Warkitting: the Drive-by Subversion of Wireless Home Routers

Alex Tsow Markus Jakobsson
School of Informatics
Indiana University
Bloomington, IN 47408
{atsow, markus}@indiana.edu

Liu Yang Susanne Wetzel
Department of Computer Science
Stevens Institute of Technology
Hoboken, NJ 07030
{lyang, swetzel}@cs.stevens.edu

Abstract

In this paper we introduce the notion of *warkitting* as the drive-by subversion of wireless home routers through unauthorized access by mobile WiFi clients. We describe how such attacks can be performed, evaluate the vulnerability of currently deployed wireless routers based on experimental data, and examine the impact of these attacks on Internet fraud. Our analysis shows that it is possible in practice to carry out warkitting attacks with low cost equipment widely available today and that the volume of credential theft possible through warkitting exceeds current estimates of credential theft due to phishing. We discuss how to detect a warkitting attack in progress and show how to analyze warkitted routers for evidence linking it to the attackers.

Keywords: Warkit, WAPkit, WAPjack, wireless home routers, phishing, pharming, Internet fraud, intrusion detection

1 Introduction

The convenience of wireless networking has led to widespread adoption of wireless access points in home and small business settings. While some wireless access points are administered by security conscious users, several recent studies show that a large number of wireless access points run with default settings. The default settings from the most established vendors are optimized for ease of setup instead of secure access. Most commonly, the default settings disable wireless encryption, permit wireless access to router administration, and guard administration with published passwords. With some studies reporting as many as 50% of popular wireless routers deployed on default settings [1], ease of setup coincides with mass vulnerability.

Until recently, the perceived risk of wireless routers has centered around unauthorized network and bandwidth use. However, as we illustrate in this paper, the risks are far greater. An open router is a gateway for eavesdropping, redirection to fraudulent websites, and traffic profiling. These capabilities grant the attacker nearly total control over how the network's clients interact with the Internet.

Our first contribution in this paper is to identify a new type of attack on wireless home routers. We call this attack *warkitting* which combines the notions of *wardriving* and *rootkits*. While the phrase may suggest the drive-by subversion of the firmware kernel only, we use it more generally to refer to any malicious alteration to the wireless access point's configuration or firmware over the wireless connection. Warkitting attacks allow for many malicious activities. For example, a compromised router may redirect a legitimate website login request to a malicious server carrying out a man-in-the-middle attack [2]. Consequently, a successful login will disclose the client's login credentials to the man-in-the-middle server. While other security weaknesses (e.g., buffer overflow) may pose a threat to a wireless router, an open administrative web interface is far more dangerous and can be more easily exploited by an attacker.

In order to carry out a warkitting attack, the attacker first discovers vulnerable wireless routers through wardriving (i.e., the discovery and mapping of wireless access points) or by retrieving the necessary data from existing WiFi access point databases such as WiGLE [3] or WiFiMaps [4]. Then, the attacker will engage in one of two types of malicious behavior. In the first type, the attacker subverts the router firmware of the Wireless Access Point (WAP); we refer to this type of abuse as *WAPkitting*. WAPkitting allows the attacker to completely control the actions of subverted routers within the limits of their hardware specification. In the second type of malicious behavior, the attacker engages in malicious configuration of firmware settings, but makes no modification to the firmware itself. This type of attack has the same effect as DNS poisoning attacks [5], i.e., allow connections to be hijacked and rerouted without the user's knowledge. We refer to this type of WAP abuse as *WAPjacking*. WAPjacking is less powerful than WAPkitting in that it does not challenge the existing control software but maliciously alters its configuration. We make the distinction between the three terms as follows: WAPkitting and WAPjacking are independent of the means of infection, and specify the relative modifications done to a WAP upon corruption. Warkitting, on the other hand, does not specify the type of WAP alteration, but does relate to how infection occurs.

A second contribution of this paper is to evaluate the extent to which currently deployed wireless routers are vulnerable to warkitting. In our tests, we use specific techniques to detect the models of open routers without compromising any of their private information. Our tests indicate that 10% of wireless routers are susceptible to WAPjacking and 4.4% of wireless routers are vulnerable to WAPkitting. In addition, our analysis shows that the volume of credential theft possible through warkitting exceeds the current estimates of credential theft due to phishing.

Our third contribution is to propose methods to detect a warkitting attack in progress and to analyze warkitted routers for evidence to identify attackers. In particular, we propose to identify WAPjacked or WAPkitted routers through direct firmware analysis and external behavior analysis. We furthermore discuss various techniques to determine both the malicious attacking devices as well as details on the compromise itself (e.g., time and method).

Outline. The remainder of the paper is organized as follows: Section 2 discusses related work. Section 3 defines two different classes of router compromise and how an attacker may execute them on a large scale. Section 4 presents our experimental data on the vulnerability of current wireless networks. Section 5 discusses the impact of large scale warkitting on Internet fraud and compares it to known phishing statistics. Section 6 provides a discussion on how to identify and secure forensic evidence in case of a compromised router. Section 7 concludes the article and points to future work.

2 Related Work

Studies. As the number of wireless clients increases, so does demand for wireless access points. Wardriving, or WiFi mapping, is a peer-driven activity that discovers wireless networks and records them by GPS coordinates. Technical details of WiFi mapping can be found in [6, 7]. WiGLE [3] distributes software for WiFi mapping and also maintains a database of observed wireless networks. Since their founding in 2001, they have collected over 6 million entries in their database that are corroborated by over 300 million observations. WiFiMaps [4] is another high profile wardriving database. Started in 2003, the database contains information on over 400,000 access points. For each wireless access point, the data collection includes geographic location, MAC address, SSID, broadcast channel, and encryption method.

Shah and Sandvig [1] conducted a wireless study that leverages the WiFiMaps database to illustrate the regulatory effects of software default settings. Of the three settings—use of encryption, broadcast channel, and SSID—they found that among owners of Linksys home wireless routers 50.3% changed none of the default settings, 24.8% changed one, 18.6% changed two, and only 6.3% changed all three default settings. We interpret the study as a general result on default settings, not just on the three settings that they evaluate. Although they did not test for passwords, we infer that they also have high rates of default usage. This study is an empirical followup to Shah and Kesan's earlier theoretical paper on the properties of computer code that shape public policy [8], which identifies default settings as one way to control how people use computer systems.

Hottell, Carter, and Deniszczuk [9] conducted another vulnerability study. Instead of relying on existing databases, they gathered their own data through wardriving in order to avoid stale data. They hypothesized that router security is correlated with income, education, and population density. However, their study of nearly 2,500 access points discovered no significant correlation with these demographic factors, but instead found that usability had the largest impact on security settings.

