eVoting - A Perspective on Security

Damon Small

eVoting – A Perspective on Security
GSEC Practical Assignment 1.2f
Damon J. Small
January 2002

Much has been written about how technology can be used to improve the voting process in the United States, especially since the Presidential Election of 2000. There are some fairly obvious security issues that must be addressed, and some that are not so obvious. Analyzing an election from a procedural standpoint, there are essentially three events that must take place:

- Identification of the voter
- Casting of the ballot
- Counting of the ballot

While a fairly innocuous process on the surface, these events, when taken together, form a very complex situation from an IT and security perspective. This paper will discuss how these objectives can be met with technology, and in contrast from some of the other literature that is out there, it will go beyond the theoretical and discuss in pragmatic terms what should be done to get from current state to "eVoting." Keep in mind that, given this practical approach to implementing electronic voting, we are differentiating eVoting from voting via the Internet. Included are the important first steps that must be taken in order to gain technical experience and public trust in these new systems; therefore, the systems discussed will not be a radical departure from our current voting procedures, but rather an enhancement to them with new devices.

Identification of the Voter

This process varies somewhat from state to state, so in this description we use the Commonwealth of Virginia's process as the model. In traditional voting systems, the voter must be registered in the Precinct in which he or she lives. The registration process, which verifies the voter's place of residence and eligibility to vote, is typically linked to some other accepted form of identification, such as a Driver's License or social security number. Once registered, the voter is placed in a database, or more precisely, a long list of eligible voters. This can be, but is not necessarily, maintained by the State as it is in Virginia. Prior to voting day, separate lists for each District and Precinct are printed for each voting location. Given that the act of voting is necessarily tied to a person's geographical location within a State, it makes sense that each District would have a separate list containing only its voters. The problem is that by relying on a hardcopy printout of registered voters makes validating the voter on Election Day very much a manual process, and manual processes can be tricky and prone to human error. A typical scenario might be that the voter reports to his voting location with Driver's License and Voter Registration card in hand, whereupon he stands in the appropriate line for a poll worker to look at his papers, find his name in the list, and check it off. At the end of the day, tallying the total numbers of registered voters that actually voted is also a manual process, with the same inbuilt problems as in the

validation process.

Clearly, keeping this information in an electronic format would solve many problems, but how do we ensure that the database itself remains intact and secure? One way to begin answering this is to consider a rule of thumb - the three boundaries of security: 1) Network Perimeter, 2) Host Perimeter, and 3) Physical Perimeter. Beginning with the Network Perimeter, or the outermost connection of our voting system at a particular Precinct, we need to first decide *how*, or even *if*, it will communicate with a central office. One of the inherent problems when discussing voting and technology is that it is impossible to say with any degree of certainty whether or not a particular voting location will have the ability to connect to a network. Considering that voters often report to local high school gymnasiums, we cannot even assume that an analog dialup line will be within reach. An obvious answer to this problem might be wireless technologies, such as IEEE 802.11b. Unfortunately, the encryption algorithms, known as Wired Equivalent Privacy (WEP), used to secure 802.11b wireless transmissions has come under fire, and are no longer considered truly secure[i,ii].

One possible security measure to overcome the limitations of the 802.11b WEP protocol is to tunnel all wireless communications using a Virtual Private Network (VPN) solution[iii]. That being said, the nature of the data in the poll books is fairly static. There are always last minute registrations (where allowed) and changes, but in general, once the poll books are generated, they don't change much. Because of legal restrictions, the votes are not tallied until after the polls close anyway, so the need for real-time connections from the voting location to some central location in minimal. Given that any security analyst would say that unnecessary services should be disabled, and since the risks and feasibility limitations outweigh the potential gains of an eVoting implementation with today's technology, the prudent decision is to maintain each voting location as separate sites, not connected to a central network at all. The one constant in the technological universe is change, so it is certainly reasonable to expect that we will move towards having interconnected voting sites, but we must learn to walk before we can run. For our purposes then, we will consider laying technology on top of our pre-existing infrastructure, with each poll location being a separate, and disconnected Local Area Network (LAN). There is still much to consider in terms of the security of such a system, however.

