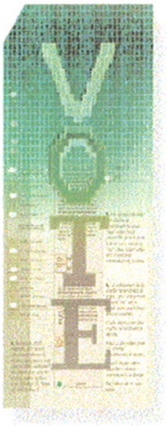
By David Jefferson, Aviel D. Rubin, Barbara Simons, *and* David Wagner

Analyzing Internet Voting Security

An extensive assessment of a proposed Internet-based voting system.

The Secure Electronic Registration and Voting Experiment (SERVE) is an Internet-based voting system built by Accenture and its subcontractors for the U.S. Department of Defense FVAP (Federal Voting Assistance Program).[1] FVAP's mission is to reduce voting barriers for all citizens covered by the Uniformed and Overseas Citizens Absentee Voting Act (UOCAVA), namely U.S. citizens who are members of the military services, their family members, and nonresident U.S. citizens. SERVE is intended to allow UOCAVA voters both to register to vote and to vote via the Internet, from anywhere in the world. It is meant to be a complete, Independent Testing Authority-qualified and state-certified voting system that collects real votes.



To participate, an eligible voter first enrolls in the SERVE program. After enrollment, the voter may register to vote, and then vote in one or two short sessions from any Internet-connected PC. The PC must run a Microsoft Windows operating system and either the Internet Explorer or Netscape Web browser. The browser must be configured to enable JavaScript, along with either Java or ActiveX scripting, and session cookies; no additional hardware or software is required.

When a person registers online to vote, his or her information is stored on the central Web server for later retrieval by the Local Election Official (LEO), at which point the LEO updates its database. When a person votes in the election, the completed ballot is stored on the central server and later downloaded by the LEO, who stores it for canvass. The communication between the user's Web browser and the central server is protected using the Secure Socket Layer (SSL) protocol. Once that connection is established, an ActiveX control is downloaded to the voter's PC and run to provide functionality not available in current browsers.

[1] The authors are four of a group of eight computer scientists and security experts who reviewed the Pentagon's $22 million SERVE program for voting over the Internet. Shortly after the release of the full report in January 2004 [3], the Pentagon decided not to implement SERVE in the 2004 election, citing security concerns. It is still possible that a program similar to SERVE could be proposed for future elections. This article, derived from the full report, describes the security issues the authors identified with SERVE, most of which apply to Internet voting in general. To simplify the presentation, the present tense is used, even though there are no longer current plans to use SERVE in any election.

Besides being restricted to overseas voters and military personnel, the 2004 trial SERVE was to be limited to people who vote in one of 50 counties in the seven states (Arkansas, Florida, Hawaii, North Carolina, South Carolina, Utah, and Washington) that agreed to participate. The 2004 trial was expected to handle up to 100,000 votes over the course of the year (as many votes as a small state), including both the primaries and the general election. By comparison, approximately 100 million votes were cast in the 2000 general election. One goal was to determine if a similar system might be suitable for expansion to all six million UOCAVA voters.

Limited though it is, SERVE is a real voting system. Systems similar to SERVE might eventually be offered by Accenture or other vendors for use by all voters, instead of just a limited population. For these reasons we analyze SERVE not as an experiment, but as a real voting system whose use could be significantly expanded in future years.

BECAUSE THE INTERNET IS INDEPENDENT OF NATIONAL BOUNDARIES, AN ELECTION HELD OVER THE INTERNET IS VULNERABLE TO ATTACKS FROM ANYWHERE IN THE WORLD.

Our Recommendations

Our findings and recommendations are summarized here.

Direct-recording electronic (DRE) voting systems have been widely criticized for various deficiencies and security vulnerabilities: that their software is totally closed and proprietary; that the software undergoes insufficient scrutiny during qualification and certification; that DREs are especially vulnerable to various forms of insider (programmer) attacks; and that DREs have no voter-verified audit trails (paper or otherwise) that could largely circumvent these problems. All of these criticisms of DREs apply directly to SERVE as well [4].

In addition, because SERVE is an Internet- and PC-based system, it has numerous other fundamental security problems that leave it vulnerable to a variety of well-known cyber-attacks (denial-of-service attacks, spoofing, viral attacks on voter PCs, and so forth), any one of which could be catastrophic.

Such attacks could occur on a large scale, and could be launched by anyone in the world, ranging from a disaffected lone individual to a well-financed enemy agency outside the reach of U.S. law. These attacks could result in large-scale, selective voter disenfranchisement, and/or privacy violation, and/or vote buying and selling, and/or vote switching even to the extent of reversing the outcome of many elections at once, including the presidential election. Some of the attacks could succeed and yet go completely undetected. Even if detected and neutralized, such attacks could have a devastating effect on public confidence in elections.

It is impossible to estimate the probability of a successful cyber-attack; but we show that the attacks we are most concerned about are quite easy to perpetrate. In some cases there are kits readily available on the Internet that could be modified or used directly for attacking an election. And we must consider the sentiment that a U.S. general election offers one of the most tempting targets for cyber-attack in the history of the Internet, whether the attacker's motive is overtly political or simply self-aggrandizement.

The vulnerabilities we describe cannot be fixed by design changes or bug fixes in SERVE. Instead, they are fundamental in the architecture of the Internet and of the PC hardware and software that is ubiquitous today. The vulnerabilities cannot be eliminated for the foreseeable future without a wholesale redesign and replacement of much of the hardware and software security systems in the PC and the Internet, or else some unforeseen radical security breakthrough(s).

We have examined numerous variations on SERVE; however, all such variations suffer from the same kinds of fundamental vulnerabilities. Regrettably, we cannot recommend any of them. We do suggest a kiosk architecture as a starting point for designing an alternative voting system with similar aims to SERVE, but which does not rely on the Internet or on unsecured PC software (see [3], Appendix C.)

A seemingly successful voting experiment in a U.S. presidential election involving seven states would likely be viewed by most people as strong evidence that SERVE is a reliable, robust, and secure voting system. Such an outcome would encourage expansion of the program by FVAP in future elections, or the marketing of the same voting system by vendors to jurisdictions all over the U.S., and other countries as well.

However, the fact that no successful attack is detected does not mean that none occurred. Many attacks, especially if cleverly hidden, would be extremely difficult to detect, even in cases when they change the outcome of a major election. A "successful" trial of SERVE in 2004 is the top of a slippery slope toward even more vulnerable systems in the future. (The existence of SERVE was cited as justification for Internet voting in the Michigan Democratic caucuses earlier this year.)

Like the proponents of SERVE, we believe that there should be better support for voting for members of the military overseas. Still, because the danger of successful, large-scale attacks is so great, we reluctantly recommend shutting down the development of SERVE immediately and not attempting anything like it in the future until the security problems of the PC and the Internet are resolved. The remainder of this article explains some of the reasoning behind these conclusions.

Vulnerabilities in SERVE

Because the Internet is independent of national boundaries, an election held over the Internet is vulnerable to attacks from anywhere in the world. Not only could a political party attempt to manipulate an election by attacking SERVE, but so could individual hackers, criminals, terrorists, and even other countries. There is no need to postulate a large conspiracy or highly sophisticated adversaries; many of the attacks we describe could be mounted by lone individuals with college-level training in computer programming. Here, we give a short description of a variety of attacks that can be mounted against SERVE.

Lack of Voter-Verified Audit Trail and Insider Attacks. Paperless DRE voting systems have been widely criticized because they are unauditable. There is no way for a voter to verify that the vote recorded inside the machine is the same as the vote he or she entered and saw displayed on the machine's screen; if serious problems subsequently occur in the canvass of the votes (which happens all too frequently), there is no independent audit trail of the votes to help resolve the problem. Voter verification is the only readily available effective defense against programmed insider attacks. Every argument about the need for voter verification and auditability that has been made about DREs applies essentially unchanged to SERVE (see www.notablesoftware.com/evote, www.verifiedvoting.org, and [1]).

Privacy. The privacy of SERVE ballots is protected using encryption. When the ballot is cast, it is encrypted during transmission over the Internet and decrypted at the central server. Once received, the ballot is separated from the voter's identity and the anonymous ballot is reencrypted so that only the LEO of the voter's district can read it. These encrypted ballots are stored at the central server and can be downloaded (in randomly reordered form) upon request by LEOs.

This architecture introduces several privacy risks. First, a LEO could deduce how voters in his or her precinct have voted by downloading votes from SERVE so frequently that the LEO gets at most one new vote and voter name each time. This would allow a curious LEO to infer how each individual voted. Second, the brief existence of cleartext ballots on the server introduces a risk that SERVE system administrators could view how individuals voted. Likewise, if SERVE machines were subverted by hackers, the privacy of all votes could be compromised. Third, SERVE's retention of encrypted ballots for 18 months or longer could compromise voter privacy if this information were to land in the wrong hands and old system keys were exposed.

Vote Buying/Selling. Vote selling is a problem in all elections, but it is a special concern for Internet voting, since large-scale vote buying and selling can be automated. During the 2000 presidential election, several Web sites were created to facilitate vote swapping between Gore and Nader voters. While the Gore/Nader swapping depended on the honor system and no money changed hands, a similar approach could be used with SERVE to provide enforced vote swapping or vote bartering services, or to purchase votes from SERVE voters.

The most straightforward vote-buying scheme would involve the selling of personally identifiable information and the voter's password or private key. One possible defense would be for SERVE to prohibit the submission of multiple votes from the same Internet address. This is not a strong defense, however, because a purchaser of votes could fool SERVE into thinking the votes were coming from different addresses, and because legitimate users often appear to come from the same IP address.

Another approach to vote buying would be for the buyer to provide the seller with a version of the SERVE ActiveX component that is modified to ensure the voting is done according to the wishes of the vote purchaser. There does not seem to be any way for SERVE to defend against this style of vote buying.

Large-Scale Impact. When voting is conducted at physical precincts on mechanical devices or with

Threat	Skill needed	Consequences	Realistic?	Countermeasures
Denial of-service attack (various kinds)	low	disenfranchisement (possibly selective disenfranchisement)	Common on the Internet.	No simple tools; requires hours of work by network engineers; launchable from anywhere in the world.
Trojan horse attack on PC to prevent voting	low	disenfranchisement	There are a million ways to make a complex transaction such as voting fail.	Can mitigate risk with careful control of PC software; reason for failure may never be diagnosed.
On-screen electioneering	low	voter annoyance, frustration, distraction, improper influence	Trivial with today's Web.	Nothing voter can do to prevent it: requires new law.
Spoofing of SERVE (various kinds)	low	vote theft, privacy compromise, disenfranchised voters	Web spoofing is common and relatively easy.	None exists; likely to go undetected; launchable by anyone in the world.
Client tampering	low	disenfranchisement	One example: change permissions on cookie file. Many other trivial examples.	None exists for all possible mechanisms. Too difficult to anticipate all attacks; likely never diagnosed.
Insider attack on system servers	medium	complete compromise of election	Insider attacks are the most common, dangerous, and difficult to detect of all security violations.	None within SERVE architecture; voter-verified ballots needed; likely undetected.
Automated vote buying/selling	medium	disruption of democracy	Very realistic, since voter willingly participates.	None exists; buyers may be out of reach of U.S. law.
Coercion	medium	disruption of democracy	Harder to deploy than vote buying/selling, but man-in-the-middle attacks make it achievable with average skill.	None exists; likely to go undetected.
SERVE-specific virus	medium or high	vote theft, privacy compromise, disenfranchised voters	Some attacks require only experimentation with SERVE; others require leak of SERVE specs or code and resourceful attacker.	Virus-checking software can catch known viruses, but not new ones; likely to go undetected.
Trojan horse attack on PC to change votes or spy on them	high	vote theft, privacy compromise	Widely available spyware would be a good starting point.	Can mitigate risk with careful control of PC software; harder to control at cybercafe or other institutionally managed networks; likely to go undetected.

Major vulnerabilities identified in SERVE.

paper ballots, whatever vote manipulation occurs happens on a relatively small scale. By contrast, since SERVE is vulnerable to many different types of attacks, a significant percentage of votes cast over the Internet could be vulnerable. A single malicious party could potentially affect tens of thousands of votes cast through SERVE, whereas it is extremely unlikely that any single person could conduct vote fraud on such a large scale in existing nonelectronic elections. The table appearing here summarizes the major vulnerabilities we have identified in SERVE, along with our assessment of those vulnerabilities. The table describes, for each potential threat to SERVE, what skill is required by the attacker, the consequences of a successful attack, how realistic the attack is, and what countermeasures might be used to thwart the attack.

The remainder of this article details three of the most important of the vulnerabilities in SERVE; for further information see the original report [3].

Lack of Control of the Voting Environment

Perhaps the greatest challenge with Internet voting arises from the fact that electoral authorities do not have control over all the equipment used by voters. Since SERVE's voters can vote on their own computers or on computers controlled by others, third parties might be able to gain control of a large number of computers used for voting. Such attacks could result in the loss of voter privacy, disenfranchisement, or vote alteration without anyone, including the voter and election officials, noticing or detecting any problem.

The Computers. Voters' personal computers are unlikely to be as carefully defended as corporate ones, and hence voters' machines are especially susceptible to attack. Attacks can be easily automated; hackers routinely scan thousands or even millions of computers in search of those that are easiest to compromise. A relatively easy way to disenfranchise the voter is to disable ActiveX or Web cookies so that it is no longer possible to vote through SERVE. Alternatively, a malicious third party could cast an unauthorized ballot that appears to come from the voter.

A shared computer, for example at a cybercafe or public library, is even more insecure. The owner, the system administrator, or even a prior visitor could have installed remote spying or subversion software. Voting from workplaces entails similar risks. One study found that 62% of major U.S. corporations monitor employee's Internet connections, and more than one-third store and review files on employees' computers (see www.amanet.org/research/pdfs/ems_short2001.pdf).

The Software. Preinstalled software applications also pose risks. Backdoors placed in software and activated when a user tries to vote could invisibly monitor or subvert the voting process. Software security vulnerabilities could allow a remote attacker to take complete control of a computer using remote control software such as PCAnywhere or BackOrifice. Successful penetration of even well-defended computers is routine.

Viruses and Worms. One of the most dangerous forms of remote attack is a virus or worm that spreads itself and contains a malicious payload designed to take control of machines and wreak havoc with a future election [7]. Since virus-checking software defends against only previously known viruses, virus checkers often are unable to keep up with the spread of new viruses and worms. In 2001, the Code Red worm infected 360,000 computers in 14 hours, and in 2003 the Slammer worm [5] brought down many ATM machines and compromised many Internet hosts (see securityresponse.symantec.com/avcenter/venc/data/w32.sqlexp.worm.html). Modern worms are even more virulent, are often spread by multiple methods, are able to bypass firewalls and other defenses, and can be difficult to analyze. For example, it took quite a while to determine that SoBig.F was a Trojan horse designed to plant spam engines (see securityresponse.symantec.com/avcenter/venc/data/w32.sobig.f@mm.html).

Attackers can build new viruses, or modify existing viruses sufficiently that they will avoid detection. Virus construction kits are available on the Internet. In addition, attackers have the advantage that they can test new versions of viruses using the same publicly available virus checkers that potential victims use, thus confirming that the virus will not be detected before its release.

Web Sites. A dangerous hybrid attack involves placing malicious content on specially chosen Web sites. For instance, an attacker with a vendetta against one candidate might booby-trap the Web site of that candidate, so that those who visit the candidate's Web site are unable to vote using SERVE. Such selective disenfranchisement might eliminate several hundreds or thousands of votes for a candidate, enough to throw the election to his or her opponent.

