BUILDING A DARK PICONET UPON BLUETOOTH INTERFACES OF COMPUTERS

Rodney Owens, Weichao Wang
Department of Software and Information Systems
University of North Carolina at Charlotte, NC USA
Email: rvowens@uncc.edu and weichaowang@uncc.edu

***Abstract*—*In this paper we demonstrate an attack scenario in which Bluetooth enabled computers are remotely controlled by an attacker without any security software detecting the connection. We describe in detail the methods to deliver malware, evade detection, elevate permissions, and transport critical information out of the network via Bluetooth connections. A prototype system using state-of-the-art operating systems and security software is built to show the practicability of the attack. We also study different mitigation strategies along with their downside. Security improvements for similar scenarios are also discussed.*

I. INTRODUCTION

With the fast development of pervasive computing applications, Bluetooth devices have quickly gained popularity because of several highly desirable properties. Their uniform interfaces remove any concerns of users about the compatibility issues. With the low power consumption (from $1mw$ to $100mw$) and decent transmission speed (up to 3 Mbits/sec for version 2.0) of the technique, Bluetooth promises to interconnect almost any mobile devices without hurting their battery lives. The technique has been embedded into many Personal Area Network (PAN) devices such as cellular phones, PDAs, and GPS. Presently, personal computer manufacturers are starting to include built-in bluetooth adapters into their portable computers. Dell's Latitude E6400 XFR, a portable computer designed specifically for military applications, has a "Dell Wireless 370 Bluetooth Module" as an optional add-on. The wide adoption of Bluetooth technologies has also attracted a lot of research efforts on its security. Many of these efforts focus on the authentication and data transmission procedures such as the pairing [1]–[3] and traffic relay [4] functions.

However, the vulnerabilities of Bluetooth interfaces on corporate or military computers and their impacts on system security have not been extensively studied and several factors contribute to this deficiency. First, Bluetooth interfaces have a short transmission range, which means an attacker would have to be physically close to a computer to connect to it. This reduces the probability of an external attack. Second, Bluetooth uses frequency hopping to improve data transmission robustness and it is not easy to monitor the communication channels. While there are tools such as NetStumbler [5] to listen for wireless access points, tools to listen for Bluetooth communications in a large range are scarce. Third, Bluetooth is embedded in so many electronic devices that the military would be hard pressed to try to audit all the traffic. Because of these reasons, many believe that Bluetooth is only a threat to personal data and does not pose serious threats to corporate or military information.

To demonstrate the impacts of Bluetooth vulnerabilities on the safety of a military network, in this paper we will illustrate an attack to establish a dark piconet upon Bluetooth interfaces of well protected computers and steal critical information from them. Before presenting the details of the attack and mitigation techniques, we first describe a scenario of the attack. During a Joint Warfighting operation, a group of fully rugged, Bluetooth enabled computers, such as GETAC V100, are setup to form the infantry battalion Command and Control (C2) system. An enemy solider can use a long range BlueSniper rifle to aim at the computers from a mile away. She/He will then be able to deliver malware to the computers, construct a dark piconet upon the Bluetooth interfaces, and steal the attack plan from them. The impacts will be much more severe than simply eavesdropping on the Bluetooth channels of the computers. The example in the paper will show that a malicious party can even take over the control of the computer through the dark piconet.

As a complete procedure of the attack, we will first show mechanisms to infect individual computers and penetrate local firewalls. To avoid detection by anti-virus systems, we propose to change the binary files of the malware to generate altered signatures. Mechanisms are designed to allow multiple infected computers with Bluetooth interfaces to form a piconet. A long range Bluetooth connection is used to link the attacker and the piconet to conduct subsequent attacks. During the whole procedure, we will avoid operations on the closely monitored network interfaces such as Ethernet or IEEE 802.11 adapters. The attack will conduct a series of configuration changes on the Bluetooth interfaces that are all normal operations.

As a concrete example, we implement a prototype of the

attacking system. The target computers are equipped with state-of-the-art operating systems and anti-virus software. We embed the malicious code into the device driver of an off-the-shelf USB-Bluetooth device so that the computers can be infected through supply chain attacks. A long range powerless Bluetooth transceiver is built to provide out-of-band connection between the infected computers and the attacker. We manipulate the executable file of a known malicious code so that the altered signature can avoid detection by anti-virus systems. With all these hardware and software tools, we demonstrate the establishment of the dark network and two subsequent attacks: data stealing and domain administrator compromise.