Given our assumptions about each polling location being a stand-alone, disconnected network, a copy of the database would have to be present at each. This presents our first major security concern - physical security of the database of registered voters. Even if you have the most secure system in the world, if unauthorized persons can get to the physical machine on which the data lives, it can be compromised. The solution to this dilemma need not be glamorous - keeping the poll book machines under the watchful eye of a poll worker or locked in a safe of sorts may suffice. The bottom line is that the computer that the poll book lives on must be kept from the hands and eyes of unauthorized personnel.

So far we have matched the voter's Driver's License (or other form of ID) against the

database of registered voters, but how do we know that the voter really is who he says he is? Furthermore, how do we ensure that the vote that is cast is really cast by the person whom we have identified? In other words, do we have non-reputable proof of the voter's ID, and the vote that he cast? Non-repudiation is a key concept in terms of security and data integrity. The solution to this problem must begin with the form of identification the voter presents at the poll location. Systems already in place in some States could be modified to help with this issue. Upon receiving a Driver's License in the State of Texas, the driver's fingerprint is electronically scanned and stored in a Department of Public Safety database to verify the identity of the driver[iv]. In the Commonwealth of Virginia, motorists can register to vote when they apply for a Driver's License[v]. If we combine these two systems, we could capture a fingerprint and have it on record with both the Department of Motor Vehicles, and the Registrar of Voters. This information can then be stored on the magnetic strip, or a Smart Card-enabled Driver's License[vi] or National ID Card[vii]. Then, at the point at which the voter checks in on Election Day, biometric scanners can be used to check the fingerprint against the one that is on record or stored on the Smart Card. This would provide the necessary proof that this person is who he says he is.

Casting the Ballot

Now that we know who the voter is, we are ready for him or her to cast a ballot. One of the first key issues to consider is the fact that, despite having spent so much effort validating the voter's identity, we must not associate that identity with the cast ballot. In our Democratic society, after all, the secret ballot is key. Because of this, it is necessary to NOT use the same Smart Card-enabled Driver's License that was used in the identification of the voter. At this point, we already know who the voter is. Our concern shifts to providing the voter with the appropriate ballot, making sure the voter votes once and only once, and counting the ballot. The ballot information, such as in which District a voter lives in a multi-district precinct, or whether the voter needs assistive technologies in order to cast a vote, can be stored on a separate, reusable Smart Card that is issued to the voter after the ID check. There are many varieties of eVoting booths already in existence[viii], and many run on industry standard PC-based hardware. This is both good and bad news. Using what is essentially COTS (Commercial Off The Shelf) hardware and software will keep prices down, which is important for local governments, but also means that any well-known security vulnerabilities that exist in the Operating System (OS) or applications used on these devices could be exploited by a malicious voter. Although not a true security measure, the "security by obscurity" provided by proprietary systems is better than no security at all. Therefore, physical security and the configuration of the OS and applications are critical.

Starting with physical security, the voting booth itself must have a perimeter. In traditional client-server environments, servers are protected from the black hats by locking them away in a data center where they can't be reached. This is not an option in our case since the voter will necessarily need to interact closely with the device. Except in cases where assistive technologies are used, the chassis of the voting

machine should be locked to ensure that all external interfaces are inaccessible. These include serial, parallel, PS/2, SCSI, and NIC ports. The only means of interfacing to the device should be through a touch screen-enabled application, and the Smart Card reader. Imagine how embarrassing it would be if all a voter had to do to access the database was plug in a keyboard and hit CTRL-ALT-DEL. The case of the chassis itself should also be locked to prevent access to the motherboard and internal peripherals, and if floppy or CD drives are present, they too should be locked and disabled.

