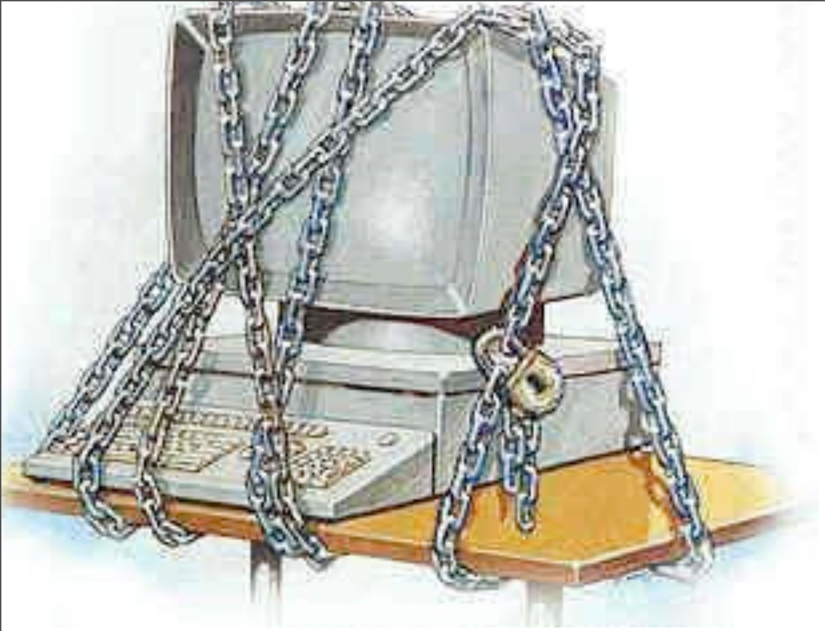
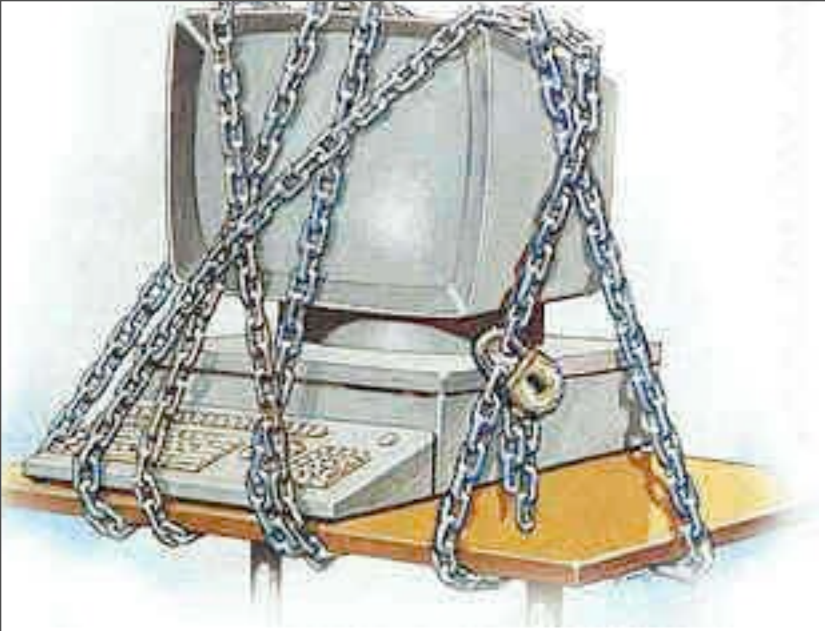


Computer Security

Stephen Checkoway

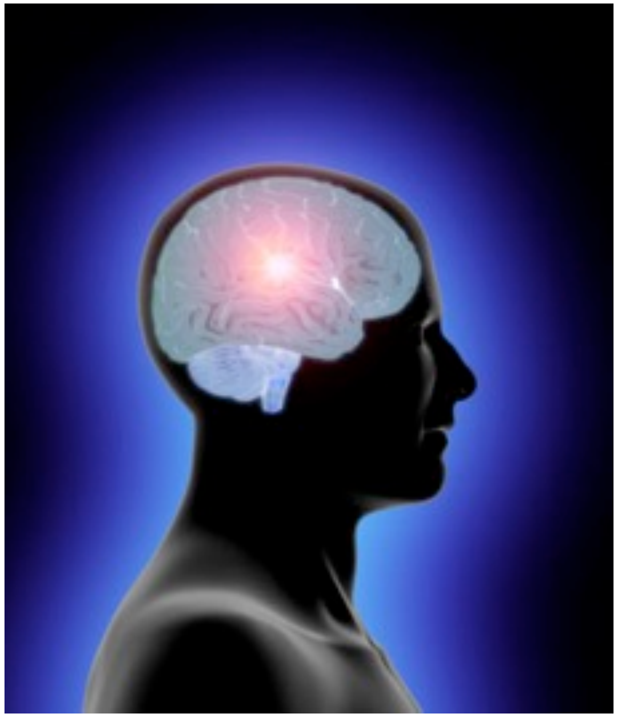


~~Computer Security~~

Stephen Checkoway

The Security Mindset

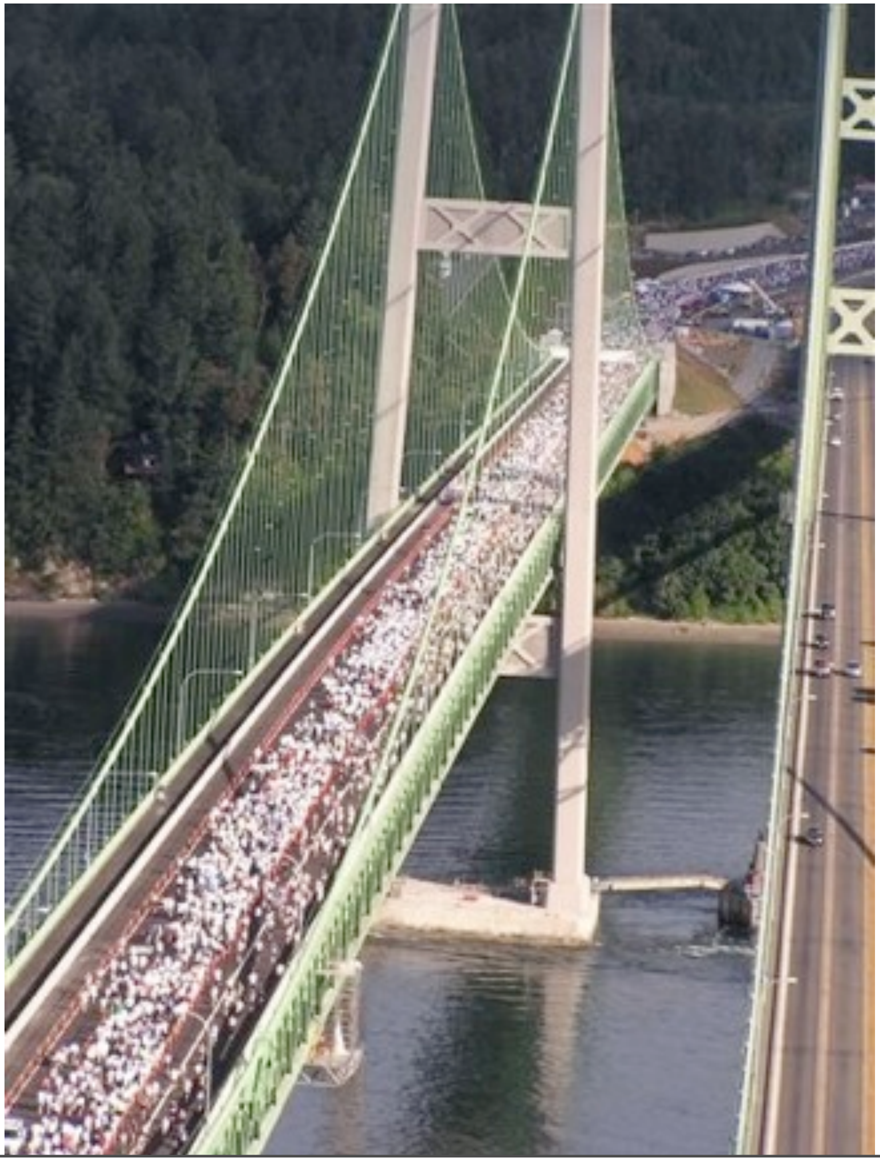
Stephen Checkoway



Talk Outline

- Security mindset overview
- 4 concrete examples of where a security mindset is needed, but was lacking
- My research at UCSD
- Where to go from here

Engineering Mindset vs. Security Mindset



Security Mindset Failure 1

Problem: Prevent people on no-fly lists from flying.