Spoofing and Man-in-the-Middle Attacks

In man-in-the-middle attacks the adversary interposes itself between legitimate communicating parties and simulates each party to the other. To simplify the discussion in the context of this article, we focus primarily on ways that a man-in-the-middle attack can subvert voter privacy, although the same general technique can be used for other attacks, such as vote buying.

The use of SSL does little to mitigate man-in-the-middle attacks on privacy. Any man-in-the-middle could act as an SSL gateway, forwarding application data between the voter and the vote server unaltered. The attacker could see all of the traffic by decrypting and reencrypting as communications pass between the two. In effect, the attacker would communicate using two SSL sessions, one between itself and the voter, and the other between itself and the vote server, and neither would know that there was a problem. These attacks are possible because the voter's browser does not verify that it is talking to the real SERVE Web server—only that it is talking to someone in possession of a valid SSL certificate (who could be an attacker).

Man-in-the-middle attacks also could be used to disenfranchise voters by spoofing the entire interaction with the voter. SERVE has some safeguards in place, but they assume the voter knows exactly what to expect from the voting experience; it is likely that an attacker could create a voting experience the voter would believe is real. Similarly, voters could be led to believe they registered successfully, when in fact they were communicating directly with an adversary instead of the legitimate registration server. The voters would discover when attempting to vote that they were not registered, but at that point there might be nothing they could do to resolve the situation.

PERHAPS THE GREATEST CHALLENGE WITH INTERNET VOTING ARISES FROM THE FACT THAT ELECTORAL AUTHORITIES DO NOT HAVE CONTROL OVER ALL THE EQUIPMENT USED BY VOTERS.

Perhaps the most serious consequence of man-in-the-middle attacks is that attackers could engage in election fraud by spoofing the voting server and observing how a particular voter votes. If the vote is to the attacker's liking, the voter is redirected to SERVE's legitimate voting site. If the attacker does not like the vote, then the entire voting session is spoofed; in this case, the user thinks he or she has voted, but in fact the vote will not be received or counted by SERVE.

Denial-of-Service Attacks

Attacks in which legitimate users are prevented from using the system by malicious activity such as overloading the election Web server are known as denial-of-service attacks. A particularly nasty variant of denial-of-service attack is the distributed-denial-of-service (DDoS) attack. In this scenario, an attacker typically takes control of many computers in advance by spreading a custom-crafted virus or

WE EXPECT THAT DENIAL-OF-SERVICE ATTACKS COULD DISENFRANCHISE A SUBSTANTIAL FRACTION OF THE SERVE POPULATION, AND THERE SEEMS TO BE LITTLE THAT SERVE CAN DO TO DEFEND AGAINST SUCH ATTACKS.

worm. In computer security jargon, the compromised machines are often known as zombies or slaves, because the attacker leaves behind hidden software that causes infected machines to blindly obey subsequent commands from the attacker. Automated tools for mounting DDoS attacks have been circulating among the hacker community since at least 1999 [2], and hackers routinely amass large zombie networks of compromised machines. In February 2000, major DDoS attacks were mounted against several high-profile Web sites, including CNN, Yahoo and eBay. It was later discovered that these damaging attacks had been perpetrated by a lone teenager located outside the U.S.

Since 2000, DDoS attacks have become routine. One study recorded over 10,000 denial-of-service attacks during a three-week period in 2001 [6]. The Code Red worm, for example, contained code to mount a DDoS attack on the White House Web site. (Fortunately, the DDoS attack was deflected at the last minute.) In 2003, an Internet election in Canada was disrupted by a denial-of-service attack on Election Day. These are not isolated examples; it is all too easy to mount DDoS attacks, and the culprits are rarely caught.

If an attacker were to mount a large-scale denial-of-service attack that renders SERVE's voting service unavailable on Election Day, it would call into question the validity of the election and effectively disenfranchise large numbers of UOCAVA voters. Alternatively, network services could be knocked out or degraded for areas where a particular demographic is known to vote for a particular party, possibly modifying the outcome of the election. Detection of a selective disenfranchisement attack might be possible, but it is not clear how to respond—once polls close, there may be no good choices. We expect that denial-of-service attacks could disenfranchise a substantial fraction of the SERVE population, and there seems to be little that SERVE can do to defend against such attacks.

Conclusion

Because of space constraints, we have mentioned only a few of the possible attacks. These attacks depend on fundamental vulnerabilities in the current PC architecture (malicious code, for example) and in the Internet (such as spoofing and denial-of-service attacks). These attacks can be launched by anyone in the world, and in many cases may be successful while remaining completely undetected. Consequently, we conclude that Internet voting in general, and SERVE in particular, cannot be made secure for use in real elections for the foreseeable future. (See the full report [3].)

References

1. *California Secretary of State Ad Hoc Touchscreen Voting Task Force Report*; www.ss.ca.gov/elections/taskforce_report.htm.
2. Houle, K.J. and Weaver, G.M. *Trends in Denial of Service Attack Technology.* Technical Report, CERT Coordination Center (Oct. 2001).
3. Jefferson, D.R., Rubin, A.D., Simons, B., and Wagner, D. *A Security Analysis of the Secure Electronic Registration and Voting Experiment (SERVE)*; www.servesecurityreport.org/.
4. Kohno, T., Stubblefield, A., Rubin, A.D., Wallach, D.S. Analysis of an electronic voting system. *IEEE Security and Privacy* (2004).
5. Moore, D., Paxson, V., Savage, S., Shannon, C., Staniford, S., and Weaver, N. Inside the Slammer worm. *IEEE Security and Privacy* (2003).
6. Moore, D., Voelker, G.M., and Savage, S. Inferring Internet denial-of-service activity. *Usenix Security* (2001).
7. Staniford, S., Paxson, V., and Weaver, N. How to own the Internet in your spare time. *Usenix Security* (2002).

David Jefferson (d_jefferson@yahoo.com) is a computer scientist at Lawrence Livermore National Laboratory chair of the Technical Advisory Board for the Secretary of State of California

Aviel D. Rubin (rubin@jhu.edu) is a professor of Computer Science and the technical director of the Information Security Institute at Johns Hopkins University.

Barbara Simons (simons@acm.org) is an independent technology consultant, retired from IBM Research, and a past president of ACM.

David Wagner (daw@cs.berkeley.edu) is an assistant professor of Computer Science at the University of California at Berkeley.

In *Proc. 16th Conference on Financial Cryptography & Data Security*, Feb. 2012

Attacking the Washington, D.C. Internet Voting System

Scott Wolchok, Eric Wustrow, Dawn Isabel, and J. Alex Halderman

The University of Michigan, Ann Arbor
{swolchok,ewust,dki,jhalderm}@umich.edu

Abstract. In 2010, Washington, D.C. developed an Internet voting pilot project that was intended to allow overseas absentee voters to cast their ballots using a website. Prior to deploying the system in the general election, the District held a unique public trial: a mock election during which anyone was invited to test the system or attempt to compromise its security. This paper describes our experience participating in this trial. Within 48 hours of the system going live, we had gained near-complete control of the election server. We successfully changed every vote and revealed almost every secret ballot. Election officials did not detect our intrusion for nearly two business days—and might have remained unaware for far longer had we not deliberately left a prominent clue. This case study—the first (to our knowledge) to analyze the security of a government Internet voting system from the perspective of an attacker in a realistic pre-election deployment—attempts to illuminate the practical challenges of securing online voting as practiced today by a growing number of jurisdictions.

Keywords: Internet voting, e-voting, penetration testing, case studies

1 Introduction

Conducting elections for public office over the Internet raises grave security risks. A web-based voting system needs to maintain both the integrity of the election result and the secrecy of voters' choices, it must remain available and uncompromised on an open network, and it has to serve voters connecting from untrusted clients. Many security researchers have cataloged threats to Internet voting (e.g. [11,15]), even as others have proposed systems and protocols that may be steps to solutions someday (e.g. [6,12]); meanwhile, a growing number of states and countries have been charging ahead with systems to collect votes online. Estonia [1] and Switzerland [2] have already adopted online voting for national elections. As of 2010, 19 U.S. states employed some form of Internet voting [5], and at least 12 more were reportedly considering adopting it [4].

Among the jurisdictions considering Internet voting, one of the most enthusiastic proponents was the District of Columbia. In 2010, the Washington, D.C. Board of Elections and Ethics (BOEE) embarked on a Federally-funded pilot project that sought to allow overseas voters registered in the District to vote over the web starting with the November 2010 general election [16]. Though the D.C. system, officially known as the "D.C. Digital Vote-by-Mail Service," was technologically similar to parallel efforts in other states, BOEE officials adopted a unique and laudable level of transparency. The system was developed as an open source project, in partnership with the nonprofit Open Source Digital Voting (OSDV) Foundation [3]. Most significantly, prior to collecting real votes with the system, the District chose to operate a mock election and allow members of the public to test its functionality and security.

We participated in this test, which ran for four days in September and October 2010. Our objective was to approach the system as real attackers would: starting from publicly available information, we looked for weaknesses that would allow us to seize control, unmask secret ballots, and alter the outcome of the mock election. Our simulated attack succeeded at each of these goals and prompted the D.C. BOEE to discontinue its plans to deploy digital ballot return in the November election.

In this paper, we provide a case study of the security of an Internet voting system that, absent our participation, might have been deployed in real elections. Though some prior investigations have analyzed the security of proposed Internet voting systems by reviewing their designs or source code, this is the first instance of which we are aware where researchers have been permitted to attempt attacks on such a system in a realistic deployment intended for use in a general election.

We hope our experiences with the D.C. system will aid future research on secure Internet voting. In particular, we address several little-understood practical aspects of the problem, including the exploitability of implementation errors in carefully developed systems and the ability of election officials to detect, respond, and recover from attacks. Our successful penetration supports the widely held view among security researchers that web-based electronic voting faces high risks of vulnerability, and it cautions against the position of many vendors and election officials who claim that the technology can readily be made safe.

The remainder of this paper is organized as follows: Section 2 introduces the architecture and user interface of the Digital Vote-By-Mail System. In Section 3, we describe how we found and exploited vulnerabilities in the web application software to compromise the mock election. Section 4 describes further vulnerabilities that we found and exploited in low-level network components. Section 5 discusses implications of our case study for other Internet voting systems and future public trials. We survey related work in Section 6 and conclude in Section 7.

2 Background: The D.C. Digital Vote-By-Mail System

Architecture The Digital Vote-by-Mail (DVBM) system is built around an open-source web application[1] developed in partnership with the D.C. BOEE by the OSDV Foundation's TrustTheVote project[2]. The software uses the popular Ruby on Rails framework and is hosted on top of the Apache web server and the

[1] http://github.com/trustthevote/DCdigitalVBM/

[2] http://trustthevote.org

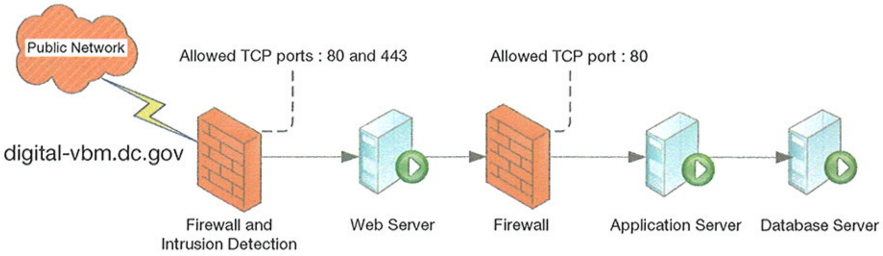


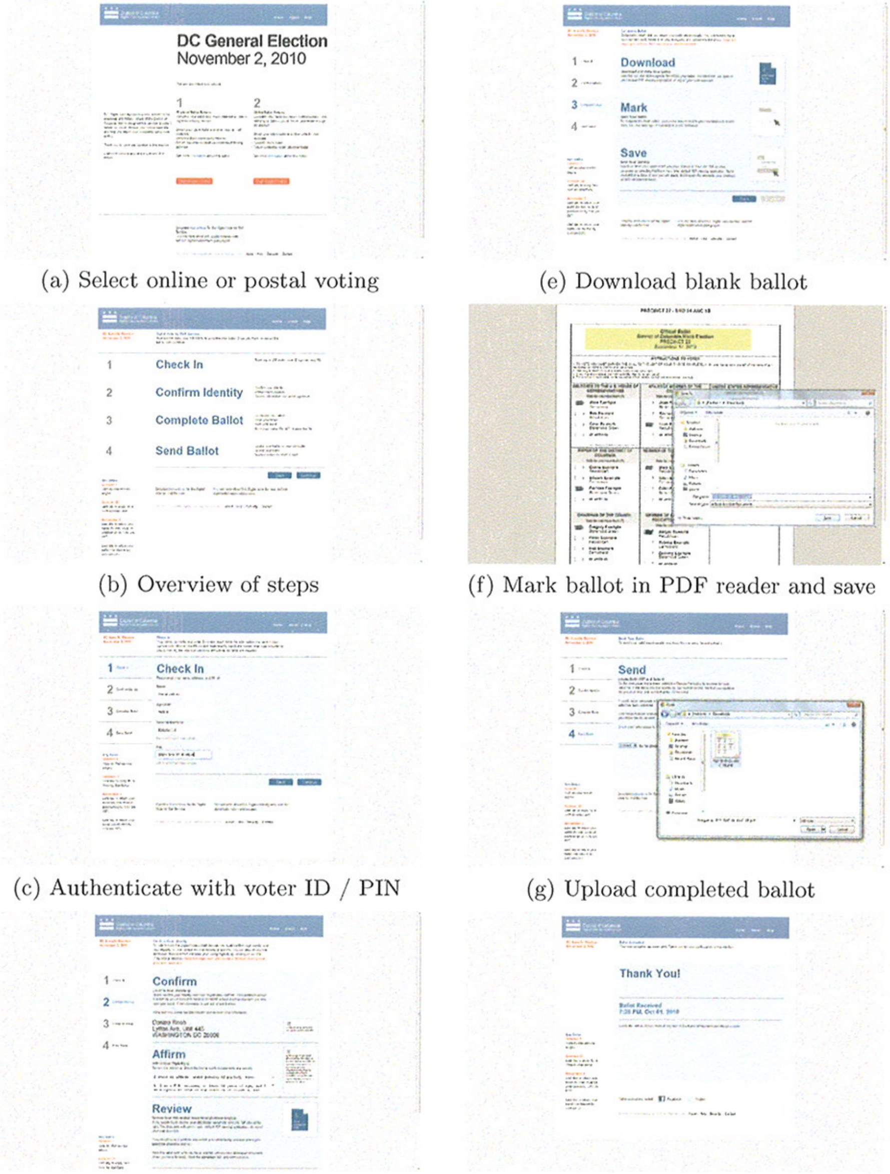
Fig. 1: **Network architecture** — The front-end web server receives HTTPS requests from users and reverse-proxies them to the application server, which hosts the DVBM election software and stores both blank and completed ballots. A MySQL database server stores voter credentials and tracks voted ballots. Multiple firewalls reduce the attack surface and complicate attacks by disallowing outbound TCP connections. The intrusion detection system in front of the web server proved ineffective, as it was unable to decrypt the HTTPS connections that carried our exploit. (Adapted from http://www.dcboee.us/DVM/Visio-BOEE.pdf.)