In terms of the software on the device, the lack of a connection to the Internet makes them much less susceptible to malicious code that use email or web browsers as the attack vector. This is no reason for complacency, though, and all latest security patches and updates from the OS and application manufacturers should be applied (and kept up to date). Since this is a challenging task when there potentially are a few thousand machines involved, it will be necessary to use imaging software to deploy a fully hardened OS load, or some other software management package. Fortunately, voting tends to be a seasonal business, so there should be ample downtime between elections to make any necessary updates to the electronic polls. Any and all unnecessary services and applications should be disabled. At this point, we will make another assumption. For our purposes, we will assume that our eVoting machine will run a Windows-based Operating System. Given this, and given the fact that Windows 95/98/ME is designed for use by consumers at home, and not in environments where security is critical, it is strongly recommended that Windows 2000 be used. This way, System Policies can be employed to further lock-down the OS. For instance, the following restrictions should be applied using the Windows Policy Editor (PolEdit.exe), Group Policy Objects (GPO), and Windows 2000 Resource Kit Utilities:

- Disable Desktop Icons: In the event that the voting application crashes, the voter will be presented with a “blank” desktop, rather than a regular Windows 2000 desktop.
- Disable “Run” from the Start Menu: This will prevent a user from launching new programs or commands from the Run Dialogue Box
- Disable the Task Manager: This will prevent users from spawning new processes from the “New Task” dialogue box
- Enable “Run only allowed applications”: This feature will prevent any executable from running, except those that are listed. For instance, this feature could be configured such that only necessary systems applications and the voting application itself can run, so that in the unlikely event that malicious software gets on the voting machine, it will not be allowed to be executed. This will also prevent a user from accessing the file system through the Internet Explorer “back door.”

These are just a few examples of steps that can be taken to further harden the local OS of the voting machine. The point is that since the voter will interact directly with the unit, a default configuration is not acceptable, and measures to give access to only that which is necessary to cast a ballot must be taken.

Other key concepts in the voting process is that each eligible voter must cast one and only one vote, the ballot must be tamper-proof, and the ballot must remain secret. These goals can be achieved through the use of digital signatures, homomorphic encryption, and zero knowledge proofs[ix]. Digital signatures are used to validate the authenticity of documents, and are used quite regularly in industry today. In fact, digital signatures are now legally recognized[x], and carry the same strength as hand-written signatures. Homomorphic encryption and zero knowledge proofs ensure that the voter's answers have not been altered, but at the same time keep the identity of the voter secret. C. Andrew Neff (2000) provides a good description of these technologies:

> "Homomorphic Encryption is a special kind of encryption having the property that the sum of two encrypted numbers is always equal to the encryption of their sum. This means that anyone can compute and verify the "encrypted sum" of a collection of encrypted values. Because the data is encrypted, this same person will not know what numbers are actually encrypted, either in the original values, or the final sum, but he will know that whatever the unencrypted values are, they maintain the sum/total relationship.
>
> The concept of zero knowledge proof can be illustrated with an example. Suppose two people wish to play a card game that depends on one of them the player holding a single card that must be either an ace or a two. In order to continue, the second person the dealer must know with absolute certainty that this card is one or the other, but he must not know which. Revealing the card is not an option for the player since he needs to keep the contents of the card secret, but the dealer can't trust the player at his word either. One way out of the dilemma is to have the player, under the dealer's inspection, remove seven *other* cards from the deck and put them aside face down. The player makes sure that these cards are all the remaining aces and twos. The dealer is now given the opportunity to inspect the remaining 44 cards, looking for an ace or a two. If he finds none, he can be certain that the player's original card must have been one of the missing four aces or one of the missing four twos. At this point, the seven removed cards can be shuffled back into the deck, and the game can continue. This is a fairly simple example, but more complicated mathematical constructs have been invented to achieve the same kind of goal in more complicated situations like electronic voting. In essence, the goal is always to allow one participant to do something *secretly*, yet provide an indisputable *proof of honesty.*[xi]"

Counting the Ballot

All this being said, we can now discuss the counting of the ballot. On the relative scale, even a long ballot is a fairly simple dataset. There are a finite number of candidates and issues, and the voter can vote "yes" or "no" (or abstain) to them. So, in the database world, the data itself is not very complex, even considering write-ins. Keeping the data intact in a distributed environment is decidedly trickier, though. In the security world, not all solutions are technological; many times, policies and procedures can help keep our data intact. Such is the case in this scenario. Whether the voting machines are networked or not, there will still be many databases of ballots that need to be tallied. In effect, we are facing the challenge of exporting many copies of a ballot database and combining them later at a central location. Programmatically there are many ways that this process can be streamlined, but the biggest risk to be assumed is the human factor – the "sneaker-netting" of the databases from the polling location to the central office. This solution, then, is decidedly low-tech.

