Phishing with Consumer Electronics – Malicious Home Routers

Alex Tsow
School of Informatics
Indiana University
atsow@indiana.edu

ABSTRACT

This short paper describes an attack that exploits the online marketplace's susceptibility to covert fraud, opaqueness of embedded software, and social engineering to hijack account access and ultimately steal money. The attacker introduces a fatal security flaw into a trusted embedded system (e.g. computer motherboard, network interface card, network router, cell phone), distributes it through the online marketplace at a plausible bargain, and then exploits the security flaw to steal information. Unlike conventional fraud, consumer risk far exceeds the price of the good.

As proof of concept, the firmware on a wireless home router is replaced by an open source embedded operating system. Once installed, its DNS server is reconfigured to selectively spoof domain resolution. This instance of malicious embedded software is discussed in depth, including implementation details, attack extensions, and countermeasures.

1. INTRODUCTION

Phishing attacks combine technology and social engineering to gain access to restricted information. The most common phishing attacks today send mass email directing the victim to a web site of some perceived authority. These web sites typically spoof online banks, government agencies, electronic payment firms, and virtual marketplaces. The fraudulent web page collects information from the victim under the guise of "authentication," "security," or "account update." Some of these compromised hosts simply download malware onto clients rather than collect information directly.

In the generalized view of phishing, the delivery mechanism need not be email, the veil of legitimacy need not come from an online host, and the bait need not be credential confirmation. This paper identifies a phishing variant that distributes attractively priced "fake" hardware through the online marketplace. The "fake" hardware is a communications device in which its embedded software has been maliciously modified; e.g. a cell phone that discloses its current GPS coordinates at the behest of the attacker.

Demand for security has lead to the integration of cryptography in many communications systems. The resulting systems are based on powerful microcomputers that, when co-opted, can execute sophisticated resource-expensive attacks. The embedded software, or *firmware*, that controls these systems eludes scan by malware detectors, and remains unrecognized by the consumer public as a potential host for malicious behavior.

Bugs due to time-to-market pressure, evolving data standards, and security fixes demand field upgradability for embedded software. Moreover, there are several consumer embedded systems for which there are open source firmware distributions: home network appliances (routers, storage, print servers), cell phones (Motorola), computer motherboards (the Linux BIOS project, slimline Open Firmware), and digital music players (iPodLinux, RockBox). Admittedly some of these projects lag behind the current market, but several new cell phones and network appliances are presently supported. While open source firmware is not a requirement for compromising embedded systems, it confers the attacker with an expedient platform for experimentation and development.

Eliminating open source projects does not eliminate the attack. Insiders can collude with an attacker providing access to technical blueprints, passwords, signing keys, and proprietary interfaces. In some ways this makes the attack more effective, because the technical secrecy will be promoted as grounds for trust.

This paper demonstrates an instance of the hardware "spoofing" by maliciously reconfiguring a wireless home router. The router implements a *pharming* attack in which DNS lookups are selectively misdirected to malicious web sites. Opportune targets for pharming attacks include the usual phishing subjects: online banks, software update services, electronic payment services, etc.

Besides stealing online authentication credentials, a spoofed server has access to data stored as cookies for a particular domain. Cookies regularly contain innocuous data, however a visit to one poorly coded (yet legitimate) web site could store clear text personal information in cookies. Less sensitive private information like internet searches, names, email and IP addresses commonly show up in cookie repositories.

Target web sites use SSL (via `https`) in conjunction with certified public keys to authenticate themselves to their clients. In principle this should prevent successful pharming attacks, however the requisite human computer interaction technology for effective use of this cryptographic protocol is not well understood, let alone widely deployed. Users frequently overlook browser frame padlocks indicating an `https` session [7, 16]. Other times a padlock in the browser display area suffices to convince users of a secure connection. In some contexts people "click through" warning after warning to proceed with a browsing session.

WWW2006, May 22–26, 2006, Edinburgh, UK.
.

