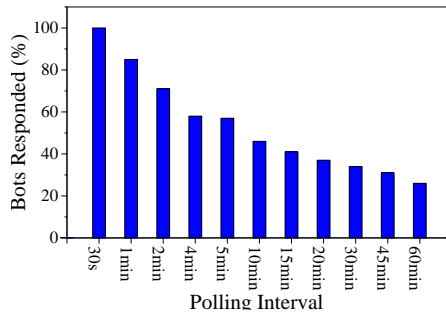
Fig. 6. Effect of varying polling intervals on static seed nodes.
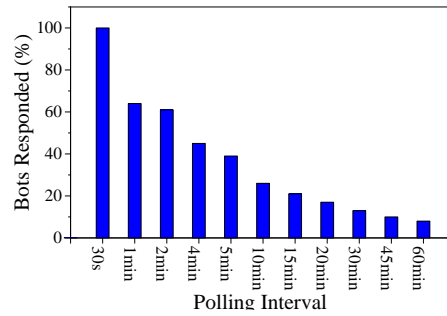

Fig. 7. Effect of varying polling intervals on mobile seed nodes.

net to a specific task. Accordingly, a botmaster may wish to select nodes of differing popularity to best meet their mission.

Polling Interval We define *polling interval* as the time between two consecutive polls conducted by a Bluetooth device looking for other devices in its proximity. Figures 6 and 7 show the effect of the variation in polling interval on the PoD. The experiment is performed on the NUS data set, which has a base polling interval of 30 seconds, as the polling granularity of the MIT dataset is too coarse to offer interesting results. The results are presented with polling interval of 30 seconds as the base reference; all other results are shown relative to this case. The botmaster is able to control the polling interval at each bot. A low polling interval would diminish stealthiness as the bot may become visible to the user of the victim devices: more frequently probing by the device has an adverse effect on the battery life of the device. This observable behavior might alert the user about the presence of a malicious bot on the device, thus exposing the botnet. On the other hand, increasing the polling interval improves the stealth of the botnet, but reduces the number of devices the botmaster can spread his command to in the desired time.

We repeated the experiment both for the static nodes placed at strategic points on the NUS campus and for the mobile nodes that were free to roam around. As shown by the graphs in Figure 6 and 7, there is a clear decrease in the PoD with increasing polling interval. This result is expected because with longer polling interval, there is greater possibility of missing devices that come in the proximity of the polling device at a time between two subsequent polls when the device is not polling. Moreover, the percentage drop in the PoD is less for static nodes as compared to the mobile nodes: both static and mobile nodes show the highest PoD for the base case with polling interval of 30 seconds; for polling interval of 1 minute, this goes down to 85% for static nodes and much lower 65% for mobile nodes. One possible reasoning behind a much lower drop for the case of static nodes is that people traverse popular spots multiple times and spend relatively longer time at such locations, hence there is a stronger possibility of a successful command transfer to the corresponding devices carried by these people.
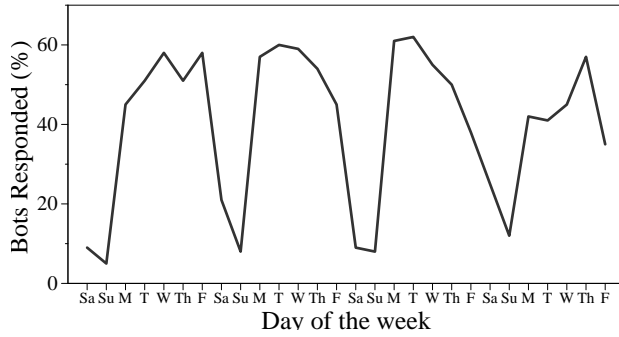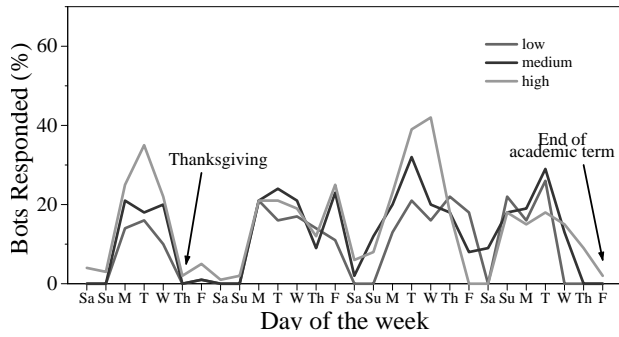
Fig. 8. Weekday Effect: Static nodes

