From: Matt Bishop
To: votingsystemguidelines@eac.gov
Subject: Comments on the UOCAVA Pilot Program Testing Requirements
Date: 04/30/2010 01:44 PM

Hello,

These comments on the UOCAVA Pilot Program Testing Requirements come from the UC Davis electronic voting research group.

We hope they are helpful.

Sincerely,

Matt Bishop

====================
2.1: Basis for the error rates? Why 1 in 10,000,000 ballot positions and 1 in 500,000 ballot positions? How do these compare with non-electronic rates?

2.1.3.2a: Define "entry". If I vote for candidate A, then change it to B, must A then B be recorded or will it record only B?

2.1.4: the requirements of section 2.1 do not appear to be related to telecommunications. For example, 2.1.3.2a says the voting systems shall record every entry made by the user, so does this mean the telecommunications components must record and/or transmit every entry made by the user? If so, this seems to compromise voter privacy.

2.2.2: What exactly does "approaching" mean? How far away from the limit do you alert the user?

2.3.2.3: This does not cover a provisional ballot. Add an item that says, if the law in the jurisdiction for which the system is intended disallows provisionals being cast on the system, the system must link the reason for the provisional ballot and the ballot to the voter's identity without violating the privacy of the voter.

2.5.1.5:Add something here for handling provisional ballots, if they can be cast on the system.

2.6: is the goal of the audit to audit the *election* or to audit the *system* (two different types of audits, with different information needed)? Please specify which, and if both, indicate which audit needs what information.

2.6.2.1: exported to what? Paper? Another system? In some XML format? In binary? What is the purpose of this -- that will control what format(s) the export is to be in.

Section 3: this assumes voters do not need assistive devices such as audio (for blind people) or ways to help color-blind people get the information in 3.6j. We think you should include requirements for disabled voters.

3.5.1.2: This may not be possible (California's 2003 gubernatorial recall election had 135 candidates for governor; how would you put those on one screen and meet the requirements in 3.4.1.2a?); what happens then?

4.3.1.2: define "small" (it should be related to simplicity of function and not to number of lines of code)

4.4: is the intent to disallow threads or multi-process code? If so, state this explicitly, because this will affect many graphics and other types of programming used in voting systems.

4.6.1c and d: if the vendor writes and sells a compiler to 1 other company, or donates it to a university, is it COTS?

4.7.2.1: why these errors? For example, heap overflow errors are not listed.

Section 5: The justifications are weak. Tie them to a process model -- how do the requirements of the process of the pilot program drive these? Also, nowhere do I see a need for the statement of a comprehensive security policy; in light of 5.2.1.5, this is critical.

5.1.1.1: if the intent is for the vendor to do this, say so; otherwise, it sounds like election officials may define personnel roles not put in the system by the manufacturer.

5.1.2.4: What does "modifying a higher-privilege process" mean -- changing what is in its memory, or affecting its execution (for example, changing data that the higher-level process reads so that process will do something other than what it should)? The two are very different.

5.1.2.5: The term "operating system privileged account" is unclear -- do you mean Administrator on Windows (or root on Unix/Linux)? If so, say something like "system level account" or "system administrator account", or if you mean with the privileges of the operating system (ie, all privileges), say the voting system SHALL require its execution with at most user-level privileges or as a user level account.

5.2.1.1: You need to justify the "1,000,000" number -- why that and not 5,000,000 or 100,000? Did you get it from experience with remote voting by paper and/or fax?

5.2.1.6: This says "SHALL allow". That implies to me that the specification is optional. If you mean it not to be (and I think the "per" clause may intend this), say "SHALL require".

5.2.1.X: You should consider requiring confidentiality between the voting system server and the vote capture device. Otherwise, if the system is to relay ballots individually after each voter has voted, this could compromise anonymity of the ballot. Or, you can say that no ballots or vote counts may be transmitted until voting is complete.

5.3: please replace "random number" with "random or pseudorandom number" unless you really mean to require voting systems to use truly random (ie, physically random) numbers. Also, you should specify "cryptographically random/pseudorandom", because otherwise a linear congruential generator would be acceptable -- and those have been broken. (You might specify FISP 140-2 approved RNGs, as in 5.3.2.4, if that is what you mean.)