Considering the significant impacts of the investigated attack on computer system security, we study both hardware and software based countermeasures. The software based approaches focus on restricting the privileges of third party codes and monitoring non-traditional communication channels. The hardware based approach will allow manufactures to install an antenna switch to every Bluetooth device so that users can easily control its functionality. Analysis shows that many of these mechanisms will impact the usability of some applications and further research is needed. We also discuss mechanisms to generalize the proposed approach to security improvement strategies.

The remainder of the paper is organized as follows. In Section II, we review related work on Bluetooth technique and its safety. In Section III, we describe mechanisms to infect individual computers with malware and avoid detection by anti-virus systems. These computers can then form a localized network and transmit critical information to attackers. In Section IV, a concrete example of the attack on off-the-shelf systems is described. We investigate mechanisms to defend against such attacks in Section V. Finally, in Section VI we conclude the paper and discuss future extensions.

II. Related Work

Bluetooth Technique and Piconet

Bluetooth communication occurs in the unlicensed frequency of 2.4GHz. The transceiver utilizes frequency hopping to reduce interference and fading. The communication channel can support both data (asynchronous) and voice (synchronous) traffic. Bluetooth technology has enabled the establishment of a piconet [6]. A piconet is a short range (about 10 meters) star topology network of Bluetooth devices. It operates in a master-slave relationship, with a single computer as the master and up to seven devices as slaves. In addition to these seven slaves, a computer can have up to 255 "parked" slaves. Parked slaves are devices that do not actively communicate on the network but can be called upon by the master. To create a piconet, two devices need to authenticate each other with a PIN code. Once the authentication succeeds, one device may request to connect to the other via a virtual network adapter. Although a single hop Bluetooth network can cover only a limited area, investigators have developed various protocols to form multi-hop scatternets of these devices [7], [8].

Bluetooth Security

Considering the weak processing capabilities of many Bluetooth devices, the safety of the technique has attracted a lot of research efforts from the very beginning. The guide to Bluetooth security published by NIST [9] finds that the technique is susceptible to DoS attacks, eavesdropping, man-in-the-middle attacks [1], message modification, and resource misappropriation. For example, Jakobsson and Wetzel [10] discovered flaws in the pairing protocol and encryption scheme. The Bluebug attack [11] on cellular phones shows that stealing information and abusing connections are possible. Passive and active attacks on PIN based pairing have been investigated in [3], [12]. In [13], the authors investigate the security of the newly proposed Simple Pairing protocol that involves Diffie-Hellman-based key establishment and relies on a human visual channel for authentication. Self-installing and self-propagating worms [14], [15] such as Lasco.A on Bluetooth devices have also been designed.

The technique of Bluesniping [16] enables attackers to identify and communicate with Bluetooth devices from a long distance. In an experiment conducted in 2004, investigators successfully extended the range of Class 2 Bluetooth radios to more than 1 mile. The attack studied in this paper is beyond traffic sniffing. It will use the Bluetooth channel to compromise and take over the control of the computers.

III. Building a Dark Piconet upon Bluetooth Interfaces

In this section, we describe an attack to infect the Bluetooth interfaces of computers. These devices will then form a self-organized network and transfer critical information to attackers through a long range Bluetooth connection. While we use `netcat.exe` as our example in this paper, the proposed attack can be easily applied to other software packages.

A. Infecting a Single Computer

In this part, we discuss how publicly available executables can be used to install malicious binaries as system services, thus making them run with the privileges of `SYSTEM`. We further supplement this subsection with details on how to make these malicious binaries undetectable by anti-virus software.