To start, access to the data must be controlled. A very discreet number of personnel

should have administrative access to the poll booth machines, and to the key that unlocks that hardware. If the poll booth machine were taken from the polling location to a centralized location for tallying, then the same process that tracks and audits ballot cards today can be used. If the data were exported to some removable media, then we would need to supplement these procedures such that the media are separately handled from the devices. Essentially, there needs to be detailed handling instructions and procedures to ensure that once the carbon-based elements of the voting system (i.e., the human beings) are engaged, there are no questions as to what will happen to the data.

In summary, we approach the first steps of eVoting by breaking the voting process down into its three most basic components: Identifying the voter, casting a ballot, and counting a ballot. To this we have applied security components from the Network, Host, and Physical Perimeters. Namely, the network perimeter needs to be secured by not taking unnecessary risks, and not using controversial wireless protocols. The host and physical perimeters are secured through the use of local security policies and by physically locking the chassis. Additional physical perimeter controls are put in place through the use of policies and procedures when handling removable media that contain the homomorphically encrypted ballots themselves. By breaking the complex voting process down and applying these security principles, it is possible to see how we might begin to move from hanging chads to digital eVotes. These first steps are not the end, but rather the first necessary steps towards our ultimate goal of someday being able to vote from a more convenient location with total confidence that that will be counted.

References

[i] Kozup, Chris "Secure Your WLAN Now" December 28, 2001 URL: http://techupdate.zdnet.com/techupdate/stories/main/0,14179,2835133,00.html?chkpt=zdnnhwrttu ZDNet Tech Update

[ii] Blackwell, Gerry "Serious WLAN Security Threats, Part I" January 2002 URL: http://www.80211-planet.com/columns/article/0,4000,1781_949891,00.html 802.11 Planet Insights

[iii] Blackwell, Gerry "Serious WLAN Security Threats, Part II" January 2002 URL: http://www.80211-planet.com/columns/article/0,,1781_947571,00.html 802.11 Planet Insights

[iv] Texas Department of Public Safety "Thumbprint Capture for Driver Licenses and Identification Cards" 2000 URL: http://www.txdps.state.tx.us/administration/driver_licensing_control/license_issuance/thumbprints.htm

[v] Virginia Department of Motor Vehicles "Apply to Register to Vote" 2001 URL: http://www.dmv.state.va.us/webdoc/general/vote.asp

[vi] Bridis, Ted "Study of Smart Drivers Licenses Begins" Januray 7, 2002 URL: http://dailynews.yahoo.com/h/ap/20020107/tc/identity_cards_1.html Associated Press

[vii] O'Harrow, Robert Jr., and Krim, Jonathan "National ID Card Gaining Support" December 17, 2001 URL: http://www.washingtonpost.com/wp-dyn/articles/A52300-2001Dec16.html Washington Post

[viii] Cranor, Lorrie "Electronic Voting Host List" Last Updated January 19, 2001 URL: http://www.bsos.umd.edu/gvpt/its/EvoteSp01/links.html#vendor

[ix] Neff, C. Andrew "Conducting a Universally Verifiable Electronic Election Using Homomorphic Encryption" November 2000 URL: http://www.votehere.net/VH-Content-v2.0/homomorphicsystemdescription.pdf (currently offline) VoteHere.net

[x] Charles, Deborah "Clinton Signs Digital Signature Bill" June 30, 2000 URL: http://www.zdnet.com/zdnn/stories/news/0,4586,2597132,00.html Reuters

[xi] Neff, C. Andrew "Conducting a Universally Verifiable Electronic Election Using Homomorphic Encryption" November 2000 URL: http://www.votehere.net/VH-Content-v2.0/homomorphicsystemdescription.pdf (currently offline) VoteHere.net

