Saving Democracy from E-voting

Joseph Kiniry
DTU and DemTech
August 2013

Today's Talk

- background
- the state of modern digital elections
- independent audits by scientists
- verified election systems

Background

Democracy Matters

- we view democracy as a critical system
- we work very hard to ensure that elections are fair and free (particularly if technology is inappropriately injected into them by lobbied politicians)
- armchair quarterbacking is easy though, as the world is going to use computers in elections whether we like it or not
- so we hack, for good, for democracy

Hacking

- sometimes you have to get the attention of the authorities so that they listen to you (the public employee researcher) rather than the vendors
- The Netherlands (KOA, RIES, Nedap)
- Ireland (Nedap/PowerVote)
- Denmark (stopped just-in-time) via the hard work of the DemTech project and other IT activists in Denmark

Denmark's State-of-Affairs

- computers are used in elections everywhere but during voting
- Municipalities have been asking for years to experiment with computer-based voting
- various standard claims made initially about improving accessibility, accuracy, timeliness of the result, cost
- bill L-132 introduced by the Ministry
- bill's focus is on open-ended, under-defined trials of ballot printers with possible computer-supported tallying
- feedback from election and technology experts nearly uniformly negative
- feedback from disabled organizations, Municipalities, vendors, and IT trade organizations positive
- debate on bill—and its death—occurred in March 2013

The State of Modern Digital Elections

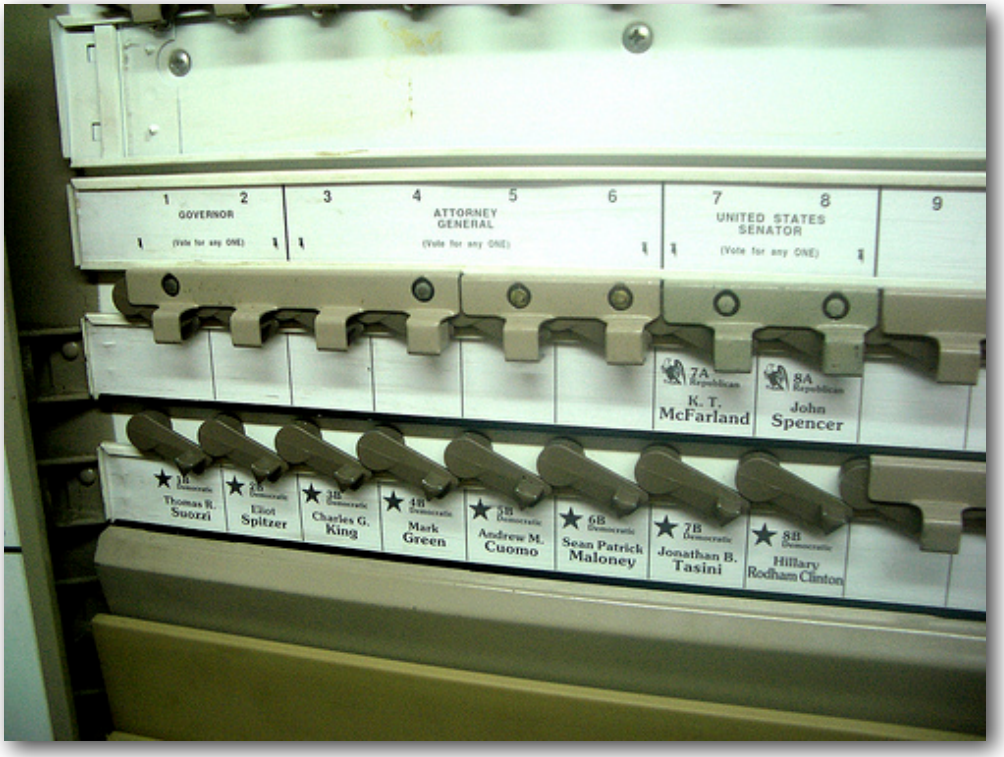
“Traditional” Digital Elections

- a short history of digital elections
- take a traditional paper ballot-based election that has been refined for decades
- add mechanical devices to make fraud more difficult, decrease the number of spoiled ballots, and increase accessibility for voters with various impairments (literacy, vision, mobility, etc.)
- mechanical device manufacturers go out of business or move on to computer-based equipment
- buy new equipment to fulfill the requirements mentioned above, but at higher cost and a loss of transparency