Furthermore, many trustworthy web sites (news organizations, search engines) do not use SSL since they do not collect personal data. *Semantic attacks*, a more subtle manipulation, employ disinformation through reputable channels. For example, one attack uses multiple trusted news sources to report "election postponed" based on the client's browsing habits.

A router serving the home, small office, or local hotspot environment mediates all communications between its clients and the internet. Anyone connecting to the internet through this router is a potential victim, regardless of platform. In home and small office settings, victims are limited in number, however the storefront hotspot presents a gold mine of activity – potentially yielding hundreds of victims per week.

2. RELATED WORKS

One of the first mass attacks on embedded software was performed by the Chernobyl virus in 1999 [5]. The goal of this malware is purely destruction. It attempts to erase the hard disk and overwrite the BIOS at specified dates. Cell phones have also become targets for worms [4] with the first reports in the wild in 2004. The same author in 2003 predicted infectious malware for the Linksys line of home routers, switches and wireless access points [3].

Arbaugh, Farber, and Smith [2] implement a cryptographic access control system, AEGIS, to ensure that only sanctioned bootstrapping firmware can be installed on the host platform.

This paper explores a variant of email based phishing [9], where distribution occurs through online market places and hardware is "spoofed" by maliciously compromising its embedded software. While much work has been done to detect web site spoofing and to create secure authentication protocols, their effective interaction with human agents is a subject of ongoing research:

Wu, Miller, and Garfinkel [16] present a user study showing that people regularly disregard toolbar warnings when the content of the page is good enough. Another user study by Dhamija, Tygar, and Hearst [7] shows that `https` and browser frame padlock icons (among other indicators) frequently escape consideration in user assessments of web page authenticity. In other work, they propose and implement dynamic security skins [6] which uses a combination of visual hashing and photographic images to create an evident and trusted path between the user and login window.

Stamm and Jakobsson [14] conduct an experiment that distributes a link to a clever video clip through a social network. The link require users to accept self signed Java policy certificate[1] for the full viewing experience; 50% of those visiting the site accepted it. Browser warnings do not indicate the resulting scope of access and mislead users about the authenticity of the certificate.

Cookie theft is one of the more worrisome results of pharming. Attackers can spoof users by presenting stolen cookies to a server; even worse, cookie sometimes directly store personal information. Attempts to provide user authentication, data integrity, and confidentiality within the existing cookie paradigm are discussed in [13]. Unfortunately, the strong authentication methods depend on prior server knowledge of a user's public key.

[1]This allows embedded Java applets a level access on par with the user's, including writing and executing programs.

3. PHISHING WITH MALICIOUS HARDWARE

3.1 Adversarial Model

We make four assumptions about an attacker, $\mathcal{A}$, who compromises firmware in an embedded system: $\mathcal{A}$ has unrestricted physical access to the target device for a short period of time. $\mathcal{A}$ can control all messages that the device receives and intercept all messages that the device sends. $\mathcal{A}$ has in-depth knowledge of the device's hardware/software architecture. $\mathcal{A}$ knows access passcodes necessary to change the device's firmware.

This model gives rise to multiple contexts along each of the four attack requirements. Each property could be generally attainable or available to insiders only. The following table classifies example scenarios according to this decomposition:

	Insider access	General access
Physical	Device at work	Device at home
I/O	Proprietary interfaces	Ethernet/USB
Technical Blueprints	closed source	open source
Passcodes	requires OEM Signed firmware	arbitrary firmware

For instance, $\mathcal{A}$ may have insider access to cell phones through a coatchecking job. The target cell phones run on open source firmware, but require a proprietary wire to upload software. In this instance, the phone's owner has not locked the phone with a password. This illustrates an insider / insider / public / public case of the firmware attack.