The easiest way to gain complete control of a Microsoft Windows computer based on Windows NT is to create a backdoor that runs as a system service. Microsoft provides two executable files, namely `instsrv.exe` and `srvany.exe`, that when run together, can be used to install any third-party executable file as a system service. The syntax for these commands is as follows:

```
INSTSRV.EXE <service name> SRVANY.EXE

reg add ''HKLM\SYSTEM\CurrentControlSet
 \Services\<service name>\Parameters''

reg add ''HKLM\SYSTEM\CurrentControlSet
 \Services\<service name>\Parameters'' /v
 Application /d <malicious binary file>
```

These commands need either local administrator or domain administrator privileges to execute. While different methods such as Trojan can be used to inject the malicious system service, in this paper we plan to conduct a supply chain attack to trick users. We propose to embed the malicious code into a device driver. Therefore, when the device driver is installed, the new system service will be added. At the same time, we will also embed scripts into the driver so that if the infected computer is equipped with a Bluetooth interface, its PIN number will be configured to a pre-determined code. In our example attack, we extract a windows version of `netcat` [17] to run as the new system service so that a long range Bluetooth connection can be established to steal critical information.

Using a device driver also means that a particular population of users can be targeted, such as USB VOIP adapter drivers for telemarketing or serial data bus drivers for on-board data handling (OBDH) subsystems in military equipments. The most vulnerable of these would be corporate or government users because computer equipment is purchased in large quantities of a homogeneous type. Worse yet, military users are at greatest risk because they would be the target of terrorist funded computer hardware manufacturing. Should this malware be built into the drivers of any device targeted for military use, it would be able to gain control of the local computer and activate any disabled bluetooth interfaces.

When we try to add `netcat` as the system service, there are two difficulties that we need to overcome. First, we need to bypass the firewall configuration of the infected computer so that data transmission through the Bluetooth adapter will not be blocked or trigger an alert, while still allowing the firewall to block the malware on other network adapters. Second, since previous research has shown that `netcat` can be used to conduct malicious activities, we must alter the executable file so that it can avoid detection by anti-virus software. Below we describe schemes to solve these problems.

Since `netcat` can work as both initiator and responder of a connection, we need to adopt different methods to fool the local firewall system. If we have configured `netcat` to actively dial out to a preset IP address when it is installed, we do not need to make any changes to the firewall rules. The disadvantage of this method, however, is that we lose some configuration flexibility. At the same time, the persistent attempt of the Bluetooth adapter to establish a connection may attract the attention of the IDS system.

On the contrary, if we configure `netcat` to passively wait for a connection, we have more control over the time, duration, and type of the communication. We have to embed script into the malcode so that the firewall rules will be updated to allow the penetration. For example, the command for Windows Vista to achieve this goal is as follows:

```
netsh firewall set allowedprogram program=
  <malicious binary> name=''Windows Update''
  scope=custom addresses=<atker's BL adapter>
```

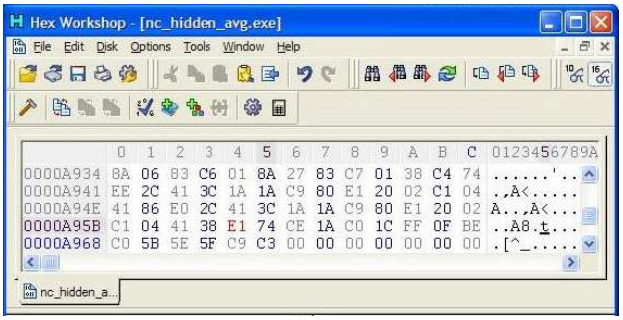
Here we name the firewall rule as "Windows Update" since most people will not touch this configuration in fear they break Windows' ability to update. Similarly, in Windows XP, this same firewall rule can be created by editing the registry keys under 'GloballyOpenPorts' and 'AuthorizedApplications'. Under this configuration, the open port is linked to the Bluetooth adapter. If the system administrator monitors only the Ethernet and WiFi connections, she/he will not identify any suspicious activities or unauthorized open ports.

The last problem that we need to solve during the infection of a single computer is the detection of our activities by security software. To thwart intrusion detection systems (IDS), we could make the malware search for known IDS binaries, terminate their execution, and even replace the IDS binaries with altered ones. Anti-virus software, however, is much more difficult to terminate without the user or the operating system detecting the abnormality. Fortunately, many anti-virus systems can detect only known malware based on their signatures. Therefore, several methods can be adopted to avoid detection.

Previous research has shown that altering the definition database of the anti-virus product can effectively restrict its detection capabilities. This method, however, may raise alarms from the anti-virus program. Another option we could take would be to use polymorphic techniques [18], [19] for computer worms, many of which can be reversed or detected by intelligent scanners [20], [21]. Since in this attack we are not seeking self-propagation of the malcode to a large number of computers, we propose to directly edit the executable file of the malicious code to change its signature while preserving the functionality. This approach is very straightforward and practical since many malicious

codes have reserved some blocks for such purposes.