Lever Machines

Lever Machines

Lever Machines



Optical Scanners

State of Connecticut
Official Ballot

Fairfield, Connecticut	Municipal Election	November 8, 2011

Voting District 1
Sheet 1 of 1

OFFICE → / PARTY ↓	1 First Selectman	2 Selectman	3 Town Clerk	4 Board of Finance Vote for Any Two	5 Board of Finance Vote for Any Two	6 Board of Education Vote for Any Five Five to be Elected No More Than Three From One Party	7 Board of Education	8 Board of Education	9 Board of Education	10 Board of Education	11 Board of Assessment Appeals Full Term	12 Board of Assessment Appeals To Fill Vacancy for Two Years	13 Town Plan and Zoning Commission Four Year Term Vote for Any Two	14 Town Plan and Zoning Commission Four Year Term Vote for Any Two	15 Town Plan and Zoning Commission Two Year Term
DEMOCRATIC	○ 1A Michael C. Tetreau	○ 2A Cristin McCarthy Vahey	3A	○ 4A Elaine G. Gaffney	○ 5A Patrick D. McCabe	○ 6A Stacey Zahn	○ 7A Jessica Gerber	○ 8A Jennifer Maxon Kennelly	○ 9A Neal Fink	○ 10A Philip Dwyer	○ 11A Stephen E. Tower	○ 12A Timothy P. Sheehan	○ 13A Patricia M. Jacobson	○ 14A Sally E. Parker	○ 15A Anita Rappoport
REPUBLICAN	○ 1B Rob Bellitto	○ 2B James Walsh	○ 3B Betsy P. Browne	○ 4B Christopher W. DeWitt	○ 5B Thomas Flynn	○ 6B Paul Fattibene	○ 7B John Convertito	8B	9B	10B	○ 11B David D'Ausilio	○ 12B Laura M. Devlin	○ 13B Matthew Wagner	○ 14B Bryan LeClerc	○ 15B Douglas Soutar
INDEPENDENT	○ 1C Hugh F. Dolan	○ 2C Deanna R. Polizzo	3C	○ 4C Laura Incerto	5C	○ 6C Amanda Parks	○ 7C Hugh Joseph Donnelly	8C	9C	10C	11C	12C	13C	14C	15C
GREEN	1D	2D	3D	4D	5D	6D	7D	8D	9D	10D	11D	12D	13D	14D	15D
WORKING FAMILIES	1E	2E	3E	4E	5E	6E	7E	○ 8E Jennifer Maxon Kennelly	9E	10E	11E	12E	13E	14E	15E
WRITE-IN VOTES →	○ 1F	○ 2F	○ 3F	○ 4F	○ 5F	○ 6F	○ 7F	○ 8F	○ 9F	○ 10F	○ 11F	○ 12F	○ 13F	○ 14F	○ 15F

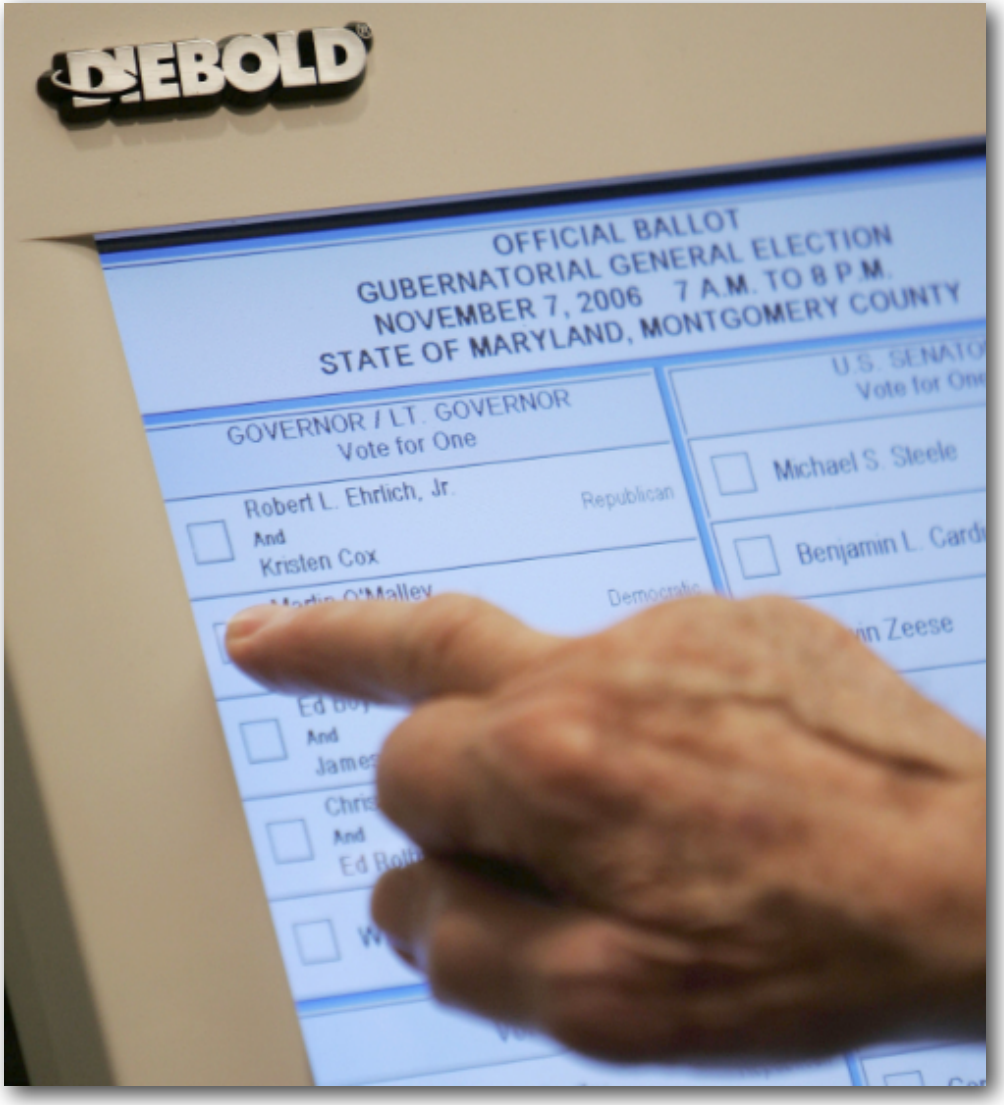
Be sure to complete your vote on the reverse side of this ballot.