3.2 Spoofing honest electronics

Embedded software is an effective place to hide malicious behavior. It is outside the domain of conventional malware detection. Spyware, virus, and worm detection typically take place on client file systems and RAM. New malware detection efforts analyze internet traffic to stop its spread. Neither of these methods detect malicious embedded software. The first model simply doesn't (or can't) scan the EEPROM of a cell phone, a network router, or other embedded systems. The second model reduces the spread of infectious malware, but does not diagnose infected systems.

Many embedded systems targeted at the consumer market have an appliance-like status. They are expected to function correctly out of the box with a minimum of setup. Firmware may be upgraded at service centers or by savvy owners, however consumer products must be able to work well enough for the technically disinterested user. Because of these prevailing consumer attitudes, malicious appliances are beyond the scope of conceivability for many, and therefore endowed with a level of trust absent from personal computers.

Field upgradeable embedded systems generally exhibit no physical evidence of modification after a firmware upgrade. There is no red light indicating that non OEM software controls the system. By all physical examination the compromised hardware appears in new condition.

3.3 Distribution

The online marketplace provides a powerful distribution medium for maliciously compromised hardware. While more expensive than email distribution, it is arguably more effective. High percentages of phishing related email are effec-

tively marked as spam due to header analysis, destroying their credibility. However, online advertisements are available to millions. Only interested users look at the posting. It is unnecessary to coerce attention since the victim approaches the seller.

Online marketplaces connect buyers with sellers. They do not authenticate either party's identity, product warranty or quality. Consequently, the vast majority of auctions carry a *caveat emptor* policy. Merchandise frequently sells "as is" with minimal disclosure about its true condition. One could improve trust by offering a shill return policy: returns accepted within 14 days for a 15% restocking fee ($10 minimum, shipping non-refundable). If the victim uses the product, the attacker potentially benefits from the stolen information, and gets to redeploy the system on another victim.

Reputation systems in the online marketplace help buyers and sellers gauge the trustworthiness in the *caveat emptor* context. These systems principally measure transaction satisfaction: Did the buyer pay in a timely manner? Did the seller deliver in a timely manner? Was the item fundamentally misrepresented? Phishing with malicious embedded systems clearly violates this last criterion, however stealthy malware may *never* be known to the victim. Coupled with pressure to reciprocate positive feedback, the victim will very likely rate the transaction positively. Unlike other fraudulent online sales, this attack's stealthiness will ensure high trust ratings for the seller. Also unlike conventional fraud, the buyer's risk far exceeds the purchase price and delivery fees. The attacker recoups his loss on the "good deal" when exploiting the security hole to access private information.

4. A HOME PHARMING APPLIANCE

This paper's central example of hardware spoofing is a wireless home network router. Our prototype implements a basic pharming attack to selectively misresolve the client domain name requests. It is an example where the four adversarial requirements are all publicly attainable. Physical access is achieved through purchase. All communications to this device go through open standards: ethernet, WiFi, serial port, and JTAG (a factory diagnostic port). Technical details are well documented through open source firmware projects. Firmware upgrades are neither limited to company drivers, nor password protected when new.

4.1 The system context

In general, we assume that the attacker, $\mathcal{A}$, has complete control over the router's incoming and outgoing network traffic, but cannot decrypt encrypted data. While the router can control the communications flow as the $\mathcal{A}$ desires, it is computationally bound. Computationally intensive extensions to the pharming attack need to carefully schedule processing to avoid implausible timing delays. $\mathcal{A}$ controls the appearance and actions of the web administration interface. Administrator access to the firmware update feature would simulate user feedback for the upgrade process and then claim failure for some made up reason. Other functionality, such as WEP/WPA, firewalling, is left intact in both function and appearance.

As a proof of principle, we replace the firmware on a Linksys WRT54GS version 4. The Linksys runs a 200Mhz Broadcom 5352 SoC that includes a MIPS instruction set core processor, 16 MB of RAM, 4 MB of flash memory, 802.11g network interface, and a 4 port fast ethernet switch. The factory embedded software is a version of Linux. Independent review of the corresponding source code has spawned the OpenWRT project [12], an enthusiast developed Linux distribution for the Linksys WRT54G(S) series of routers.