Kuo, Goh, and Tang [10] conducted a usability study on wireless network configurations. They first interviewed human subjects to determine their expertise and then observed their interactions with one of three wireless router setup procedures: one from the Linksys WRT54G, one from the Netgear WGT624, and their study's prototype which configures a modified Linksys WRT54G. The prototype's setup interface used a goal-oriented wizard that automated as much of the technical setup as possible. They found that the differences in the configuration interfaces did not significantly impact high expertise users. Among those users with low expertise, however, the prototype interface outperformed the commercial interfaces by a factor of 2 to 3 in the number of security mechanisms being configured successfully.

A similar study by Stephano and Groth affirmed that goal-oriented setup interfaces lead to a more secure setup of a wireless router [11]. Their study also showed that for one of the vendors, 3 of 11 subjects failed to change the administrative password when using the respective setup interface. In this case, the setup program always displays fifteen placeholder characters in the password box to obfuscate the actual length of the current password. In particular, this applies to the initial setup where the current password is set to the empty string by default. Subjects who did not replace the default password said that they did not want to "break the router" by changing a setting that appeared to have already been set.

Embedded Systems and Firmware. Today, third-party firmware is available for many embedded systems. For example, the LinuxBIOS project provides a firmware replacement for the power-on bootstrapping subsystem found on many PC motherboards [12]. In many cases, third-party firmware provides more features, has better performance and provides a user with more control over the system.

The popular Linksys WRT54G originally used a version of Linux as its embedded operating system (though the most recent versions use VxWorks), whose intellectual property license mandates Linksys to make all modifications to the open-source distribution of this system public. In turn, this has spurred the development of many third-party firmware distributions for these routers: OpenWRT [13], DD-WRT [14], Sveasoft [15], WiFi-Box [16], HyperWRT [17], eWRT [18], Freifunk [19], Rupan [20], and Tofu [21].

Since embedded software is field-rewritable in many cases, it is also vulnerable to malicious alteration. One of the first mass attacks on firmware was the Chernobyl virus in 1999 [22]. This destructive virus targeted desktop PCs, erasing their hard disks and overwriting their BIOS at specified dates. Researchers have proposed several approaches to firmware protection. Adelstein, Stillerman and Kozen [23] identified non-destructive malware in *Open Firmware* boot platforms as a threat and proposed a code analyzer which checks for malicious code at load time and prevents flagged code from running. Arbaugh, Farber, and Smith [24] implemented a cryptographic access control system, AEGIS, to ensure that only sanctioned bootstrapping firmware can be installed on the host platform.

Tsow identified a fraudulent attack where compromised consumer electronics can be resold as benign devices through the online marketplace [2]. As a principal example, he showed how to subvert the firmware on a home wireless router by replacing it with a third-party open-source package, OpenWRT [13], and configuring its internal DNS server to selectively resolve domain names in a fraudulent manner. The attack pushed clients to phony websites, yet retained correct URLs, setting them up for credential theft.

3 Warkitting

As noted earlier, we use the phrase *warkitting* to generally refer to attacks in which the wireless access point's configuration or firmware is modified over the wireless connection. While WAP-jacking targets a wireless router's configuration settings, WAPkitting compromises the firmware itself.

3.1 WAPjacking

WAPjacking changes the settings of existing firmware to bring some benefit to the attacker. While configurable parameters vary among router models and manufacturers, most routers destined for the home market allow users to select a DNS server, enable administrative access via the Internet, log usage statistics, send usage reports to an email address, and control the traffic routing.

Changing the DNS server to one controlled by the attacker is one way to quietly misdirect legitimate link requests to fraudulent hosts. DNS poisoning, or *pharming* [5], is more efficient than phishing in bringing victims to phony websites because clients navigate to the sites by their own initiative, not at the behest of an email. Pharming attacks withstand the scrutiny of many anti-phishing tools because their analysis assumes correct domain name resolution. Stealing authentication credentials such as username and password becomes a trivial task. For example, when `www.mybank.com` is requested by the client's web browser, the wireless router asks the malicious DNS server for the IP address, and receives the address of a duplicate host that is controlled by the attacker. The victim's login results in the disclosure of website credentials; most commonly this means username and password, however man-in-the-middle attacks will discover personalized data in more elaborate login processes such as PassMark's SiteKey [25][1]. Alternatively, the malicious DNS could profile traffic. Instead of fraudulently resolving a domain name, it could simply log all DNS requests. The log is an aggregate profile of which domains their clients visit, and when they visit them. For home routers, the space of clients is small, so the loss of precision due to aggregation is limited. In conjunction with a known email address, this behavioral profiling makes spear phishing [26, 27] much easier. This sort of attack could also be employed by stalkers or burglars to help identify when their victims are home.

Some wireless access points allow direct control over their routing tables. This is a gateway for local man-in-the-middle attacks. The default route could be set to a computer with unauthorized access to the local wireless network. This computer could log and duplicate traffic, then forward it to an arbitrary computer on the Internet. This malicious client need not be the attacker sitting in a car full of antennas, but could simply be a neighbor's subverted computer.

Many routers log usage statistics, including MAC addresses of connected clients. Whether spread over a wide area or tailored to the hotspots of an individual, this attack compromises location privacy. The ability to track the MAC address (a unique identifier) tells the attacker when an individual is at home, at work, or at a favored public hotspot.

Enabling Internet administration of router settings gives the attacker an opportunity to dynamically change the configuration details. Subverted home routers need to work in conjunction with external malicious hosts, such as fraudulent DNS servers. Large networks of compromised Internet hosts controlled by individual operators, known as *botnets* [28], have commoditized the hosting of malicious agents. For instance, the attacker can rent botnet nodes to host fraudulent DNS servers. As botnet nodes are discovered and shut down, the DNS settings of the compromised routers can be altered over the Internet to reference other bots in the network. Thus, dynamic alteration of administrative settings gives the attacker's support network a degree of takedown resistance.

Ultimately, all WAPjacking modifications are detectable by the router's owner. If the owner has reason to audit the router configuration, all of these malicious settings can be revealed upon inspection after administrative login. If administrative login is not possible because the attacker has changed the password, most access points have hardware reset buttons that return settings to their factory defaults. However, user studies indicate that average users are reluctant to interfere with a system that appears to be working correctly [11]. Consequently, while easily detectable in principle, a WAPjacked router will likely go undetected due to practical matters.

Although WAPjacking is clearly detectable by external analysis, its space of vulnerable routers is very general. A router's underlying implementation does not matter because the attacker never tries to subvert its control. Wireless access points implemented with embedded software, special purpose hardware, or trusted computing platforms are all vulnerable as long as administration is possible through wireless clients and the necessary authentication credentials are known.

