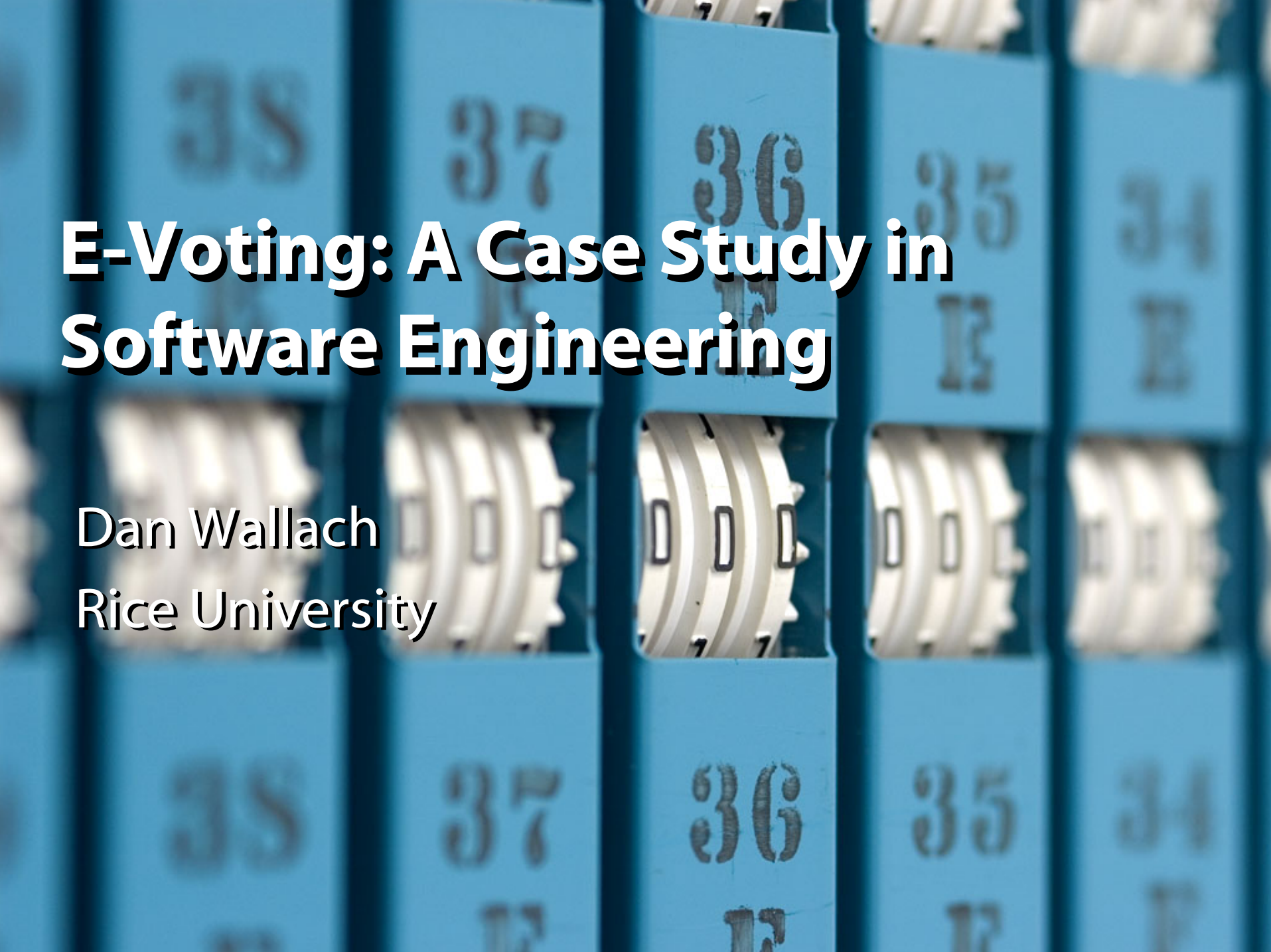


E-Voting: A Case Study in Software Engineering

Dan Wallach
Rice University

What elections are all about

Task: Measure voter intent

Goal: Convince the loser that he/she actually lost!

How?

ACCURATE

How can technology improve elections?