4.2 Basic Pharming attack

Once installed, OpenWRT supports login via `ssh`. This shell provides a standard UNIX interface with file editing through `vi`. DNS spoofing is one of the most expedient attacks to configure. OpenWRT uses the `dnsmasq` server to manage domain name resolution and DHCP leases. The malicious configuration sets the

```
address=/victimdomain.com/X.X.X.X
```

option to resolve the `victimdomain.com` to the dotted quad `X.X.X.X`. All subsequent requests for `victimdomain.com` resolve to `X.X.X.X`. In addition to `address`, the option

```
alias=<old-ip>,<new-ip>[,<mask>]
```

rewrites downstream DNS replies matching `<old-ip>` modulo the mask as `<new-ip>` (replacing numbers for mask bits only); this enables the router to hijack entire subnets.

Anti-phishing tools have limited utility in the presence of phoney domain name resolution. The three prevailing approaches to detecting phoney web sites are server stored reputation databases, locally constructed white lists, and information oriented detection. The first two methods depend exclusively on domain name resolution for database lookup and white/black list lookup. Pharming renders these methods entirely ineffective because the pre-resolution links are correct. The information or content based analysis also depend heavily on link analysis, but may recognize phishing attacks in which login fields are presented in a non SSL connection. However, document obfuscation could reduce the effectiveness of automatic recognition of password requests.

The system runs a `crond` background daemon to process scheduled tasks at particular times of day. For instance, DNS spoofing could be scheduled to begin at 5pm and end 9am to avoid detection during normal business hours.

4.3 Attack extensions

Self signed certificates

One variant is to get the victim to accept a self-signed certificate. The router may offer a self signed SSL certificate to anyone attempting to access its administrative pages. This certificate would later be used to start `https` sessions with the login pages for the spoofed domains. Since web sites change their security policies frequently, spoofed hosts could make entry contingent on acceptance of SSL or even Java policy certificates. Once the victim accepts a Java policy certificate, an embedded Javascript or Java applet may place malware directly onto the victim's file system. Router based pharming greatly aids this kind of attack because it can misdirect *any* request to a malicious web site. Unlike standard phishing attacks that bait the victim into clicking on a link, the attacker exerts no influence on the victim's desire to request the legitimate URL. We hypothesize that this psychological difference results in higher self-signed certificate acceptance rate.

Spying

An easy malicious behavior to configure in the default OpenWRT installation is DNS query logging; it is a simple configuration flag in the `dnsmasq` server. SIGUSR1 signals cause `dnsmasq` to dump its cache to the system log, while SIGINT signals cause the DNS cache to clear. This information approximates the aggregate browsing habits of network clients. The `crond` process could coordinate periodic DNS cache dumps to the system log. The router then posts this data to the attacker during subsequent misdirection.

Cookies can be stolen either through pharming or packet sniffing. Clients fulfill cookie requests when the origin server's hostname matches the cookie's `Domain` attribute *and* the cookie's `Secure` attribute is clear. In this case, browser responds to the cookie request sending values in clear text. These cookies are vulnerable to packet sniffing, and need not utilize pharming for theft.

If the `Secure` attribute is set, then the connection must meet a standard of trust as determined by the client. For Mozilla Firefox, this standard is connection via `https`. The combination of pushing self signed SSL certificates (to satisfy the "secure connection" requirement) and pharming (to satisfy the domain name requirement) results in cookie theft through a man in the middle attack.

Other data is also vulnerable to packet sniffing. POP and IMAP email clients frequently send passwords in the clear. Search queries and link request logging (from the packet sniffing level instead of DNS lookup level) can help to build a contextual dossier for subsequent social engineering.

Delaying detection of fraudulent transactions

The 2006 Identity Theft Survey Consumer Report [10] shows that fraudulent transaction detection strongly influences consumer cost. When the victim monitors account activity through electronic records, the survey found that fraudulent activity was detected in an average of 10 days – 12 days earlier than when account activity is monitored through paper records. Moreover, fraud amounts were 42% higher for those who monitored their transactions by paper instead of electronically.