3.2 WAPkitting

In a WAPkitting attack, external software seizes control from the router's firmware. While most easily accomplished by exploiting open administrative access, WAPkitting can theoretically proceed by more traditional means such as buffer overflow [29]. Almost all wireless home routers are based on integrated systems on chip (SoC) which combine a general purpose microprocessor, RAM, a small amount of permanent storage, a wireless network interface, and a wired network switch. An embedded operating system coordinates the execution of the firewall, routing, domain name resolution, and other services. Like a desktop computer, these operating systems are vulnerable to malware, including total subversion. The web administration page on most home wireless routers is not only a configuration interface, but also a firmware replacement interface. None of the widely deployed systems known to us limit the upgrades to digitally signed firmware. While this measure would prevent upgrade interfaces from replacing legitimate firmware with WAPkits (since they are not signed by the manufacturer), it would not seal vulnerabilities due to software errors.

The ability to install arbitrary control software on a wireless router opens unlimited possibilities to an attacker. Once installed, the router has the ability to tell the user one thing while doing another. To disguise its presence, a complete firmware compromise could emulate the manufacturer web administration interface, forward the critical settings to a malicious agent, and selectively ignore commands. Moreover, the hardware reset button on these devices only clears the NVRAM, a small section of the memory that stores the settings registry for factory firmware but not necessarily the malicious firmware. The malicious software can detect NVRAM resets and behave accordingly. Above all, it will never let the user re-flash the firmware, although it may appear to do so.

Since firmware compromises give attackers complete control, more attacks are possible. Generally speaking, the WAPkitted router could perform any kind of man-in-the-middle attack. Many online login pages are vulnerable to a particular kind of man-in-the-middle attack called *SSL double-posting*, or *trawl phishing* [30]. A potential target's website presents its login forms over an unencrypted *http* session. Normally, when the username and password are submitted, a javascript launches an encrypted SSL session to post this data to the server. In the SSL double-post attack, the WAPkitted router modifies the HTML login page (it can do this since the login page is sent in cleartext) to post login credentials to both the legitimate server *and* a covert server controlled by the attacker.

Another man-in-the-middle attack is called *race pharming* [31]. Rather than spoofing sensitive websites by maliciously resolving DNS queries, this technique uses a race condition in TCP handshakes to hijack sessions with legitimate servers. In particular, a WAPkitted wireless router could use race pharming to perform man-in-the-middle attacks on clients that are part of a different, but nearby, wireless network.

The widespread availability of third-party open-source firmware makes some models of wireless routers particularly vulnerable. Attackers can leverage these legitimate software efforts to become platforms for malicious firmware development. In terms of time investment, open-source firmware provides WAPkit developers with the path of least resistance. The development tools and package management systems aid rapid malware construction. Of course, open-source firmware is not a prerequisite, but a convenient facilitator, for WAPkit production.

Maintaining closed-source firmware mitigates the urgency of the warkitting threat, but does not eliminate it. *Closed-source* is much weaker than *secret*; it confers the developer with legal recourse for misuse, but does not protect the source from unauthorized access. In the most powerful set of assumptions, we assume the attacker can find out how underlying hardware and firmware work on *every* platform. This parallels Kerckhoff's principle in cryptography where security must not rest on the secrecy of an algorithm, protocol, or implementation instance.

A determined adversary may circumvent attempts to keep firmware code secret by exploiting a router's JTAG (Joint Test Action Group) interface [32]. JTAG is the primary debugging and testing interface for many embedded systems. Many wireless routers ship with functional JTAG interfaces intact, although access usually requires router disassembly and soldering a cable directly to its printed circuit board. While establishing the physical connection is a delicate procedure, JTAG offers the ability to read and write the contents of a router's internal persistent storage. Indeed, this is how the first wave of third-party firmware replacements were installed on wireless routers. It is also how third-party hobbyists have circumvented limits imposed by closed-source firmware. Most recently, the boot loader on the Linksys WRT54Gv5, which uses the closed-source VxWorks embedded operating system, has been reversed engineered to allow straightforward third-party firmware replacement. Prior to this advance, the WRT54Gv5 had been considered a dead-end for enthusiasts wanting to customize their router. While JTAG manipulation clearly requires more access than a mobile attacker can get, it provides a platform for reverse-engineering existing firmware and detailed testing of malicious firmware alterations.

3.3 Warkitting Equipment and Strategies

Warkitting uses wardriving equipment to discover open wireless networks. This equipment is inexpensive. A basic setup uses a computer with wireless network interface. Network detection software is free [33, 34]. A mobile attacker may extend his connection range by purchasing or constructing a specialized antenna. For the purposes of avoiding suspicion, the attacker will script network detection and subversion; this way the attacker minimally engages his computer. He could roam with his laptop concealed in a backpack as it subverts vulnerable routers.

Since the attacker need not interact with the computer during an expedition, he could adapt an inexpensive wireless router to detect and subvert victims using open-source firmware. A suitable wireless router has a wireless interface, a good antenna, and a small but adequately powerful general purpose computer. Some models even include USB ports that can connect to external storage devices. Inexpensive USB flash drives can host extra subversion software if the router's internal flash memory is too confined. The total cost for this equipment was less than $100 in the summer of 2006. For example, the Asus WL-500g has 16MB of RAM, 4MB of flash memory, and a USB port for around $75. For an organized operation, a dozen people could warkit a city for less than $1000 in equipment costs.

In the interest of efficiently harvesting routers, the attacker may first execute a WAPjacking subversion that opens the router for Internet administration, and then follow up with a firmware replacement via the Internet. Firmware flashing takes one to two minutes per router, while enabling Internet administration takes less than ten seconds (there is some variance due to load). In general, the number of traveling attackers is limited and they will want to spend the smallest necessary amount of time to subvert each router. This WAPjack-then-WAPkit strategy displaces the time intensive portion to an army of botnet nodes that can perform firmware flashes in parallel over the Internet.

4 Test Results

Our vulnerability assessment has two basic goals. The first one is to determine the distribution of router models susceptible to WAPkitting versus those only susceptible to WAPjacking. The second one is to estimate the density of routers with open wireless administrative access.

Technically, evaluating the vulnerability of routers would be straightforward if there were no ethical or legal considerations when interacting with other people's property: One would first simply attempt an administrative login to the routers using a small dictionary of default passwords and record the success rate of gaining access. Upon successful administrative login, it is possible to retrieve information on the model of the router, version of the firmware, and what (if any) changes have been made to the default settings of the configuration. In case the communication in the wireless network is protected through encryption[2], the encryption key must be recovered before attempting an administrative login. If the wireless network filters clients based on their MAC address, conducting the experiment then requires spoofing the MAC address of a connected or recently connected client. Packet control information such as the MAC address is never encrypted—even in networks where the communication is protected by encryption.

Our experiment takes a more constrained approach. We limit our scans to the properties broadcast by wireless networks. Prior studies whose data collection methodologies have been considered non-invasive [1, 9] have recorded four properties of wireless access points: *MAC address*, *encryption setting*, *SSID*, and *broadcast channel*. These studies have collected some of their data using NetStumbler [34], a wireless network detection package. Netstumbler interacts with networks at the data-link layer [35] broadcasting requests for networks to voluntarily identify themselves according to the ANSI/IEEE 802.11 specification [36]. The legal limits of acceptable interaction with private wireless networks at higher levels, including the network and transport layer, remains unclear. Thus, we limit our vulnerability analysis to the four properties above.