To accomplish this task, we first install sample anti-virus programs on a test computer. We then slowly remove the unimportant bits from the malware, one hex digit at a time, until the anti-virus software can no longer detect it. We now know where a sensitive hex digit is for this signature and need to alter only this digit. We will also test the functionality of the malicious code to make sure that it still satisfies our requirements. If the expected functionality is not preserved, we can restart the procedure from another block.



(a)

(b)

Fig. 1. Change binary file of `netcat` to avoid detection by AVG. (a) Binary file manipulator. (b) Detection result of AVG.

Using `netcat` as an example, we successfully update its binary file to avoid detection by three popular anti-virus packages while preserving its basic functionality. In Figure 1, a binary form of `netcat` is modified. The changed value is highlighted in red on the Hex Workshop screen. By changing one byte from 'E0' to 'E1', we manipulate the signature of `netcat` and it can no longer be detected by AVG Anti-Virus.

B. Building a Piconet of Infected Computers

To reduce the chance that the connection between the infected computer and the attacker is detected, we propose to use a long range Bluetooth connection to link them. This method avoids those closely monitored channels such as Ethernet and WiFi adapters. However, it poses some difficulty to information transmission since the infected computer may not have a line-of-sight to the Bluesniping device of the attacker. To solve this problem, the malware will prepare the infected computers to form a piconet. Below we first discuss the configuration of the Bluetooth interfaces. Mechanisms to determine the master-slave relationship and build secure data transmission will then be introduced.

The latest versions of the Microsoft Windows operating system do not easily allow the SYSTEM service to pair the computer with Bluetooth devices since pairing requires user intervention. We go around this problem by scripting keyboard inputs into an executable file that communicates with the malware running as the SYSTEM service. Using this scheme, we are able to interact with the user's desktop and add a flag to the Bluetooth device's broadcast ID. We can pair the computer with other infected computers which possess the same flag. In the situation where a computer has two Bluetooth interfaces, the malware could be configured to add a second flag to represent a piconet ID code. These computers could then act as piconet "routers" to create a dark scatternet.

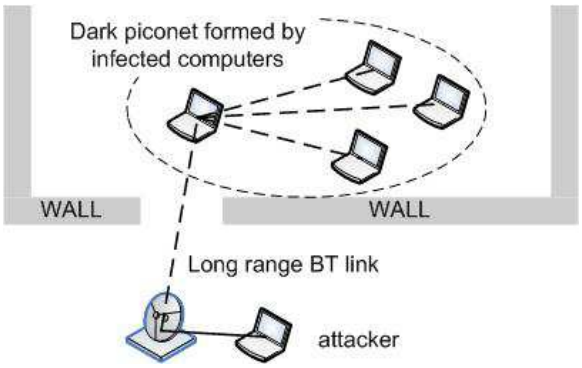


Fig. 2. The attack scenario.

The purpose of the dark piconet is to allow the attacker to steal critical information on multiple infected computers from a safe distance even when she/he has line-of-sight to only one of them. Bearing this in mind, we make the computer that has the strongest outside signal the master device. Since the infected computers do not have this information beforehand, the malware can further be created to search for a "master" flag, along with the "infection" flag, in the device name. When both flags are set, the malware "knows" that this computer can directly connect to the attacker and all other infected computers within the transmission range will try to pair with it. This "master" flag will be inserted by the attacker manually once we are able to connect to one infected computer.

Figure 2 illustrates the described scenario. Once we have

connected to the master and accepted the pairing requests from all other infected machines, we will park all of them. We will then connect to these slaves alternatively and send instructions to their SYSTEM command prompt.

IV. Experimental Results

To demonstrate the practicability of the attack, we have built a prototype system and conducted field tests. Below we first describe the system setup. Two examples of subsequent attacks on the infected computer systems are then described.