- Anonymity / privacy of voter
- Integrity of vote records / final tally
  - Software correctness / robustness
  - Tamper-resistance
- Human factors / accessibility for voters
- Procedural compliance / robustness

Voting technology glossary

Precinct-based optical scanner

Direct Recording Electronic (DRE)

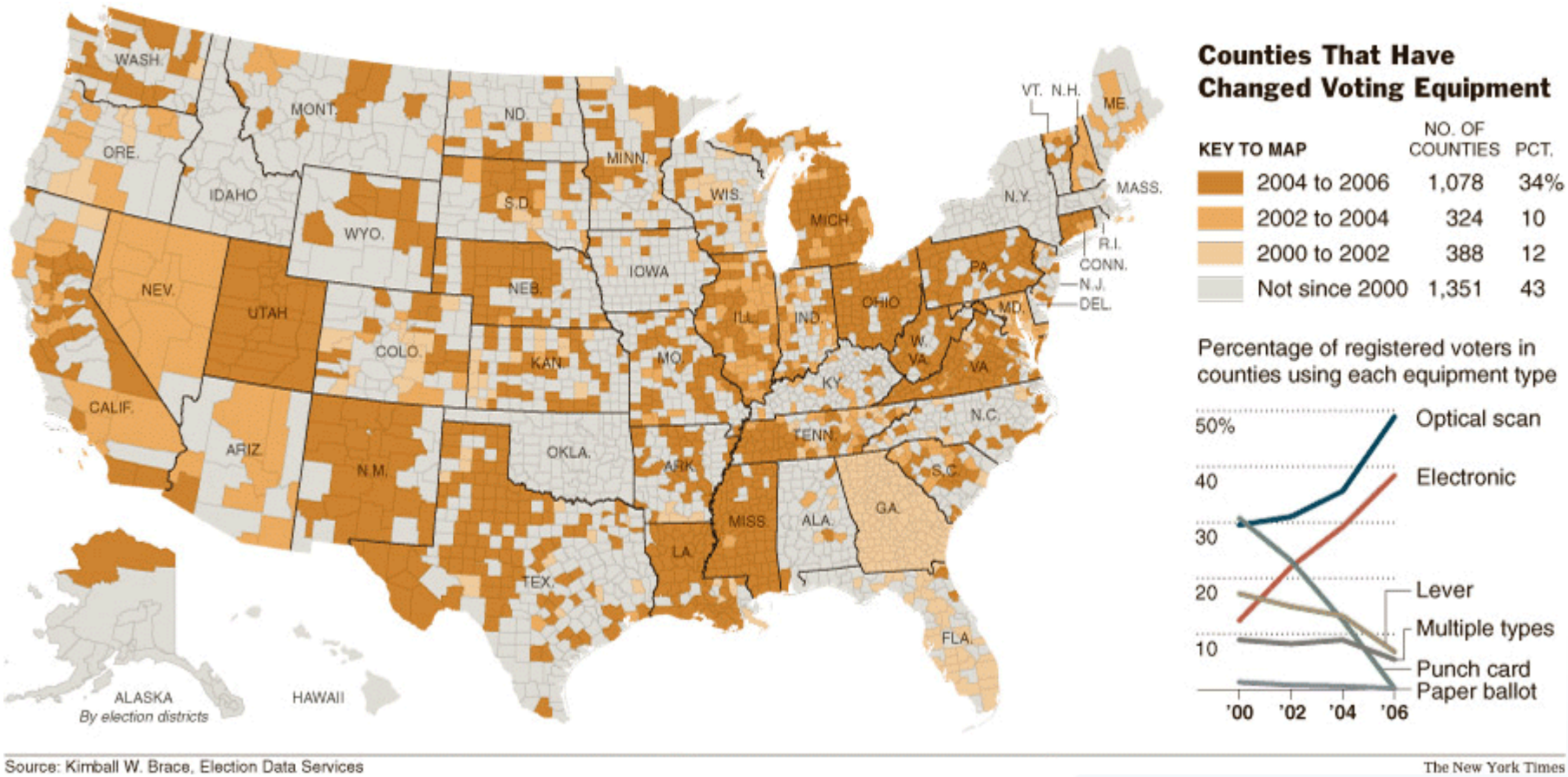
ACCURATE

Voting technology glossary

Voter-verifiable paper audit trail (VVPAT)

ACCURATE

Voting Machine Adoption



Source: Kimball W. Brace, Election Data Services

The New York Times

ACCURATE

Risk 1: Anonymity

Resistance to voter bribery / coercion

- First addressed with "Australian ballot," 1850's
- Inherent weakness of mail-in ballots (or Internet voting)

Still of concern today

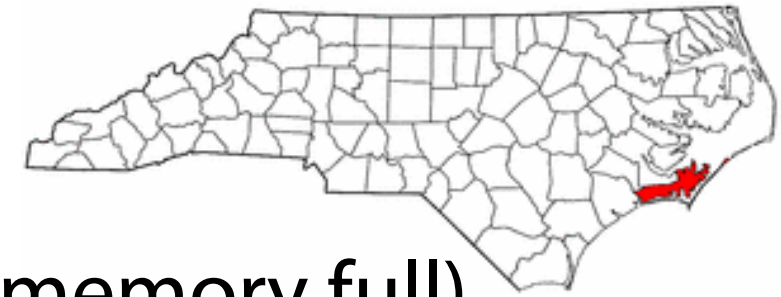
- Chain voting, pattern voting, camera-phones
- Votes recorded in order (paper-roll VVPAT)
- Timing issues (write-in votes in TX-22)

Risk 2: Bugs

Carteret County, NC

Nov. 2004 election

4438 votes lost (machine memory full)



Not uncommon issues:

- Hardware / smartcard / battery failures
- Inconsistent tallies (operator error?)