We postulate that routers which use default settings for SSID, encryption, and broadcast channel are likely to also use the default web administration password. Prior studies on default settings and usability lend partial support to this claim [1, 8, 11], however, the extent to which this holds is unknown. An unpublished study of a small set of unauthorized wireless routers in a privately owned and operated network found that more than 60% of the routers with default settings for SSID, encryption, and broadcast channel also used a default administrative password.[3] The small size and context of the data set precludes confident generalization. Nevertheless, we will use this result as a starting point for our analysis.

Default administrative password usage exposes wireless networks to WAPjacking, but not necessarily WAPkitting. A wireless access point is at high risk for WAPkitting if it is supported by third-party open-source firmware replacements and if the firmware upgrade interface is accessible through the local wireless network. We estimate the rate of vulnerability to WAPkitting by inferring the manufacturer *and* model of wireless access points from their MAC addresses. Every wireless access point broadcasts its uniquely assigned 6 byte MAC address. The first three bytes of the default MAC address, known as the *Organizational Unique Identifier* (OUI) [37], are assigned by the IEEE and identify the manufacturer. The device's manufacturer uniquely assigns the remaining three bytes in an unspecified manner. While a MAC address determines a wireless access point's model and version because of its uniqueness, the mapping is not public knowledge.

We have collected a small database of 197 unique (MAC address, model) pairs. We assume that MAC addresses are assigned sequentially by manufacturers, and that a match on the first four bytes of the MAC address accurately determines the model of the router. For example, if the collected data identifies the MAC address 00:00:00:11:11:11 as model "A" (where 00:00:00 is the IEEE-assigned manufacturer identifier, and 11:11:11 is the manufacturer-assigned portion), then our mapping assumes that MAC addresses of the form 00:00:00:11:XX:XX also correspond to model "A". With high probability this assumption is conservative since it leaves only two bytes, or 65,536 unique assignments, for each four byte prefix. In particular, our assumption that the first four bytes identify the model did not contradict any of our 197 pairs.

In our experiment we detect wireless networks using the four properties above using only a commodity laptop computer and PCMCIA WiFi network interface. The subject area measures roughly 1000 meters by 120 meters spanning three adjacent parallel streets for a length of seven blocks in the greater New York Metropolitan area. We use this data and our MAC address mapping to examine model and manufacturer distributions. Table 1 summarizes our results. Over the course of one hour we detected 790 unique wireless networks. The four largest manufacturers make up 77% of our collected data. Linksys was by far the largest with 44.1%. The model determination procedure described above identified the models of 138 routers among the 790. Generalizing the model distribution of the 138 identified routers to the entire sample, we estimate that 267 wireless routers are either Linksys WRT54G or WRT54GS, which are well-supported by third-party open-source firmware.

The experiment revealed 132 routers (16.7%) that broadcast default settings for all three recorded properties. These are the most likely ones to also be using the default administrative password. Taking the 60% rate of default password usage as a baseline,[3] we estimate that 79 of these routers are vulnerable to quick WAPjacking (which takes less than ten seconds) through the web administration interface. Moreover, our experiment identified 69 Linksys WRT54G or WRT54GS routers of which 33 are using default settings for all three properties.

Manufacturer	Man. Total (% of Total)	Model Name	Model Count (% of Man. Total)	Default Count	TPOSF
Linksys	348 (44.1%)	WRT54G	51 (14.7%)	26	Yes
		WRT54GS	18 (5.2%)	7	Yes
		BEFW11S4	12 (3.4%)	6	-
		WRT54GX	4 (1.1%)	-	Limited
		WRK54G	3 (0.9%)	-	-
		BEFSR11	1 (0.3%)	-	-
		WRT54GC	1 (0.3%)	-	-
		Not Identified	258 (74.1%)	36	-
Netgear	120 (15.2%)	WGR614	26 (21.7%)	7	Limited
		WGT624	6 (5%)	4	Limited
		MR814	2 (1.7%)	-	-
		WPN824	1 (0.8%)	-	Limited
		Not Identified	85 (70.8%)	12	-
Apple	74 (9.4%)	None Identified		6	-
Dlink	66 (8.44%)	DI-524	6 (9.0%)	1	Limited
		DI-614+	2 (3.0%)	1	-
		DI-808HV	2 (3.0%)	-	-
		DI-514	1 (1.5%)	1	-
		DI-624	1 (1.5%)	1	-
		WBR-1310	1 (1.5%)	1	-
		Not Identified	53 (80.3%)	8	-
Belkin	45 (5.7%)	None Identified		7	-
AboCom	32 (4.1%)	None Identified		-	-
Cisco	22 (2.8%)	None Identified		-	-
Microsoft	16 (2.0%)	None Identified		1	-
EpiGram	15 (1.9%)	None Identified		-	-
GemTeck	7 (0.9%)	None Identified		4	-
Buffalo	4 (0.5%)	None Identified		-	-
Delta	3 (0.4%)	None Identified		-	-
Others	38 (4.8%)	None Identified		3	-

Table 1: This table shows detected wireless networks in a 0.12 km^2 area of the greater New York Metropolitan Area Using an incomplete MAC-address-to-model mapping, we have identified 138 routers. The mapping does *not* identify all MAC addresses of a given model, so the set of unidentified routers certainly contains more instances of the listed models. The two largest sets of identified models are the Linksys WRT54G and WRT54GS, well supported by third party open source firmware (*TPOSF*). By applying the model distribution in the fourth column of the 138 identified routers, we estimate that 267 wireless routers are either Linksys WRT54G or WRT54GS. Also, more than 240 routers are expected to have "Limited" TPOSF support. A total of 132 (16.7%) wireless routers use default settings for all detected factors, the highest possible risk category in our analysis for subversion by unauthorized administrative access.

Among the 138 routers with identified models, 43 of them have limited third-party open-source firmware support. If applying this rate to the 790 routers, more than 240 routers are expected to have limited third-party open-source firmware support. *Limited* support means that some versions of the model are not supported, that some features are unstable, or that the installation procedure is complex. While imperfect, limited open-source support still gives attackers an advantage in developing WAPkits.

5 Impact

This section examines the impact of warkitting on Internet fraud. Today, phishing is perceived as one of the biggest threats with respect to Internet fraud. In the following we show that warkitting attacks may be even more dangerous. In part, this is due to the fact that WAPkitting and WAP-jacking circumvent the need to bait victims with emails by quietly altering the firmware or configuration of wireless routers. Consequently, these new attacks render conventional anti-phishing wisdom—never click on a link but instead enter the web address you intend to reach into your browser—useless.