The malicious router in the home or small office setting (as opposed to the hotspot setting) provides the primary internet access for some set of clients. When such a client monitors account activity, either the network router or the spoofed pharming server can delete fraudulent transactions from electronic records, forestalling detection. The result is a more profitable attack.

4.4 Sustainability

Cost to Attacker

The startup costs for malicious hardware phishing through the online marketplace are high compared to conventional email phishing. Retail price of the router used in this paper is $99, however it is commonly discounted 20-30%. Assume that bulk purchases can be made for a price of $75 per unit. A quick scan of completed auctions at one popular venue between the dates 2/2/2006 and 2/9/06 shows 145 wireless routers matching the search phrase "linksys 802.11g router." Of these, all but 14 sold. Thus there is a sufficiently large market for wireless routers to make the logistics of selling them a full time job.

Listing fees are insignificant. For the sake of computation, let $5 be a gross upper bound on per router selling costs through online marketplaces. To compute a pessimistic lower bound on the cost of reselling the malicious routers, assume that routers sell for an average of $30. Then it costs $50 ($75 new acquisition, plus $5 listing, less $30 selling price) per router to put into circulation. While this method is expensive, the online marketplace disseminates a reliably high number of routers over a wide area.

Hit rate

A gross estimate of phishing success rate is derived from the finding that 3% of the 8.9 million identity theft victims attribute the information loss to phishing [10]. This puts the total phishing victims in 2005 at 267,000, or roughly a 5135 people per week hit rate for the combined efforts of all phishers. Fraud victims per week triples when expanding the cause from phishing to computer-related disclosures (viruses, hacking, spyware, and phishing). This gives a plausible upper bound on phishing's effectiveness, since people can not reliably distinguish the cause of information loss given the lack of transparency in computer technology.

As noted above, the 131 of the wireless routers closely matching the description of this paper's demonstration sold in a week. Other brands use a similarly exploitable architecture (although this is far from universal). Over the same period of time there were 872 auctions for routers matching the the query "802.11g router." This indicates high potential for circulating compromised routers in volume. While far more expensive pricewise, cost in time should be compared to spam based phishing and context aware phishing since one hit (about $2,100 for account misuse) could cover the cost of circulating a week's worth of routers.

Assume that each compromised router produces an average of 3 identity theft victims (the occasional hotspot, multiple user households and small offices), and an individual sells 15 routers a week. Then the number of harvested victims is 45, around .88% of the total number of victims attributed to phishing. Of course these are made up numbers, but illustrates the potential impact due to a *single* attacker.

Financial Gain to Attacker

Assume that the attacker is able to acquire 45 new victims a week as stipulated above. In 2005, the average amount per identity fraud instance was $6383. This suggests a yearly gross of

$$45 \times 52 \times \$6,383 = \$14,936,220$$

for a modestly sized operation. At 15 routers a week, the yearly expenditures for circulating the routers is $39,000, based on the cost of $50 above.

Identity theft survey data [15] shows that on average fraud amount due to *new account & other fraud* ($10,200) is roughly five times higher than fraud amount due to *misuse of existing accounts* ($2,100). A malicious router potentially collects far more personal information than email based phishing due to its omnipresent eavesdropping. This extra information makes it easier to pursue the new account & other fraud category than one bite phishing (e.g. email), thereby increasing the expected fraud amount per victim. Moreover, multiple accounts are subject to hijacking, and the router may elude blame for the information disclosure for quite some time given the opaqueness of computer technology, opening the victim to multiple frauds a year.

Consider a worst case estimate where: no victim is robbed more than once, the fraud amount is due to account misuse ($2,100), and the distribution costs are high ($120 per router, i.e. free to victim). The yearly gross is still $4,914,000, with a distribution cost of $81,000.