Fig. 9. Weekday Effect: Dynamic nodes

However, for mobile nodes, since even the polling device is in motion, this phenomenon of repeatedly coming in contact may not occur.

The graphs also show that even with a higher polling interval, a considerable percentage of the devices can still receive the botnet command. For example, even for a polling interval as high as 60 minutes, the PoD values are still significant – 26% for static nodes and 9% for dynamic nodes.

In essence, by choosing the polling interval for his bot victims and carefully placing Bluetooth devices at strategic locations, the botmaster can achieve a balance between the PoD and stealthiness for his desired use of the botnet. Additionally, the botmaster can direct his bots to use *adaptive* polling to attain such a balance: a bot would aggressively search for other devices by lower its polling interval when the bot device is in motion (possibly during morning and evening hours) and would switch to a higher polling interval at other times of the day when the device is static.

Weekday Patterns Humans follow daily and weekly patterns that also greatly influence the command propagation rate. Figure 8 presents the number of daily encounters broken down by the day of the week when they occur. This graph confirms the intuition that more encounters occur on week days than on weekend days. We observed similar

behavior for both MIT and NUS data sets, and also for both static and mobile seed nodes (Figure 8 and 9). Additionally, such behavior is independent of the popularity of the node (Figure 9).

These observations support our argument that repetitive nature of human routines can be leveraged by mobile phone-based botnets. With a better understanding of such weekday patterns of the targeted devices, the botmaster can be more effective in propagating the commands faster. For example, it would be beneficial to issue a command on a weekday rather than a weekend. Special considerations should also be made for days such as Thanksgiving and the end of an academic term (Figure 9) [2].

Each set of contacted nodes generated for different weeks for one particular node has 70% of the nodes as common in all result sets. That means around 50 nodes were contacted every time the command was issued to a particular node, out of the 70 nodes contacted each time. Out of the remaining 30%, most of the nodes were present in more than one result set but not in all. This shows that there is low weekly fluctuations and that a botmaster can control most of the botnet effectively over any given week.

6   Modeling the Public Transport System

In this section, we perform a large-scale simulation to demonstrate the viability of a mobile phone-based botnet in a larger real-world setting. Our tests demonstrate that communication in such a botnet maintaining the previously discussed characteristics even in a large-scale environment and provides the botmaster with reasonable tradeoffs between botnet response time and stealthiness.

We simulate the rush hour period on a typical weekday at New York City's Grand Central Terminal [1]. Grand Central is one of the busiest stations in the city – it not only serves the second busiest subway station in the city, but also serves the Metro North trains from upstate New York. Consequently, the subway station receives passenger traffic both from people coming to NYC using the Metro North trains and from local commuters. We use this setting to simulate bots that reside on the cellular phones of individuals traversing the station as per their daily routines.

We use publicly available data sources for our simulations to model the station and to estimate the mobility patterns of the commuters. We also use probabilistic distributions for various parameters to allow variation in the individual behavior of bots. We acknowledge that accurately predicting patterns in human movement is difficult, however, approximations can be made by carefully analyzing different sources of publicly available data and statistics. When certain statistics are not available, we make conservative estimates.

A person arriving at Grand Central Terminal will typically have the following movement pattern: he or she arrives at Grand Central either by taking a Metro North train or by entering the station through one of the entrances. He or she traverses the station to reach the desired subway platform and then waits for a random time at the platform before the train arrives. He or she boards one of the train cars and therein remains in a static position till the train reaches his desired destination.