Risk 3: Software insecurity

Most studied: Diebold AccuVote-TS / TSx

- Poor software engineering
- Incorrect cryptography / protocols
- Possible for voters to cast multiple votes
- Vulnerable to malicious software upgrades

Tadayoshi Kohno, Adam Stubblefield, Aviel D. Rubin and Dan S. Wallach, *Analysis of an Electronic Voting System*, IEEE Security & Privacy 2004.

How *not* to encrypt data

```
#define DESKEY
  ((des_key*)"F2654hD4")
```

One key for every voting machine, everywhere

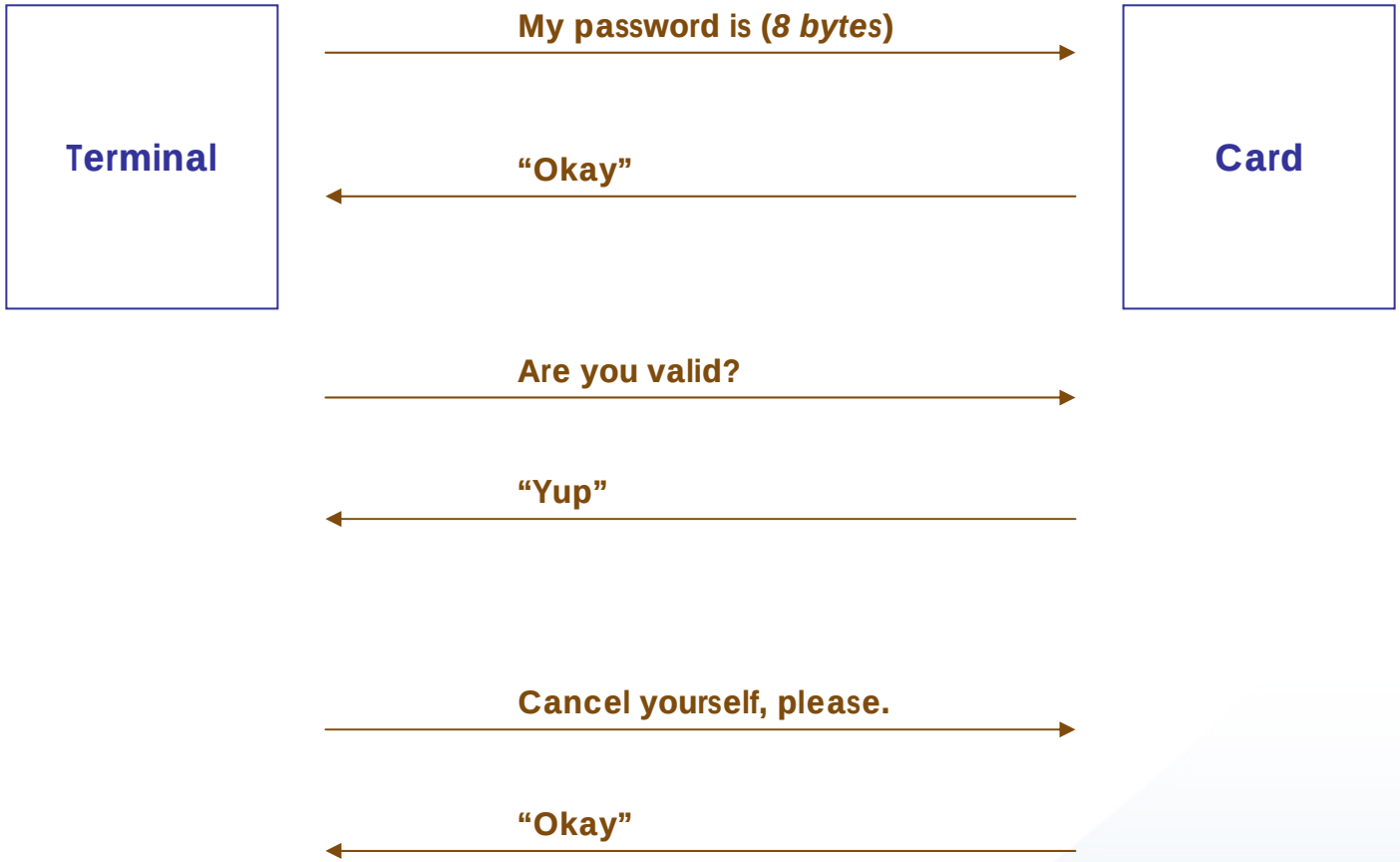
Doug Jones (Iowa official) found this in 1997

- Still present in current systems!