MySQL relational database. Global election state (such as registered voters' names, addresses, hashed credentials, and precinct-ballot mappings, as well as which voters have voted) is stored in the MySQL database. Voted ballots are encrypted and stored in the filesystem. User session state, including the user ID and whether the ballot being cast is digital or physical, is stored in an encrypted session cookie on the user's browser.

Electronic ballots are served as PDF files which voters fill out using a PDF reader and upload back to the server. To safeguard ballot secrecy, the server encrypts completed ballots with a public key whose corresponding private key is held offline by voting officials. Encrypted ballots are stored on the server until after the election, when officials transfer them to a non-networked computer (the "crypto workstation"), decrypt them using the private key, and print them for counting alongside mail-in absentee ballots.

Figure 1 shows the network architecture deployed for the mock election. HTTPS web requests are interpreted by the web server over TCP port 443. The web server then performs the HTTP request on the user's behalf to the application server, which runs the DVBM application software. The web server, application server, and a MySQL database server all run Linux. Firewalls prevent outbound connections from the web and application servers. Since the web server and application server run on separate machines, a compromise of the application server will not by itself allow an attacker to steal the HTTPS private key.

Voter experience The DVBM system was intended to be available to all military and overseas voters registered in the District. Months prior to the election, each eligible voter received a letter by postal mail containing credentials for the system. These credentials contained the voter ID number, registered name, residence ZIP code, and a 16-character hexadecimal personal identification number (PIN). One



(a) Select online or postal voting

(e) Download blank ballot

(b) Overview of steps

(f) Mark ballot in PDF reader and save

(c) Authenticate with voter ID / PIN

(g) Upload completed ballot

(d) "Affirm" identity

(h) "Thank you" screen

Fig. 2: **Screenshots of the D.C. voting system** show a typical voter's workflow. After opting to digitally return the ballot (a), the voter receives instructions (b) then enters credentials provided by postal mail (c) and attests to his identity (d). He then downloads a PDF file of the ballot (e), selects candidates and saves the file (f), and uploads the completed ballot to the server (g), which returns a confirmation screen (h).

instance of this letter is shown in Figure 5. The letters instructed voters to visit the D.C. Internet voting system website, which guided them through the voting process.

Figure 2 depicts the steps of the online voting user interface. Upon arrival, the voter selects between a digital or postal ballot return. Next, the voter is presented with an overview of the voting process. The voter then logs in with the credentials provided in the mail, and confirms his or her identity. Next, the voter is presented with a blank ballot in PDF format. In the postal return option, the voter simply prints out the ballot, marks it, and mails it to the provided address. For the digital return, the voter marks the ballot electronically using a PDF reader, and saves the ballot to his or her computer. The voter then uploads the marked ballot to the D.C. Internet voting system, which reports that the vote has been recorded by displaying a "Thank You" page. If voters try to log in a second time to cast another ballot, they are redirected to the final Thank You page, disallowing them from voting again.

3 Attacking the Web Application

In this section, we describe vulnerabilities we discovered and exploited in the DVBM server application. Our search for vulnerabilities was primarily conducted by manual inspection of the web application's source code, guided by a focus on the application's attack surface. In particular, we concentrated on voter login, ballot upload and handling, database communication, and other network activity. The fact that the application was open source expedited our search, but motivated attackers could have found vulnerabilities without the source code using alternative methods. For example, one might attack voter login fields, ballot contents, ballot filenames, or session cookies, by either fuzzing or more direct code injection attacks such as embedding snippets of SQL, shell commands, and popular scripting languages with detectable side effects.

3.1 Shell-injection vulnerability

After a few hours of examination, we found a shell injection vulnerability that eventually allowed us to compromise the web application server. The vulnerability was located in the code for encrypting voted ballots uploaded by users. The server stores uploaded ballots in a temporary location on disk, and the DVBM application executes the gpg command to encrypt the file, using the following code:

```
run("gpg", "--trust-model always -o
    \"#{File.expand_path(dst.path)}\" -e -r
    \"#{@recipient}\" \"#{File.expand_path(src.path)}\"")
```

The run method invoked by this code concatenates its first and second arguments, collapses multiple whitespace characters into single characters, and then executes the command string using Ruby's backtick operator, which passes the provided command to the shell. The Paperclip[3] Rails plugin, which the application uses to handle file uploads, preserves the extension of the uploaded ballot file, and no filtering is performed on this extension, so the result of `File.expand_path(src.path)` is attacker controlled. Unfortunately, in the Bash shell used on the server, double quotes do not prevent the evaluation of shell metacharacters, and so a ballot named `foo.$(cmd)` will result in the execution of `cmd` with the privileges of the web application.

The current release of the Paperclip plugin at the time of our analysis (late September 2010) was version 2.3.3. It appears that a similar vulnerability in Paperclip's built-in `run` method was fixed on April 30, 2010[4]. The first release containing the patch was version 2.3.2, which was tagged in the Paperclip Git repository on June 8, 2010. The degree of similarity between the DVBM application's custom `run` method and the Paperclip `run` method suggests that the DVBM application's implementation is a custom "stripped-down" version of Paperclip's, contrary to the D.C. BOEE's assertion that "a new version of [Paperclip] that had not been fully tested had been released and included in the deployed software" and "did not perform filename checks as expected." [14] Indeed, if DVBM had used the Paperclip `run` method together with an up-to-date version of the Paperclip library, this specific vulnerability would not have been included in the software. The resulting attack serves as a reminder that a small, seemingly minor engineering mistake in practically any layer of the software stack can result in total system compromise.

When we tested the shell injection vulnerability on the mock election server, we discovered that outbound network traffic from the test system was filtered, rendering traditional shellcode and exfiltration attempts (e.g., `nc umich.edu 1234 < /tmp/ballot.pdf`) ineffective. However, we were able to exfiltrate data by writing output to the `images` directory on the compromised server, where it could be retrieved with any HTTP client. To expedite crafting our shell commands, we developed an exploit compiler and a shell-like interface that, on each command, creates a maliciously named ballot file, submits the ballot to the victim server, and retrieves the output from its chosen URL under `/images`.

Interestingly, although the DVBM system included an intrusion detection system (IDS) device, it was deployed in front of the web server and was not configured to intercept and monitor the contents of the encrypted HTTPS connections that carried our attack. Although configuring the IDS with the necessary TLS certificates would no doubt have been labor intensive, failure to do so resulted in a large "blind spot" for the D.C. system administrators.

3.2 Attack payloads

We exploited the shell injection vulnerability to carry out several attacks that illustrate the devastating effects attackers could have during a real election if they gained a similar level of access:

[3] https://github.com/thoughtbot/paperclip

[4] The patch in question is available at https://github.com/thoughtbot/paperclip/commit/724cc7. It modifies `run` to properly quote its arguments using single quotes.

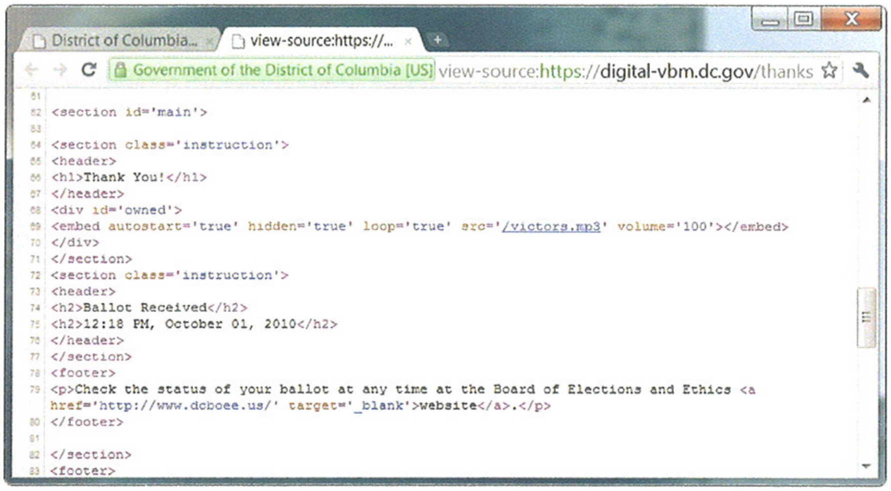
Stealing secrets We retrieved several cryptographic secrets from the application server, including the public key used for encrypting ballots. Despite the use of the term "public key," this key should actually be kept secret, since it allows attackers to substitute arbitrary ballots in place of actual cast ballots should they gain access to the storage device. We also gained access to the database by finding credentials in the bash history file (`mysql -h 10.1.143.75 -udvbm -pP@ssw0rd`).

Changing past and future votes We used the stolen public key to replace all of the encrypted ballot files on the server at the time of our intrusion with a forged ballot of our choosing. In addition, we modified the ballot-processing function to append any subsequently voted ballots to a `.tar` file in the publicly accessible `images` directory (where we could later retrieve them) and replace the originals with our forged ballot. Recovery from this attack is difficult; there is little hope for protecting future ballots from this level of compromise, since the code that processes the ballots is itself suspect. Using backups to ensure that compromises are not able to affect ballots cast prior to the compromise may conflict with ballot secrecy in the event that the backup itself is compromised.

Revealing past and future votes One of the main goals of a voting system is to protect ballot secrecy, which means not only preventing an attacker of the system from determining how a voter voted, but also preventing a voter from willingly revealing their cast ballot to a third party, even if they are coerced or incentivized to do so. While any absentee system that allows voters to vote where they choose allows a voter to reveal his or her vote voluntarily, our attack on the D.C. system allowed us to violate ballot secrecy and determine how nearly all voters voted.

Our modifications to the ballot processing function allowed us to learn the contents of ballots cast following our intrusion. Revealing ballots cast prior to our intrusion was more difficult, because the system was designed to store these ballots in encrypted form, and we did not have the private key needed to decipher them. However, we found that the Paperclip Rails plugin used to handle file uploads stored each ballot file in the `/tmp` directory before it was encrypted. The web application did not remove these unencrypted files, allowing us to recover them. While these ballots do not explicitly specify the voter's ID, they do indicate the precinct and time of voting, and we were able to associate them with voters by using login events and ballot filenames recorded in the server application logs. Thus, we could violate the secret ballot for past and future voters.

Discovering that real voter credentials were exposed In addition to decrypted ballots, we noticed that the `/tmp` directory also contained uploaded files that were not PDF ballots but other kinds of files apparently used to exercise error handling code during testing. To our surprise, one of these files was a 937 page PDF document that contained the instruction letters sent to each of the registered voters, which included the real voters' credentials for using the system. The first page of this file is shown in Figure 5. These credentials would have allowed us (or anyone else who penetrated the insecure server) to cast votes as these citizens in


District of Columbia...
view-source:https://...
Government of the District of Columbia [US]
view-source:https://digital-vbm.dc.gov/thanks

```
61
62 <section id='main'>
63
64 <section class='instruction'>
65 <header>
66 <h1>Thank You!</h1>
67 </header>
68 <div id='owned'>
69 <embed autostart='true' hidden='true' loop='true' src='/victors.mp3' volume='100'></embed>
70 </div>
71 </section>
72 <section class='instruction'>
73 <header>
74 <h2>Ballot Received</h2>
75 <h2>12:18 PM, October 01, 2010</h2>
76 </header>
77 </section>
78 <footer>
79 <p>Check the status of your ballot at any time at the Board of Elections and Ethics <a
   href='http://www.dcboee.us/' target='_blank'>website</a>.</p>
80 </footer>
81
82 </section>
83 <footer>
```

Fig. 3: **Musical "calling card"** — We modified the Thank You page that appears at the end of the voting process to play the University of Michigan fight song, "The Victors." Nevertheless, it took two business days for officials to become aware of the infiltration. Our additions appear on lines 68–70 above.

the real D.C. election that was to begin only days after the test period. Since the system requires that these credentials be delivered via postal mail, it would be infeasible for officials to send updated ones to the voters in time for the election.

Hiding our tracks We were able to hide the evidence of our intrusion with moderate success. We downloaded the DVBM application logs, altered them to remove entries corresponding to our malicious ballot uploads, and, as our final actions, overwrote the application log with our sanitized version and removed our uploaded files from the `/tmp` and `images` directories.

Our calling card To make our control over the voting system more tangible to nontechnical users, we left a "calling card" on the final screen of the digital voting workflow: we uploaded a recording of "The Victors" (the University of Michigan fight song) and modified the confirmation page to play this recording after several seconds had elapsed, as shown in Figure 3. We hoped that this would serve as a clear demonstration that the site had been compromised, while remaining discreet enough to allow the D.C. BOEE system administrators a chance to exercise their intrusion detection and response procedures.

3.3 Other vulnerabilities and potential attacks

Our intention in participating in the trial was to play the role of a real attacker. Therefore, once we had found vulnerabilities that allowed us to compromise the system, our attention shifted to understanding and exploiting these problems. However, along the way we did uncover several additional vulnerabilities in the

DVBM web application that were not necessary for our attack. Two key system deployment tasks were not completed. First, the set of test voter credentials was not regenerated and was identical to those included in the public DVBM Git repository. While the test voter credentials were fictitious, their disclosure constituted a security problem because public testers were asked to contact the D.C. BOEE for credentials, implying that the number of credentials available to each test group was to be limited.

Similarly, the encryption key used for session cookies was unchanged from the default key published in the repository. Disclosure of the key exacerbated a second vulnerability: rather than using the Rails-provided random `session_id` to associate browser sessions with voter credentials, the DVBM developers used the `rid` value, which corresponds to the automatically incremented primary key of the registration table in the system's MySQL database. This means every integer less than or equal to the number of registered voters is guaranteed to correspond to some voter. Combining this with the known encryption key results in a *session forgery* vulnerability. An attacker can construct a valid cookie for some voter simply by choosing an arbitrary valid `rid` value. This vulnerability could have been used to submit a ballot for every voter.

Our attack was expedited because the DVBM application user had permission to write the code of the web application. Without this permission, we would have had to find and exploit a local privilege escalation vulnerability in order to make malicious changes to the application. In fact, the version of the Linux kernel running on the application server (2.6.18-194.11.4.el5) had a known local root exploit (CVE-2010-3081) that could have allowed us to gain root privileges on the machine. As we were able to carry out our attacks as the web application user, we did not need to use this exploit.

We also identified other attack strategies that we ultimately did not need to pursue. For instance, the "crypto workstation" (see Section 2) used for decrypting and tabulating ballots is not directly connected to the Internet, but attackers may be able to compromise it by exploiting vulnerabilities in PDF processing software. PDF readers are notoriously subject to security vulnerabilities; indeed, the Crypto Workstation's lack of Internet connectivity may reduce its security by delaying the application of automated updates in the time leading up to the count. If the Crypto Workstation is compromised, attackers would likely be able to rewrite ballots. Furthermore, the web application allowed uploaded PDF ballots to contain multiple pages. If the printing is done in an automated fashion without restricting printouts to a single page, an attacker could vote multiple ballots.

4 Attacking the Network Infrastructure

In addition to the web application server, we were also able to compromise network infrastructure on the pilot network. This attack was independent from our web application compromise, yet it still had serious ramifications for the real election and showed a second potential path into the system.