---

[2] verified against MIT's 2004 academic calendar.

6.1 Simulation Setup

Train Station: Grand Central Terminal is modeled as a square, with entrances and train tracks placed along its edges. The size of the station and the number of entrances used in our model are the same as those existing in the real structure [1]. We uniformly place entrances along two sides of the square. Metro North tracks (44 in total) are placed along the third side, and the fourth side has the 3 subway tracks. This design is a close approximation of the architecture of the actual terminal [1].

Train Cars: We again used publicly available information about the size of New York City's transit trains to model the trains in our simulation [3]. The number of cars are fixed and identical for all trains arriving at the subway station. While boarding a train, a person chooses one car at random. For simplicity, we assume that the commands propagate from one device to another only within one car. The train arrival times are simulated according to the known subway time schedule.

Commuter Traffic Estimates: The Metro North trains at Grand Central serve approximately 125K commuters per day [1]; approximately 150K for the subway station [2]. We assume that Metro North passengers constitute about 50% of the subway commuters. We also assume that 50% of the daily commuters take the subway during rush hours. The arrival of commuters at Grand Central is distributed over the complete rush time interval (6:30AM–9:30AM) based on a gaussian distribution with mean at 8:00AM and variance of 33%.

Phone Infections: We only consider devices that have been previously infected and a currently carried by the commuters in the terminal during these simulations. In order to estimate the number of bot victims, we consider some known statistics on cellular phones usage: about 80% of commuters in New York City carry phones according to a MNRR survey [4]. We assume that approximately 30% of these phones can be (and are) infected. We believe that this is a safe assumption based on our earlier premise that mobile software platforms and operating systems are being standardized without adequate focus on security, leaving such mobile devices vulnerable to malware infections.

Command Transfer: The time required to transfer a command from one bot-infected device to another via the Bluetooth channel is the total time taken for the four stages of the protocol, namely inquiry, connection establishment, probing and command transfer. The corresponding timeouts for the four stages are set to 10.24 (from Bluetooth specification), 5.12 (from Bluetooth specification), 0.1 and 1 seconds, respectively. We model the time for all four stages as a gaussian distribution with mean set to half of the corresponding timeout value.

Simulating Human Movement: We use a modified form of Random Landmark Model [39] to simulate the movement pattern of humans. In our simulation, the initial starting position is either one of the entrances of the train station or any track of the Metro North, and the destination is set to one of the subway tracks. These starting and ending points are randomly chosen for each individual. The speed of movement is fixed at typical walking speeds of 1, 2 and 3 meters/s. After arriving at a particular train station, a person waits at a fixed point before boarding the next arriving train. Once the entry time of an individual is determined during the initial cycle of our simulation (a cycle represents a rush hour period on any given day), we use gaussian distribution to calculate the entry time for all subsequent cycles. We limit the distribution to within
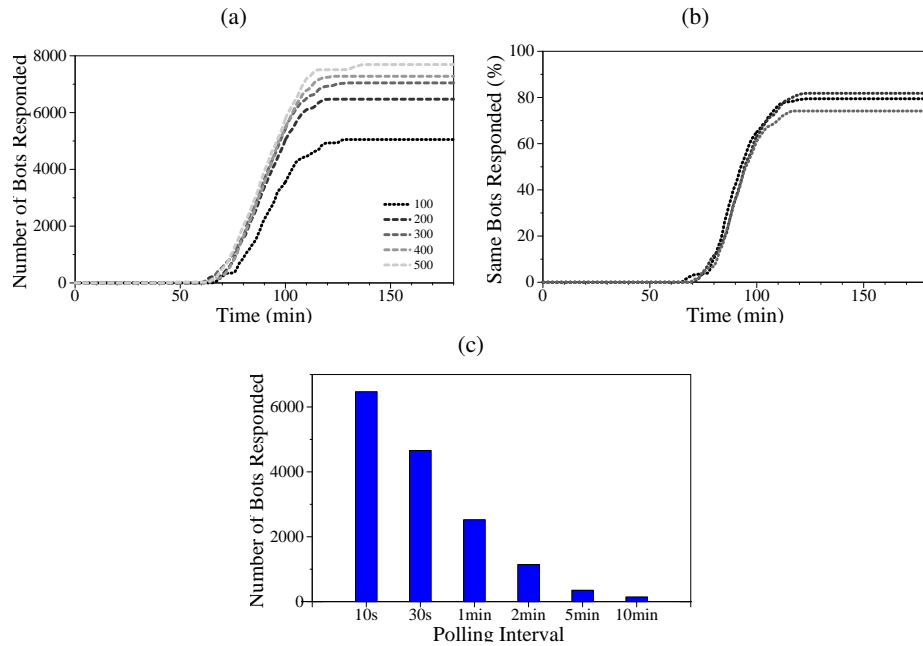
Fig. 10. Simulation results for the transport model: (a) Effect of the initial number of seeds on the number of bots receiving the command, (b) number of bot encountered repeatedly over multiple runs simulating different days, and (c) effect of varying polling interval on command propagation rate. During the rush hour commute, 30% of the total population or around 17K nodes are vulnerable. In (a), the initial seeds are chosen using the same Gaussian distribution used for the station entry time of nodes. These initial seeds are not counted in the results.