Ticket Counter

Security Checkpoint

Gate

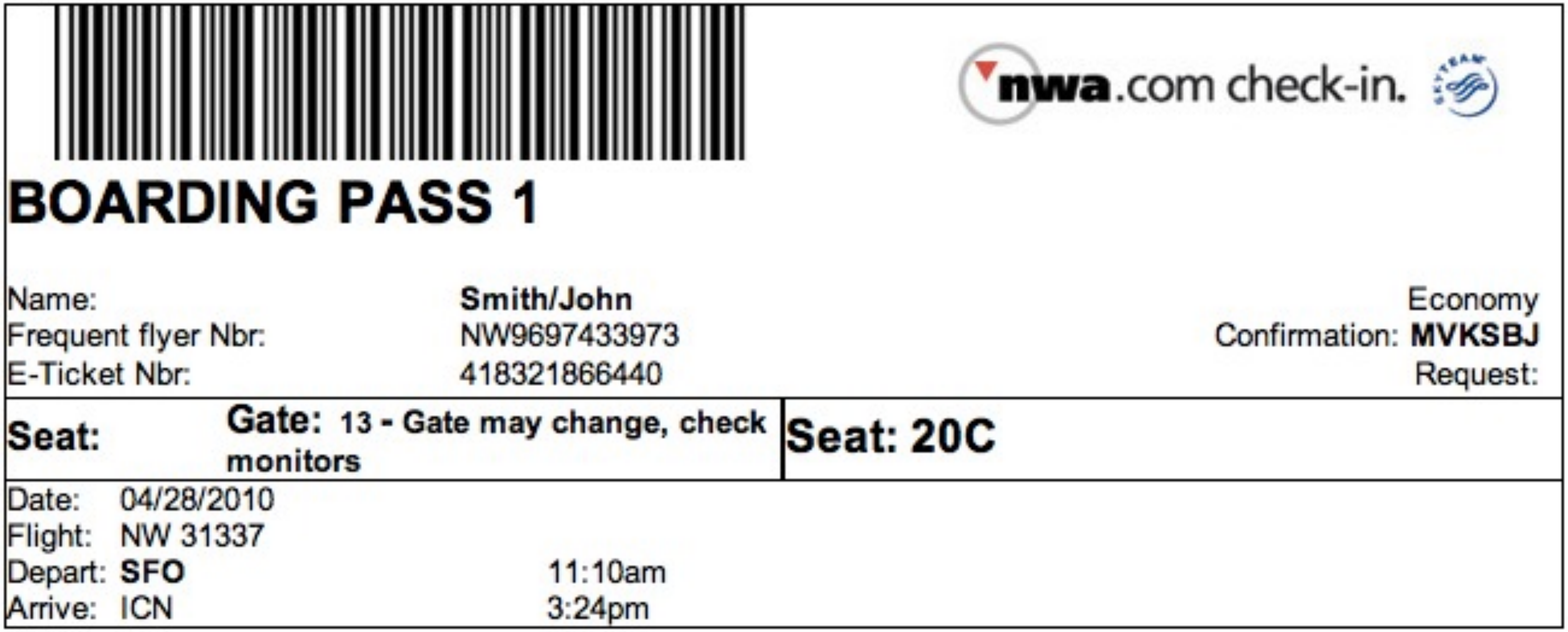
Security Mindset Failure 1

Problem: Prevent people on no-fly lists from flying.

Location	ID Valid	Boarding Pass Valid	Boarding Pass Matches ID
Ticket Counter	✓?	✓	✓
Security Checkpoint	✓?	✗	✓
Gate	✗	✓	✗

Security Mindset Failure 1

Problem: Prevent people on no-fly lists from flying.



nwa.com check-in.

BOARDING PASS 1

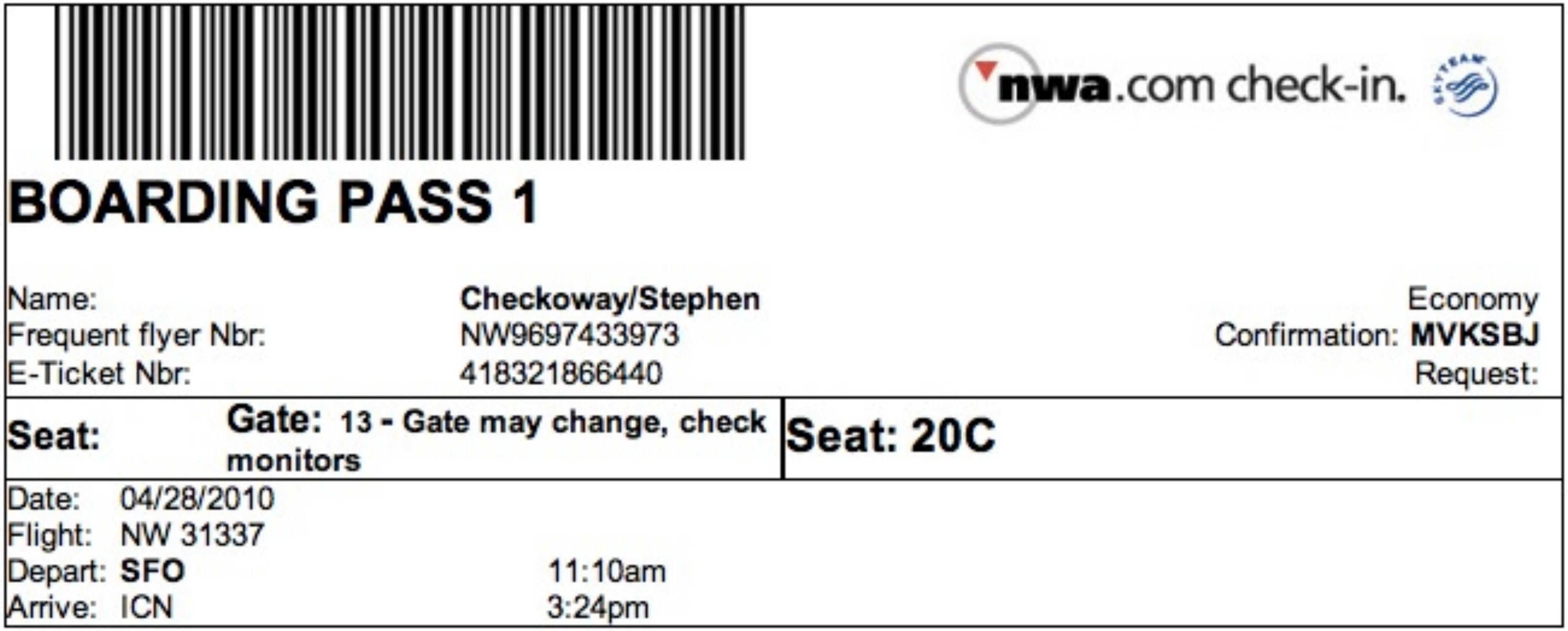
Name:	**Smith/John**	Economy
Frequent flyer Nbr:	NW9697433973	Confirmation: **MVKSBJ**
E-Ticket Nbr:	418321866440	Request:

Seat: **Gate:** **13 - Gate may change, check monitors** **Seat: 20C**

Date:	04/28/2010	
Flight:	NW 31337	
Depart:	**SFO**	11:10am
Arrive:	ICN	3:24pm

Security Mindset Failure 1

Problem: Prevent people on no-fly lists from flying.



nwa.com check-in.

BOARDING PASS 1

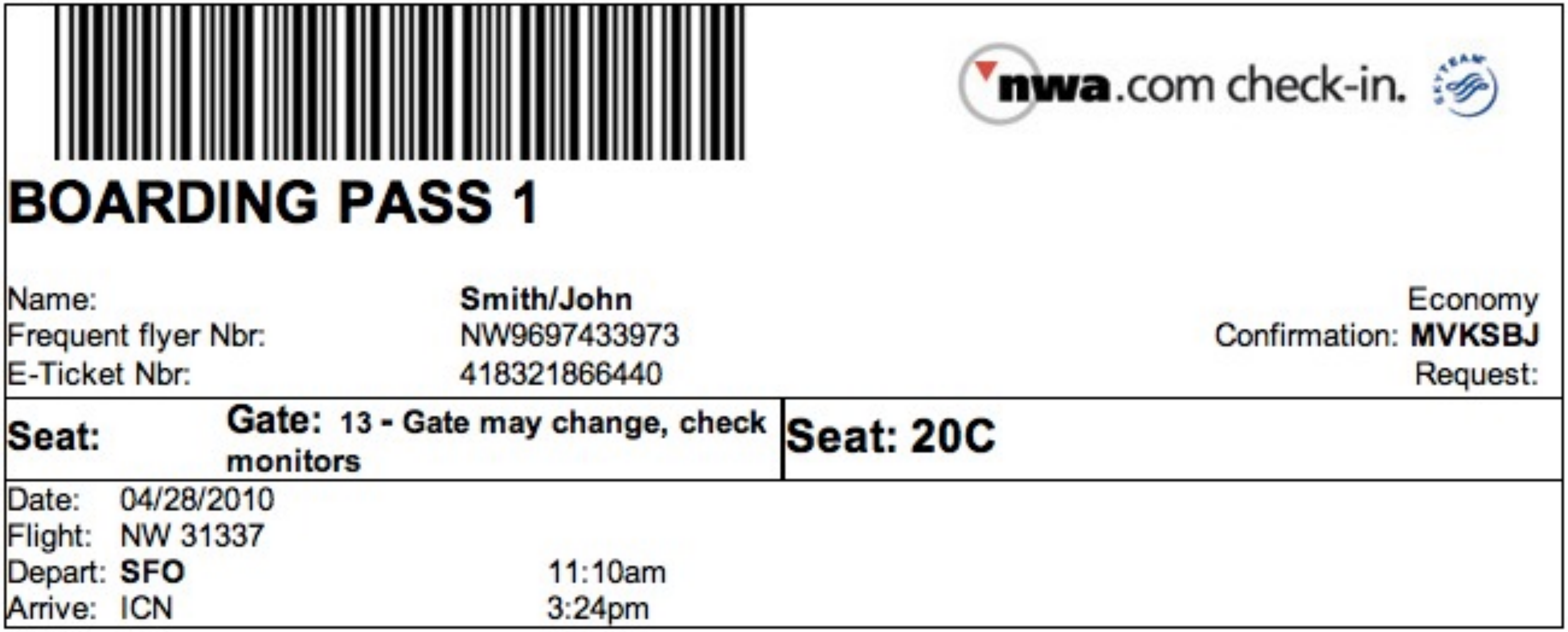
Name:	**Checkoway/Stephen**	Economy
Frequent flyer Nbr:	NW9697433973	Confirmation: **MVKSBJ**
E-Ticket Nbr:	418321866440	Request:

Seat: **Gate:** 13 - Gate may change, check monitors **Seat: 20C**

Date:	04/28/2010	
Flight:	NW 31337	
Depart:	**SFO**	11:10am
Arrive:	ICN	3:24pm

Security Mindset Failure 1

Problem: Prevent people on no-fly lists from flying.



nwa.com check-in.

BOARDING PASS 1

Name:	**Smith/John**	Economy
Frequent flyer Nbr:	NW9697433973	Confirmation: **MVKSBJ**
E-Ticket Nbr:	418321866440	Request:

Seat: **Gate:** **13 - Gate may change, check monitors** **Seat: 20C**

Date:	04/28/2010	
Flight:	NW 31337	
Depart:	**SFO**	11:10am
Arrive:	ICN	3:24pm

Security Mindset Failure 1

Problem: Prevent people on no-fly lists from flying.

What (simple) change can be made to these procedures to prevent this?

Location	ID Valid	Boarding Pass Valid	Boarding Pass Matches ID
Ticket Counter	✓?	✓	✓
Security Checkpoint	✓?	✗	✓
Gate	✗	✓	✗

Security Mindset Failure 2

Something top secret!

Security Mindset Failure 3

Engineering problem:
Read a line of input and parse it as a number.