  (DES replaced with AES, but same key)

Comparable naïveté with other vendors

Diebold's smart card protocol



Princeton study of Diebold

All physical locks use the same key

- Common for hotel mini-bars, office furniture

Implemented a voting machine virus

- Software update from memory card
- No authentication of any kind
- Infection can spread via memory cards (no networking necessary)

http://itpolicy.princeton.edu/voting/

Dutch study of Nedap ES3B

- Poor physical key security
- Easily modified ROM chips
- Easily observed RF emissions
- Demonstration: chess software

http://www.wijvertrouwenstemcomputersniet.nl/images/9/91/Es3b-en.pdf

Risk 4: Procedural failures

Poll workers have many responsibilities

- Machine setup
  - Validate date, machines zeroed, etc.
- End-of-day tallying / reporting
- Unusual events
  - “Fleeing voters” (forgot to press “cast ballot”)
  - Machine / memory card failures

Webb County (Laredo) experience

March 7, 2006: primary election

First local use of ES&S DRE machines

Margin of victory in Flores v. Lopez
was ~100 of ~50K votes (0.2%)

Significant procedural problems

Joint work with Dan Sandler

Normal event logs

Votronic	PEB#	Type	Date	Time	Event
5117865	161061	SUP	03/06/2006	16:31:12	01 Terminal clear and test
	161126	SUP	03/07/2006	07:09:37	09 Terminal open
			03/07/2006	07:13:50	13 Print zero tape
			03/07/2006	07:15:39	13 Print zero tape
	160973	SUP	03/07/2006	12:32:24	20 Normal ballot cast
			03/07/2006	16:59:19	20 Normal ballot cast
			03/07/2006	18:06:23	20 Normal ballot cast
			03/07/2006	18:25:56	20 Normal ballot cast
			03/07/2006	18:32:18	20 Normal ballot cast
			03/07/2006	18:48:54	20 Normal ballot cast
			03/07/2006	18:56:03	20 Normal ballot cast
			03/07/2006	19:01:52	20 Normal ballot cast
	161126	SUP	03/07/2006	19:39:41	10 Terminal close

ACCURATE

Issue #1: Test votes

Votronic	PEB#	Type	Date	Time	Event
5145172	161061	SUP	03/06/2006	15:04:09	01 Terminal clear and test
	161126	SUP	03/06/2006	15:19:34	09 Terminal open
	160973	SUP	03/06/2006	15:26:59	20 Normal ballot cast
			03/06/2006	15:30:39	20 Normal ballot cast
	161126	SUP	03/06/2006	15:38:37	27 Override
			03/06/2006	15:38:37	10 Terminal close

- Election was on 3/7
- 93 votes with the wrong dates
  - Four machines: clock probably set wrong
  - 26 machines: test votes included in final tally

  (one Republican ballot, one Democrat ballot, repeated on each test machine)

ACCURATE

Issue #2: Lost votes?

Votronic	PEB#	Type	Date	Time	Event
5140052	161061	SUP	03/07/2006	15:29:03	01 Terminal clear and test
	160980	SUP	03/07/2006	15:31:15	09 Terminal open
			03/07/2006	15:34:47	13 Print zero tape
			03/07/2006	15:36:36	13 Print zero tape
	160999	SUP	03/07/2006	15:56:50	20 Normal ballot cast
			03/07/2006	16:47:12	20 Normal ballot cast
			03/07/2006	18:07:29	20 Normal ballot cast
			03/07/2006	18:17:03	20 Normal ballot cast
			03/07/2006	18:37:24	22 Super ballot cancel
			03/07/2006	18:41:18	20 Normal ballot cast
			03/07/2006	18:46:23	20 Normal ballot cast
	160980	SUP	03/07/2006	19:07:14	10 Terminal close

- Most machines cleared on 3/6
- Ten machines cleared and used on 3/7

  Poll workers not supposed to do this!