Optical Scanners

Optical Scanners

DREs and VVPAT

DREs and VVPAT



DREs and VVPAT

Independent Audits by Scientists

DemTech Auditing

- scientists have been auditing commercial e-voting systems for years
  - California top-to-bottom review, D.C. hack
- DemTech is archiving and analyzing publicly released election software for correctness, security, and software engineering practices
  - examined Norway (v2), Scantegrity II, KOA
  - recently looking at Helios and Estonia
  - in the queue: Pret-a-Voter, Solon, LiquidFeedback, ACT, Victoria, VoteBox

Norway

Norway

- the Norwegian remote code voting system (lightweight verifiable elections via two channel communication)
- designed, developed, maintained, and managed by the Ministry of Local Government and Regional Development, EDB ErgoGroup, and Scytl (a Spanish firm and the top evoting vendor in the world)
- better clarity on tender and contract, large amounts (perhaps nearly 1000 pages in total) of software engineering artifacts
- >262 KLOC Java, > 36 KLOC C#, ~20 KLOC Perl, >10 KLOC other
- currently the best-in-world example of a government attempting to be transparent with regards to computer-based voting

Norwegian Software Engineering

- evidence of lightweight code standard
- evidence of lightweight use of static analysis
- evidence of knowledge of cryptography
- some unit and few system tests
- no formal specifications
- no transparency on development process or artifacts
- dozen or so security problems found

Estonia

Estonia's Software

- (some) server software published as a code drop from Sven Heiberg on 11 July to https://github.com/vvk-ehk/evalimine
- code release was mandated by government, not technologists responsible
- unclear what role activism in Estonia had
- DemTech fork for analysis in out GitHub Organization found at https://github.com/demtech/evalimine

Big Picture

- nearly typical code drop: no docs, no harness, no tests, no protocol description
- evidence of very poor software engineering practices (looks like a one man hack job)
- code is medium sized

python:	9892 (59.46%)
cpp:	6487 (38.99%)
sh:	257 (1.54%)

Documentation

- documentation coverage is embarrassing
  2.3% coverage for Python code
  1.2% coverage for C++
- the majority of comments are in reused library code, not in Estonia's code (proper).
- no non-source documentation (architecture, requirements, test plan, etc.)

Engineering Practices

- 2 assertions in C++ code
- 1 assertion in Python code (for a default case in CLO processing)
- 1 function comment (in the whole system!)
- on the other hand, the build system does use Python lint and the Python syntactic static checker
- evidence that that there perhaps is validation code, but it is not included (and as such, causes the build system to fail)

Code Borrowing

- large amount of included code is lifted from libraries and is used for handling server-side crypto (particularly cert management)
- "borrowed" code has no attribution of source, authorship, or license
  - base64 from John Walker and in the public domain
  - countLines taken from GNU's wc (!)