Prior to the start of the mock election, the D.C. BOEE released a pilot network design diagram that showed specific server models, the network configuration connecting these servers to the Internet, and a CIDR network block (8.15.195.0/26). Using Nmap, we discovered five of the possible 64 addresses in this address block to be responsive. By using Nmap's OS fingerprinting feature and manually following up with a web browser, we were able to discover a Cisco router (8.15.195.1), a Cisco VPN gateway (8.15.195.4), two networked webcams (8.15.195.11 and 8.15.195.12), and a Digi Passport 8 terminal server[5] (8.15.195.8).

4.1 Infiltrating the terminal server

The Digi Passport 8 terminal server provides an HTTP-based administrative interface. We were able to gain access using the default root password (`dbps`) obtained from an online copy of the user manual. We found that the terminal server was connected to four enterprise-class Cisco switches (which we surmised corresponded to the switches shown on the network diagram provided by the BOEE) and provided access to the switches' serial console configuration interfaces via telnet.

We hid our presence in the terminal server using a custom JavaScript rootkit, which we installed over an SSH session (the same account names and passwords used in the web interface were accepted for SSH). The rootkit concealed an additional account with administrator privileges, "dev," which we planned to use in case our attack was discovered and the passwords changed. We also used our SSH access to download the terminal server's `/etc/shadow` and `/etc/passwd` files for cracking using the "John the Ripper" password cracker[6]. After about 3.5 hours using the cracker's default settings, we recovered the secondary administrator password `cisco123` from a salted MD5 hash.

Evidence of other attackers When we inspected the terminal server's logs, we noticed that several other attackers were attempting to guess the SSH login passwords. Such attacks are widespread on the Internet, and we believe the ones we observed were not intentionally directed against the D.C. voting system. However, they provide a reminder of the hostile environment in which Internet voting applications must operate.

The first SSH attack we observed came from an IP address located in Iran (80.191.180.102), belonging to Persian Gulf University. We realized that one of the default logins to the terminal server (user: `admin`, password: `admin`) would likely be guessed by the attacker in a short period of time, and therefore decided to protect the device from further compromise that might interfere with the voting system test. We used iptables to block the offending IP addresses and changed the admin password to something much more difficult to guess. We later blocked similar attacks from IP addresses in New Jersey, India, and China.

[5] A *terminal server* is a device that attaches to other pieces of equipment and allows administrators to remotely log in and configure them.

[6] http://www.openwall.com/john/

4.2 Routers and switches

After we compromised the terminal server, we found several devices connected to its serial ports. Initially, there were four Cisco switches: a pair of Nexus 5010s and a pair of Nexus 7010s. Connecting to these serial ports through the terminal server presented us with the switches' login prompts, but previously found and default passwords were unsuccessful.

The terminal server provided built-in support for keystroke logging of serial console sessions and forwarding of logged keystrokes to a remote syslog server, which we enabled and configured to forward to one of our machines. This allowed us to observe in real time as system administrators logged in and configured the switches, and to capture the switches' administrative password, `!@#123abc`.

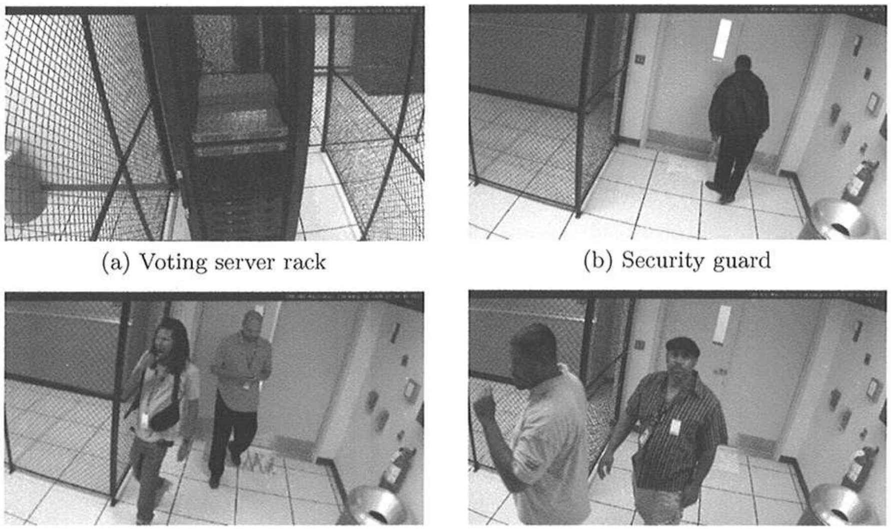
Later in the trial, four additional devices were attached to the terminal server, including a pair of Cisco ASR 9010 routers and a pair of Cisco 7606-series routers. We were again able to observe login sessions and capture passwords. At the end of the public trial, we changed the passwords on the routers and switches—effectively locking the administrators out of their own network—before alerting BOEE officials and giving them the new password.

D.C. officials later told us that the routers and switches we had infiltrated were not intended to be part of the voting system trial and were simply colocated with the DVBM servers at the District's off-site testing facility. They were, however, destined to be deployed in the core D.C. network, over which real election traffic would flow. With the access we had, we could have modified the devices' firmware to install back doors that would have given us persistent access, then later programmed them to redirect Internet voting connections to a malicious server.

4.3 Network webcams

We found a pair of webcams on the DVBM network—both publicly accessible without any password—that showed views of the server room that housed the pilot. As shown in Figure 4, one camera pointed at the entrance to the room, and we were able to observe several people enter and leave, including a security guard, several officials, and IT staff new hardware. The second camera was directed at a rack of servers.

These webcams may have been intended to increase security by allowing remote surveillance of the server room, but in practice, since they were unsecured, they had the potential to leak information that would be extremely useful to attackers. Malicious intruders viewing the cameras could learn which server architectures were deployed, identify individuals with access to the facility in order to mount social engineering attacks, and learn the pattern of security patrols in the server room. We used them to gauge whether the network administrators had discovered our attacks—when they did, their body language became noticeably more agitated.



(a) Voting server rack

(b) Security guard

(c) Typical workers, before attack

(d) Workers, after learning of attack

Fig. 4: **Unsecured network surveillance cameras** gave us a real-time view into the network operations center. We could observe whether administrators made physical changes to the servers running the voting system (*a*) and monitor the frequency of patrols by security guards (*b*). We inferred that our attack had not been detected based on the relaxed body language of workers in the facility, e.g. (*c*), which changed dramatically after the BOEE learned of our intrusion (*d*).

5 Discussion

5.1 Attack detection and recovery

After we completed our attack—including our musical calling card on the "Thank You" page—there was a delay of approximately 36 hours before election officials responded and took down the pilot servers for analysis. The attack was apparently brought to officials' attention by an email on a mailing list they monitored that curiously asked, "does anyone know what tune they play for successful voters?" Shortly after another mailing list participant recognized the music as "The Victors," officials abruptly suspended the public examination period, halting the tests five days sooner than scheduled, citing "usability issues."

Following the trial, we discussed the attack with D.C. officials. They explained that they found our modifications to the application code by comparing the disk image of the server to a previous snapshot, although this required several days of analysis. They confirmed that they were unable to see our attacks in their intrusion detection system logs, that they were unable to detect our presence in the network equipment until after the trial, and that they did not discover the attack until they noticed our intentional calling card. We believe that attack detection and recovery remain significant challenges for any Internet voting system.

5.2 Adversarial testing and mechanics of the D.C. trial

The D.C. BOEE should be commended for running a public test of their system. Their trial was a step in the right direction toward transparency in voting technology and one of the first of its kind. Nonetheless, we reiterate that adversarial testing of Internet voting applications is not necessary to show that they are likely to be weak. The architectural flaws inherent in Internet voting systems in general and the potential disastrous implications of a single vulnerability were known and expected by researchers prior to the D.C. trial [11]. We hope not to have to repeat this case study in order to highlight these limitations once again.

The key drawback to adversarial testing is that a lack of problems found in testing *does not* imply a lack of problems in the system, despite popular perception to the contrary. It is likely that testers will have more limited resources and weaker incentives than real attackers — or they may simply be less lucky. A scarcity of testers also seems to have been an issue during the D.C. trial. During our compromise of the DVBM server, we were able to view the web access logs, which revealed only a handful of attack probes from other testers, and these were limited to simple failed SQL and XSS injection attempts.

One reason for the lack of participation may have been ambiguity over the legal protections provided to testers by the BOEE. Another possible reason is that the test began on short notice — the final start date was announced only three days in advance. If such a trial must be repeated, we hope that the schedule will be set well in advance, and that legal protections for participants will be strongly in place. In addition to the short notice, the scheduled conclusion of the test was only three days before the system was planned to be opened for use by real voters. Had the test outcome been less dramatic, election officials would have had insufficient time to thoroughly evaluate testers' findings.

Despite these problems, one of the strongest logistical aspects of the D.C. trial was that access to the code — and to some extent, the architecture — was available to the testers. While some observers have suggested that this gave us an unrealistic advantage while attacking the system, there are several reasons why such transparency makes for a more realistic test. Above and beyond the potential security benefits of open source code (pressure to produce better code, feedback from community, etc.), in practice it is difficult to prevent a motivated attacker from gaining access to source code. The code could have been leaked by the authors through an explicit sale by dishonest insiders, as a result of coercion, or through a compromised developer workstation. Since highly plausible attacks such as these are outside the scope of a research evaluation, it is not only fair but realistic to provide the code to the testers.

5.3 Why Internet voting is hard

Practical Internet voting designs tend to suffer from a number of fundamental difficulties, from engineering practice to inherent architectural flaws. We feel it is important to point them out again given the continued development of Internet voting systems.

Engineering practice Both the DVBM system and the earlier prototype Internet voting system SERVE [11] were built primarily on commercial-off-the-shelf (COTS) software (which, despite the use of the term "commercial," includes most everyday open-source software). Unfortunately, the primary security paradigm for COTS developers is still "penetrate and patch." While this approach is suitable for the economic and risk environment of typical home and business users, it is not appropriate for voting applications due to the severe consequences of failure.

Inherited DRE threats Relatively simple Internet voting systems like D.C.'s DVBM strongly resemble direct recording electronic (DRE) voting machines, in that there is no independent method for auditing cast ballots. If the voting system software is corrupt, recovery is likely to be impossible, and even detection can be extremely difficult. DRE voting is highly susceptible to insider attacks as well as external compromise through security vulnerabilities. In previous work [7,8,10,13,17], the closed, proprietary nature of DREs has been held as an additional threat to security, since there is no guarantee that even the *intended* code is honest and correct. In contrast, the DVBM system was open source, but the public would have had no guarantee that the deployed voting system was actually running the published code.

Tensions between ballot secrecy and integrity One of the fundamental reasons that voting systems are hard to develop is that two fundamental goals of a secret ballot election—ballot secrecy and ballot integrity—are in tension. Indeed, the D.C. system attempted to protect integrity through the use of logs, backups and intrusion detection, yet these systems can help an intruder compromise ballot secrecy. Other security mechanisms put in place to protect ballot secrecy, such as encrypting completed ballots and avoiding incremental backups make detecting and responding to compromise much more difficult.

Architectural brittleness in web applications The main vulnerability we exploited resulted from a tiny oversight in a single line of code and could have been prevented by using single quotes instead of double quotes. Mistakes like this are all too common. They are also extremely hard to eradicate, not because of their complexity, but because of the multitude of potential places they can exist. If any one place is overlooked, an attacker may be able to leverage it to gain control of the entire system. In this sense, existing web application frameworks tend to be *brittle*. As our case study shows, the wrong choice of which type of quote to use—or countless other seemingly trivial errors—can result in an attacker controlling the outcome of an election.

Internet-based threats Internet voting exposes what might otherwise be a small, local race of little global significance to attackers from around the globe, who may act for a wide range of reasons varying from politics to financial gain to sheer malice. In addition to compromising the central voting server as we did, attackers can launch denial-of-service attacks aimed at disrupting the election, they can redirect voters to fake voting sites, and they can conduct widespread attacks on voters' client machines [9]. These threats correspond to some of the most difficult unsolved problems in Internet security and are unlikely to be overcome soon.

Comparison to online banking While Internet-based financial applications, such as online banking, share some of the threats faced by Internet voting, there is a fundamental difference in ability to deal with compromises after they have occurred. In the case of online banking, transaction records, statements, and multiple logs allow customers to detect specific fraudulent transactions and in many cases allow the bank to reverse them. Internet voting systems cannot keep such fine-grained transaction logs without violating ballot secrecy for voters. Even with these protections in place, banks suffer a significant amount of online fraud but write it off as part of the cost of doing business; fraudulent election results cannot be so easily excused.

6 Related Work

Although this is, to the best of our knowledge, the first public penetration test of an Internet voting system scheduled for use in a general election, we are not the first to caution against the adoption of Internet voting.

The most closely related work is the 2004 security analysis of the Secure Electronic Registration and Voting Experiment (SERVE) by Jefferson et al. [11]. Like the D.C. DVBM project, SERVE was an Internet voting "pilot" that was slated for use in an actual election by absentee overseas voters. Jefferson et al. reviewed the system design and pointed out many architectural and conceptual weaknesses that apply to remote Internet voting systems in general, though they did not have an opportunity to conduct a penetration test of a pilot system. On the basis of these weaknesses, Jefferson et al. recommended "shutting down the development of SERVE immediately and not attempting anything like it in the future until both the Internet and the world's home computer infrastructure have been fundamentally redesigned." We emphatically reaffirm that recommendation. Despite incremental advances in computer security in the last eight years, the fundamental architectural flaws Jefferson et al. identified remain largely the same to this day.

More recently, Esteghari and Desmedt [9] developed an attack on the Helios 2.0 [6] open-audit Internet voting system. Their attack exploits an architectural weakness in home computer infrastructure by installing a "browser rootkit" or "man-in-the-browser attack" that detects the ballot web page and modifies votes. Esteghari and Desmedt note that Helios 3.0 is capable of posting audit information to an external web server *before* ballot submission, which can, in theory, be checked using a second trusted computer to detect the action of the rootkit, but it is not clear that such a second computer will be available or a sufficiently large number of nontechnical voters will take advantage of this audit mechanism.

7 Conclusions

Our experience with the D.C. pilot system demonstrates one of the key dangers in many Internet voting designs: one small mistake in the configuration or implementation of the central voting servers or their surrounding network infrastructure can easily undermine the legitimacy of the entire election. We expect that other fielded Internet voting systems will fall prey to such problems, especially if they are developed using standard practices for mass-produced software and websites. Even if the central servers were somehow eliminated or made impervious to external attack, Internet voting is likely to be susceptible to numerous classes of threats, including sabotage from insiders and malware placed on client machines. The twin problems of building secure software affordably and preventing home computers from falling prey to malware attacks would both have to be solved before systems like D.C.'s could be seriously considered. Although new end-to-end verifiable cryptographic voting schemes have the potential to reduce the trust placed in servers and clients, these proposals are significantly more advanced than systems like D.C.'s and may prove even more difficult for developers and election officials to implement correctly. Securing Internet voting in practice will require significant fundamental advances in computer security, and we urge Internet voting proponents to reconsider deployment until and unless major breakthroughs are achieved.