ACCURATE

Issue #3: Insufficient audit data

- Many machines cleared after the election
  - Only CompactFlash memory cards remained
- Many "zero tapes" were lost
- No records for "cancelled ballots"

  (Poll workers *supposed* to keep a log)

Issue #4: Unwieldy equipment

ACCURATE

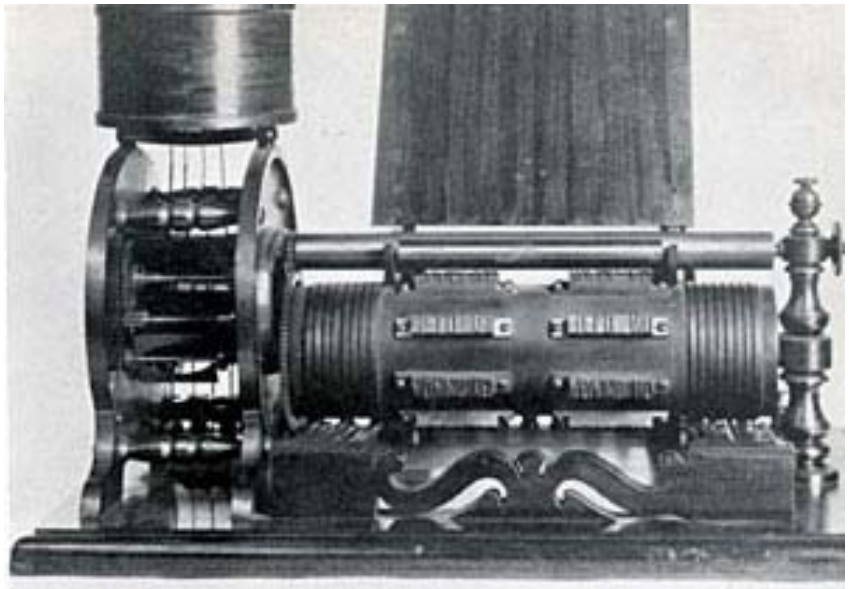
Meaningless certification

- All of these systems are “certified”
  - Clear evidence of insufficient processes/laws
- Most certification documents are secret
- Testing authorities skip “hard” tests
  - Or, no evidence of doing them properly
- No consideration of development process
- No consideration of procedural difficulties
- No oversight of testing authorities

Research: Build a better machine

Step 1: Use Moore's Law

- Computation is free
- Disk storage is infinite
- *N* is small enough that $O(N^2)$ is still cheap

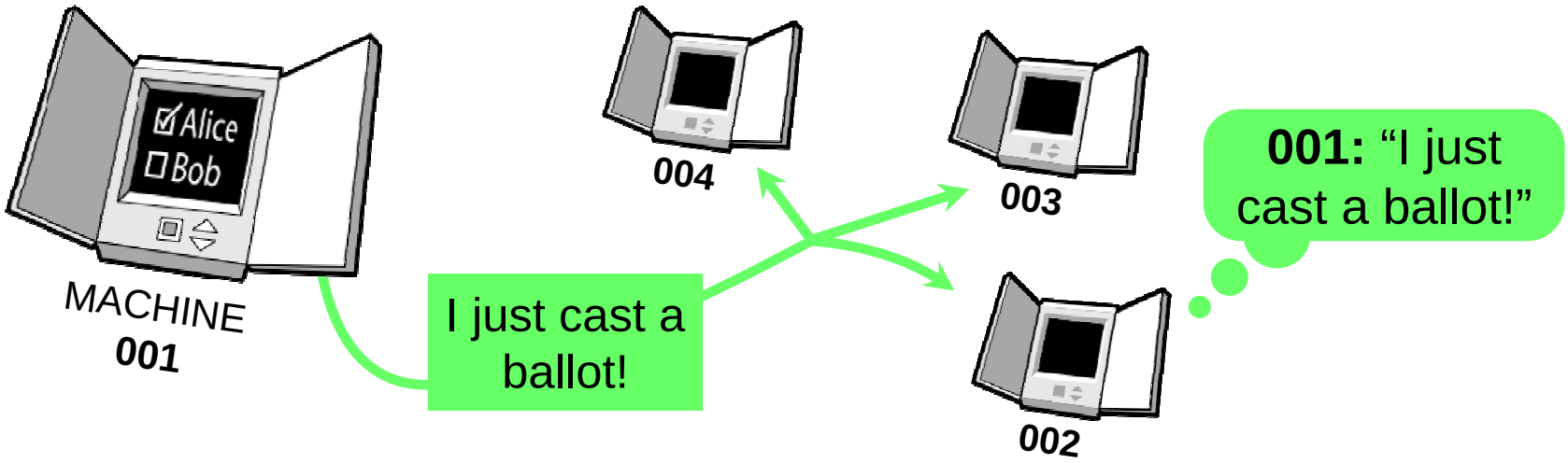


Implications?