In order to evaluate the danger of these new attacks, we estimate the number of credential thefts that can occur based on our experimental data and compare our results to statistics on credential theft due to phishing.

5.1 Threat of Warkitting

The warkitting attack can be carried out most efficiently when WAPjacking a vulnerable router. It then takes less than ten seconds to replace the administrative password, change the trusted DNS host to a malicious server, and enable Internet administration. If limited only by the speed of processing vulnerable routers, an attacker can then WAPjack 360 routers per hour. Real world factors such as intermittent connections at network boundaries may slow down progress dramatically. The attacker needs to remain in the communication range of the wireless router for the duration of the attack. Consequently, in areas of high network density an attacker may need to stay for an extended period of time. On the other hand, the performance of the WAPjacking attack can be improved considerably by parallelizing the attack using multiple router-based devices.

As stated in Section 4, in our experiment we discovered 790 unique wireless routers in one hour where 132 routers were found to use default settings for all three measurements. Moreover, 79 of those wireless routers are expected to be vulnerable to WAPjacking using the default administrative password. Consequently, detecting and WAPjacking the 79 vulnerable routers will take approximately 74 minutes. Thus, in one person month (i.e., 40 hours for 4 weeks, at a speed of covering an area of 0.12 km^2 per hour) it is possible to WAPjack at least 10,248 routers with one attacking device assuming a network density similar to the one in our experiments. If we assume that on average a router serves two different users who have sensitive credentials for five different accounts each, then each WAPjacked router reveals ten sets of credentials resulting in a total of 102,480 disclosed credentials.

The WiGLE database [3] lists 84,991 wireless networks in Manhattan in August 2006. Based on our experimental data, we estimate that 8,516 wireless routers are vulnerable to WAPjacking. With a size of 87.5 km^2 and a land area of 68%, Manhattan is 496 times larger than the area covered in our experiment. Assuming a uniform distribution of wireless routers yields a lower density than in our experiment. Thus, it is possible to completely WAPjack all vulnerable wireless routers in Manhattan in less than 4 person months using one attacking device.

5.2 Threat of Phishing

Estimating the threat posed by phishing will help to understand the actual danger of warkitting. In the following we provide estimates by generalizing statistics on phishing from two different sources [38, 39]:

The first source is the April 2005 Pew Internet & American Life Project phone survey [38]. It discovered that of the 2,201 randomly selected adults in the United States 1,295 were email users and 35% of them reported that they had received phishing emails in the past year. Among those receiving phishing emails, 2% admitted that they had responded to the solicitations and had provided the requested information. Generalizing these numbers to the U.S. population (estimated at 300 million in August 2006) suggests a rate of just over 100,000 phishing victims per month:

$$1295/2201 \times 0.35 \times 0.02 \times 300 \text{ million people } \div 12 \approx 102,965 \text{ victims}$$

The second source is the monthly numerical assessments by the Anti-Phishing Working Group (APWG) [40] recording the number of unique phishing scams over the past month. While they recorded 20,109 unique phishing attacks in May 2006 [41], there generally is limited evidence on the effectiveness of phishing attacks. One experiment that used high levels of contextual information finds a victim rate as high as 19% [42]. Because of this attack's highly focused victim selection and context, we cannot generalize its victim rates to the average attack found in the wild. However, some insights can be obtained from a social networking-based phishing experiment [39] which exposed a control group of 94 human subjects to a limited context phishing message resulting in a 16% yield. Because the phishing messages were mailed from trusted servers, they were not subject to spam filtering. Assuming that on average a unique phishing message is sent to 100,000 email users and that spam filtering is 99% effective [43, 44] on each batch of phishing messages, this will on average result in 160 victims per attack. Combining this estimate with the recorded number of phishing attacks by the APWG yields an estimate of 3.2 million phishing victims in May 2006 worldwide.[4]

5.3 Discussion

One may assume that a single phishing victim corresponds to the disclosure of one set of credentials. Then, comparing the estimates on the volume of credential theft due to warkitting with that of phishing clearly demonstrates the increased danger of warkitting attacks over phishing attacks.

In particular, the number of credentials expected to be disclosed through warkitting in Manhattan roughly match the number of credentials expected to be stolen through phishing in all of the U.S. Furthermore, the number of credentials stolen through phishing worldwide is only thirty times larger than the number of credentials disclosed through warkitting in Manhattan. Estimating the number of phishing victims at 3.2 million in May 2006 is likely to be an overstatement since 80% of phishing scams generally attack less than ten commercial brands (e.g., financial institutions) [40]. In addition, the hit rate tends to decrease when users receive an increasing number of phishing emails.

One should also note that the massive corruption of routers may serve other purposes than to steal credentials. Namely, such an attack could be mounted as part of an extortion attack—not against the owners of the routers, but of financial institutions or online service providers who might fear the existence of a tremendous botnet. Notice here that the router owners may have very low incentives to clean their devices of malware if there is no perceived threat to them. Furthermore, attacks of the type we describe may be used as a distribution network of other types of malware, to be propagated to machines connecting to affected routers. The fact that the routers can turn off any updates of resident anti-virus software of such machines makes this type of attack particularly threatening. Finally, one should note that router malware may be used as part of a politically motivated attack, i.e., plainly for doing damage to the infrastructure or to lower consumer trust in the same.

6 Forensic Analysis

An investigation that involves potentially subverted wireless routers needs to (a) positively identify WAPjacked or WAPkitted routers and the scope of its malicious activity, (b) determine the device's external collaborators, and (c) determine the details of the subversion attempt, including time and means of compromise. These questions are not independent and there could be trade-offs to make when evaluating them all. Successful evaluation also depends on data gathered from prior analysis of similar intrusions. For instance, a passive "fingerprint" library that identifies common open-source firmware-based routers by properties of their packet construction [45] would help investigators make decisions in the field, rather than in the laboratory, about the likelihood of compromise *and* consequently on the appropriate method for data capture.

6.1 Identifying Subverted Wireless Home Routers

There are two general approaches to identifying WAPkitting and WAPjacking attacks: direct firmware analysis and external behavioral analysis.

Direct firmware analysis extracts an image of the router's persistent storage to determine the presence of WAPkitted firmware and WAPjacked settings. The comparison to factory default images and previously captured WAPkits will conclusively determine the presence of either malicious firmware or malicious settings. Moreover, if a previously unknown WAPkit is discovered, investigators can use a combination of control flow analysis and controlled laboratory simulations to profile its behavioral characteristics. Scanning tools such as *nmap* [46] or its passive counterpart *p0f* [47, 45] analyze the packet construction and traffic timing (among many other attributes) to identify network nodes, including hardware, operating system, and running applications. Investigators can add the results of their firmware analysis to fingerprint libraries focused on detecting WAPkits.