Security Mindset Failure 3

Engineering problem:
Read a line of input and parse it as a number.

```
int get_int()
{
    char buffer[12];
    gets( buffer );
    return atoi( buffer );
}
```

Security Mindset Failure 3

Engineering problem:
Read a line of input and parse it as a number.

Input	Output
`"-37"`	`-37`
`"25753abc"`	`25753`
`"12345678901234567890"`	`???`

Security Mindset Failure 3

Engineering problem:
Read a line of input and parse it as a number.

```
int get_int()
{
    char buffer[12];
    gets( buffer );
    return atoi( buffer );
}
```

Security Mindset Failure 3

Engineering problem:
Read a line of input and parse it as a number.

```
int get_int()
{
    char buffer[12];
    gets( buffer );
    return atoi( buffer );
}
```

Fixed length

Security Mindset Failure 3

Engineering problem:
Read a line of input and parse it as a number.

```
int get_int()
{
    char buffer[12];
    gets( buffer );
    return atoi( buffer );
}
```

Fixed length

Security mindset:
What input can I give to make this program misbehave?

Security Mindset Failure 3

Engineering problem:
Read a line of input and parse it as a number.

```
int get_int()
{
    char buffer[12];
    gets( buffer );
    return atoi( buffer );
}
```

Fixed length

No safe usage

Security mindset:
What input can I give to make this program misbehave?

A Crash Course on Stack Smashing

A Crash Course on Stack Smashing

- Watching a movie (Hackers, of course)

A Crash Course on Stack Smashing

- Watching a movie (Hackers, of course)
    - Making dinner

A Crash Course on Stack Smashing

- Watching a movie (Hackers, of course)
  - Making dinner
    - Telephone call

A Crash Course on Stack Smashing

- Watching a movie (Hackers, of course)
    - Making dinner
        - Telephone call
    - Resume making dinner (from where you left off)

A Crash Course on Stack Smashing

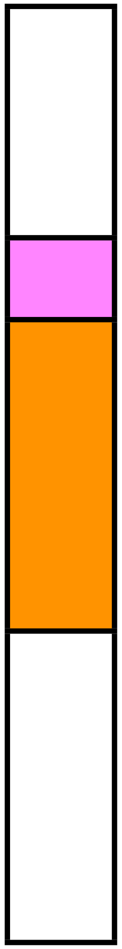
- Watching a movie (Hackers, of course)
    - Making dinner
        - Telephone call
    - Resume making dinner (from where you left off)
- Resume watching the movie (from where you left off)

A Crash Course on Stack Smashing

- Watching a movie (Hackers, of course)
    - Making dinner
        - Telephone call
    - Resume making dinner (from where you left off)
- Resume watching the movie (from where you left off)
- Need to keep track of where we were in each task: write down where you are in each task before beginning the next—use a stack!

A Crash Course on Stack Smashing

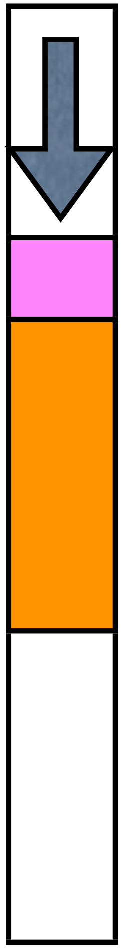
Stack



- Program needs to remember it is doing
- Stack grows down
- Save location before function call
- Local variables

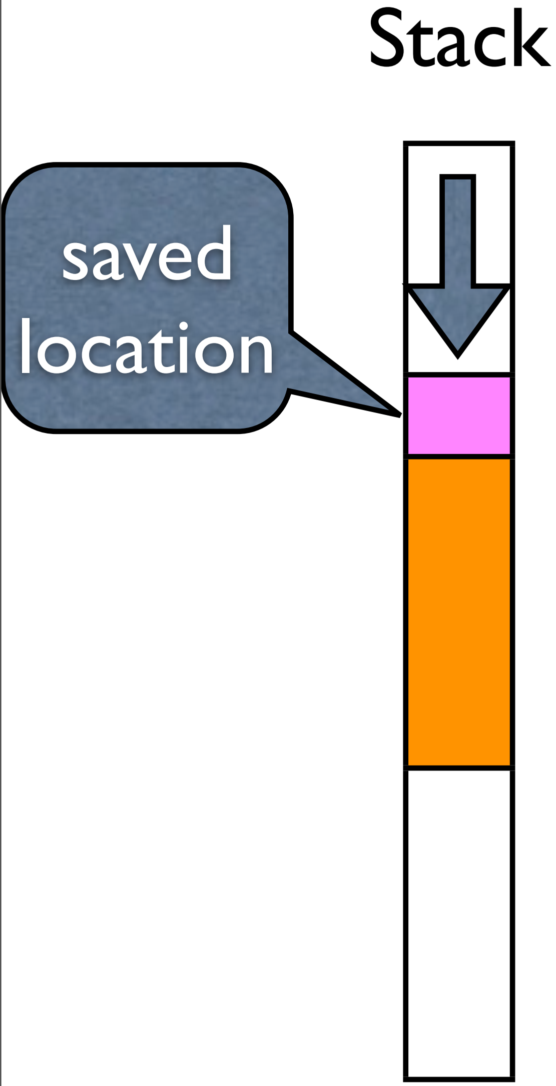
A Crash Course on Stack Smashing

Stack



- Program needs to remember it is doing
- Stack grows down
- Save location before function call
- Local variables

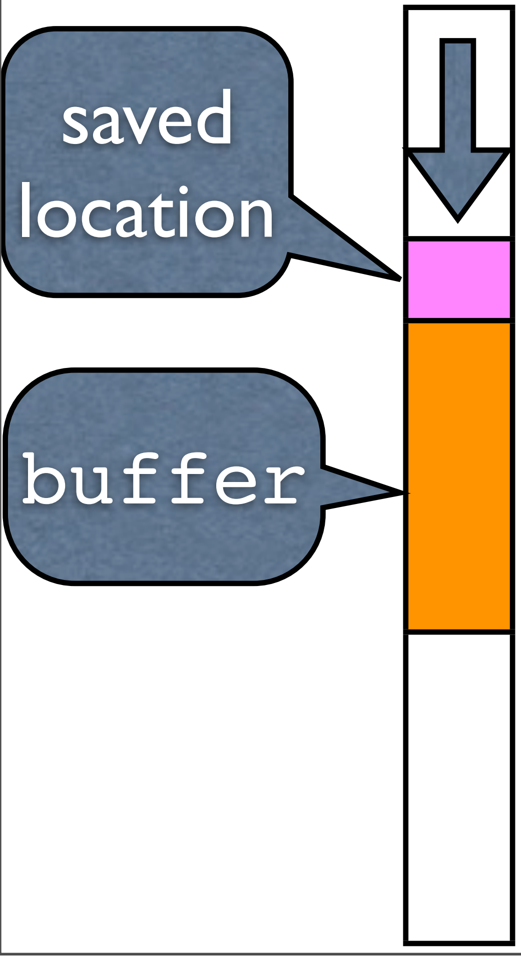
A Crash Course on Stack Smashing



- Program needs to remember it is doing
- Stack grows down
- Save location before function call
- Local variables

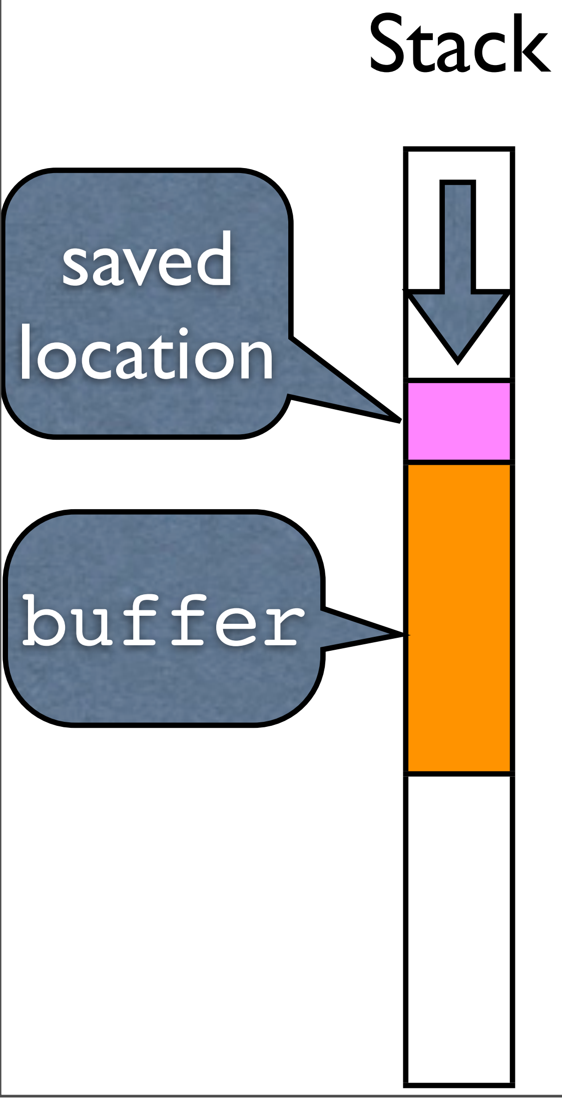
A Crash Course on Stack Smashing

Stack

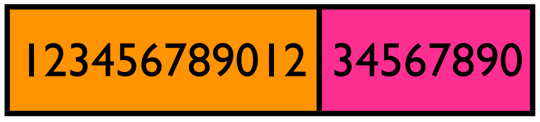


- Program needs to remember it is doing
- Stack grows down
- Save location before function call
- Local variables