A. Experiment Setup

In our experiment environment, we install Microsoft Windows Vista Ultimate on two HP/Compaq 6715b laptops. One laptop is then attached to a Windows 2008 domain. Snort 2.8.3.2 and Symantec Antivirus (program version 10.2.0.276, scan engine 81.3.0.13) are installed on the computer to protect the domain member from our malware. We alter a binary version of `netcat` via the method described in Section III.A so that it cannot be detected by Symantec, AVG, and Kaspersky. We manipulate the device driver of a rocketfish RF-BCDM4 Bluetooth micro adapter so that the malware can be installed on the target machine through the setup procedure.

Fig. 3. Prototype system of the Bluesniping gun.

As illustrated in Figure 3, we use a parabolic dish to build a Bluesniping gun. This Bluetooth enabled device is built upon a Dish Network antenna and a Kensington Model K33348B Bluetooth adapter. It uses only the power provided by a USB interface of the attacker's computer.

We conduct the field test in a university campus. The victim laptop is placed close to a window on the third floor of a normal college building, while the attacker's computer and Bluesniping gun are located on the top level of a parking garage. The attacking computer and parabolic dish are located about 60 meters away from the building and they have line-of-sight to the window of the victim machine.

B. Examined Attacks

Stealing Information from Computers

The attacker can connect to the victim computer via its own copy of `netcat`. She/He can then list the currently running processes and terminate the Snort process. Symantec Antivirus poses no threat at this point because of the manipulation of the binary file. The attacker then maps a network drive back to her/his computer and proceeds to copy documents, pictures, and any files she/he wants from the victim. Should the attacker have thought a mapped drive would have been noticeable, she/he could simply have chosen to display the contents of sensitive text files on her/his display via `netcat`.

Compromising Domain Administrator Account

In a Microsoft Windows domain, individual domain member computers usually do not hold very valuable information. Most information is stored on either the domain controller or a domain file server. In order to pull the files she/he wants from these secured servers, the attacker needs to acquire the domain administrator's network credentials. Using the method described in Section III.A, she/he can modify `whosthere.exe` from the Pass-the-Hash toolkit [22] so that anti-virus software cannot detect it. By repeatedly forcing the currently logged in users to logout, the attacker will attract the domain administrator to login. She/He will then get the administrator's password hash with `whosthere.exe`. With this information, the attacker can upload a modified version of `msvctl.exe` and `msvctl.dll` [23] and create a session with the domain administrator's permissions. The attacker will be able to map a network drive from the domain controller to the victim's machine and proceed to copy any files she/he wants from the domain controller or any other domain members. Our experiment shows that the administrator's password hash will be valid for more than 24 hours. This leaves a period of time that is long enough for most subsequent attacks.

V. Mitigation Techniques

In this section we investigate techniques to defend against the attack described above. We will introduce software based and hardware based approaches respectively.

A. Software Based Mitigation Strategies

The software based mitigation strategies can be divided into two classes: disabling and monitoring. For the first class, a practical scheme is to prevent local administrator permissions from being given to non-network administrative users. In this way, the system service cannot be easily added into the computer. Another mitigation strategy is to prevent the malware from convincing the operating system that the user has confirmed the connections. This solution

can be achieved by disabling a program's ability to emulate keyboard and mouse movements. This approach, however, may degrade the usability of some applications. For example, the "On-Screen Keyboard" in Windows allows a user to emulate keyboard control by clicking on buttons on the screen. This program is especially useful when the computer's keyboard fails or when the user has limited control over her/his hands.

The system administrator can monitor different aspects of the computer to defend against the described attack. For example, if software can intelligently alert administrators to changes that they do not intentionally make to the system, the behaviors of many malicious codes can be detected. We can also make intrusion detection systems Bluetooth aware so that they can monitor connections and traffic on these channels. This scheme is effective as long as the attacker cannot easily disable this monitoring task even when she/he gains the SYSTEM privileges.

A network administrator could monitor changes to the list of computer services to prevent access to the SYSTEM user. This scheme, however, has its own problems. Software manufactures enjoy the ability to access the SYSTEM user. Should the Trojan be attached to a hardware driver, an administrator could be easily fooled into allowing a malicious system service to be installed. With the fast increase in the diversity of computer peripherals, most network administrators will not be able to keep up with all the system services installed.

B. Hardware Based Mitigation Techniques

The software based solution could not provide guaranteed security at this time since the SYSTEM user of Microsoft Windows computers has unlimited control of the computers and administrator user accounts. In the worst case, the SYSTEM user is capable of preventing any human controlled user accounts from logging into the computer. Therefore, hardware or hybrid based methods are expected.