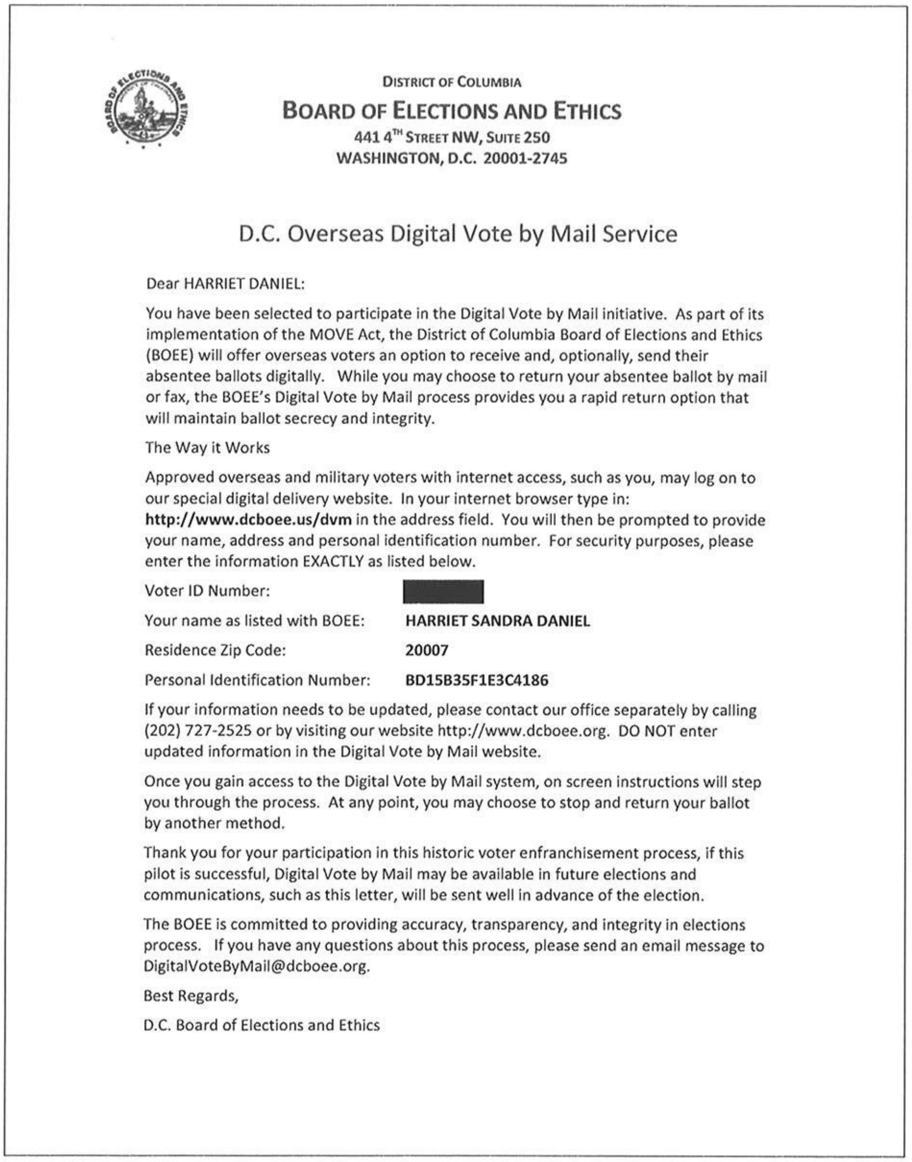
Acknowledgments

We are grateful to the many people who helped make this work possible, including Jeremy Epstein, Susannah Goodman, Nadia Heninger, David Jefferson, Bryan Sivak, Pamela Smith, David Robinson, and especially Joseph Lorenzo Hall. We thank the anonymous reviewers for their constructive feedback and Konstantin Beznosov for shepherding this paper to publication. The authors also wish to thank Rokey Suleman and Paul Stenbjorn of the D.C. BOEE for having the courage to allow the public to test its voting system.

References

1. Internet voting in Estonia. Vabariigi Valimiskomisjon. http://www.vvk.ee/public/dok/Internet_Voting_in_Estonia.pdf, Feb. 2007.
2. Uncovering the veil on Geneva's Internet voting solution. Republique Et Canton De Geneve http://www.geneve.ch/evoting/english/doc/Flash_IT_vote_electronique_SIDP_final_english.pdf, Feb. 2009.
3. District of Columbia's Board of Elections and Ethics adopts open source digital voting foundation technology to support ballot delivery. OSDV Press Release. http://osdv.org/wp-content/uploads/2010/06/osdv-press-release-final-62210.pdf, June 2010.
4. Internet voting, still in beta. The New York Times editorial. http://www.nytimes.com/2010/01/28/opinion/28thu4.html, Jan. 2010.
5. Internet voting. Verified Voting. http://www.verifiedvoting.org/article.php?list=type&type=27, May 2011.

6. Adida, B. Helios: Web-based open-audit voting. In *Proc. 17th USENIX Security Symposium* (July 2008).
7. Appel, A. W., Ginsburg, M., Hursti, H., Kernighan, B. W., Richards, C. D., Tan, G., and Venetis, P. The New Jersey voting-machine lawsuit and the AVC Advantage DRE voting machine. In *Proc. 2009 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE)* (Aug. 2009).
8. Butler, K., Enck, W., Hursti, H., McLaughlin, S., Traynor, P., and McDaniel, P. Systemic issues in the Hart InterCivic and Premier voting systems: Reflections on project EVEREST. In *Proc. 2008 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE)* (July 2008).
9. Esteghari, S., and Desmedt, Y. Exploiting the client vulnerabilities in Internet e-voting systems: Hacking Helios 2.0 as an example. In *Proc. 2010 Electronic Voting Technology Workship / Workshop on Trustworthy Elections (EVT/WOTE)* (Aug. 2010).
10. Feldman, A. J., Halderman, J. A., and Felten, E. W. Security analysis of the Diebold AccuVote-TS voting machine. In *Proc. 2007 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections (EVT/WOTE)* (Aug. 2007).
11. Jefferson, D., Rubin, A. D., Simons, B., and Wagner, D. A security analysis of the secure electronic registration and voting experiment (SERVE). http://servesecurityreport.org/paper.pdf, Jan. 2004.
12. Kiayias, A., Korman, M., and Walluck, D. An Internet voting system supporting user privacy. In *22nd Annual Computer Security Applications Conference.*
13. Kohno, T., Stubblefield, A., Rubin, A. D., and Wallach, D. S. Analysis of an electronic voting system. In *IEEE Symposium on Security and Privacy* (May 2004), pp. 27–40.
14. Rokey W. Suleman, I., McGhie, K. W., Togo D. West, J., and Lowery, C. Making reform a reality: An after-action report on implementation of the Omnibus Election Reform Act. DCBOEE. http://www.dcboee.org/popup.asp?url=/pdf_files/nr_687.pdf, Feb. 2011.
15. Rubin, A. Security considerations for remote electronic voting over the Internet. http://avirubin.com/e-voting.security.html.
16. Stenbjorn, P. An overview and design rationale memo. DCBOEE. http://www.dcboee.us/dvm/DCdVBM-DesignRationale-v3.pdf, Sept. 2010.
17. Wolchok, S., Wustrow, E., Halderman, J. A., Prasad, H. K., Kankipati, A., Sakhamuri, S. K., Yagati, V., and Gonggrijp, R. Security analysis of India's electronic voting machines. In *Proc. 17th ACM Conference on Computer and Communications Security (CCS)* (Oct. 2010).



DISTRICT OF COLUMBIA
BOARD OF ELECTIONS AND ETHICS
441 4TH STREET NW, SUITE 250
WASHINGTON, D.C. 20001-2745

D.C. Overseas Digital Vote by Mail Service

Dear HARRIET DANIEL:

You have been selected to participate in the Digital Vote by Mail initiative. As part of its implementation of the MOVE Act, the District of Columbia Board of Elections and Ethics (BOEE) will offer overseas voters an option to receive and, optionally, send their absentee ballots digitally. While you may choose to return your absentee ballot by mail or fax, the BOEE's Digital Vote by Mail process provides you a rapid return option that will maintain ballot secrecy and integrity.

The Way it Works

Approved overseas and military voters with internet access, such as you, may log on to our special digital delivery website. In your internet browser type in: **http://www.dcboee.us/dvm** in the address field. You will then be prompted to provide your name, address and personal identification number. For security purposes, please enter the information EXACTLY as listed below.

Voter ID Number:	[redacted]
Your name as listed with BOEE:	**HARRIET SANDRA DANIEL**
Residence Zip Code:	**20007**
Personal Identification Number:	**BD15B35F1E3C4186**

If your information needs to be updated, please contact our office separately by calling (202) 727-2525 or by visiting our website http://www.dcboee.org. DO NOT enter updated information in the Digital Vote by Mail website.

Once you gain access to the Digital Vote by Mail system, on screen instructions will step you through the process. At any point, you may choose to stop and return your ballot by another method.

Thank you for your participation in this historic voter enfranchisement process, if this pilot is successful, Digital Vote by Mail may be available in future elections and communications, such as this letter, will be sent well in advance of the election.

The BOEE is committed to providing accuracy, transparency, and integrity in elections process. If you have any questions about this process, please send an email message to DigitalVoteByMail@dcboee.org.

Best Regards,

D.C. Board of Elections and Ethics

Fig. 5: **Voter instructions and credentials** — D.C. overseas voters received letters like this, containing instructions and credentials for using the online voting system. The letters, which were mailed prior to the pilot test, assert that the system would "maintain ballot secrecy and integrity." After we infiltrated the pilot server, we discovered a PDF file, apparently uploaded during testing, that contained all 937 letters sent to actual voters, including the secret credentials. (This is the first page from that file; we have redacted the voter ID number for privacy.) It would have been impossible for D.C. to provide new credentials to all voters in time for the upcoming election.

CONTRIBUTED ARTICLES

Internet Voting in the U.S.

By Barbara Simons, Douglas W. Jones

Communications of the ACM, Vol. 55 No. 10, Pages 68-77
10.1145/2347736.2347754
Comments (2)

The assertion that Internet voting is the wave of the future has become commonplace. We frequently are asked, "If I can bank online, why can't I vote online?" The question assumes that online banking is safe and secure. However, banks routinely and quietly replenish funds lost to online fraud in order to maintain public confidence.

Key Insights

PRINT

- [illegible]t voting is fundamentally insecure.
- Most people do not associate widely publicized computer viruses and worms with Internet voting.
- Internet voting is being pushed in many countries by vendors, election officials, and well-meaning people who do not understand the risks.

We are told Internet voting would help citizens living abroad or in the military who currently have difficulty voting. Recent federal legislation to improve the voting process for overseas citizens is a response to that problem. The legislation, which has eliminated most delays, requires states to provide downloadable blank ballots but does not require the insecure return of voted ballots.

Yet another claim is that email voting is safer than Web-based voting, but no email program in widespread use today provides direct support for encrypted email. As a result, attachments are generally sent in the clear, and email ballots are easy to intercept and inspect, violating voters' right to a secret ballot. Intercepted ballots may be modified or discarded without forwarding. Moreover, the ease with which a `From` header can be forged means it is relatively simple to produce large numbers of forged ballots. These special risks faced by email ballots are in addition to the general risks posed by all Internet-based voting schemes.[17]

Many advocates also maintain that Internet voting will increase voter participation, save money, and is safe. We find the safety argument surprising in light of frequent government warnings of cybersecurity threats and news of powerful government-developed viruses. We see little benefit in measures that might improve voter turnout while casting doubt on the integrity of the results.[a]

Almost all the arguments on behalf of Internet voting ignore a critical risk Internet-based voting shares with all computerized voting—wholesale theft. In the days of hand-counted paper ballots, election theft was conducted at the retail level by operatives at polling places and local election offices. By contrast, introduction of computers into the voting process created the threat that elections can be stolen by inserting malware into code on large numbers of machines. The situation is even more dangerous with Internet voting, since both the central servers and the voters' computers are potentially under attack from everywhere.

Despite the serious threats it poses to election integrity, Internet voting is being used in several countries and U.S. states, and there is increasing public pressure to adopt it elsewhere. We examine some of these threats, in the hope of encouraging the technical community to oppose Internet voting unless and until the threats are eliminated.

D.C. pilot test Internet voting has generally been deployed without being subjected to public testing prior to use. To the best of our knowledge, the only exception was a "digital vote by mail" pilot project in Washington, D.C. in 2010. In June of that year, the Open Source Digital Voting Foundation announced that it had been selected by the District of Columbia Board of Elections and Ethics (BOEE) to support a project to allow Internet voting for military and overseas voters, starting with the upcoming September primary. The BOEE had optimistically planned a "public review period" in advance of the primary in which everyone was invited to try to attack the system in a mock election. While the system was not ready for the primary, a public test was eventually scheduled to run from September 28 to October 6, with midterm election voting scheduled to begin October 11 or 12.

The break-in. By October 1 people testing the system reported hearing the University of Michigan fight song following a 15-second pause after they submitted their ballots.[6,44] A Michigan team had taken over the system within 36 hours of the start of the tests by exploiting a shell-injection vulnerability, thereby gaining almost total control over the BOEE server. The attackers remained in control for two business days, until the BOEE halted the test after noon on October 1. An attacker intent on subverting a real election would not leave such an obvious calling card. The delay between the break-in and the shutdown of the system reveals how difficult it is to determine that a break-in has occurred, even when the "culprits" announce themselves with music.

On October 5, Michigan professor Alex Halderman revealed that, in addition to installing the fight song, his team had changed ballots cast prior to their intrusion, had rigged the system to alter subsequently cast ballots, and could violate voters' secret ballot rights. That day the BOEE restarted the test with the song removed. Testers were told to print out and mail in their ballots, instead of returning them over the Internet. **Figure 1** is the hacked ballot, with write-in candidates selected by the Michigan team.

Halderman was the star of an October 8 oversight hearing, where he dropped additional bombshells. From the start, his team had control of the network infrastructure for the pilot project. The team used the default master password from the owner's manuals, which had not been changed, for the routers and switches, thereby gaining control of the infrastructure and obtaining an alternative way to steal votes in a real election. Control of the network also enabled the team to watch network operators configure and test the equipment. When they discovered that a pair of security cameras in the BOEE data center was connected to the pilot system and unprotected, the team used the cameras to watch the system operators. As proof, Halderman brought some security-camera photos to the hearing. Halderman even discovered a file used to test the system that consisted of copies of all 937 letters sent to real voters. The letters included voter names, IDs, and 16-character PINs for authentication in the real Internet election. While the team could already change voter selections, inclusion of unencrypted PINs in a file used for testing demonstrates that the BOEE did not understand the fundamental principles of computer security. The PINs would have allowed the team or any other intruder to cast ballots for actual voters. Finally, Halderman found evidence of attempted break-ins that appeared to be from China and Iran. Since the attempts involved trying to guess the network logins, the Michigan team changed the previously unchanged defaults (user: *admin*, password: *admin*). Whether or not they were intentionally directed at the D.C. voting system, the attempts showed how dangerous the Internet can be, with sophisticated adversaries from around the world constantly trying to break in to systems.

Implications of the attack. The D.C. incursion illustrates how Internet voting can be attacked from anywhere. Most complex software systems have an abundance of vulnerabilities, with attackers needing to exploit just one. Moreover, all attacks except those specifically targeting the designated BOEE election network were out of bounds in the pilot test. Examples of non-allowed attacks included client-side malware; denial-of-service attacks; attacks against ISPs; and DNS, routing, and other network attacks. Attackers in a real election would not have felt bound by such constraints. Once the Michigan team had changed all the votes, it was impossible for D.C. officials to reconstruct the original ballots. In a close race, attackers might control the outcome without risk of detection. It took more than a day for D.C. officials to realize their system had been successfully attacked, despite the musical calling card. By the time officials discovered the attack, it was too late to recover from it.