- Never delete anything, ever
- Digitally sign everything
- Store redundant copies everywhere

Networked voting systems

Local network not Internet



Massive redundancy, but is it secure?

Example: “Protective counters”

Defense against ballot stuffing

- Lever machines: visible mechanical counter
- Diebold has a text file on the flash card
- Our system: records every vote ever cast

Network data handling

Two classes of data: events and votes

- Events are public: sign and log everything
- Timeline entanglement to preserve history (*Maniatis and Baker '02*)

Need to preserve anonymity of votes

- Option 1: Assume a trusted network
- Option 2: Encrypt the votes

Network vote storage?

Issue: who gets to decrypt?

- Requirements vary from state to state

If local precinct needs vote totals

- Homomorphic encryption (allows computation of vote totals)

If local precinct needs individual ballots

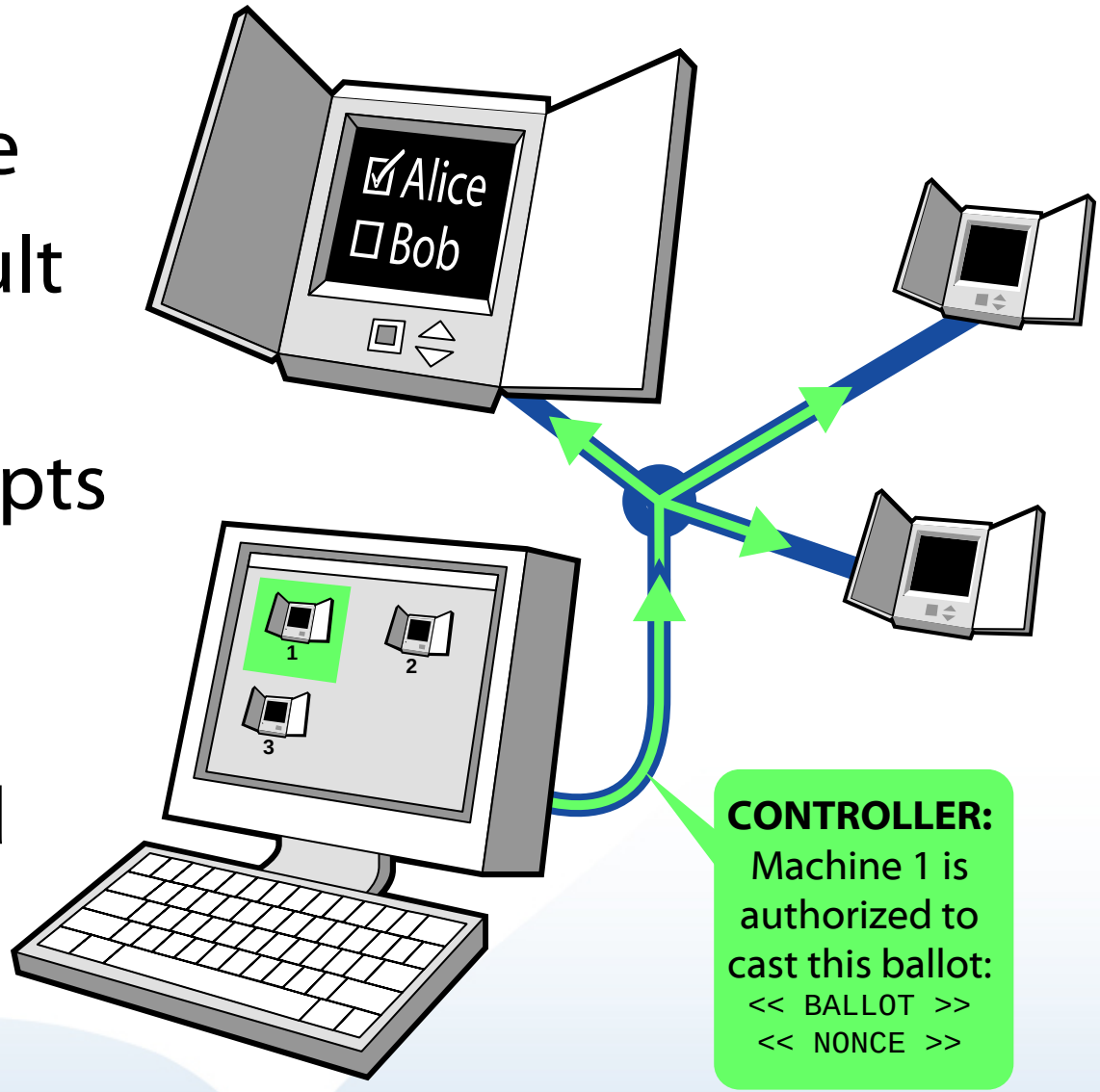
- Verifiable mix networks

Work in progress…

Voting protocol

1. Authorize vote
2. Broadcast result (encrypted)
3. Console decrypts and tallies

Others just record



Plaintext votes (local store)

Votes should not be in the order cast

- Option #1: randomize the order
- Option #2: sort the ballots (Chaum)

Simple solution: One sorted list per election

- Election IDs need to be globally unique

Pragmatic benefits

Admin console shows status of all machines

- Votes cast, battery running low, etc.