A software/hardware hybrid method to prevent attackers from gaining remote access to Bluetooth devices would be to include a software antenna kill switch. This software controlled switch on Bluetooth adapters can be turned off via a software signal, but must be turned on by physically pushing a small button on the device. If the software signal could be sent by a network device, then all Bluetooth devices on the network could be deactivated on a certain interval. Users who use Bluetooth can be educated on the advantage of this requirement and learn to push the "reset" button before attempting to use a Bluetooth device. A similar software approach called BT-Guard that can change the Bluetooth status automatically according to users' settings has been designed for cellular phones. A disadvantage of this approach still lies in the ability of malware to disable the proper reception of a software kill signal directed to a computer's Bluetooth adapter. At the same time, if attackers can impersonate the system administrator to issue a fake 'kill' signal, DoS attacks on Bluetooth devices can be conducted.

C. Generalization of the Lesson

While the attack investigated in this paper is a specific example, the lesson that we learn can be generalized. Since most security mechanisms are built upon some assumptions, identifying and evaluating those assumptions in security-sensitive scenarios will allow us to have a better understanding of the system safety. For example, people always assume that the hardware components of computers will deliver the promised functionality. This assumption, however, will be voided under supply chain attacks. For example, in 2008 the FBI seized an estimated $3.5 million worth of fake cisco network devices that were sold to the US Navy, Marine Corps, and Air Force. If the manufactures had embedded some backdoors into the hardware, they could have disabled the network connections of these equipments by sending a single packet.

In addition to monitoring and disabling, we can also thwart the threats by hiding the real hardware/software configurations of the systems. This goal can be achieved through diversifying the platforms [24], [25] or virtualization of the resources [26], [27]. Mechanisms to integrate these techniques to form a comprehensive security system will be studied in future work.

VI. Summary and future work

In this paper we demonstrate a possible attack scenario in which Bluetooth enabled computers are remotely controlled by an attacker without any security software detecting the connection. We describe, in detail, the methods of delivering malware, evading detection, elevating permissions, and transporting information out of the network via Bluetooth connections. A prototype system on state-of-the-art operating systems and security software is built to show the practicability of the proposed attack. We also study different mitigation strategies along with their downside.

Immediate extensions to our investigation consist of the following aspects. First, we will study more deeply into the mitigation strategies, along with the testing of prototyped software and hardware solutions for preventing this dark piconet from becoming a danger to military networks. Second, this investigation shows that some components in computer systems that are assumed to be hard to attack may become unconventional targets of malicious activities. We will study the security of these components so that we can provide a more comprehensive protection to the computer systems.

ACKNOWLEDGEMENT

The authors would like to thank UNCC Faculty Research Grants (FRG) for their support. The authors would like to thank Bill Harden and Chris Turner of Yadkinville, NC for their help in constructing the directional bluetooth antenna.

REFERENCES

[1] K. Haataja and K. Hypponen, "Man-in-the-middle attacks on bluetooth: a comparative analysis, a novel attack, and countermeasures," in *International Symposium on Communications, Control and Signal Processing (ISCCSP)*, 2008, pp. 1096–1102.

[2] L. Nguyen, R. Safavi-Naini, W. Susilo, and T. Wyosocki, "Secure authorization, access control and data integrity in bluetooth," in *IEEE International Conference on Networks (ICON)*, 2002, pp. 428–433.

[3] Y. Shaked and A. Wool, "Cracking the bluetooth pin," in *MobiSys '05: Proceedings of the 3rd international conference on Mobile systems, applications, and services*, 2005, pp. 39–50.

[4] A. Levi, E. Cetintas, M. Aydos, C. K. Koc, and M. U. Caglayan, "Relay attacks on bluetooth authentication and solutions," in *International Symposium on Computer and Information Sciences (ISCIS)*, 2004, pp. 278–288.

[5] C. Hurley, M. Puchol, R. Rogers, and F. Thornton, *WarDriving: Drive, Detect, Defend, A Guide to Wireless Security*. Syngress, 2004.

[6] R. Morrow, *Bluetooth: Operation and Use*. McGraw-Hill Professional, 2002.