The BOEE had intended to accept voted ballots over the Internet. If there had been no pilot test or if the Michigan team had not participated, members of the military and civilians living abroad who vote in Washington, D.C. would have been voting over a highly vulnerable system. The BOEE did the right thing (for a municipality determined to deploy Internet voting) by setting up a public test. It also learned an important lesson from the test and ultimately canceled the Internet-ballot-return portion. Voters were instead allowed to download blank ballots from the Web and print and return them by postal mail. Unfortunately, other states have not been as responsible. In the upcoming 2012 U.S. election, 33 states will allow some kind of Internet voting, including at least one Web-based Internet pilot project, and the return of voted ballots over the Internet through email attachment or fax, without first encouraging independent experts to test their systems.[42]

One of us (Jones) has consulted with several election offices, including the BOEE. He observed it to be above average, in terms of both physical and human resources, suggesting that the mistakes found by the Michigan team were not the result of isolated incompetence, but are typical of the best we can expect under current conditions. Likewise, Halderman has said that the quality of the D.C. source code seemed much better than the closed-source electronic voting systems he has examined. Security is difficult, and even organizations with security expertise have been successfully attacked. Given that elections offices are under-resourced, have many other problems to worry about, lack security expertise, and are highly decentralized, it is completely unrealistic to expect extraordinary security competence from them.

The Case for Internet Voting

Despite warnings from independent studies and commissions, as well as sensational news stories about hacking and viruses, some widely held misconceptions about Internet voting persist: It saves money and increases voter turnout; Web-based voting is more secure than postal voting or voting by email or fax; because banking and purchasing can be done over the Internet, voting can be done safely over the Internet; and Internet voting is inevitable—the wave of the future. We discuss the first three points in the following sections and the fourth in the sidebar "**Internet Voting and E-Commerce Compared**." Regarding the inevitability of Internet voting, some of the most outspoken Internet voting opponents are highly respected computer security experts. Our goal is to convince you that secure Internet voting is unachievable for the foreseeable future and therefore, we sincerely hope, not inevitable.

Saves money. The cost of Internet voting, especially up-front charges, can be steep. For example, 2009 cost estimates from Internet voting vendor Everyone Counts were so large that a legislative proposal in Washington state to allow Internet voting for military and civilian voters was killed in committee. The estimated costs, obtained by John Gideon of VotersUnite, included proposed up-front costs ranging from $2.5 million to $4.44 million. After that, each county would have been hit with an annual license fee of $20,000-$120,000, plus $2-$7 per overseas voter.[5]

In the March 2011 election in the state of New South Wales, Australia, 46,864 people voted on an Internet voting system called iVotes, also an Everyone Counts product.[33] The development and implementation costs for using iVotes in the election exceeded $3.5 million (Australian dollars), resulting in a cost of about $74 per vote cast. By contrast, the average cost for all forms of voting in the same election was $8 per vote, though the cost per Internet vote would have decreased if amortized over more voters.

Increases turnout. Internet voting does not necessarily increase turnout. Everyone Counts ran an Internet-based election in Swindon, U.K., in 2007 and a local election in Honolulu, HI, in 2009 where votes were cast only by Internet or telephone. The Electoral Commission, established by the U.K. Parliament, determined that Internet voting in Swindon had a negligible effect on turnout; meanwhile, in Honolulu there was an 83% drop in turnout compared to a similar election in 2007.[22,40] We know of no rigorous study of the impact of Internet voting on turnout; conducting such a study would be difficult, since turnout can vary enormously from election to election. But even if Internet voting could increase turnout, the increase would be irrelevant if the election results were at risk of corruption by insecure Internet use.

Web-based voting is more secure. Verifiability and transparency are critical aspects of any election, especially if it involves a secret ballot. It is fundamentally impossible for anyone, even election officials, to directly oversee or observe the tabulation of an Internet-based election, including one that is Web-based. A software bug or an attack could cause an election outcome to be wrong because either the tabulation is incorrect or the voters' selections were modified. To address such risks, we need to determine after an election that the technology operated correctly and the declared winner actually won.

We can verify the results of a paper-based election by auditing a sample of the cast ballots or, in the extreme, by recounting all of them. Such an audit or recount must involve a secure, observable chain of custody of the ballots, something impossible with current Internet voting technology. Allowing voters to print copies of their ballots for personal use is meaningless, because these copies may not match the electronic versions used in computing the results.

Military Voting

Members of uniformed services and their families and non-military citizens living overseas are called UOCAVA voters, after the U.S. Uniformed and Overseas Citizens Absentee Voting Act of 1986 (**http://www.fvap.gov/reference/laws/uocava.html (http://www.fvap.gov/reference/laws/uocava.html)**). They have long complained that absentee ballots are never delivered or their returned voted ballots arrive too late to be counted, concerns used to justify the push for Internet voting at both the state and federal levels. A widely discussed solution is to have the military run its own centralized Internet voting system over its high-security infrastructure. This is a bad idea for at least two reasons: First, it runs counter to the principle of civilian control over the military and creates the potential that the military might control the vote. Second, it is unrealistic and unwise to even consider connecting unsecure Web servers run by local election officials to a military network that is supposed to maintain a high level of security. Some supporters of Internet voting for the military have noted that postal mail ballots are also not secure. While it is true that all forms of remote voting pose security problems, Internet voting can be attacked by anyone from anywhere, something that is not the case for postal ballots. In addition, the Internet can be used for wholesale attacks on large numbers of voters, whereas attacks on postal ballots are inherently confined to a retail scale.

Two projects for UOCAVA voters are noteworthy: SERVE, killed in 2004, and Operation BRAVO, implemented in the 2008 U.S. presidential election:

SERVE. The Secure Electronic Registration and Voting Experiment, or SERVE (**www.fvap.gov/resources/media/serve.pdf (http://www.fvap.gov/resources/media/serve.pdf)**), was the most ambitious project to date intended for use by UOCAVA voters. The goal of the $22 million project was to allow registration and voting over the Internet in the 2004 primaries and general election. Participation by states and counties within those states was voluntary. Voters could use any Windows computer, either their own or a public computer, like those found in libraries and cyber-cafés. Voters were responsible for the security of whatever computers they used. The vendor was Accenture.

In 2003, a group of experts called the Security Peer Review Group was assembled by the Federal Voting Assistance Program (FVAP) to evaluate SERVE; FVAP was charged with facilitating voting for all UOCAVA voters. Following two three-day meetings with FVAP and the lead technical staff of SERVE, the four computer scientists who attended both meetings, including one of us (Simons), released a report, the conclusion of which said: "Because the danger of successful, large-scale attacks is so great, we reluctantly recommend shutting down the development of SERVE immediately and not attempting anything like it in the future until both the Internet and the world's home computer infrastructure have been fundamentally redesigned, or some other unforeseen security breakthroughs appear."[18]

When the report was issued in early 2004, 50 counties in seven states—Arkansas, Florida, Hawaii, North Carolina, South Carolina, Utah, and Washington—were planning to participate in SERVE. FVAP had estimated the maximum overall vote total would be approximately 100,000, including primaries and the general election. On January 30, 2004 Deputy Secretary of Defense Paul Wolfowitz said the Pentagon "...will not be using the SERVE Internet voting project in view of the inability to assure legitimacy of votes that would be cast using the system, which thereby brings into doubt the integrity of election results."[43] SERVE was subsequently terminated.

Operation BRAVO. In 2008, Operation BRAVO, or Bring Remote Access to Voters Overseas, provided Internet voting from secure kiosks for residents of Okaloosa County, FL. Unlike previous pilot projects, these kiosks were equipped with printers to create paper voter-choice records of voters' ballots. Voters could verify the records before leaving the kiosk, after which the records were flown back to Okaloosa County for manual reconciliation with the ballots sent over an Internet-based virtual private network. Small discrepancies in the ballot count were uncovered by law professor Martha Mahoney of the University of Miami, but, as of August 2012, BRAVO had yet to release a formal report explaining the discrepancies.[26] The vendor was Scytl.

The Okaloosa County experiment concerned only a single county. Expanding kiosk-based Internet voting for all service members would be very difficult, since the system would have to deal with tens of thousands of different ballot styles and conflicting state rules governing ballot presentation, requirements that would also add significantly to the cost.

The MOVE Act. Instead of Internet voting, why not allow remote voters to download a blank ballot from the Internet, print it, and return the voted ballots by mail? If the blank ballots are available early enough, most voted ballots should arrive in time to be counted. Such a system might not have the pizzazz of Internet voting but would have fewer security issues and almost certainly involve less cost. That is one of the reforms dictated by the 2009 Military and Overseas Voter Empowerment, or MOVE, Act. Written to address the problems of UOCAVA voters, MOVE requires states to make blank ballots available electronically at least 45 days prior to any federal election; UOCAVA voters may also request and receive voter-registration and absentee-ballot applications electronically.

The Military Postal Service Agency analyzed the handling of absentee ballots during the 2010 general election,[29] finding problems with getting postal ballots to members of the military, though paper ballots were generally returned quickly. Many had been electronically downloaded, filled out by service members, and returned by postal mail. The average postal delay for returned ballots was 5.2 days, well ahead of the seven-day limit set by the MOVE Act; 92% of absentee ballots were delivered within seven days of acceptance at overseas Military Post Offices (MPOs). Only 118 out of 23,900 voted ballots, most likely from Afghanistan or Iraq, took 20 or more days to be returned from an MPO. The time to get a voted ballot from a service member to an MPO ranged from two to 20 days. Therefore, if election officials provide downloadable blank ballots at least 45 days before an election, essentially all members of the military should be able to return their voted paper ballots in time to be counted.

Risks

Not satisfied with the significant speed-up provided by MOVE, Internet-voting advocates continue to call for the return of voted ballots through the Internet, either as email attachments or as some kind of Web form. Doing either securely would require solving some of the most intractable problems in cybersecurity:

The server. In the 2010 D.C. pilot project, University of Michigan graduate students attacked the election server over the Internet. Independent hackers, political operatives, foreign governments, and terrorists could also mount such attacks. Local election officials with little or no expertise in computer security have little hope of defending themselves.

Corporate and government vulnerability. Many corporations and government agencies store sensitive or classified information on their computers, sharing with election officials the goal of defending against attackers who might steal or alter such information. Despite large staffs of security professionals with significant resources, computers in major corporations and government agencies have been attacked successfully. For example, a 2008 survey of approximately 1,000 large organizations worldwide found the average loss per organization from intellectual property cybertheft was about $4.6 million.[19] A December 2009 report from the Computer Security Institute (**http://gocsi.com (http://gocsi.com)**) surveying 443 U.S. companies and government agencies found 64% had reported malware infections during the preceding year.[36]

A major China-based Internet attack on Google and many other companies in late 2009 showed that even major corporate sites are vulnerable. The attack targeted Gmail accounts of Chinese human-rights activists and Google's own intellectual property, including software-development systems.[31] As many as 34 companies were targeted, including Adobe, Juniper Networks, defense contractor Northrop-Grumman, major security supplier Symantec, and Yahoo!.[41] The attacked companies have vastly more security expertise and resources than local election officials or today's relatively small Internet voting vendors. The attacks used email that appeared to come from trusted sources, so victims would be tricked into clicking on a link or opening an attachment. Then, using a vulnerability in Microsoft's Internet Explorer browser, the attacker would download and install malware that took complete control of the compromised systems.

George Kurtz, executive vice president and worldwide chief technology officer of McAfee, an Internet security company, expressed dismay at the implications: "All I can say is wow. The world has changed. Everyone's threat model now needs to be adapted to the new reality of these advanced persistent threats. In addition to worrying about Eastern European cyber-criminals trying to siphon off credit card databases, you have to focus on protecting all of your core intellectual property, private nonfinancial customer information and anything else of intangible value."[23]

Government sites have also been vulnerable. In a March 2010 address to the RSA Security Conference, FBI director Robert S. Mueller said the FBI's computer network had been penetrated and the attackers had "corrupted data."[31] Later that year, General Michael Hayden, former director of both the CIA and the NSA, said: "The modern-day bank robber isn't speeding up to a suburban bank with weapons drawn and notes passed to the teller. He's on the Web taking things of value from you and me."[13]

Finally, malware that appears to be government-generated has been used to obtain critical intelligence, as in the case of the Flame virus, and, for targeted attacks, Stuxnet. Both were widely reported to have been developed by the governments of Israel and the U.S., with Stuxnet apparently created to attack Iran's nuclear facilities.[32,38] Similar tools could allow a foreign power to attack or subvert an Internet election anywhere.

Insider attacks. While many security discussions focus on outsider attacks, insider attacks might be even more dangerous. A risk of any computerized voting, including Internet voting, is that one or more insiders (programmers, election officials, volunteers, or vendors to whom the election is outsourced) could rig an election by manipulating election software. Since computerized voting is an opportunity for wholesale rigging through software used by large numbers of voters, the size of the conspiracy needed to win an election is greatly reduced, as is the risk of being caught.

An attacker could add a back door to the system, with or without the vendor's knowledge. In general, no amount of testing can be relied on to reveal the presence of a back door. A thorough code review (not required by current law) can sometimes do this, but code reviews cannot reliably distinguish between an innocent mistake and intentional malware. A trusted insider (such as former CIA agent Aldrich Ames[h]) can do tremendous damage, even if eventually caught.

The client. Since malware can infect public or privately owned machines linked to the Internet without the owner's knowledge or permission, client-side malware designed to steal an election poses significant risks for ballots cast from voters' computers. These risks include credential theft, copying of the ballot to a third party, and modification of the ballot before encryption, as well as outright prevention of voting. Machines can be infected in many ways, including downloading documents with malicious macros, browser plugins, or improper security settings.

Furthermore, millions of computers are already connected to botnets. In 2010, the FBI reported the Mariposa botnet may have infected eight million to 12 million computers worldwide.[9] The virus used to create the botnet could steal credit-card data and online-banking passwords, as well as launch a denial-of-service attack; the creator of the virus also sold customized versions with augmented features. A Microsoft report estimated that in the first half of 2010 more that 2.2 million U.S. Windows PCs were in botnets.[4]

Those wishing to rig elections need not build new botnets. Many botnets used for financial fraud are available for rent. It would not take a large staff to alter existing malware to attack elections, and it would not be out of character for existing malware developers to offer ready-to-customize election-rigging malware as soon as Internet voting were to enter widespread use.

The sheer number of potential attacks and the difficulty of preventing any of them increase the vulnerability of Internet-based elections. In light of the many successful attacks against governments, major banks, and the world's technology leaders, it should be relatively easy to entrap large numbers of voters who are not technologists. Once a voter's computer is infected, all bets are off. Malware can make the computer display a ballot image that represents the voter's intent correctly, even as it sends something entirely different over the Internet. That is, it is the virus that votes, not the voter. The voter never knows, because it is impossible for the voter to see what is actually sent.