Admin console tells poll workers what to do

- Less opportunity for poll-worker error

Voting machines are interchangeable

- Add/remove machines on the fly

Software tampering?

Secure bootstrapping / attestation

- Machines can “challenge” each other
- Just log the result, resolve conflicts later

Burn software to ROM (not Flash)

- Ballot definition downloaded for each vote
  - State-specific rules part of the ballot definition
- Less need for software upgrades

And, of course, VVPAT

- Printed ballots should take legal precedence

Software engineering

Strong type systems are security mechanisms

- No concerns about buffer overflows
- Narrow public interfaces between modules
- Easy to verify using *grep*

Less is more

- Diebold: ~35K lines of C++ (plus Windows CE)
- Yee '06: 400 lines of Python (*pygame*, SDL, …)
- Our current prototype: ~4K lines of Java

Recount / auditing process

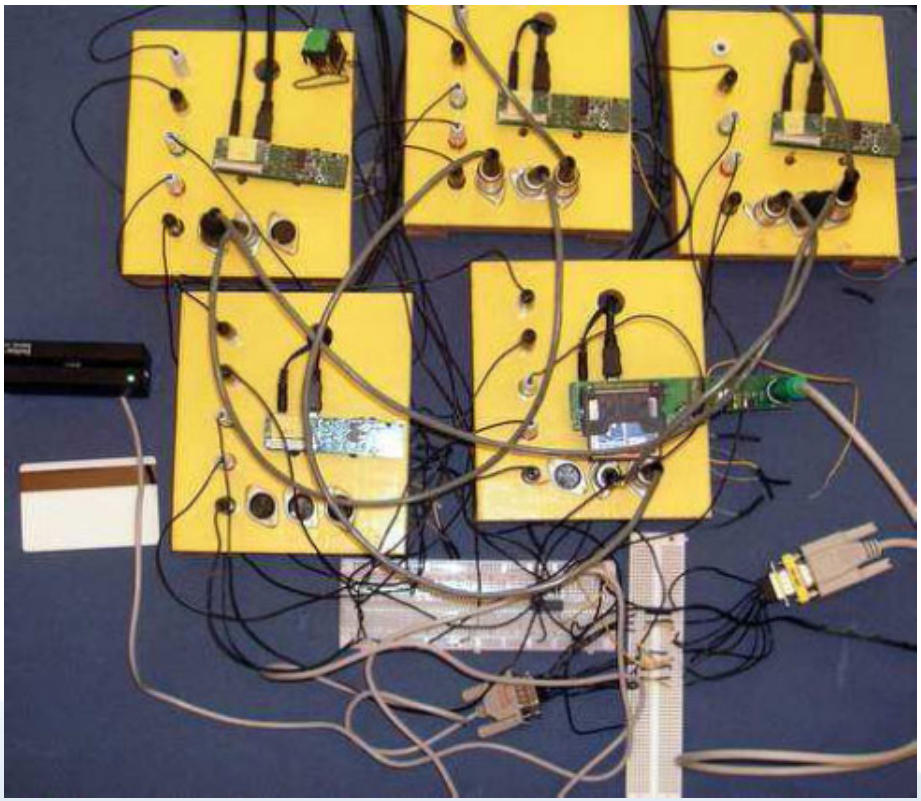
1. Tally the votes from the admin consoles
2. Sample the VVPAT: ensure consistency
3. Sample the machines: ensure consistency