Main Issues Found

- horrible engineering practices
  - not even an attempt at rigor or quality
- vote auditing does not exist
  - vote validity is a stub and only logged
  - malformed votes will be logged and stored and detected during decryption

Vote Analysis

```
def analyze(ik, vote, votebox):

    #   TODO: implement security checks
    #   such as verifying the correct size
    #   of the encrypted vote

    return []
```

Helios

Helios Big Picture

- E2E verifiable elections system
- best-in-breed of extant systems
- (strong) software independence arguments sometimes made against making E2E systems open source or used as arguments against the need for rigorous engineering

Helios Hacking

- at the end of June we setup a Helios server and ran a demonstration election
- asked everyone to either vote normally or to do some gentle, obvious, first-attempt hacking on their clients
- the first thing we tried worked: the Helios verifier reported the election as being 'ok' but the ballot box had been stuffed

Error/Hack Details

From http://documentation.heliosvoting.org/verification-specs/helios-v3-verification-specs

Thus, to verify a <ZK_PROOF_0..max> on a <ELGAMAL_CIPHERTEXT>, the following steps are taken.

```
verify_disjunctive_0..max_proof(ciphertext, max,
                                disjunctive_proof,
                                public_key):
for i in range(max+1):
  # the proof for plaintext "i"
  if not verify_proof(ciphertext, i, disjunctive_proof[i],
public_key):
    return False
..etc...
```

Note that the length of disjunctive_proof is never checked.

Election System Validation Redux

- examine open source academic systems, opened government election systems, and leaked commercial systems for evidence of validation techniques
  - e.g., unit and system testing, peer code review, traceability, static analysis, etc.
- a typical system has zero unit tests, a handful of system tests (re-run past election data as regression), no evidence of peer-review, no traceability between code artifacts and requirements, no static analysis
- the best-in-world systems have dozens or hundreds of hand-written unit and system tests and follow very traditional, manual validation techniques of decades ago

Verified Election Systems

Verified Election Systems

- some researchers have focused for just over a decade on verified election systems
- the KOA tally system (NL list-based vote counting systems used in EU elections)
- Votail (IE PR-STV vote counting system)
- Uilioch (model-finder-based system test generation)
- DiVS (DK vote counting system) and DVL (DK digital voter list system)
- Nijmegen system for Scotland (Scottish PR-STV vote counting system implemented in functional)
- Naish's unpublished work (AU STV & list-based vote counting system implemented in Prolog)

Verification, Generally

- translate law text into formally specified static and dynamic models (discovering many consistency errors in law as a result)
- formal concept analysis, architecture specifications, data models, contracts, scenarios, events, ownership, abstract state machines, protocols
- other groups: logic programming, purely functional programming languages, applied pi calculus
- my group: EBON system specification, low-level specifications written in the Java Modeling Language (JML), formal validation and verification to EAL level ~6
- new work includes use of verification technologies for .Net languages from MSR, FRAMA-C for verification of C, investigations into the use of old-school formal methods like Z, VDM, and (Event-)B and SPARK/Ada

V^4

The DemTech V^4 System

- conform to Danish law (thus, mandatorily only a traditional VVPAT kiosk)
- completely open development process, development artifacts, all dependent technology, etc. (all on GitHub)
- off-the-shelf, low-cost, single-election use hardware
  - 8-bit micro-controllers @ $4 each, 3D printed ballots, low-cost line printers, goal cost of system under $50 each
  - battery-powered & completely fault tolerant
  - formal validation and verification of all software artifacts (domain analysis, data model, architecture, all protocols, source code, and object code)

The DemTech V^4 System

- verified precinct-based tallying
- crypto-based verifiable election system
- foundation for citizen-supplied voting systems and citizen-supported digital elections
- provides a open platform for later experimentation and research in evoting
  - voting for the disabled
  - logic-based election schemes
  - verified election logging for audits

The DemTech V^4 System

- crypto-based verifiable election system
  - entire hardware platform and certified software system are public on GitHub
  - initial state of system known and public
  - election administrator or voter can request hardware and software audit receipt at any time with user-supplied salt
  - receipt encodes complete state of hardware and software, including running homomorphic tally of system

Open Election Systems

- interested in learning more? come join us and hack, for good, for democracy
- any expertise level and areas welcome
- we need the participation of the digital electorate both pre-election (system development and certification) and on election day (as election volunteers)
- developers with experience in Java, C, micro-controllers, and hacking are valuable

Conclusion

- secret, proprietary election systems will never be trusted by activists nor the public
- election outcomes and electorate trust are at high risk due to shoddy engineering
- rigorously engineering and verified election systems are the “high-end” solution
- open question: is it appropriate to trade the electorate’s comprehension of the election apparatus for strong security guarantees?

demtech.dk