Once the decision has been made to directly analyze the firmware, investigators should disrupt the WAPkit's control of the suspected device by disconnecting power. Interfering with a device's behavior while under WAPkit control could result in destruction of evidence; the WAPkit could detect analysis attempts and erase the contents of its flash memory. Control should not be returned to the router firmware unless subsequent analysis shows that the router is no longer compromised. Thus, the firmware extraction method should not rely on the firmware itself. The JTAG interface [32] (see Section 3.2) is an ideal tool to extract flash memory contents, including the NVRAM, operating system kernel, applications, and filesystem, without ever engaging the WAPkit. While manufacturers will be able to provide the most effective JTAG manipulation software, there also may be the possibility to use open-source software which exists for some of the Linksys routers [48]. In either case, a special cable connects the device's motherboard to a host PC running the extraction software. Engaging the JTAG extraction software immediately after power-on stops the boot-up phase and saves the entire contents of the flash memory to the PC.

The process for firmware extraction is time-consuming and requires investigators to interfere with the behavior of subverted routers. The takedown of many suspicious networks in a small time window could signal to the malicious network's supporting hosts (e.g., fraudulent DNS servers and web hosts) that their discovery is imminent. When investigators need to identify subverted wireless access points without interfering with their Internet-visible behavior, they can resort to external behavioral analysis. As mentioned above, characteristics of low-level packet construction and timing can identify implementation details about a particular node. The most sophisticated attackers will do their best to match default firmware implementation fingerprints, so this kind of evaluation may have limited use.

Another type of analysis, intrusion detection, identifies malicious or suspicious activity rather than implementation details. Most network intrusion detection systems (NIDS) focus on *signature* [49, 50] or sometimes *anomaly* detection [51] by analyzing traffic on a single network interface. Signature-based NIDSs analyze packet content using fixed strings or limited pattern languages such as variants of regular expressions. Anomaly-based NIDSs compare traffic against "typical" behavior. The difficulty in modeling typical behavior leads to high false positive rates and has hindered adoption of anomaly-based NIDSs. Since these systems are designed for high bandwidth communications ($>$1 Gb/s), their analysis must be fast.

Home wireless routers are vastly simpler and slower than the high-speed routers that comprise the Internet. Their routing decisions are limited to broadcasting on the local area network and forwarding traffic to its wide area network (WAN) uplink. Thus, tracking a connection is trivial since traffic only proceeds over one interface. Moreover, WAN bandwidth is typically limited to 1-6 Mbps download and 0.5-3 Mbps for upload. In this situation, the low bandwidth and trivial routing make anomaly-based intrusion detection that *compares traffic on both the LAN and the WAN* technologically feasible. To our knowledge, there are no existing systems for detecting transformation of network traffic through nodes.

A commodity PC with three network interfaces—a wireless interface to record LAN traffic, an

Ethernet card to monitor WAN traffic, and a third interface to cross-check external queries, such as DNS—should suffice for passively auditing a suspected router. Packet logging devices that record both incoming and outgoing traffic will discover discrepancies in DNS responses, socket request IP addresses, and downloaded data, as well as unexpected connections with the router that are not initiated by any wireless client. These systems will allow investigators to audit suspicious devices in the field without disrupting their service.

6.2 Determining a Wireless Router's Role in Malicious Networks

Compromised wireless routers act in conjunction with other machines to commit Internet fraud, eavesdrop, and covertly disclose findings to those who exploit them. Records of connected clients, DNS caches, DNS IP addresses, and email addresses help to identify victim clients, bogus websites, DNS servers, and other collaborators. Code analysis and controlled firmware testing on virtual machines can inform investigators about the methods that were used to contact foreign hosts, and the information transmitted between the two.

Some data of interest is part of the router's filesystem, particularly in the case of WAPjacking. In this case, all of the malicious settings are stored in the system's flash memory. Details about where the information is stored and how much is exposed through the administration interface varies by manufacturer. When extracting the firmware, the entire filesystem should be mined. Files that were deleted (e.g., as a result of clearing logs) could still exist on a WAPjacked router [52]. Filesystem analyzers will recover this data since standard deletion does not overwrite prior contents but merely marks it as available for overwriting.

Important and transient information such as recently connected clients and the local DNS cache may reside only in the router's RAM. Disconnecting power will cause this data to be lost. There may be bugs in the system that allow an investigator to subvert the control software and extract the contents of RAM, however, such exploits are hard to discover. In early versions of the Linksys WRT54G firmware, there is an administrative web-scripting vulnerability that allows clients to execute system commands with root access [53].

While the data storage location in WAPjacked routers is determined by manufacturer settings, sophisticated WAPkits will try to store the sensitive network information of the attack (as opposed to the sensitive user information) in RAM only. This concealment strategy has limits since investigators can fully extract the firmware image. The WAPkit must contact, or be contacted by, an external agent to implant those sensitive values. Simply encrypting the filesystem cannot protect WAPkit data since this requires the internal storage of the decryption key. Investigators will discover the key and decrypt the sensitive values during their direct firmware analysis. The router's communication with covert external agents is difficult to hide. The external host may try to authenticate the router but investigators can easily corrupt this process by learning the authentication secrets from the extracted filesystem. Investigators can even run captured WAPkits on virtual machines to learn the contents of the communication with the external host.

Running recovered firmware on virtual machines that emulate router hardware gives observers direct access to the internal control and data structures of the firmware (particularly those in RAM). In addition to providing insight to a WAPkit's internal control and data structures, virtual machines let investigators interact with external hosts in real time. In particular, a virtual machine that runs the WAPkit grants clear access to all data transmitted to and from external hosts—even in the presence of encryption. The drawback of virtual machines is that they need to be deployed in a controlled setting; one cannot use a virtual machine to extract memory resident data from a WAPkit-controlled device in the field.

Finally, the passive traffic auditor described in the previous section will capture DNS queries to fraudulent servers, connections to fraudulent websites, and communication with other third-party hosts. In case of pharming, investigators can mine cheating DNS servers by sending them lookup queries. This can identify fraudulent web pages *before* they are detected by users. However, sophisticated servers can counter this by resolving names to fraudulent host addresses only when queried by certain IP addresses. Thus, mining a cheating DNS server requires simulation of undetected operation including usage of the victim's IP address.

Investigators can discover data about fraudulent websites by looking through the web browser histories of clients recently connected to the router. Fraudulent websites are unlikely to use an encrypted *https* connection, as is good practice when displaying login pages, since web browsers require presentation of a verified SSL certificate. Examination of browser caches reveals whether or not https was utilized. This will help identify if, and when, clients have disclosed login credentials to a fraudulent source. If a fraudulent website offers an unverified SSL certificate, the web browser alerts the user to this situation by querying him for certificate acceptance. If accepted, the browser may store it in a file of user-accepted certificates. This file can also help investigators to determine occurrences of fraudulent login.