10 minutes of the entry time of the initial cycle. It effectively represents a regular daily commute for an individual, who boards a train almost at the same time every day.

6.2 Simulation Results

Our results show that public transportation can act as an ideal environment for a botmaster to effectively pass commands to a large bot population. With a relatively small initial seed of command-carrying nodes, the botnet commands can reach a considerable number of bots (Figure 10a). Such seeds can be created by the botmaster using other alternate mechanisms, for example, by planting static nodes at popular spots near the station or by dropping commands at the nodes traveling to Grand Central in the Metro North trains. We observe that more seeds allow commands to propagate faster, although the propagation rate speedup is modest beyond a threshold. This suggests that the botmaster can achieve considerable coverage even when the number of seeding bots is low: for a initial seed of 200, the command can be transferred to about 6500 nodes within the rush hour, which forms more than 35% of the population that can be infected. Note that these numbers correspond to only nodes that receive commands at Grand Central

Terminal; these nodes will further propagate the commands to other nodes when they visit their work, school, homes, etc. Such results are out of scope for this simulation.

Our transport simulation model reinforces our premise that humans follow routines in their day-to-day movements, which can be exploited for botnet C&C. Figure 10b shows that a command propagation cycle typically encounters more than 70% of the same nodes every day at rush hour. These numbers would be much higher in a more closed setting like offices where the movement of employees is typical more restricted and as a result, the Bluetooth devices come in constant contact with each other for longer periods of time.

Our results also demonstrate that by carefully specifying the polling interval for the bots, a botmaster can balance the latency experienced by the botnet and stealthiness of the bots (Figure 10c). Keeping the interval to a lower value results in higher propagation rates: for a polling interval of 10 seconds, the command propagates to about 6500 bots, which constitutes about 36% of the total vulnerable machines. With an increased polling interval, the propagation rate drops substantially with only 14% of the bots receiving command for a polling interval of 1 minute; this value drops drastically to about 1% for 5 minutes. One reasoning behind such a drastic drop is the dynamic nature of the transportation system: individuals come in contact with each other for much shorter intervals of time (as compared to the academic environment represented by the MIT dataset). Therefore, for larger polling interval, higher number of victims are missed between subsequent polls.

The transport model and simulation results reiterate and reinforce the trace-based analysis discussed in the previous section. These results show that public networks fulfill the requirements and the premises underlying the creation of a mobile phone-based botnet using Bluetooth as its communication channel. They also demonstrate that command propagation numbers would be much higher if botmasters target such large-scale public networks.