A Crash Course on Stack Smashing

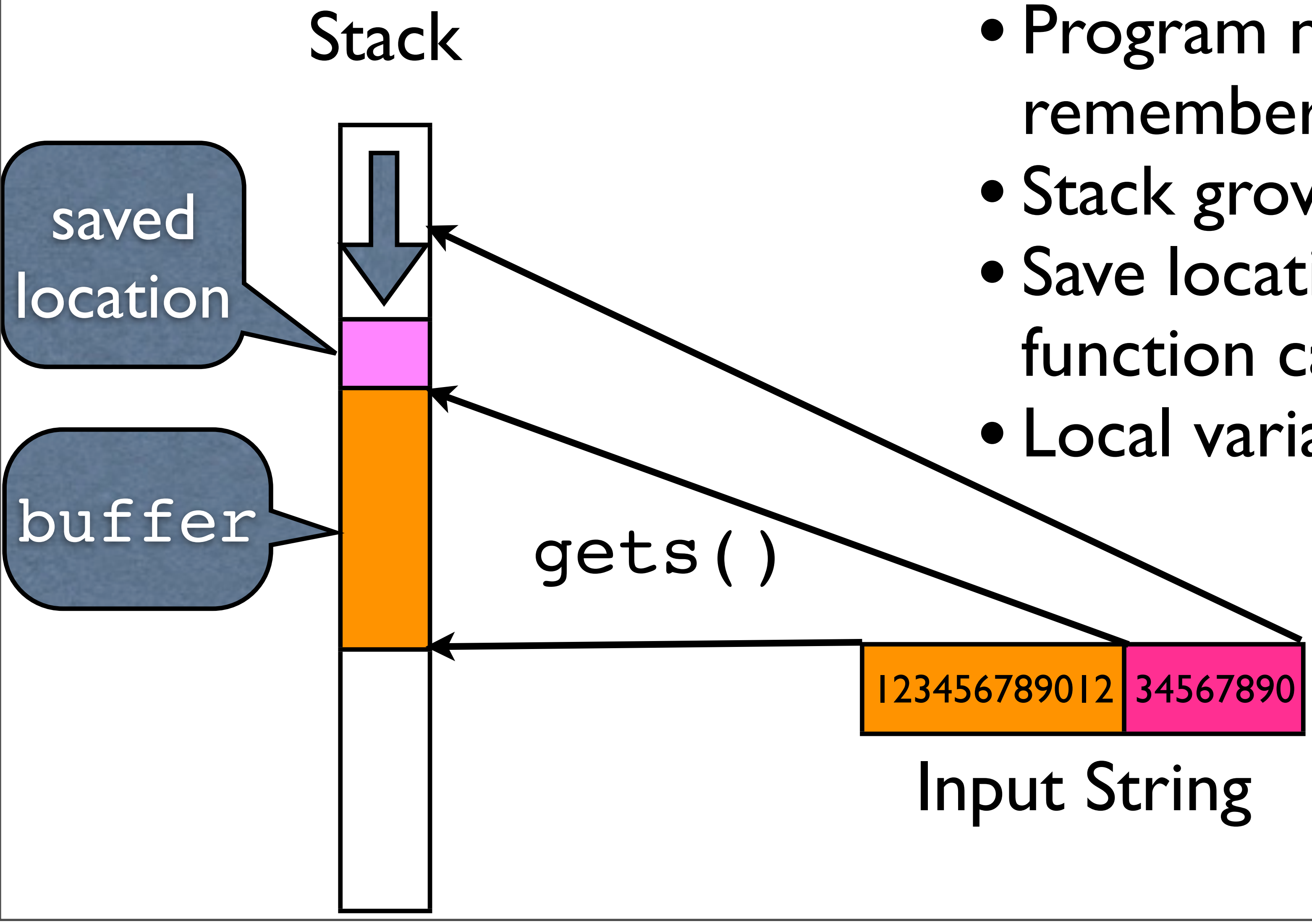


- Program needs to remember it is doing
- Stack grows down
- Save location before function call
- Local variables



Input String

A Crash Course on Stack Smashing



- Program needs to remember it is doing
- Stack grows down
- Save location before function call
- Local variables

Security Mindset Failure 3

`gets()` is not safe, but how bad is it?

- Arbitrary code execution
- Manipulate program's data
- Invoke other programs

Security Mindset Failure 3

Buffer overflows are kinds of memory-safety errors
Others include:

- Stack overflows
- Heap overflows
- Dangling pointers
- Uninitialized memory

Security Mindset Failure 3

Buffer overflows are kinds of memory-safety errors
Others include:

- Stack overflows
- Heap overflows
- Dangling pointers
- Uninitialized memory

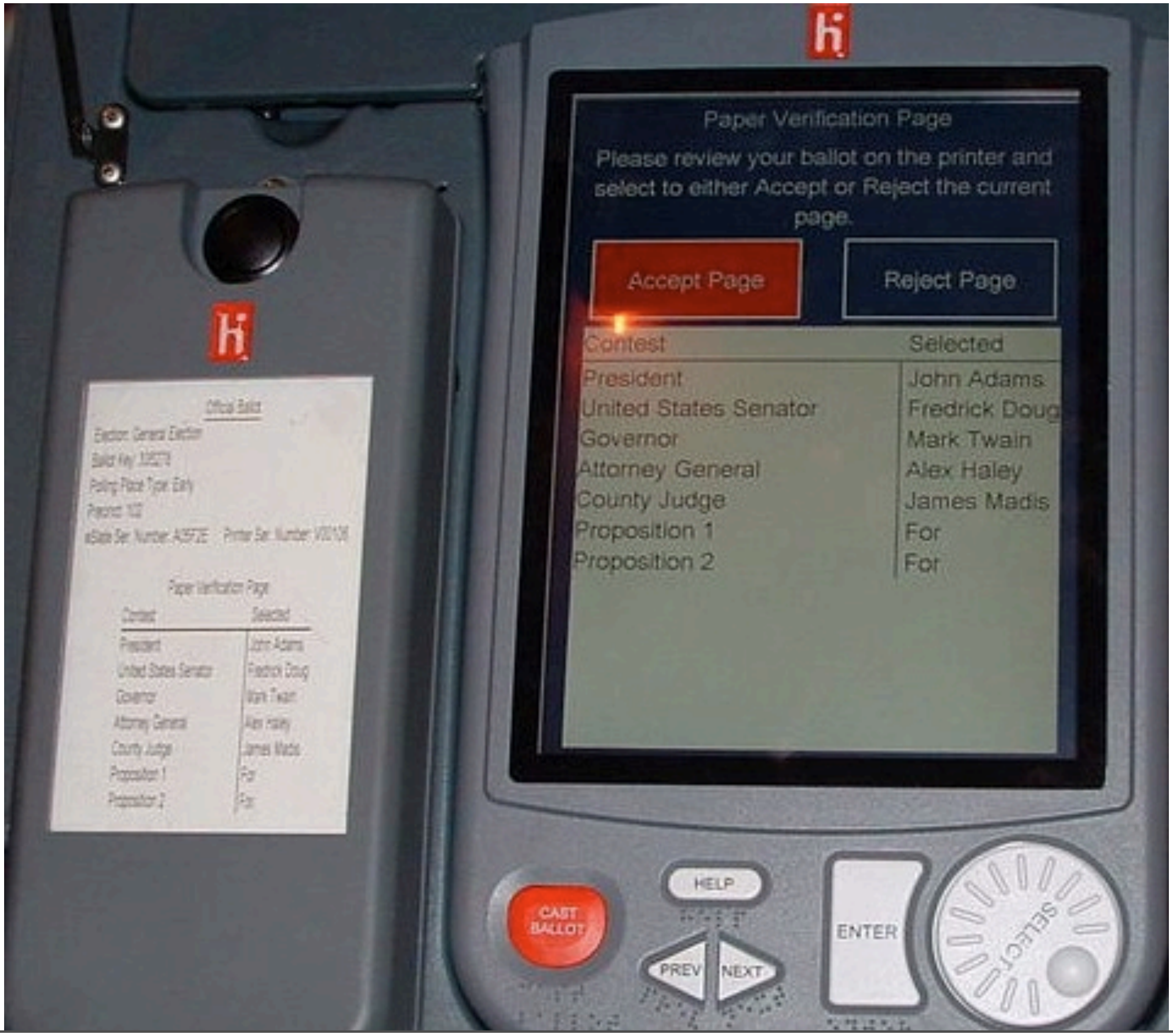
What can be done to prevent buffer overflows?

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.



Official Ballot

Election: General Election

Ballot Key: 335274

Polling Place Type: Early

Precinct: 102

eSlate Ser. Number: A05F2E Printer Ser. Number: V00106

Paper Verification Page

Contest	Selected
President	John Adams
United States Senator	Fredrick Doug
Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