6.3 Discovering the Time and Method of Attack

When a compromised router is discovered and investigators need more information to place the time of attack, the logs of nearby wireless access points, particularly the ones that withstood compromise, may indicate the time and path of a warkitting expedition. Records often indicate failed administrative login attempts and MAC addresses of recently connected clients. However, few home users enable logging on their wireless networks.

If parties with a particular interest in preventing warkitting, including private institutions, homeowners associations, and law enforcement, do not want to depend on the log files of nearby wireless routers, they can deploy passive wireless sensors that simply record all visible traffic. When spread over a wide area, real time analysis can detect warkitting activity in progress. The warkitting traffic patterns are easy to characterize as repeated administrative login attempts for a wide variety of different routers. Regular timing of failed login attempts and uniform actions following successful logins provide further evidence of warkitting by scripting. Just as the attacker can use inexpensive routers to execute subversion scripts, defenders can deploy inexpensive routers to behave as passive sensors. The passive wireless network detector Kismet [33] (which is recognized as official package by the OpenWRT [13]) is well-suited for this task. Once configured, these routers will forward all visible traffic to a system that records and analyzes it for anomalous behavior. While promising from an intrusion detection point of view, passively and promiscuously collecting network traffic is a serious privacy concern for clients.

Honeypots are systems designed to attract and analyze attacks by deliberately emulating vulnerable systems [54]. The "vulnerable" systems are closely monitored virtual machines that record the inner workings of the attack. This strategy avoids many privacy issues since they only report traffic on their own networks. Since all honeypot access is unauthorized, they only record attack data. Honeypots that emulate vulnerable wireless networks can be implemented with a single computer running multiple wireless interfaces and virtual machine processes. Such a honeypot notifies authorities about unauthorized administrative access. Notification need not to result in immediate apprehension. Instead, it may simply mark a time and place to examine surveillance camera video or other sources of identification. For example, in New York City, a profitable warkitting target due to its wealth and large number of wireless networks, one also finds an extensive system of surveillance cameras. Geographically distributed collections of honeypots will give investigators multiple sources for cross-referencing. The more often "persons of interest" are spotted in a locality, the more likely they are to be complicit in the attack. Honeypots could further employ strategies to slow warkitting progress by simulating new networks as old ones are apparently compromised and simulating weak network signals at inopportune moments.

7 Conclusion

We have described and analyzed a threat to the routing infrastructure that could have a substantial security impact on society unless it is addressed. Namely, wireless reconfiguration and malicious firmware upgrading of consumer routers may be used as a tool for identity fraud, client surveillance, extortion, and more. Our experimental evidence shows that a large number of routers are vulnerable to both kinds of subversion. The impact of the new threat exceeds current fears about vulnerabilities in wireless networking which are primarily focused on unauthorized network use and bandwidth stealing. When used for identity theft, the efforts of a small warkitting organization in a densely populated metropolitan area can produce a volume of stolen credentials that rivals national and global estimates on the number of phishing victims.

This paper concludes with a call for tool and countermeasure development, using the strategies of the previous section as a starting point. Since third-party open-source firmware replacements represent the most expedient platform for WAPkit development, investigators can leverage existing scanning tools and begin building fingerprint libraries to identify the popular distributions; this should identify the first generation of WAPkits. The next priority is to build a system that identifies malicious behavior by comparing the input and output traffic of consumer wireless routers. This auditing system will identify more compromised routers in the field than fingerprint scanning because it classifies network behavior rather than implementation. Finally, the deployment of honeypots or wireless networks that passively monitor traffic will detect warkitting expeditions in progress. These systems will rely on virtual machines to emulate consumer wireless router hardware and anomaly-based intrusion detection systems to analyze wide area traffic collection.

8 Acknowledgments

We would like to thank several people for their time and effort in making this article possible: Kun Qian for her assistance in conducting the experiment, Jared Cordasco for his help with proofreading, Zhuowei Li and Camilo Viecco for their perspectives on intrusion detection technology, Jonathan Baumann for his insights on wireless vulnerabilities, the anonymous reviewers for their content direction, and Judie Mulholland for her attentive and patient guidance during the editing process.

Notes

[1]This login method sends a personalized image in response to a particular username. The goal is to authenticate the server to the user prior to password entry. However, a man-in-the-middle attacker defeats this method with bidirectional data forwarding between client and server.

[2]According to [55], 85% of wireless routers currently deployed in Seattle rely on the WEP security mechanism (Wireless Equivalent Privacy) which is known to exhibit serious vulnerabilities [56, 57].

[3]For more information, please contact the authors by email.

[4]This estimate is a worldwide figure since it is not possible to place geographic boundaries on the recipients of phishing messages.

References

[1] Rajiv Shah and Christian Sandvig. Software defaults as de facto regulation: The case of wireless aps. In *The 33rd Research Conference on Communication, Information and Internet Policy*, Arlington, Virginia, USA, September 2005.

[2] Alex Tsow. Phishing with consumer electronics: Malicious home routers. In *Models of Trust for the Web, a workshop at the 15th International World Wide Web Conference (WWW2006)*, Edinburgh, Scotland, May 2006.

[3] WIGLE.NET. `http://www.wigle.net`, Retrieved in August 2006.

[4] WiFiMaps.com. `http://www.wifimaps.com`, Retrieved in April 2006.

[5] Minaxi Gupta. *Phishing and Countermeasures: Understanding the increasing problem of electronic identity theft*, chapter Pharming and Client Side Attacks. Wiley, 2006.

[6] Simon Byers and Dave Kormann. 802.11b access point mapping. *Communications of the ACM*, 46(5):41–46, 2003.

[7] Hendrik Speck and Niko Bender. Mapping wireless networks - SSID WLAN(D)SCAPE. In *Advances in Wired and Wireless Communication, 2004 IEEE/Sarnoff Symposium*, pages 43–47, Apr 2004.

[8] Rajiv C. Shah and Jay P. Kesan. Manipulating the governance characteristics of code. *Info*, 5(4):3–9, 2003.

[9] Matthew Hottell, Drew Carter, and Matthew Deniszczuk. Predictors of home-based wireless security. In *The Fifth Workshop on the Economics of Information Security*, 2006.

[10] Cynthia Kuo, Vincent Goh, and Adrian Tang. Design and evaluation method for secure 802.22 network configuration (poster). In *The 2005 Symposium On Usable Privacy and Security*, 2005.

[11] Amanda Stephano and Dennis P. Groth. USEable security: Interface design strategies for improving security. In *3rd International Workshop on Visualization for Computer Security*, Fairfax County, Virginia, USA, Nov 2006.

[12] Welcome to LinuxBIOS. `http://www.linuxbios.org`, Retrieved in April 2006.

[13] OpenWrtDocs. `http://wiki.openwrt.org/OpenWrtDocs`, Retrieved in April 2006.