In summary the startup costs are high for this attack, however the stream of regular victims and magnitude of corresponding fraud dwarf the distribution costs.

Management of non-monetary risks

The attacker may incur substantial non-monetary risks when implementing this scheme. The primary concern is exposure. Purchasing routers in bulk could raise suspicion. The plan above entails a relatively modest number (15) of router purchases per week. A computer criminal need not sell the routers through a single personal account. The diligent attacker will control many accounts, possibly reusing the accounts of her victims to buy and sell small numbers of routers.

Another concern is the relatively long attack lifetime. Phishing servers remain online for about 5 to 6 days before vanishing [1], yet the malicious firmware resides on the router indefinitely. This does not imply that the malicious hosts referenced by the router's pharming attack also stay online indefinitely. Although the pharming attack implemented in the demonstration is static, compromised routers can communicate with agents of the attacker through ssh connections for dynamic updates to compromised host listings. The fraudulent hosts retain their short online lifetimes under this scheme.

If the attacker has a large network of compromised routers, then her apprehension by law enforcement should begin the reversion of compromised without revealing their IP addresses. She can use a botnet to implement a dead (wo)man's switch. In normal circumstances the botnet receives periodic "safety" messages. In the absence of these messages, the botnet spams appropriately disguised "revert" commands to the IPv4 address space. The reversion to factory firmware need not be complete though. While manufacturer firmware often has sufficient vulnerabilities, the reversion could configure the manufacturer firmware for straightforward reinfection (e.g., set firewall policy to accept remote administration through an unusual port). This has the advantage of not disclosing the nature of the malware to investigators. It will simply appear vulnerable.

The biggest concern is actually executing the identity fraud. Cash transfers out of existing accounts are quick, but tend to be for lower dollar values than new account fraud as noted earlier. New account fraud seems more promising for actually purchasing goods since the attacker will be able to control the registered mailing address and avoid detection for a longer period of time. For maximal impact, the fraudster should empty the existing accounts last using cash transfers.

5. COUNTERMEASURES

Malicious firmware poses some serious threats, however, we are not helpless to prevent them. This section examines some methods to counter the general problem, and then some methods that mitigate the malicious network router.

5.1 General countermeasures

Accessibility to firmware is obscure, but not secure. These properties discourage trust. The firmware upgradability channels should be evident to the consumer, and moreover should implement effective access control. These processors have sufficient power to check digital signatures. One solution uses a hard-wired bootstrapping process to check digitally signed firmware against an onboard manufacturer public key, just as in [2]. This addition limits firmware changes to those sanctioned by the manufacturer.

In the absence of tamper proof or tamper evident hardware, a knowledgeable and determined attacker could replace the chips holding either the bootstrapping program or the manufacturer's public key (assuming that these are not integrated into the SoC silicon). Moreover, part of the appeal for many technologically savvy consumers is the ability to control the hardware in novel ways. One solution makes the digital signature check bypassable using an circuit board jumper, while using a tamper evident exterior. Third party firmware is still installable, yet the hardware can no longer be represented as within factory specification. This solution also appeals to a meticulous customer who sees third party firmware as more trustworthy.

5.2 Pharming countermeasures

In context of identity theft, the principal threat is accepting a self-signed SSL certificate. Once accepted, the spoofed host's login page can be an exact copy of the authentic page over an SSL connection. The semi-weary user, while fooled by the certificate, observes the `https` link in the address bar and the padlock icon in the browser frame and believes that the transaction is legitimate. An immediate practical solution is to set the default policy on self signed certificates to reject. A finer grained approach limits self signed certificate rejection to a client side list of critical web sites.

Many phishing toolbars check for an `https` session when a login page is detected. This detection is not straightforward. HTML obfuscation techniques can hide the intended use of web pages by using graphics in place of text, changing the names of the form fields, and choosing perverse style sheets. This includes many of the same techniques that phishers use to subvert content analysis filters on mass phishing email.