Since antivirus software works by checking for known viruses and worms, whenever a new virus appears, the anti-virus software must be updated. There can be many days or even weeks between the time the virus is initially distributed and when it is recognized and analyzed. After that, the virus fix must be distributed, and victims must disinfect their machines. Because antivirus software has limited capability for recognizing unknown malware, a new virus or worm may well escape detection for a while. Even if detected, removal can be difficult, as most PC owners who have had to deal with adware and spyware are aware. A 2007 study found that antivirus software has become less effective over time, with recognition of malware by most commercial antivirus software falling from 40%–50% at the beginning of 2007 to 20%–30% by the end of that year.[12] Another set of experiments conducted at the University of Michigan showed the number of malware samples detected decreased significantly as the malware became more current; when the malware was only one week old, the detection rate was very low.[34] Given the limitations of antivirus software, an effective attack would be to distribute election-stealing malware far in advance of the election. If the malware were to spread silently, it could infect a large number of machines before being detected, if it is detected at all. Moreover, it might be impossible to determine which votes are modified or even which computers are infected.

The Conficker worm illustrates the risk malware poses to Internet elections. Having rapidly infected from nine million to 15 million machines in 2009, Conficker could "call home" for more instructions, so the unknown creator of Conficker could instruct infected machines to install additional malware remotely without the computer owner's knowledge.[2] The new instructions might target specific candidates and elections shortly before a vote.

While many viruses and worms are planted without the computer owner's knowledge, users can be duped into downloading highly questionable software. In August 2009 a spam message circulated, saying "If You dont [sic] like Obama come here, you can help to ddos [Distributed Denial of Service] his site with your installs." CNET News reported that people who clicked on the email link were offered money in exchange for downloading the software; they were even told to return to the Web site for updates if their virus-detection software deleted their first download.[30] While the source of the software is not known, the goal could have been to disrupt sites associated with President Barack Obama, to engage in identity theft, or even to infect machines of Obama opponents, something that could be especially useful if Internet voting were to become an option in the U.S.

Threat example: The Zeus virus. The Zeus virus illustrates how a virus can manipulate what a voter sees and change the voter's selection. While Zeus has been used mainly to steal money, it would not be difficult to re-program it to steal votes.

In April 2009, malicious software was discovered in Paul McCartney's Web site that redirected visitors to an IP address in Amsterdam in order to exploit vulnerabilities on the victims' machines to install the Zeus virus.[16] The infection, planted shortly before McCartney's New York reunion concert with Ringo Starr, was timed to catch as many victims as possible before discovery.

The German edition of Wikipedia was another source of infection.[14] A bogus Wikipedia article about another dangerous piece of malware contained a link to software that would supposedly fix the problem. However, anyone who downloaded the "fix" was actually downloading a copy of Zeus. In 2009 it was estimated by security firm Damballa that Zeus had infected about 3.6 million PCs in the U.S. alone.[28]

Zeus was built to steal money from online financial accounts. When victims would visit their banks' Web sites, Zeus would copy their credentials and send them to a remote location where they would be used to steal from their accounts. Zeus could even forge financial statements so victims would see no evidence of the theft when checking their online statements.[39] Victims typically learned of the theft only when financial transactions failed to clear due to insufficient funds, at which point it was too late to retrieve the money.

The Zeus virus also spoofed verification systems used by Visa and MasterCard when enrolling new users[7] (see **Figure 2**), thereby obtaining sensitive information (such as Social Security numbers, card numbers, and PINs) from unknowing victims who would think they were providing the information to the real bank. This information, sent to the attacker's computers, would be used to defraud the victims.

Yet another attack was reported in August 2010 by Internet security firm M86 Security; the report said that about 3,000 bank customers in the U.K. were victimized by a form of the Zeus virus. The announcement accompanying the report's release, which did not provide the bank's name, said the following about the attack:[25] "Unprotected customers were infected by a Trojan—which managed to avoid detection by traditional anti-virus software—while browsing the Internet. The Trojan, a Zeus v3, steals the customer's online banking ID and hijacks their online banking sessions. It then checks the account balance and, if the account balance is bigger than GBP 800 value, it issues a money transfer transaction... From July 5, the cyber criminals have successfully stolen GBP 675,000 (c. USD 1,077,000) and the attack is still progressing."

On September 29, 2010, the U.K. Police Central e-crime Unit announced the arrest of 19 individuals accused of using Zeus to steal $6 million from thousands of victims over a three-month period.[24] To this day, new Zeus attacks continue to be discovered; for example, in October 2010, *Computerworld* reported that Zeus was attacking Charles Schwab investment accounts,[20] with victims' machines infected by links to malicious sites hidden in bogus LinkedIn reminders. There is even a criminal service that will compile a Zeus binary for a fee.[10]

Impersonating the election server. Another Internet risk involves Website spoofing. Because counterfeit sites can be made to look like legitimate sites, spoofing can fool victims into revealing sensitive personal information. With Internet voting, spoofing can be used to trick voters into thinking they have actually voted when in fact they have not, while also collecting authentication codes and voters' intended ballots, a violation of the right to a secret ballot.

Phishing involves email messages that appear to be from a legitimate organization, such as a credit-card company. The phony message contains an authentic-looking link that appears to go to a legitimate site but actually goes to a spoofed site. When such email messages and Web sites are well designed, victims end up providing sensitive information, such as credit-card numbers. Phishing is usually used to steal personal information, but can also be used to trick voters into voting on a spoofed Web site. Phishing is a powerful tool for amplifying the power of spoofing, though its effectiveness can be reduced if voters are instructed to always type in the full URL of the voting Web site, instead of just clicking on links.

A counterfeit voting site can conduct a man-in-the-middle attack. In its simplest form, the counterfeit site relies entirely on the real site for content, monitoring and occasionally editing the information flow between the voter and the real election server. This allows the attacker to intercept information, such as passwords and votes, and potentially to alter votes. A more complex counterfeit could simulate a voting session, then use the credentials collected from the voter at a later time to cast a forged ballot. Monitoring the IP addresses from which ballots are cast is not a defense, since multiple voters might share the same IP address for legitimate reasons.

A common way to avoid counterfeit Web sites is to rely on a certificate authority (CA) to authenticate sites. If the browser does not recognize the issuer of a certificate, it will ask if the user still wants to access the site. A user who does not understand the significance of the browser's question may naïvely ignore it and access a counterfeit site.

Even when voters are careful to visit only sites they believe are legitimate, they could still be victimized. First, it is possible to trick many browsers into going to the attacker's, rather than to the legitimate, site.[45] Second, some CAs do not validate the identities of sites they vouch for.[35] Third, an attack on the CA can create fake SSL certificates, as happened to DigiNotar, a Dutch CA.[21] Finally, an attack on the routing infrastructure of the Internet could divert voters to a counterfeit voting site without their noticing the diversion.[27]

Denial-of-service attacks. There are many documented instances of Distributed Denial-of-Service (DDoS) attacks. For example, the massive 2007 DDoS attack on Estonia and the attacks on the Republic of Georgia during the 2008 Russo-Georgian war all originated in Russia. Other victims of DDoS attacks include Amazon, eBay, Facebook, Google, Twitter, and Yahoo!. Politically motivated DDoS attacks, like the one on Wikileaks in 2010 and a reprisal by Anonymous against MasterCard, have become relatively common.

A DDoS attack could prevent certain groups from voting or even disrupt an entire election, as probably occurred in a 2003 leadership vote by the New Democratic Party (NDP) in Canada. Internet voting for the NDP election lasted from January 2 until the party convention January 25, 2003. Coincidentally, on January 25, the same day the Slammer worm was attacking large numbers of (unpatched) Windows 2000 servers on the Internet, the NDP voting site was reportedly down or effectively unusable for hours.[3]

Due to the secrecy surrounding the technical aspects of the NDP election, we do not know if the NDP voting site was brought down by a DDoS attack or by the Slammer worm. The vendor, **election.com (http://election.com)** , claimed to have patched the servers against Slammer and maintained that it experienced a denial-of-service attack. Unfortunately, **election.com (http://election.com)** provided neither logs nor other proof that its servers were patched, nor did it permit expert examination of its records. There was no transparency and hence no way for an independent outsider to determine what had happened.

Not having learned from the 2003 attack, the NDP suffered a massive DDoS attack during its March 2012 leadership election. The NDP was so ill prepared that people attending the party conference were unable to vote during the attack, as no back-up paper had been provided. Once again, there was no independent examination or report.

Loss of the secret ballot. All forms of remote voting diminish ballot secrecy and increase the risk of coercion and vote selling simply because they eliminate voting booths. Internet voting decreases secrecy still further. States that allow the return of voted ballots by fax or email attachments have been asking voters to sign statements relinquishing the right to a secret ballot. Mix nets and other cryptographic schemes can mimic the secrecy protections of the double envelopes traditionally used to partially preserve ballot secrecy in postal voting, but they do not protect against client-side attacks.

The threat to eliminate the secret ballot for a class of voters is disturbing for several reasons: First, it renders these voters second-class citizens, deprived of a right other citizens take for granted. Second, there is no need to eliminate the secret ballot for overseas voters, as we discussed earlier. Third, and most important, ballot-secrecy protection is more than an individual right; it is a systemic requirement, essential for fair, honest elections. Without ballot secrecy, voters, especially those in hierarchical organizations, such as the military, may be subject to coercion. An election where some voters can be pressured to vote a particular way is not a free and fair election.

Bribery. Finally, we cannot rule out the threat of old-fashioned bribery. National races in the U.S. cost vast sums—a small fraction of which would be an exceedingly large bribe and more than enough to cover the cost of attacks, such as the one on the 2010 pilot D.C. voting system, as well as others on voters' computers. Halderman said his team's attack would have cost less than $50,000 at generous consulting rates.

Other Countries

We have focused on Internet voting in the U.S., but Internet voting has been used in several other countries, including Estonia and Switzerland, neither of which protects against malware on voters' computers, and Norway in 2011.[6] The Netherlands provided an Internet voting option in its 2006 parliamentary elections, but Internet voting was subsequently banned, largely because of work by a group called "We Don't Trust Voting Computers." The U.K. tried Internet voting on a pilot basis in 2007, but the U.K. Electoral Commission recommended against further e-voting pilot projects until a range of issues had been addressed.[40]

Far Future

Systems like Helios[15] and Remotegrity[37] use encryption to allow voters to verify that their ballots were accurately received and counted. Unfortunately, cryptography does not protect Internet-based elections against DDoS attacks, spoofing, coercion, design flaws, and many kinds of ordinary software bugs.[8] Recounts on these cryptographic voting systems cannot recover from such threats. While these systems have been used for some small Internet elections, the consensus in the cryptographic community is that they are not ready for use in a major election. Ben Adida, creator of Helios, wrote in 2011: "The one problem I don't know how to address with Helios is client-side security...We now have documented evidence...that viruses like Stuxnet that corrupt nuclear power plants by spreading from one Windows machine to the other have been built...So if you run a very large-scale election for a president of a G8 country, why wouldn't we see a similar scenario? Certainly, it's worth just as much money; it's worth just as much strategically...All the ability doesn't change the fact that a client-side corruption in my browser can flip my vote even before it's encrypted, and if we...must have a lot of voters verify their process, I think we're going to lose, because most voters don't quite do that yet."[1] Note that while Helios can detect DDoS attacks, network attacks, and several other types of attacks mentioned here, it cannot prevent, diagnose, or fix them.

Perhaps eventually a paperless cryptographic Internet voting system will be developed that is sufficiently secure, accurate, usable, and transparent to be used in major elections. Until then, the conclusion of the National Commission on Federal Election Reform, co-chaired by Presidents Gerald R. Ford and Jimmy Carter in 2001, still stands, that Internet voting "is an idea whose time most certainly has not yet come."[11]

Conclusion

Proposals for conducting voting pilot projects using real elections continue to reappear in the U.S. and elsewhere, apparently independent of warnings from computer-security experts. While the appeal of Internet voting is obvious, the risks are not, at least to many decision makers. Computer professionals have an obligation to explain these risks.

Pilot projects are routinely declared successes, regardless of any problems encountered. However, it is dangerous to draw conclusions from a "successful" Internet voting pilot project. There is little reason to attack a small pilot project, and a malicious player might refrain from attacking a major election until the new technology is entrenched. Having claimed success, independent of proof of the accuracy of the pilot project, Internet-voting vendors and enthusiasts routinely push to extend Internet voting to a broader group of voters, thereby seriously undermining election security. Computer professionals must object to pilot projects that do not plan for an assessment of the integrity of the election and a public reporting of any discrepancies encountered.

Unlike legitimate computer-security experts, malicious attackers are not likely to publicize their attacks, just as credit-card thieves do not openly advertise their thefts. When election officials and policymakers ask for proof that a voting system has been attacked, it is important to keep in mind that detecting well-devised attacks is inherently difficult. The burden of proof that a voting system has not been attacked should fall on those making the claim, not the other way around.

Ultimately, the balance between the integrity of election technology on the one hand and convenience on the other is both a public-policy and a technological issue. Decision makers must be warned of all the risks in order to craft wise policy.

Acknowledgment

We are grateful to the referees who provided us with excellent recommendations.

References

1. Adida, B. Panelist remarks at panel on Internet voting. *Electronic Voting Technology Workshop/Workshop on Trustworthy Elections* (San Francisco, Aug. 9, 2011); **http://www.usenix.org/events/evtwote11/stream/benaloh_panel/index.html (http://www.usenix.org/events/evtwote11/stream/benaloh_panel/index.html)**

2. Bowden, M. The enemy within. *The Atlantic* (June 2010); **http://www.theatlantic.com/magazine/archive/2010/06/the-enemy-within/8098/ (http://www.theatlantic.com/magazine/archive/2010/06/the-enemy-within/8098/)**

3. CBC News. Computer vandal delays leadership vote (Jan. 25, 2003); **http://www.cbc.ca/news/story/2003/01/25/ndp_delay030125.html (http://www.cbc.ca/news/story/2003/01/25/ndp_delay030125.html)**

4. Claburn, T. Microsoft Finds U.S. Leads In Botnets. *InformationWeek* (Oct. 14, 2010); **http://www.informationweek.com/security/vulnerabilities/microsoft-finds-us-leads-in-botnets/227800051 (http://www.informationweek.com/security/vulnerabilities/microsoft-finds-us-leads-in-botnets/227800051)**

5. DeGregorio, P. *UOCAVA Voting Scoping Strategy*. Washington Secretary of State Public Record, Jan. 18, 2009; **http://www.votersunite.org/info/WA-PRR-scopingstrategy.pdf (http://www.votersunite.org/info/WA-PRR-scopingstrategy.pdf)**

6. District of Columbia and Halderman, J.A. Thank you to voters (hacked ballot acknowledgment with Michigan fight song); **https://jhalderm.com/pub/dc/thanks/**

7. Dunn, J.E. Trojan attacks credit cards of 15 U.S. banks. *Techworld* (July 14, 2010).

8. Estehghari, S. and Desmedt, Y. Exploiting the client vulnerabilities in Internet e-voting systems: Hacking Helios 2.0 as an example. *2010 Electronic Voting Technology Workshop/Workshop on Trustworthy Elections* (Washington D.C., Aug. 9, 2010); **http://static.usenix.org/events/evtwote10/tech/full_papers/Estehghari.pdf (http://static.usenix.org/events/evtwote10/tech/full_papers/Estehghari.pdf)**

9. FBI. *FBI, Slovenian and Spanish Police Arrest Mariposo Botnet Creator, Operators*. Press Release, July 28, 2010; **http://www.fbi.gov/news/pressrel/press-releases/fbi-slovenian-and-spanish-police-arrest-maripora-botnet-creator-operators/ (http://www.fbi.gov/news/pressrel/press-releases/fbi-slovenian-and-spanish-police-arrest-maripora-botnet-creator-operators/)**