[14] DDWRT. `http://www.dd-wrt.com/dd-wrtv2/index.php`, Retrieved in April 2006.

[15] SVEASOFT. `http://www.sveasoft.com/modules/phpBB2/viewtopic.php?t=16132`, Retrieved in April 2006.

[16] WIFI-BOX - WRT54G(s) GPL firmware. `http://sourceforge.net/projects/wifi-box/`, Retrieved in April 2006.

[17] HyperWRT. `http://www.hyperwrt.org`, Retrieved in April 2006.

[18] Ewrt - open-source WRT54G/GS firmware. `http://www.portless.net/menu/ewrt/`, Retrieved in April 2006.

[19] Freifunk firmware (English). `http://freifunk.net/wiki/FreifunkFirmwareEnglish`, Retrieved in April 2006.

[20] `http://bagupremier.free.fr/wrt/G/`, Retrieved in April 2006.

[21] HyperWRT +Tofu firmware. `http://www.polarcloud.com/tofu`, Retrieved in April 2006.

[22] CERT. Incident note IN-99-03. `http://www.cert.org/incident_notes/IN-99-03.html`, April 1999.

[23] Frank Adelstein, Dexter Kozen, and Matt Stillerman. Malicious code detection for open firmware. In *Proc. 18th Computer Security Applications Conf. (ACSAC'02)*, pages 403–412, December 2002.

[24] William A. Arbaugh, David J. Farber, and Jonathan M. Smith. A secure and reliable bootstrap architecture. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, page 65, Washington, DC, USA, 1997. IEEE Computer Society.

[25] BankOfAmerica. `http://www.bankofamerica.com`, Retrieved in August 2006.

[26] Markus Jakobsson. Modeling and preventing phishing attacks. In *Phishing Panel in Financial Cryptography '05*, 2005.

[27] The IBM global business security index report in 2005. Technical report, IBM, 2005.

[28] David Dagon, Guofei Gu, Cliff Zou, Julian Grizzard, Sanjeev Dwivedi, Wenke Lee, and Richard Lipton. A taxonomy of botnets. `http://www.math.tulane.edu/~tcsem/botnets/ndss_botax.pdf`, Retrieved in August 2006.

[29] Jonathan Pincus and Brandon Baker. Beyond stack smashing: Recent advances in exploiting buffer overruns. In *IEEE Security and Privacy*, Apr 2004.

[30] Steven A. Myers and Abhinav Acharya. Dynamic trawl phishing attacks with wireless home routers (in progress).

[31] Mark Meiss. *Phishing and Countermeasures: Understanding the increasing problem of electronic identity theft*, chapter Case Study: Race Pharming. Wiley, 2006.

[32] IEEE Standards Association. IEEE std 1149.1-1990 IEEE standard test access port and boundary-scan architecture -description. `http://standards.ieee.org/reading/ieee/std_public/description/testtech/1149.1-1990_desc.html`, Retrieved in August 2006.

[33] KISMET. `http://www.kismetwireless.net`, Retrieved in April 2006.

[34] NETSTUMBLER.COM. `http://netstumbler.com`, Retrieved in April 2006.

[35] Joshua Wright. Layer 2 analysis of WLAN discovery applications for intrusion detection. `http://home.jwu.edu/jwright/papers/l2-wlan-ids.pdf`, Nov 2002.

[36] IEEE 802.11 Working Group. *ANSI/IEEE Standard, 802.11. "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 1999.

[37] IEEE Standard Association. IEEE OUI and company_id assignments. `http://standards.ieee.org/regauth/oui/index.shtml`, Retrieved in April 2006.

[38] Deborah Fallows. Pew Internet & American Life Project. `http://www.pewinternet.org/pdfs/PIP_Spam_Ap05.pdf`, Retrieved in April 2006.

[39] Tom Jagatic, Nathaniel Johnson, Markus Jakobsson, and Filippo Menczer. Social phishing. *to appear in the Communications of the ACM*.

[40] APWG. Anti-phishing reports. `http://www.antiphishing.org/reports/`, Retrieved in August 2006.

[41] APWG. Phishing activity trends report. Technical report, Anti-Phishing Working Group, May 2006.

[42] Markus Jakobsson and Jacob Ratkiewicz. Designing ethical phishing experiments: A study of (rot13) ronl auction query features. In *WWW'06*, 2006.

[43] William S. Yerazunis. Sparse binary polynomial hashing and the CRM114 discriminator. In *Proc. MIT Spam Conference*, Jan 2003.

[44] Paul Graham. A plan for spam. `http://www.paulgraham.com/spam.html`, Aug 2002.

[45] Michal Zalewski. *Silence on the Wire: a Field Guide to Passive Reconnaissance and Indirect Attacks*. No Starch Press, 2005.

[46] Nmap - free security scanner for network exploration & security audits. `http://insecure.org/nmap/`, Retrieved in August 2006.

[47] The new p0f: 2.0.7 (2006-08-10). `http://lcamtuf.coredump.cx/p0f.shtml`, Aug 2006.

[48] HairyDairyMaid. WRT54G EJTAG DeBrick guide. `http://downloads.openwrt.org/utils/`, Retrieved in August 2006.

[49] `http://www.snort.com`, Retrieved in August 2006.

[50] Bro intrusion detection system - bro overview. `http://bro-ids.org`, Retrieved in August 2006.

[51] Stefan Axelsson. Research in intrusion-detection systems: A survey. Technical Report 98–17, Department of Computer Engineering, Chalmers University of Technology, Dec 1998.

[52] Giovanni Di Crescenzo, Niels Ferguson, Russell Impagliazzo, and Markus Jakobsson. How to forget a secret. In *STACS 99: 16th Annual Symposium on Theoretical Aspects of Computer Science*, 1999.

[53] Execute arbitrary code through the Ping.asp interface. `http://www.seattlewireless.net/LinksysWrt54g`, Retrieved in August 2006.

[54] David Dagon, Xinzhou Qin, Guofei Gu, Wenke Lee, Julian Grizzard, John Levine, and Henry Owen. HoneyStat: Local worm detection using honeypots. In *The 7th International Symposium on Recent Advances in Intrusion Detection (RAID 2004)*, Sophia Antipolis, France, Sep 2004.

[55] Andrea Bittau, Mark Handley, and Joshua Lackey. The final nail in WEP's coffin. In *The 2006 IEEE Symposium on Security and Privacy SP'06*, 2006.

[56] Nikita Borisov, Ian Goldberg, and David Wagner. Intercepting mobile communications: the insecurity of 802.11. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 180–189, New York, NY, USA, 2001. ACM Press.

[57] Scott R. Fluhrer, Itsik Mantin, and Adi Shamir. Weaknesses in the key scheduling algorithm of rc4. In *SAC '01: Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography*, pages 1–24, London, UK, 2001. Springer-Verlag.