If inconsistencies occur, study entire precinct

- Computer-aided auditing

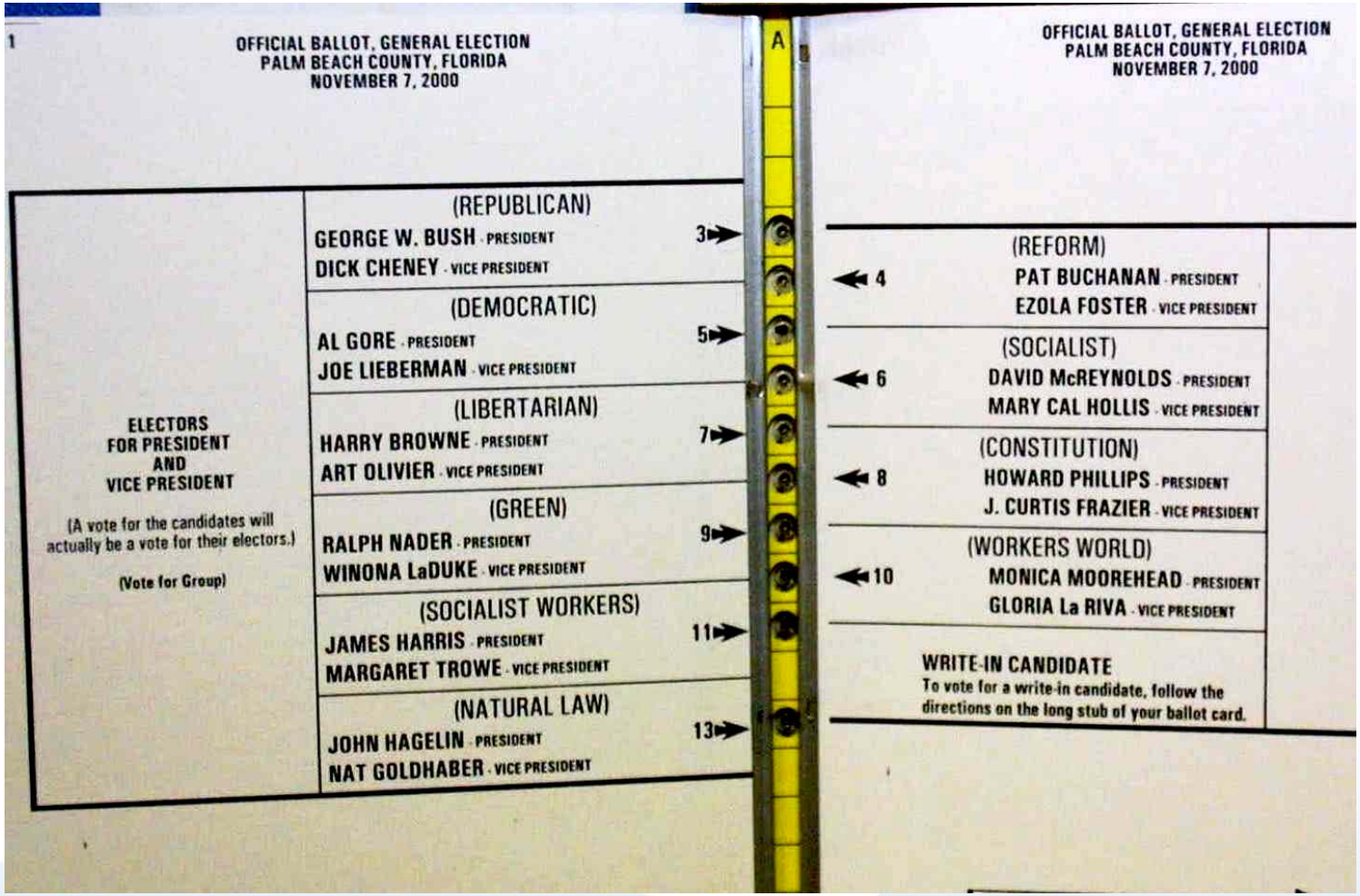
Related: hardware separation

Sastry et al., *Designing Voting Machines for Verification* (Usenix Security '06).



- Property 1: No voter session can interfere with a previous session.
- Property 2: A ballot cannot be cast without voter consent.
- Core idea: separate HW modules, reset after each vote.

Human factors matters



ACCURATE

Human factors matters

If it's not usable, it's not secure.

- Necessity: human subject experiments

Example question: do people read VVPAT?

- Need a voting machine that lies!

Joint work with Mike Byrne (Rice Psychology)

- Measure usability of voting UI features
- Poll worker usability as well

HF work in progress

- Paper ballots are most consistent across different demographics
- Education / prior experience don't help
- Error rates are stunning (1% or worse)

ACCURATE Voting Center

NSF research center, $7.5M

- PIs at U.C. Berkeley, U. Iowa, Johns Hopkins, Rice, SRI, Stanford

Research into better voting systems

- Cryptographic protocols
- Verifiable software
- Tamper resistance
- Human factors
- Policy implications

accurate-voting.org

ACCURATE

A CENTER FOR
CORRECT, USABLE,
RELIABLE, AUDITABLE,
AND TRANSPARENT ELECTIONS

ACCURATE


Conclusions

Current DRE voting systems have real problems

- Independent certification is (currently) meaningless
- Significant failures observed in the field
- Non-trivial margins of error

Good science can improve the situation

- Better software engineering
- Better auditability / fault tolerance
- Better human factors