The DNS protocol is very efficient at the cost of high vulnerability. Every machine in the DNS hierarchy is trusted to return correct results. Erroneous or malicious results are forwarded without scrutiny. *Secure DNS*, or *DNSSEC* [8, 11], is a proposal where each level of reference and lookup is digitally signed by trusted servers. The client starts out with the public key of a DNS server it trusts. Server traversal proceeds as usual, but with the addition of digital signatures for each delegation of name lookup. The lookup policy forces servers to only report on names for which they have authority, eliminating cache poisoning. This method returns a client checkable certificate of name resolution. If implemented as stated, the system will be very difficult to subvert. However, there is substantial overhead in all the signature checking. A real implementation will need to implement caching at some level for efficiency. What servers are trustable for lookups outside their authority? One should not trust public or open wireless access points since they are controlled by unknown agents. Home routers which are under the physical control of the user should be trusted. Their compromise exposes clients worse vulnerabilities than just pharming (e.g. packet sniffing, mutation, rerouting, eavesdropping). While widespread DNSSEC deployment coupled with the correct trust policies (i.e. no errant or

malicious servers are trusted) will eliminate pharming, the compromised router achieve the same effect by rerouting unencrypted `http` traffic to a man-in-the-middle host.

6. CONCLUSION

This paper serves as a call to action. Maliciously compromised embedded systems are implementable today (e.g. our demonstration). They are dangerous because of the damage they can inflict and because of misplaced consumer trust. Their distribution through online auctions is a plausibly sustainable enterprise.

7. ACKNOWLEDGEMENTS

I would like to thank Markus Jakobsson for recommending a project on malicious embedded firmware. My conversations with Bhanu Nagendra Pisupati resulted in choosing wireless routers as a promising target. I have Jean Camp's influence to thank for framing the feasibility in economic terms.

8. REFERENCES

[1] APWG. Phishing activity trends report. Technical report, Anti-Phishing Working Group, December 2005.

[2] W. A. Arbaugh, D. J. Farber, and J. M. Smith. A secure and reliable bootstrap architecture. In *SP '97: Proceedings of the 1997 IEEE Symposium on Security and Privacy*, pages 65–71, Washington, DC, USA, 1997. IEEE Computer Society.

[3] Ivan Arce. The rise of the gadgets. *IEEE Security & Privacy*, September/October 2003.

[4] Ivan Arce. The shellcode generation. *IEEE Security & Privacy*, September/October 2004.

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

[6] Rachna Dhamija and J. D. Tygar. The battle against phishing: Dynamic security skins. In *SOUPS '05: Proceedings of the 2005 symposium on Usable privacy and security*, pages 77–88, New York, NY, USA, 2005. ACM Press.

[7] Rachna Dhamija, J. D. Tygar, and Marti Hearst. Why phishing works. `http://www.sims.berkeley.edu/~rachna/papers/why_phishing_works.pdf`.

[8] D. Eastlake. Domain name security extensions. RFC 2535, March 1999.

[9] Markus Jakobsson and Steve Myers. *Phishing and Counter-measures: Understanding the Increaseing Problem of Electronic Identity Theft*. Wiley, 2006.

[10] Javelin Strategy & Research. Identity theft survey report (consumer version), 2006.

[11] Trevor Jim. Sd3: A trust management system with certified evaluation. In *IEEE Symposium on Security and Privacy*, pages 106–115, 2001.

[12] Openwrt. `http://www.openwrt.org`.

[13] Joon S. Park and Ravi Sandhu. Secure cookies on the web. *IEEE Internet Computing*, 4(4):36–44, 2000.

[14] Sid Stamm and Markus Jakobsson. Case study: Signed applets. In *Phishing and ...* [9].

[15] Synovate. Federal trade commission identity theft survey report, 2003.

[16] Min Wu, Robert Miller, and Simson Garfinkel. Do security toolbars actually prevent phishing attacks? In *CHI*, 2006.