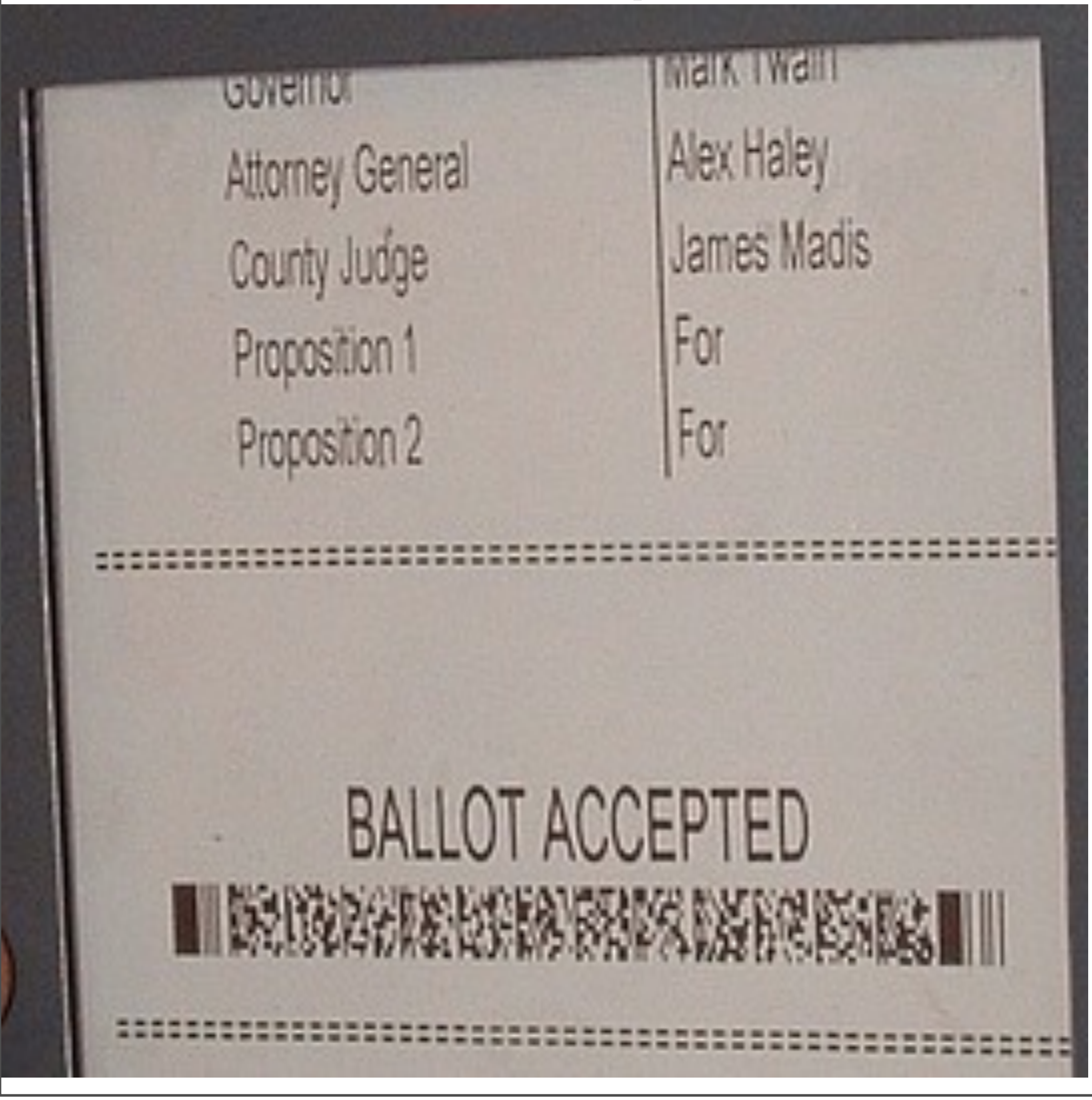
Paper Verification Page

Please review your ballot on the printer and select to either Accept or Reject the current page.

Contest	Selected
President	John Adams
United States Senator	Fredrick Doug
Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.

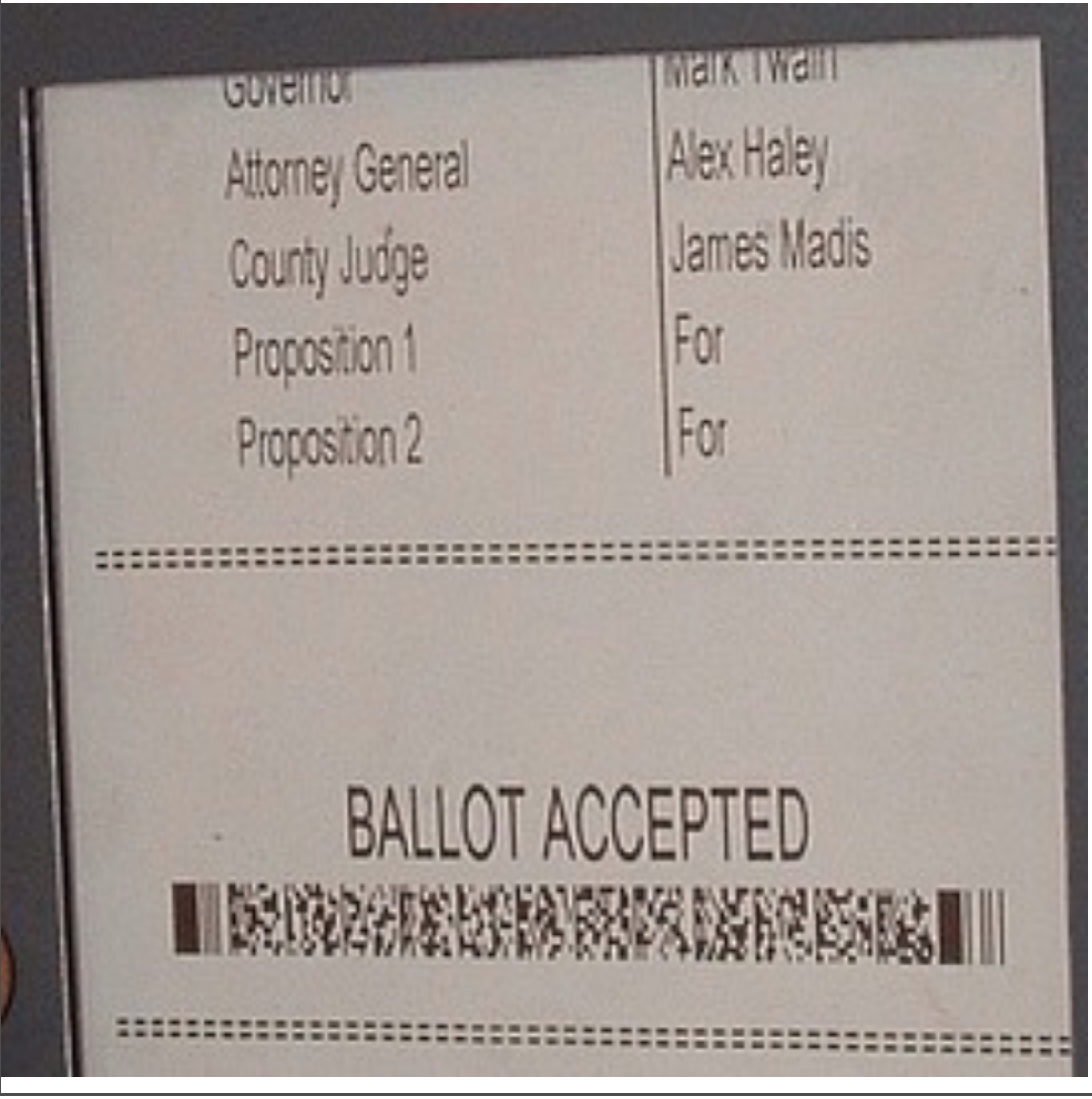


Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

BALLOT ACCEPTED

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.



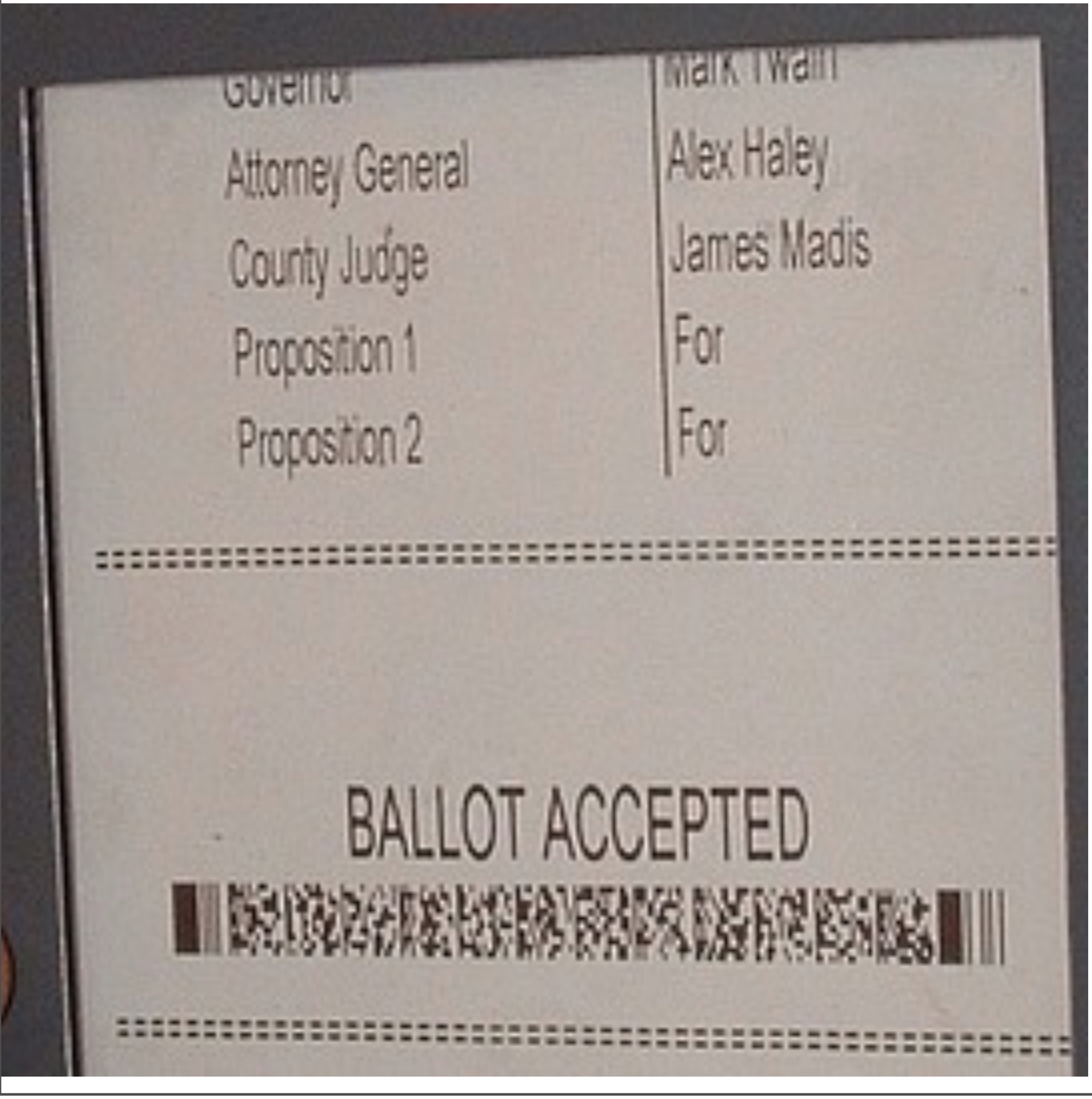
Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

BALLOT ACCEPTED

P: Human non-readable record

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.