10. Fisher, D. New Service helps attackers get Zeus botnet off the ground. *Threatpost* (Jan. 10, 2011); **http://threatpost.com/en_us/blogs/new-service-helps-attackers-get-zeus-botnet-ground-011011 (http://threatpost.com/en_us/blogs/new-service-helps-attackers-get-zeus-botnet-ground-011011)**

11. Ford, G.R. and Carter, J. *To Assure Pride and Confidence in the Electoral Process*. National Commission on Federal Election Reform, Aug. 2001; **http://fl1.findlaw.com/news.findlaw.com/hdocs/docs/election2000/electionreformrpt0801.pdf (http://fl1.findlaw.com/news.findlaw.com/hdocs/docs/election2000/electionreformrpt0801.pdf)**

12. The H Security. *Antivirus Protection Worse than a Year Ago*. Heise Media, U.K., Dec. 20, 2007; **http://www.h-online.com/security/news/item/Antivirus-protection-worse-than-a-year-ago-735697.html (http://www.h-online.com/security/news/item/Antivirus-protection-worse-than-a-year-ago-735697.html)**

13. Hayden, M. Hackers force Internet users to learn self defense. *PBS NewsHour* (Aug. 11, 2010); **http://www.pbs.org/newshour/bb/science/Jul.-dec10/cyber_08-11.html (http://www.pbs.org/newshour/bb/science/Jul.-dec10/cyber_08-11.html)**

14. Head, W. Hackers use Wikipedia to spread malware. *IT News for Australian Business* (Nov. 6, 2006); **http://www.itnews.com.au/News/67796,hackers-use-wikipedia-to-spread-malware.aspx (http://www.itnews.com.au/News/67796,hackers-use-wikipedia-to-spread-malware.aspx)**

15. Helios. **http://heliosvoting.org/ (http://heliosvoting.org/)**

16. InfoSecurity. McCartney site serves up Zeus malware. *InfoSecurity* (Apr. 8, 2009); **http://www.infosecurity-us.com/view/1178/mccartney-site-serves-up-zeus-malware/ (http://www.infosecurity-us.com/view/1178/mccartney-site-serves-up-zeus-malware/)**

17. Jefferson D. Email voting: A national security threat in government elections. VerifiedVoting blog (June 2011); **http:/blog.verifiedvoting.org/2011/06/20/1375 (http://http:/blog.verifiedvoting.org/2011/06/20/1375)**

18. Jefferson, D., Rubin, A.B., Simons, B., and Wagner, D. A *Security Analysis of the Secure Electronic Registration and Voting Experiment (SERVE)*, Jan. 20, 2004; **http://servesecurityreport.org/ (http://servesecurityreport.org/)**

19. Kanan, K., Rees, J., and Spafford, E. *Unsecured Economies: Protecting Vital Information*. Technical Report. McAfee, Inc., Santa Clara, CA, Feb. 2009; **resources.mcafee.com/content/NAunsecuredEconomiesReport (http://resources.mcafee.com/content/NAunsecuredEconomiesReport)**

20. Keizer, G. Zeus botnet gang targets Charles Schwab accounts. *Computerworld* (Oct. 16, 2010); **http://www.computerworld.com/s/article/9191479/Zeus_botnet_gang_targets_Charles_Schwab_accounts (http://www.computerworld.com/s/article/9191479/Zeus_botnet_gang_targets_Charles_Schwab_accounts)**

21. Kirk, J. Comodo hacker claims credit for DigiNotar attack. *Computerworld* (Sept. 2011); **http://www.computerworld.com/s/article/9219739/Comodo_hacker_claims_credit_for_DigiNotar_attack (http://www.computerworld.com/s/article/9219739/Comodo_hacker_claims_credit_for_DigiNotar_attack)**

22. KITV. Voting drops 83 percent in all-digital election. Honolulu, May 2009; **http://www.kitv.com/politics/19573770/detail.html (http://www.kitv.com/politics/19573770/detail.html)**

23. Kurtz, G. Operation 'Aurora' hit Google, others. McAfee Security Insights blog, Jan. 10, 2010; **http://blogs.mcafee.com/corporate/cto/operation-aurora-hit-google-others (http://blogs.mcafee.com/corporate/cto/operation-aurora-hit-google-others)**

24. Leyden, J. UK cybercops cuff 19 Zeus banking trojan suspects. *The Register* (Sept. 29, 2010); **www.theregister.co.uk/2010/09/29/zeus_cybercrime_arrests/ (http://www.theregister.co.uk/2010/09/29/zeus_cybercrime_arrests/)**

25. M86 Security. *M86 Security Labs Discovers Customers of Global Financial Institution Hit by Cybercrime*. Press Release, London, U.K., Aug. 10, 2010; **http://www.marketwire.com/press-release/m86-security-labs-discovers-customers-global-financial-institution-hit-cybercrime-1302266.htm (http://www.marketwire.com/press-release/m86-security-labs-discovers-customers-global-financial-institution-hit-cybercrime-1302266.htm)**

26. Mahoney, M.R. *Comment on Pilot Project Testing and Certification*. EAC, Washington, D.C., Apr. 2010; **http://www.eac.gov/assets/1/AssetManager/Martha%20Mahoney%20-%20Comment%20on%20Pilot%20Project%20Testing%20and%20Certification.pdf (http://www.eac.gov/assets/1/AssetManager/Martha%20Mahoney%20-%20Comment%20on%20Pilot%20Project%20Testing%20and%20Certification.pdf)**

27. Marsan, C.D. Feds to shore up net security. *Network World* (Jan. 19, 2009); **http://www.pcworld.com/businesscenter/article/157909/feds_to_shore_up_net_security.html (http://www.pcworld.com/businesscenter/article/157909/feds_to_shore_up_net_security.html)**

28. Messmer, E. America's 10 most wanted botnets. *Network World* (July 22, 2009); **http://www.networkworld.com/news/2009/072209-botnets.html (http://www.networkworld.com/news/2009/072209-botnets.html)**

29. Military Postal Service Agency. *2010 Analysis of the Military Postal System Compliance with the MOVE Act*. Washington, D.C., Aug. 2, 2011; **www.fvap.gov/resources/media/2010_MPSA_after_action_report.pdf (http://www.fvap.gov/resources/media/2010_MPSA_after_action_report.pdf)**

30. Mills, E. Spam offers to let people use their PC to attack Obama site. *CNET* (Aug. 18, 2009); **http://news.cnet.com/8301-1009_3-10312641-83.html?tag=nl.e757 (http://news.cnet.com/8301-1009_3-10312641-83.html?tag=nl.e757)**

31. Mueller III, R.S. *Prepared Remarks*. RSA Security Conference, San Francisco, Mar. 4, 2010; **http://www.fbi.gov/news/speeches/tackling-the-cyber-threat (http://www.fbi.gov/news/speeches/tackling-the-cyber-threat)**

32. Nakashima, E., Miller, G., and Tate, J. U.S., Israel developed computer virus to slow Iranian nuclear efforts, officials say. *Washington Post* (June 19, 2012); **http://www.washingtonpost.com/world/national-security/us-israel-developed-computer-virus-to-slow-iranian-nuclear-efforts-officialssay/2012/06/19/gJQA6xBPoV_story.html?wpisrc=al_national (http://www.washingtonpost.com/world/national-security/us-israel-developed-computer-virus-to-slow-iranian-nuclear-efforts-officialssay/2012/06/19/gJQA6xBPoV_story.html?wpisrc=al_national)**

33. New South Wales Electoral Commission. *Report on the Conduct of the NSW State Election 2011;* **http://www.parliament.nsw.gov.au/Prod/parlment/committee.nsf/0/67f2055c4d085409ca25795a0017cf2c/$FILE/NSW%20EC (http://www.parliament.nsw.gov.au/Prod/parlment/committee.nsf/0/67f2055c4d085409ca25795a0017cf2c/$FILE/NSW%20EC%**

34. Oberheide, J., Cooke, E., and Jahanian, F. CloudAV: N-version antivirus in the network cloud. In *Proceedings of the 17th USENIX Security Symposium* (San Jose, CA, July 28-Aug. 1, 2008), 91–106.

35. Palmer, C. *Unqualified Names in SSL Observatory.* Electronic Frontier Foundation Deeplinks blog, Apr. 5, 2011; **https://www.eff.org/deeplinks/2011/04/unqualified-names-ssl-observatory**

36. Peters, S. 14th *Annual CSI Computer Crime and Security Survey, Executive Summary.* Computer Security Institute, New York, Dec. 2009; **http://www.docstoc.com/docs/40697141 (http://www.docstoc.com/docs/40697141)**

37. Remotegrity. 2011; **https://demo.remotegrity.org/http://www.scantegrity.org/wiki/index.php/Remotegrity_Frequently_Asked_Questions**

38. Sanger, D.E. Obama order sped up wave of cyberattacks against Iran. *New York Times* (June 1, 2012); **http://www.nytimes.com/2012/06/01/world/middleeast/obama-ordered-wave-of-cyberattacks-against-iran.html?pagewanted=all (http://www.nytimes.com/2012/06/01/world/middleeast/obama-ordered-wave-of-cyberattacks-against-iran.html?pagewanted=all)**

39. Trusteer Inc. *Measuring the In-the-Wild Effectiveness of Antivirus Against Zeus.* White Paper, Sept. 14, 2009; **http://www.techrepublic.com/whitepapers/measuring-the-in-the-wild-effectiveness-of-antivirus-against-zeus/1686945/post (http://www.techrepublic.com/whitepapers/measuring-the-in-the-wild-effectiveness-of-antivirus-against-zeus/1686945/post)**

40. U.K. *Electoral Commission. Key Issues and Conclusions, May 2007 Electoral Pilot Schemes.* London, Aug. 2007; **http://www.electoralcommission.org.uk/__data/assets/electoral_commission_pdf_file/0009/16200/ICMElectoralPilotsresearc 20161__E__N__S__W__.pdf (http://www.electoralcommission.org.uk/__data/assets/electoral_commission_pdf_file/0009/16200/ICMElectoralPilotsresearch 20161__E__N__S__W__.pdf)**

41. Vascellaro, J.E. and Solomon, J. Yahoo! was also targeted in hacker attack. *Wall Street Journal* (Jan. 14, 2010); **http://online.wsj.com/article/SB10001424052748703657604575004421409691754.html (http://online.wsj.com/article/SB10001424052748703657604575004421409691754.html)**

42. Verified Voting Foundation. *Internet Voting 2012*; **http://www.verifiedvotingfoundation.org/article.php?list=type&type=27 (http://www.verifiedvotingfoundation.org/article.php?list=type&type=27)**

43. Weiss, T.R. Pentagon drops online votes for armed forces. *Computer Weekly* (Feb. 6, 2004); **http://www.computerweekly.com/news/2240054464/Pentagon-drops-online-votes-for-armed-forces (http://www.computerweekly.com/news/2240054464/Pentagon-drops-online-votes-for-armed-forces)**

44. Wolchok, S., Wustrow, E. Isabel, D., and Halderman, J.A. Attacking the Washington, D.C. Internet voting system. In *Proceedings of the 16th Conference on Financial Cryptography and Data Security* (Bonaire, Feb. 28. 2012); **http://fc12.ifca.ai/pre-proceedings/paper_79.pdf (http://fc12.ifca.ai/pre-proceedings/paper_79.pdf)**

45. Zetter, K. Vulnerabilities allow attacker to impersonate any website. ***Wired.com (http://Wired.com)*** (July 29, 2009); **http://www.wired.com/threatlevel/2009/07/kaminsky/ (http://www.wired.com/threatlevel/2009/07/kaminsky/)**

Authors

Barbara Simons (**simons@acm.org (http://delivery.acm.org/10.1145/2350000/2347754/mailto:simons@acm.org)**) is a retired IBM Research staff member, Board Chair of Verified Voting, and former ACM President.

Douglas W. Jones (**jones@cs.uiowa.edu (http://delivery.acm.org/10.1145/2350000/2347754/mailto:jones@cs.uiowa.edu)**) is an associate professor in the Department of Computer Science of the University of Iowa in Iowa City.

Footnotes

a. Portions of this article are taken from the book *Broken Ballots: Will Your Vote Count?* by Douglas W. Jones and Barbara Simons, CSLI Publications, Stanford, CA, 2012; **http://brokenballots.com (http://brokenballots.com)**

b. Ames gave the Soviet Union significant U.S. secrets resulting in the death of a number of "CIA assets."

c. Norway uses encryption, but malware on a voter's computer is still able to change votes, so long as the change is consistent with the partial proof sent to the voter or the voter does not check the partial proof.

Figures

(http://deliveryimages.acm.org/10.1145/2350000/2347754/figs/f1.jpg) Figure 1. The rigged District of Columbia ballot.

(http://deliveryimages.acm.org/10.1145/2350000/2347754/figs/f2.jpg) Figure 2. Bogus enrollment screen displayed by Zeus; screenshot by Amit Klein of Trusteer.

Sidebar: Internet Voting and E-Commerce Compared

Internet voting involves complications not found in e-commerce:

Secret ballots. Secret ballots are required by law to protect against vote buying and coercion. Ballot secrecy prohibits anyone from linking voted ballots to the voters casting them. This precludes the kind of transaction logging routinely used in e-commerce to allow reconstruction of who did what and when, should a question arise.

Receipts. Receipts, including unique transaction numbers and complete transaction descriptions, are routinely issued in e-commerce. These receipts confirm that the correct orders were placed and may be used as proof of purchase in the event of disputes. Ballot secrecy prevents issuing any documents to voters that voters could use to prove how they voted. Documents that do not provide such proof are of limited use in an audit or recount.

Malfunction and fraud. In the event of an e-commerce failure due to malfunction or fraud, there is a good chance the situation will be rectified or that the purchaser can stop a credit-card payment after noticing the discrepancy. However, if a ballot is not successfully cast on election day, the voter probably will not know and almost certainly will not be able to revote.

Vote buying and selling. Unlike commercial activities, vote buying and selling is illegal. In the 2000 U.S. presidential election between republican George W. Bush and Democrat Al Gore, an online system designed to broker Green Party candidate Ralph Nader and Gore votes was created but forced to shut down by the California attorney general. There is no evidence that any votes were actually traded. With Internet voting, voters could sell their voting credentials, perhaps even online, using a Web site designed to automatically cast their ballots.[a]

No proposed Internet voting system is able to overcome these hurdles.

[a] When family members vote on a home computer or citizens vote from a computer in a public library, multiple voters will share the same IP address; while it is possible to detect multiple votes from one IP address, it would be problematic to prohibit them.

Comments

Tim Finin

September 29, 2012 11:20

The link that is the remotegrity citation is mangled and does not work. It was probably intended to be

http://www.scantegrity.org/wiki/index.php/Remotegrity_Frequently_Asked_Questions

Mike Dell

November 08, 2012 05:03

The experiments in the Netherlands in 2004 and 2006 with Internet voting did not end with a ban. The Internet voting experimental law is still active. These experiments were conducted partly with overseas voters.

Work by the group called "We Don't Trust Voting Computers" was aimed at voting computers introduced in 1991. Their actions lead to a ban on the regular systems in use.

Electronic voting in the Netherlands dates back to 1978. The first electronic system: http://afbeeldingen.gahetna.nl/naa/thumb/1280x1280/fc078b53-3626-4426-a118-9f28b86be29c.jpg

Displaying **all 2** comments