5.1.4.3: Can appropriately anonymized/disguised/perturbed data related to the cast vote be logged? That is, if information in the CVR is disguised so it cannot be recovered (to some degree of certainty), can it be permanently recorded? Also, what does "permanently" mean -- if I store CVRs in the log, and the log will be destroyed or overwritten in 20 years, it's not "permanently stored". You should be explicit on the time limit.

5.5.1.4: How do you protect the identifier (for example, preventing masquerades)?

5.5.1.X: Suggest adding something about ballot anonymity, too.

5.6.1.5: Recommend that, if the log is not stored in a publicly documented format, such as XML, that the vendor be required to include a document describing the *internal* format as well as include a utility to export the data into a publicly documented format. Such a document should be included for the public format, to define the fields and tags used (XML is a structural format that one needs to add semantics to; you want these semantics to be public so people can understand the XML).

Table 5-2, p. 59, Voting Events: how will you prevent including a timestamp on the casting of a vote from being tied to the voter? Suggest stating explicitly that the ballot cannot be logged, that no logging entry may be tied to a ballot, or something like that.

5.8.3.2: Add that if something is physically connected, you will get a visual alarm too.

5.8.4.1: Also secure panels with screws, etc. -- a tamperproof door and lock do nothing if an adversary can unscrew the back panel and go through that.

5.8.5.1 and 5.8.7.1: the definition of "unauthorized" is vague. If I'm authorized to physically access the receptacle, and add 100 paper ballots to it, this does not appear to violate either requirement. If the intent of these requirements is to prevent ballot box stuffing, you need to prevent unauthorized *action*, not access.

5.9.2: There seems to be no requirement that the results of the OEVT be reported. This should be made explicit; in particular, *what* should be reported? To be very precise, if the OEVT finds a flaw that arises due to a misconfiguration, the OEVT need not report it, as I read this section. The problem is that you are now relying entirely on a procedural defense, and in essence saying "as the procedures will not fail, we can ignore this technical flaw." This overlooks the possibility of errors. In my experience, you want technical layers of defense to support procedural layers of defense. The proposed pilot standards do not appear to do this.

I would recommend the following approach: split the OEVT process into two parts. The first part would be to gather technical flaws *irrespective of procedural defenses*. The second part would be to apply the vendors' recommendations to those findings, and say which procedures eliminate which technical threats. That way, you can see exactly what the consequences of procedural failures are. So can the election officials who buy those systems.

5.9.2.1: it is not clear from this requirement that the voting system is to be tested *as a whole*; the scope of penetration testing should include the voting system *and* all the voting system components. (The composition of secure components is not necessary secure.)

5.9.2.2: What happens if the TDP does not "replicate the real world environment in which the voting system will be used"? Which do you follow -- the TDP or the real world environment? This came up during the California TTBR. It is best to follow the real world environment, because following only the TDP and not how the system is delivered is also *extremely* anti-election official and pro-vendor, in the sense that the vendor can send anything to the customer and add a sentence about how it should be set up -- and then it's the customer's problem. This should be a reportable vulnerability.

5.9.2.3: Please say *explicitly* that the testing team SHALL be provided full access to the TDP and anything in it. This section specified they have access to the "source code included in the TDP", implying they may not have access to other parts of the TDP (such as a penetration study report by the vendor). THis severely impacts the ability of a testing team to be thorough.

5.9.2.4: Please say explicitly that changing 1 ballot in such a way as to not interfere with the election results is a result that should be reported. This section strongly implies it is *not*.

5.9.2.4d: Justify this. It seems to me these should be equal -- certainly compromising ballot secrecy raises the issue of people voting a certain way, because others will know how they voted -- and this may lead to changing the outcome of an election. Same for non-selective DoS attacks, because you can exclude people who come in the afternoon (for example). Also, what exactly is a "selective" DoS attack; in the above, I'm assuming it targets a particular voter.

5.9.2.7: eliminate the "generating the reports of the testing results". If the reports are like FISMA reports, the time writing them will dominate the actual testing. Make the testing take the time.