Note that our observations are necessarily conservative. A more in depth approach could model the movement of people throughout a city throughout an entire day. Such a model would demonstrate that there would be other opportunities for messages to be passed. However, realistically modelling the movements of every person in a city is extremely difficult. By focusing on transportation hubs, we are able to demonstrate that these kinds of botnets are possible and simply note that improvements to the propagation of command and control messages can be expected given other repetitive daily interactions.

7 Defensive Strategies

The modeling and simulation in the previous sections have shown that mobile phone-based botnets can plausibly use Bluetooth as a communications channel. While increasing the latency of C&C messages, this approach significantly reduces the probability that defenders can observe or disrupt these networks. The use of traditional intervention techniques is unlikely to help protect cellular providers against such activity. Careful modifications that consider the patterns of interaction discussed in this paper, however, are likely to prove highly effective.

Like the move towards heterogeneity in platform architectures and operating systems, we argue that software patching mechanisms in this space will begin to mirror the desktop world. In particular, providers will likely be able to help push critical patches to devices. Such a mechanism would provide a number of benefits. For instance, whereas the majority of phone users have never installed software updates, the provider could help them do so with minimal interaction. Such improvements could not only reduce vulnerability to infection, but also improve the services available to users.

Our propagation analysis of mobile phone-based botnet behavior provides some key insights into developing effective remediation strategies. Our results show that botnet command propagation is at its peak when infected devices have a high probability of being in proximity of each other – weekday mornings. Bots are logically less likely to encounter each other later at night and during weekends. Achieving the widest propagation of commands, even if for a time delayed event (one launched later using the loosely synchronized clock provided by the network), therefore requires the botmaster to send messages at specific times of the day. If this brief time window is missed, bots will be forced to communicate with each other over the network.

Remediation can therefore be most successful if launched when bots are least able to coordinate and defend against such efforts. In particular, evenings and weekends provide the most significant periods in which bots are unlikely to communicate via Bluetooth. Such periods also represent the ideal time periods for providers to push updates. In particular, a high density of mobile users in a single location significantly limits the rate with which such updates can be disseminated – previous work has shown the limitations of cellular resources and how attempts to communicate with a large number of users in a single cell can accidentally [32] or intentionally [33] deny service. Accordingly, by pushing patches when customers are least likely to be using their phones and bots are least likely to be able to warn each other without using the provider's network, the provider can more effectively combat such botnets.

Detection mechanisms in this space also hold interesting possibilities. Whereas desktop systems generally rely on installed scanning tools (e.g., antivirus) or network-based IDSs to identify infection, mobile phones have the potential to leverage exciting new mechanisms. In particular, devices plugged into desktops to synchronize information can also be put through a more rigorous set of tests to determine whether or not malware is present. This process can include up-to-the-minute updates from the provider, which can help the more powerful desktop-based software tailor its investigation.

Knowledge of the limitations of a mobile phone-based botnet using Bluetooth helps to mitigate the advantages near-field communications affords in these systems. When used in conjunction with the proposed mitigation infrastructure, such systems are more combatable than initially supposed.

8 Conclusion

Mobile phones are becoming increasingly able to perform critical tasks. However, such devices are also becoming increasing susceptible to infection. Whereas a number of other researchers have investigated the characteristics of such infection, this paper instead attempts to determine whether such devices can be used to support a botnet. In

particular, this work tests whether or not such a botnet can effectively communicate using near-field communications to avoid traditional detection mechanisms. In this paper, we demonstrate that such a botnet is possible due to the largely repetitive mobility patterns found in human behavior. Over the course of a day, we show that commands can be consistently disseminated to over 66% of infected nodes. We then leverage such observations to develop more effective patching and mitigation strategies used to either isolate infected devices or force those whose infection is unknown to reveal themselves. While such botnets have not been observed yet in the wild, this work demonstrates that their eventual existence should be anticipated.