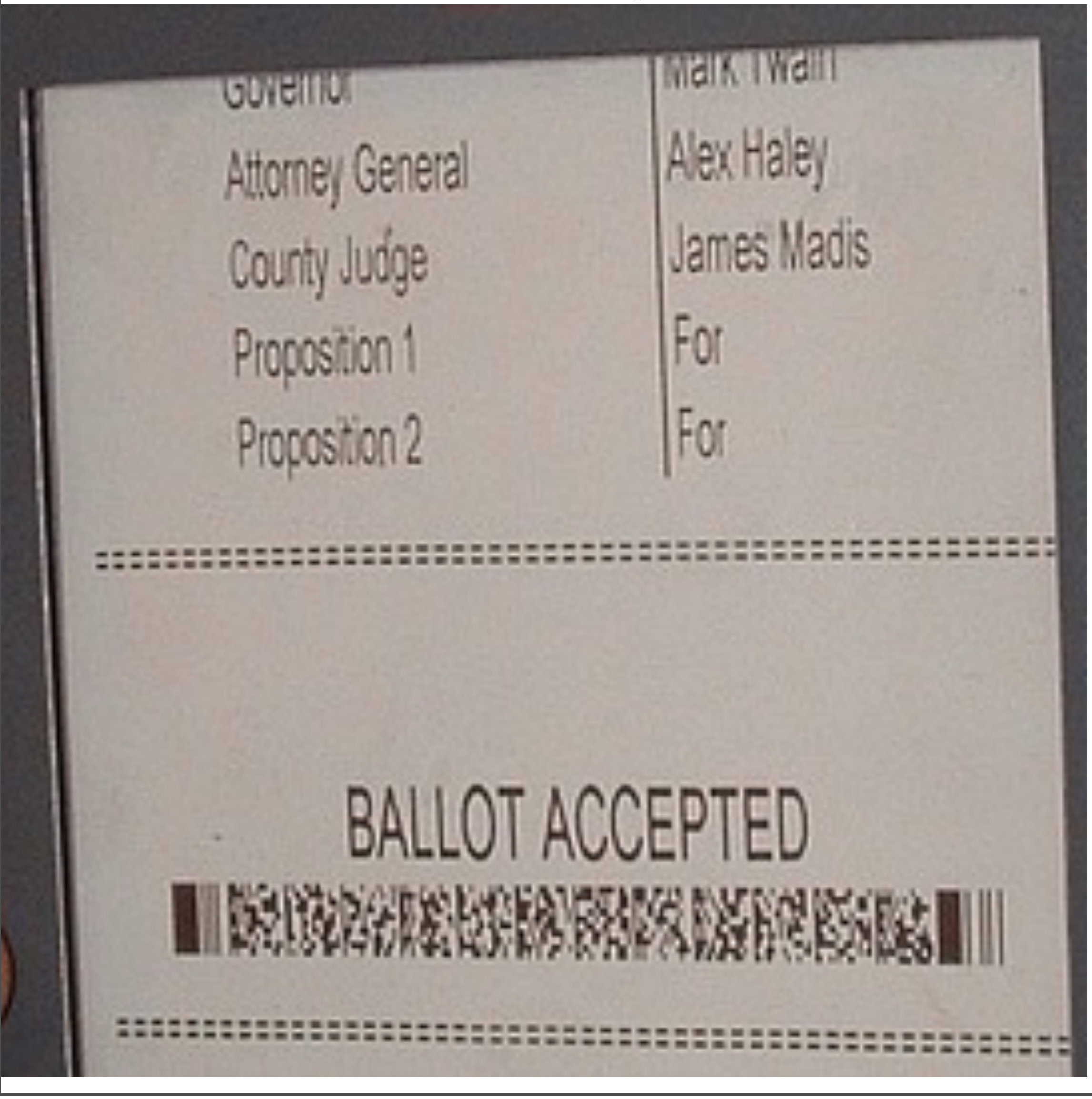
Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

BALLOT ACCEPTED

P: Human non-readable record
S: Add paper record

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.



Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

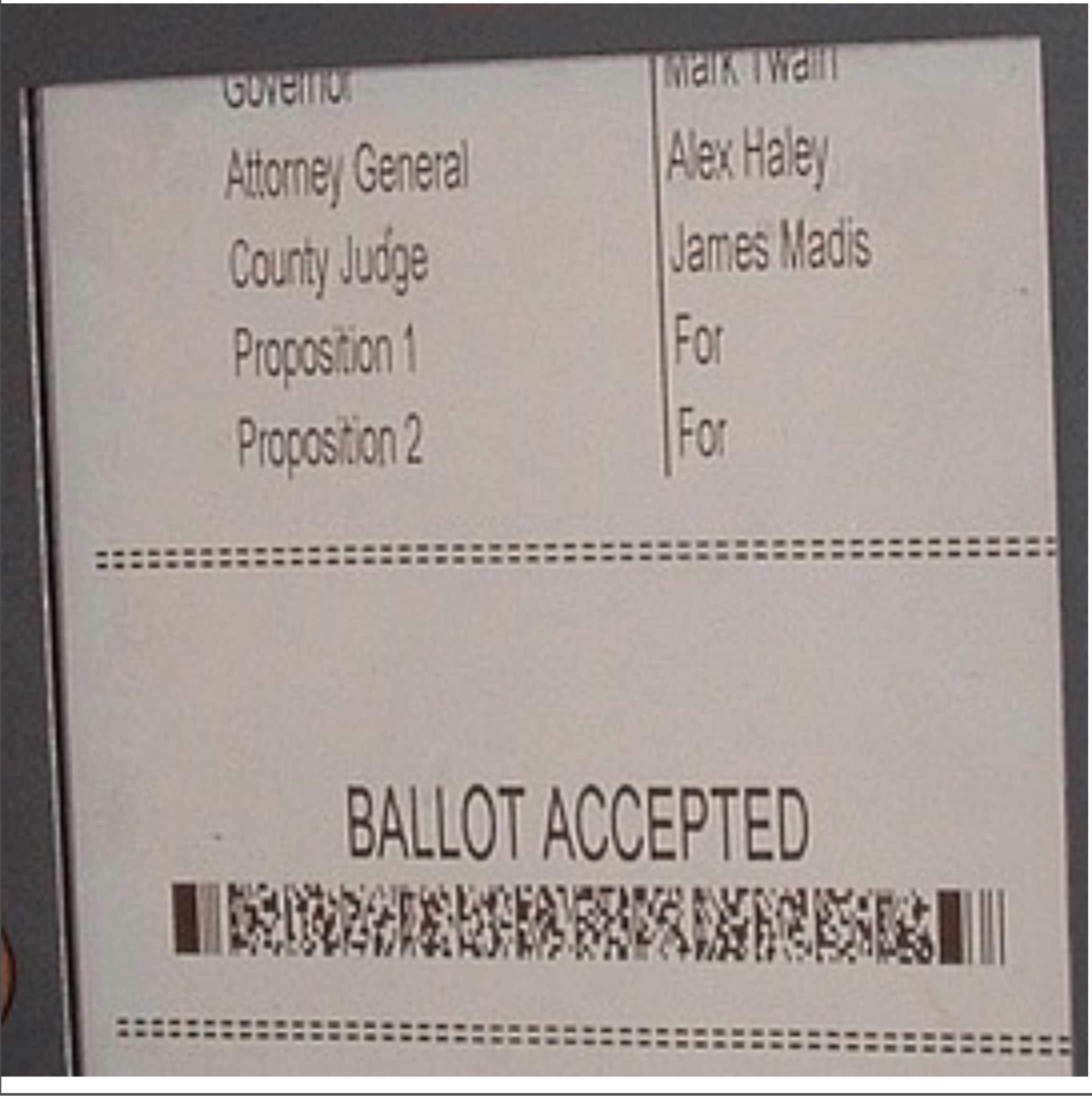
BALLOT ACCEPTED

P: Human non-readable record
S: Add paper record

P: Paper record slow to count

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.



Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

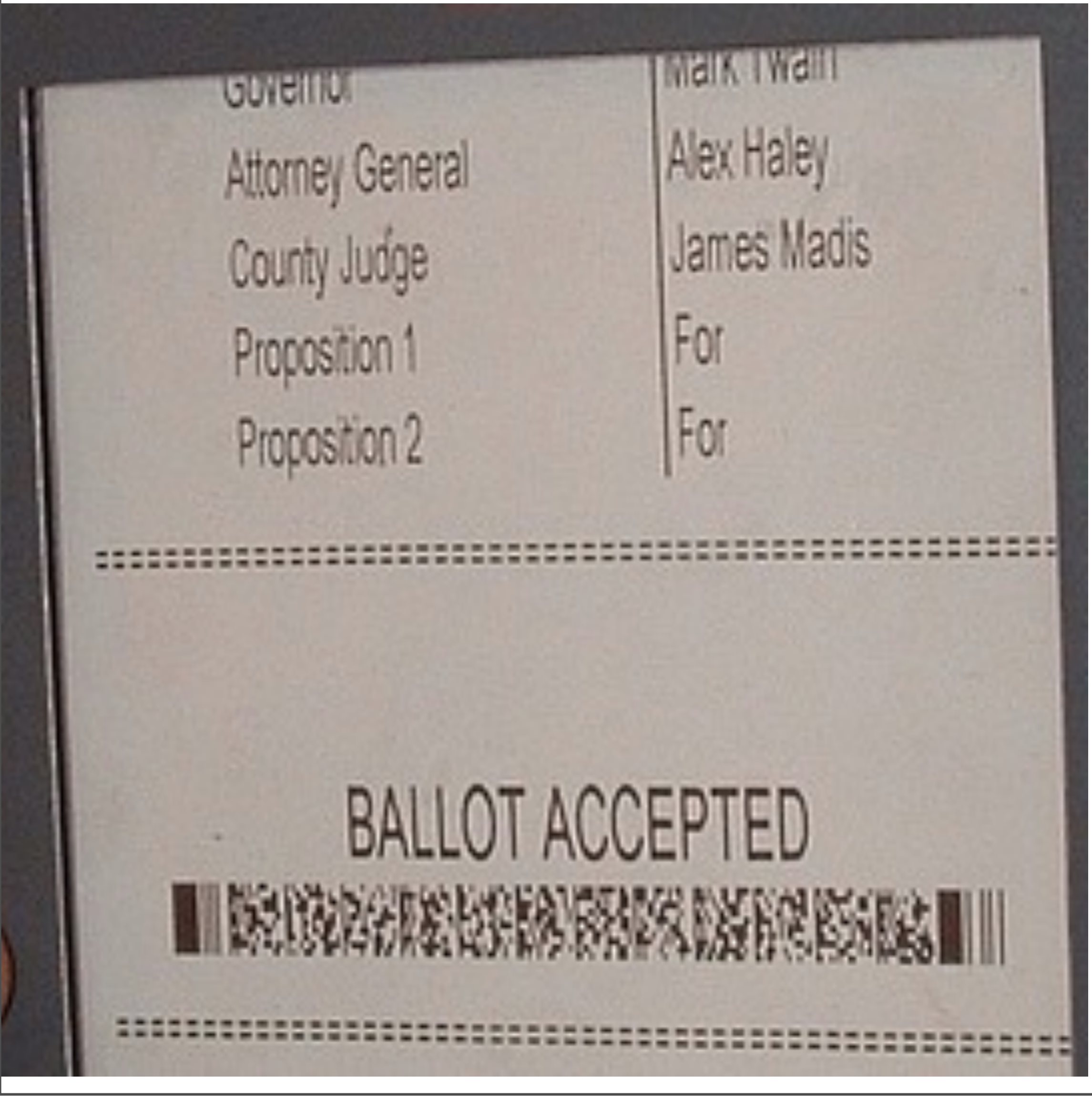
BALLOT ACCEPTED

P: Human non-readable record
S: Add paper record

P: Paper record slow to count
S: Add machine-readable record

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.



Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

BALLOT ACCEPTED

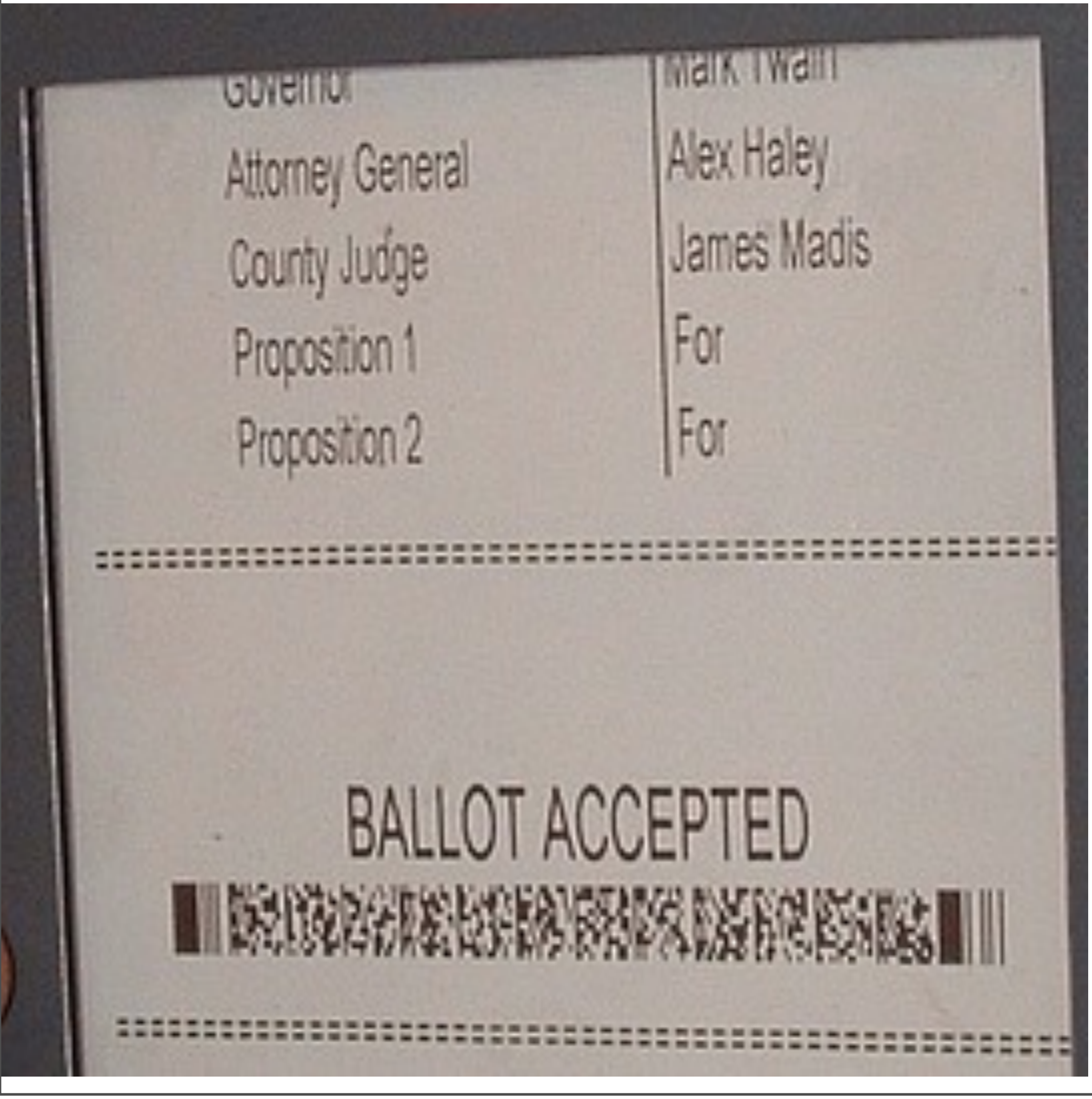
P: Human non-readable record
S: Add paper record

P: Paper record slow to count
S: Add machine-readable record

Q: How can we exploit this?

Security Mindset Failure 4

Problem: Verify votes recorded match votes cast.



Governor	Mark Twain
Attorney General	Alex Haley
County Judge	James Madis
Proposition 1	For
Proposition 2	For

BALLOT ACCEPTED

P: Human non-readable record
S: Add paper record

P: Paper record slow to count
S: Add machine-readable record

Q: How can we exploit this?

Q: What can be done?

Computer Voting

Hanging Chad

After the 2000 election, Congress allocated billions of dollars for computer voting machines (and other election-related expenses)

Computer Voting

Computer voting machines were broken again and again (here are just a few)

Study	Vendors	Year
Checkoway et al. (UCSD)	Sequoia	2009
Appel et al.	Sequoia	2008
EVEREST	ES&S, Hart, Premier	2007
California TTBR	Hart, Premier, Sequoia	2007
Feldman et al.	Diebold	2006
Hursti	Diebold	2006
Kohno et al. (UCSD)	Diebold	2003

Response

> *The proposed 'red team' concept also contemplates giving attackers access to source code, which is unrealistic and dangerous if not strictly controlled by test protocols. It is the considered opinion of election officials and information technology professionals that ANY system can be attacked if source code is made available. We urge the Secretary of State not to engage in any practice that will jeopardize the integrity of our voting systems.*
>
> – California Association of Clerks and Election Officials, 2007

Response

No computer system could pass the assault made by your team of computer scientists. In fact, I think my 9 and 12-year-old kids could find ways to break into the voting equipment if they had unfettered access.

– Santa Cruz County Clerk Gail Pellerin, 2007

By any standard – academic or common sense – the study is unrealistic and inaccurate.

– Diebold Election Systems, 2006

Company officials have said the researchers were given unusual access to the machines that real-world hackers could never gain.

– Mercury News, 2007

… is highly improbable in a real-world election.

– Hart InterCivic, 2007

Recap of Reasoning

Recap of Reasoning

- Florida debacle causes move to computers

Recap of Reasoning

- Florida debacle causes move to computers
- Computer scientists show computer voting machines are flawed

Recap of Reasoning

- Florida debacle causes move to computers
- Computer scientists show computer voting machines are flawed
- Response: Keep source code private and there are no problems

My Research

Is it practical to hack a voting machine without “unreasonable” access?