Last Updated: July 20th, 2019

Upcoming SANS Training

Click here to view a list of all SANS Courses

SANS Pen Test Hackfest Europe Summit & Training 2019	Berlin, DE	Jul 22, 2019 - Jul 28, 2019	Live Event
DFIR Summit & Training 2019	Austin, TXUS	Jul 25, 2019 - Aug 01, 2019	Live Event
SANS Riyadh July 2019	Riyadh, SA	Jul 28, 2019 - Aug 01, 2019	Live Event
SANS Boston Summer 2019	Boston, MAUS	Jul 29, 2019 - Aug 03, 2019	Live Event
SANS July Malaysia 2019	Kuala Lumpur, MY	Jul 29, 2019 - Aug 03, 2019	Live Event
SANS Crystal City 2019	Arlington, VAUS	Aug 05, 2019 - Aug 10, 2019	Live Event
SANS Melbourne 2019	Melbourne, AU	Aug 05, 2019 - Aug 10, 2019	Live Event
Security Awareness Summit & Training 2019	San Diego, CAUS	Aug 05, 2019 - Aug 14, 2019	Live Event
SANS London August 2019	London, GB	Aug 05, 2019 - Aug 10, 2019	Live Event
Supply Chain Cybersecurity Summit & Training 2019	Arlington, VAUS	Aug 12, 2019 - Aug 19, 2019	Live Event
SANS Prague August 2019	Prague, CZ	Aug 12, 2019 - Aug 17, 2019	Live Event
SANS Minneapolis 2019	Minneapolis, MNUS	Aug 12, 2019 - Aug 17, 2019	Live Event
SANS San Jose 2019	San Jose, CAUS	Aug 12, 2019 - Aug 17, 2019	Live Event
SANS MGT516 Beta Three 2019	Arlington, VAUS	Aug 19, 2019 - Aug 23, 2019	Live Event
SANS Amsterdam August 2019	Amsterdam, NL	Aug 19, 2019 - Aug 24, 2019	Live Event
SANS Virginia Beach 2019	Virginia Beach, VAUS	Aug 19, 2019 - Aug 30, 2019	Live Event
SANS Chicago 2019	Chicago, ILUS	Aug 19, 2019 - Aug 24, 2019	Live Event
SANS Tampa-Clearwater 2019	Clearwater, FLUS	Aug 25, 2019 - Aug 30, 2019	Live Event
SANS New York City 2019	New York, NYUS	Aug 25, 2019 - Aug 30, 2019	Live Event
SANS Copenhagen August 2019	Copenhagen, DK	Aug 26, 2019 - Aug 31, 2019	Live Event
SANS Hyderabad 2019	Hyderabad, IN	Aug 26, 2019 - Aug 31, 2019	Live Event
SANS Philippines 2019	Manila, PH	Sep 02, 2019 - Sep 07, 2019	Live Event
SANS Brussels September 2019	Brussels, BE	Sep 02, 2019 - Sep 07, 2019	Live Event
SANS Munich September 2019	Munich, DE	Sep 02, 2019 - Sep 07, 2019	Live Event
SANS Canberra Spring 2019	Canberra, AU	Sep 02, 2019 - Sep 21, 2019	Live Event
SANS Network Security 2019	Las Vegas, NVUS	Sep 09, 2019 - Sep 16, 2019	Live Event
SANS Oslo September 2019	Oslo, NO	Sep 09, 2019 - Sep 14, 2019	Live Event
SANS Dubai September 2019	Dubai, AE	Sep 14, 2019 - Sep 19, 2019	Live Event
SANS Rome September 2019	Rome, IT	Sep 16, 2019 - Sep 21, 2019	Live Event
SANS Paris September 2019	Paris, FR	Sep 16, 2019 - Sep 21, 2019	Live Event
SANS Raleigh 2019	Raleigh, NCUS	Sep 16, 2019 - Sep 21, 2019	Live Event
Oil & Gas Cybersecurity Summit & Training 2019	Houston, TXUS	Sep 16, 2019 - Sep 22, 2019	Live Event
SANS San Francisco Summer 2019	OnlineCAUS	Jul 22, 2019 - Jul 27, 2019	Live Event
SANS OnDemand	Books & MP3s OnlyUS	Anytime	Self Paced