Acknowledgments

References

1. Grand Central Terminal. `http://en.wikipedia.org/wiki/Grand_Central_Terminal`. Last accessed Feb 2, 2010.
2. MTA NYC Transit: 2007 Ridership by Subway Station. `http://www.mta.info/nyct/facts/ridership/ridership_sub.htm`. Last accessed Feb 2, 2010.
3. New York City Subway Rolling Stock. `http://www.nycsubway.org/cars/index.html`. Last accessed Feb 2, 2010.
4. Rail commuters still bothered by cell phone abuse. `http://www.trainweb.org/ct/cellsurvey.htm`. Last accessed Feb 2, 2010.
5. Sinit P2P Trojan Analysis. `http://www.lurhq.com/sinit.html`. Last accessed Feb 2, 2010.
6. P. Barford and V. Yegneswaran. An Inside Look at Botnets. *Advances in Information Security*, 2006.
7. E. Cooke, F. Jahanian, and D. Mcpherson. The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets. In *Workshop on Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, Cambridge, MA, June 2005.
8. F.-S. Corporation. F-Secure Computer Virus Descriptions: Cabir, Dec. 2004. `http://www.f-secure.com/v-descs/cabir.shtml`. Last accessed Feb 2, 2010.
9. F.-S. Corporation. F-Secure Computer Virus Descriptions: Mabir.A, Apr. 2005. `http://www.f-secure.com/v-descs/mabir.shtml`. Last accessed Feb 2, 2010.
10. D. Dagon, G. Gu, C. Lee, and W. Lee. A Taxonomy of Botnet Structures. In *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC)*, Miami, FL, Dec. 2007.
11. R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-Generation Onion Router. In *Proceedings of the 13th USENIX Security Symposium (SECURITY)*, San Diego, CA, Aug. 2004.
12. N. Eagle and A. Pentland. Reality Mining: Sensing Complex Social Systems. *Journal of Personal and Ubiquitous Computing*, 2005.

13. B. Eckman. `http://lists.sans.org/pipermail/unisog/2006-April/026261.html`. Last accessed Feb 2, 2010.

14. P. Ferrie, P. Szor, R. Stanev, and R. Mouritzen. Security Response: SymbOS.Cabir. *Symantec Corporation*, 2007.

15. C. Fleizach, M. Liljenstam, P. Johansson, G. M. Voelker, and A. Mehes. Can You Infect Me Now?: Malware Propagation in Mobile Phone Networks. In *ACM Workshop on Recurring Malcode (WORM)*, Alexandria, Virginia, USA, Nov. 2007.

16. F. C. Freiling, T. Holz, and G. Wicherski. Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks. In *Proceedings of the 10th European Symposium on Research in Computer Security (ESORICS)*, Milan, Italy, Sept. 2005.

17. G. Gu, R. Perdisci, J. Zhang, and W. Lee. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection. In *Proceedings of the 17th USENIX Security Symposium (SECURITY)*, San Jose, CA, July 2008.

18. G. Gu, P. Porras, V. Yegneswaran, M. Fong, and W. Lee. BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation. In *Proceedings of the 16th USENIX Security Symposium (SECURITY)*, Boston, MA, Aug. 2007.

19. J. P. John, A. Moshchuk, S. D. Gribble, and A. Krishnamurthy. Studying Spamming Botnets Using Botlab. In *Proceedings of the 6th Symposium on Networked Systems Design and Implementation (NSDI)*, Boston, MA, Apr. 2009.

20. S. Jung, U. Lee, A. Chang, D.-K. Cho, and M. Gerla. BlueTorrent: Cooperative Content Sharing for Bluetooth Users. In *Proceedings of the 5th IEEE International Conference on Pervasive Computing and Communications (PerCom)*, White Plains, NY, Mar. 2007.

21. M. Lactaotao. Security Information: Virus Encyclopedia: Symbos_comwar.a: Technical Details. *Trend Micro Incorporated*, 2005.

22. C. Mulliner and G. Vigna. Vulnerability Analysis of MMS User Agents. In *Proceedings of the 22rd Annual Computer Security Applications Conference (ACSAC)*, Miami Beach, FL, Dec. 2006.