[7] W.-Z. Song, Y. Wang, C. Ren, C. Wu, and X.-Y. Li, "Multi-hop scatternet formation and routing for large scale bluetooth networks," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 4, no. 5, pp. 251–268, 2009.

[8] C.-M. Yu and C.-C. HUANG, "On the architecture design and performance evaluation of a configurable blueweb network," *IEICE Trans B: Communications*, vol. E90-B, no. 5, pp. 1104–1111, 2007.

[9] K. Scarfone and J. Padgette, "Guide to bluetooth security," National Institute of Standards and Technology (NIST) Special Publication SP800-121, September 2008.

[10] M. Jakobsson and S. Wetzel, "Security weaknesses in bluetooth," in *CT-RSA 2001: Proceedings of the 2001 Conference on Topics in Cryptology*, 2001, pp. 176–191.

[11] J. Wagstaff, "Bluetooth may put you at risk of getting 'snarfed'," The Wall Street Journal, April 15th, 2004.

[12] F. Wong, F. Stajano, and J. Clulow, "Repairing the bluetooth pairing protocol," in *Proceedings of 13th International Workshop on Security Protocols*, 2005.

[13] R. Chang and V. Shmatikov, "Formal analysis of authentication in bluetooth device pairing," Tech report of Univ. Texas, FCS-ARSPA, Tech. Rep., 2007.

[14] J. Su, K. K. W. Chan, A. G. Miklas, K. Po, A. Akhavan, S. Saroiu, E. de Lara, and A. Goel, "A preliminary investigation of worm infections in a bluetooth environment," in *WORM '06: Proceedings of the 4th ACM workshop on Recurring malcode*, 2006, pp. 9–16.

[15] G. Yan and S. Eidenbenz, "Bluetooth worms: Models, dynamics, and defense implications," in *Annual Computer Security Applications Conference (ACSAC)*, 2006, pp. 245–256.

[16] P. McFedries, "Bluetooth cavities," *IEEE Spectrum*, vol. 42, no. 6, p. 88, 2005.

[17] G. Giacobbi, "The gnu netcat project," software available at http://netcat.sourceforge.net/, 2006.

[18] D. Spinellis, "Reliable identification of bounded-length viruses is np-complete," *IEEE Transactions on Information Theory*, vol. 49, no. 1, pp. 280–284, 2003.

[19] O. Kolesnikov and W. Lee, "Advanced polymorphic worms: Evading ids by blending in with normal traffic," Georgia Institute of Technology CC Technical Report GIT-CC-05-09, Tech. Rep., 2005.

[20] C. Kruegel, E. Kirda, D. Mutz, W. Robertson, and G. Vigna, "Polymorphic Worm Detection Using Structural Information of Executables," in *Proceedings of the International Symposium on Recent Advances in Intrusion Detection (RAID)*, 2005, pp. 207–226.

[21] J. Wang, l. Hamadeh, G. Kesidis, and D. J. Miller, "Polymorphic worm detection and defense: system design, experimental methodology, and data resources," in *LSAD '06: Proceedings of the 2006 SIGCOMM workshop on Large-scale attack defense*, 2006, pp. 169–176.

[22] H. Ochoa, "Pass-the-hash toolkit - docs and info," available at http://oss.coresecurity.com/pshtoolkit/doc/, 2009.

[23] J. Gumbel, "msvctl toolkit," available at http://www.truesec.com/PublicStore/Profile/Downloads.aspx?versionid=12, 2008.

[24] A. Alarifi and W. Du, "Diversify sensor nodes to improve resilience against node compromise," in *Proceedings of the ACM workshop on Security of ad hoc and sensor networks*, 2006, pp. 101–112.

[25] Y. Yang, S. Zhu, and G. Cao, "Improving sensor network immunity under worm attacks: a software diversity approach," in *Proceedings of the ACM international symposium on Mobile ad hoc networking and computing*, 2008, pp. 149–158.

[26] X. Jiang and D. Xu, "Collapsar: a vm-based architecture for network attack detention center," in *SSYM'04: Proceedings of the 13th conference on USENIX Security Symposium*, 2004.

[27] S. King and P. Chen, "Subvirt: implementing malware with virtual machines," in *IEEE Symposium on Security and Privacy*, 2006.