23. M. Nekovee. Worm Epidemics in Wireless Adhoc Networks. *Journal of Physics*, 9:189, 2007.

24. C. Pettey. Gartner Says Worldwide Mobile Gaming Revenue to Grow 50 Percent in 2007, June 2007. `http://www.gartner.com/it/page.jsp?id=507467`. Last accessed Feb 2, 2010.

25. N. Provos, P. Mavrommatis, M. A. Rajab, and F. Monrose. All Your iFRAMEs Point to Us. In *Proceedings of the 17th USENIX Security Symposium (SECURITY)*, San Jose, CA, July 2008.

26. M. Rajab, J. Zarfoss, F. Monrose, and A. Terzis. My Botnet is Bigger than Yours (Maybe, Better than Yours): Why Size Estimates Remain Challenging. In *Proceedings of the 1st Workshop on Hot Topics in Understanding Botnets (HotBots)*, Cambridge, MA, Apr. 2007.

27. A. Ramachandran and N. Feamster. Understanding the Network-Level Behavior of Spammers. In *Proceedings of the ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, Pisa, Italy, Sept. 2006.

28. C. J. Rhodes and M. Nekovee. The Opportunistic Transmission of Wireless Worms between Mobile Devices, 2008. `http://arxiv.org/abs/0802.2685`. Last accessed Feb 2, 2010.

29. Science Daily. Medical Records on Your Cell Phone: Computer Scientists Turn Cell Phones into Health Care Resource, 2006. `http://www.sciencedaily.com/videos/2006/0306-medical_records_on_your_cell_phone.htm`. Last accessed Feb 2, 2010.

30. V. Srinivasan, A. Natarajan, and M. Motani. CRAWDAD data set nus/bluetooth (v. 2007-09-03), Sept. 2007. `http://crawdad.cs.dartmouth.edu/nus/bluetooth`. Last accessed Feb 2, 2010.

31. J. Su, K. Chan, A. Miklas, K. Po, A. Akhavan, S. Saroiu, E. Lara, and A. Goel. A Preliminary Investigation of Worm Infections in a Bluetooth Environment. In *ACM Workshop on Recurring Malcode (WORM)*, Alexandria, VA, Nov. 2006.

32. P. Traynor. Characterizing the Limitations of Third-Party EAS Over Cellular Text Messaging Services. 3G Americas Whitepaper, 2008.

33. P. Traynor, W. Enck, P. McDaniel, and T. L. Porta. Exploiting Open Functionality in SMS-Capable Cellular Networks. In *Journal of Computer Security (JCS)*, 2008.

34. P. Traynor, M. Lin, M. Ongtang, V. Rao, T. Jaeger, T. La Porta, and P. McDaniel. On Cellular Botnets: Measuring the Impact of Malicious Devices on a Cellular Network Core. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, Chicago, IL, Nov. 2009.

35. R. Vamosi. Mobile phone malware in our future?, 2008. `http://news.cnet.com/8301-10789_3-10071982-57.html`. Last accessed Feb 2, 2010.

36. M. Waldman, A. D. Rubin, and L. F. Cranor. Publius: A Robust, Tamper-Evident, Censorship-Resistant Web Publishing System. In *Proceedings of the 9th USENIX Security Symposium (SECURITY)*, Denver, CO, Aug. 2000.

37. G. Yan and S. Eidenbenz. Bluetooth Worms: Models, Dynamics, and Defense Implications. In *Proceedings of the 22rd Annual Computer Security Applications Conference (ACSAC)*, Miami Beach, FL, Dec. 2006.

38. G. Yan and S. Eidenbenz. Modeling Propagation Dynamics of Bluetooth Worms. In *Proceedings of the 27th International Conference on Distributed Computing Systems (ICDCS)*, Toronto, Canada, June 2007.

39. G. Yan, H. D. Flores, L. Cuellar, N. Hengartner, S. Eidenbenz, and V. Vu. Bluetooth Worm Propagation: Mobility Pattern Matters! In *Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, Singapore, Mar. 2007.