Hint: Yes

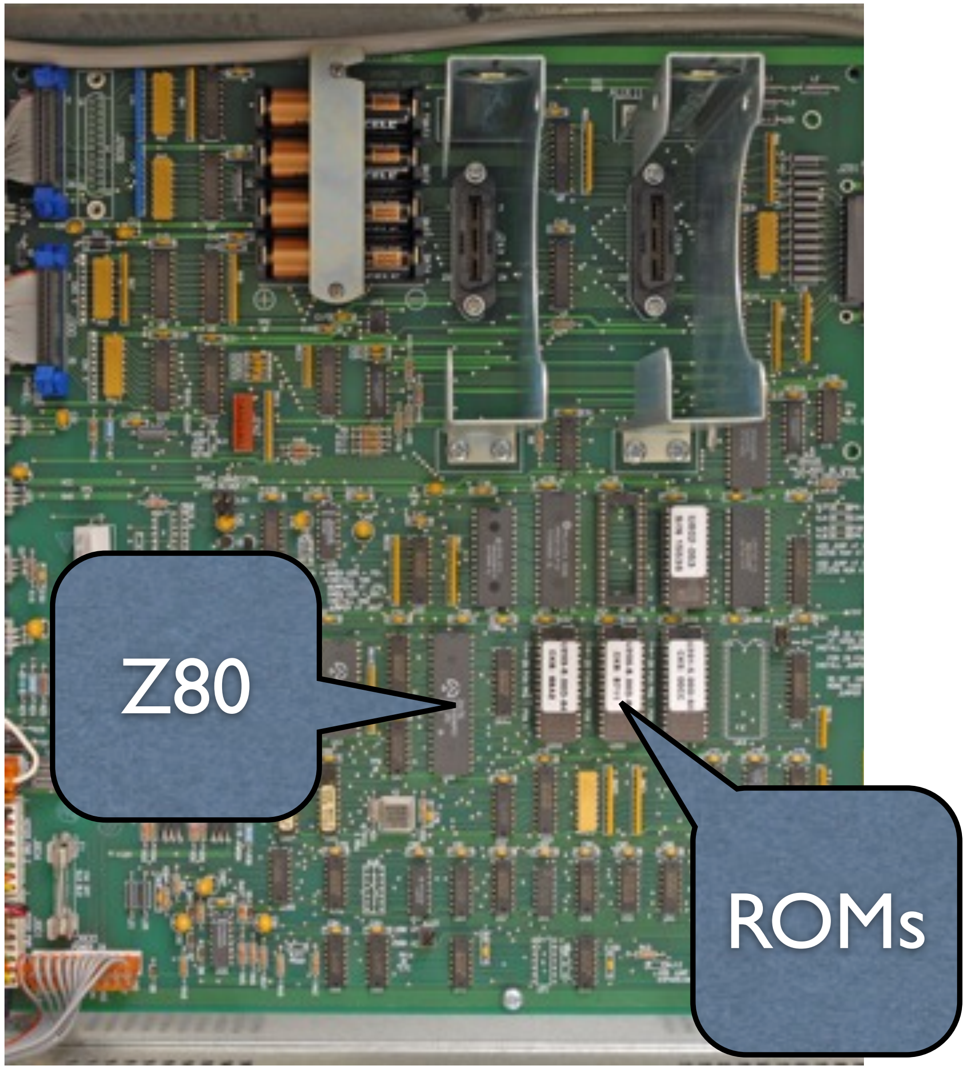
AVC Advantage

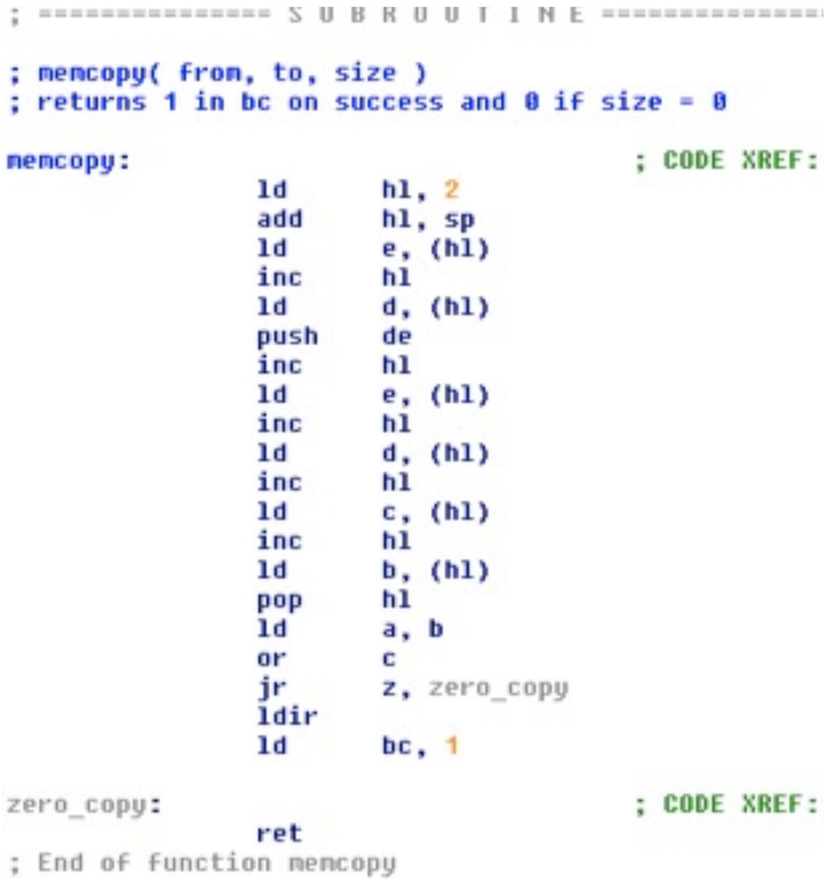
- Best-case to study
  - Only does one thing: count votes
  - Defenses against code injection
  - Also, we have access to one (in Princeton)!

Challenges

1. Understand how the machine works without source code or documentation by reverse-engineering
2. Find an exploitable bug
3. Defeat code-injection defense using recently developed techniques from system security

Reverse-Engineering





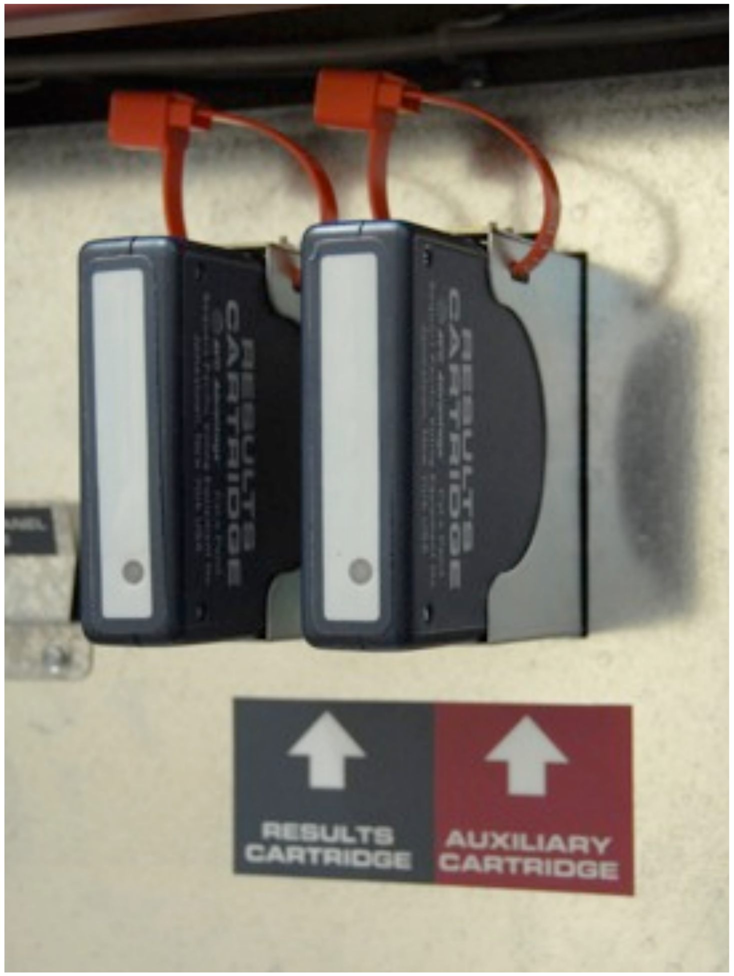
```
; =============== S U B R O U T I N E ===============

; memcopy( from, to, size )
; returns 1 in bc on success and 0 if size = 0

memcopy:                                        ; CODE XREF:
                ld      hl, 2
                add     hl, sp
                ld      e, (hl)
                inc     hl
                ld      d, (hl)
                push    de
                inc     hl
                ld      e, (hl)
                inc     hl
                ld      d, (hl)
                inc     hl
                ld      c, (hl)
                inc     hl
                ld      b, (hl)
                pop     hl
                ld      a, b
                or      c
                jr      z, zero_copy
                ldir
                ld      bc, 1

zero_copy:                                      ; CODE XREF:
                ret
; End of function memcopy
```

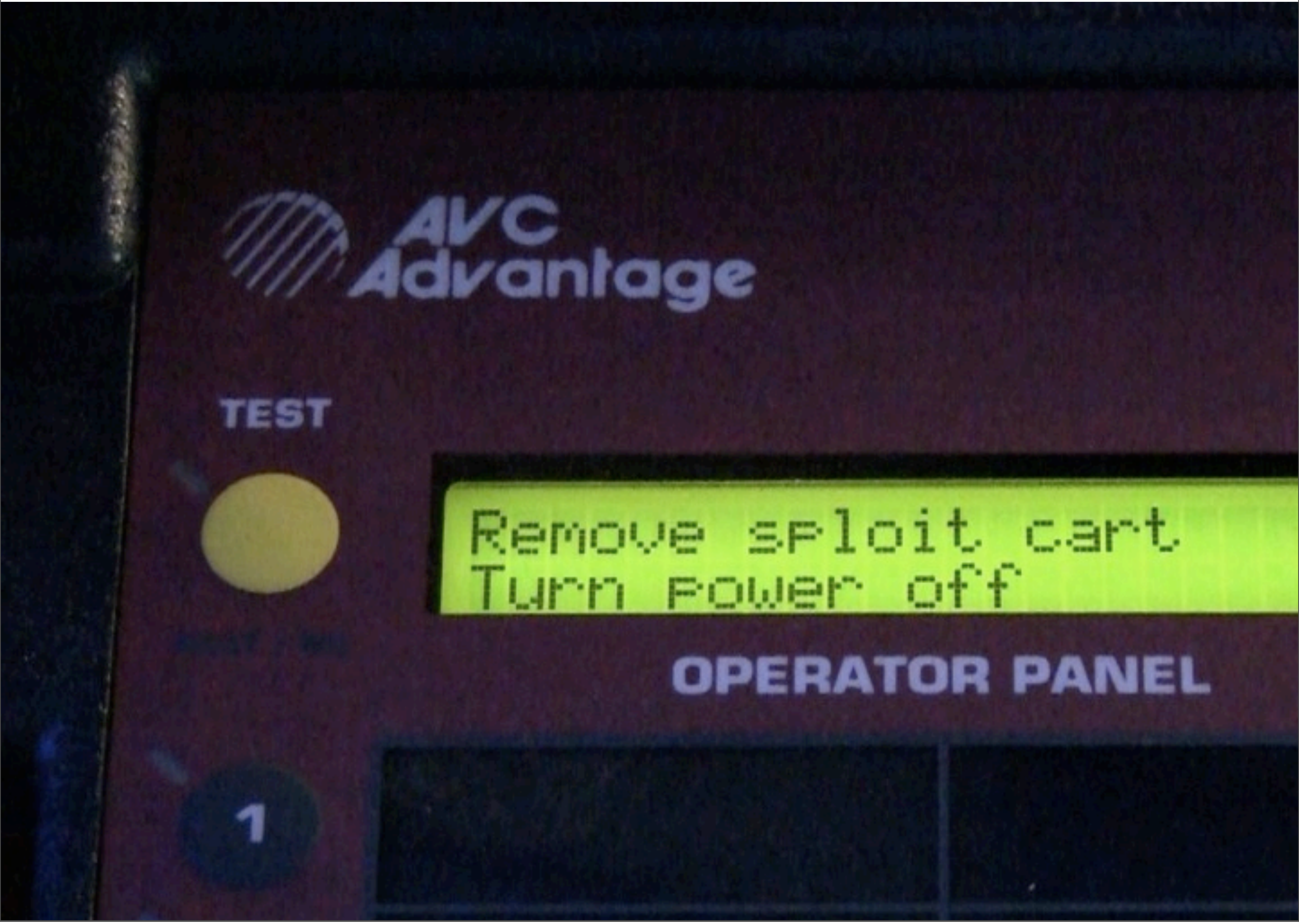
Exploit

- Classic stack-smashing buffer overflow
  - Roughly a dozen bytes overwritten
  - Exploit code needs to be in memory
- For now, assume we can inject code

Vote-Stealing Attack

- Gain physical access before election
- Malicious auxiliary cartridge contains new program
- Trigger exploitable bug
- Follow instructions

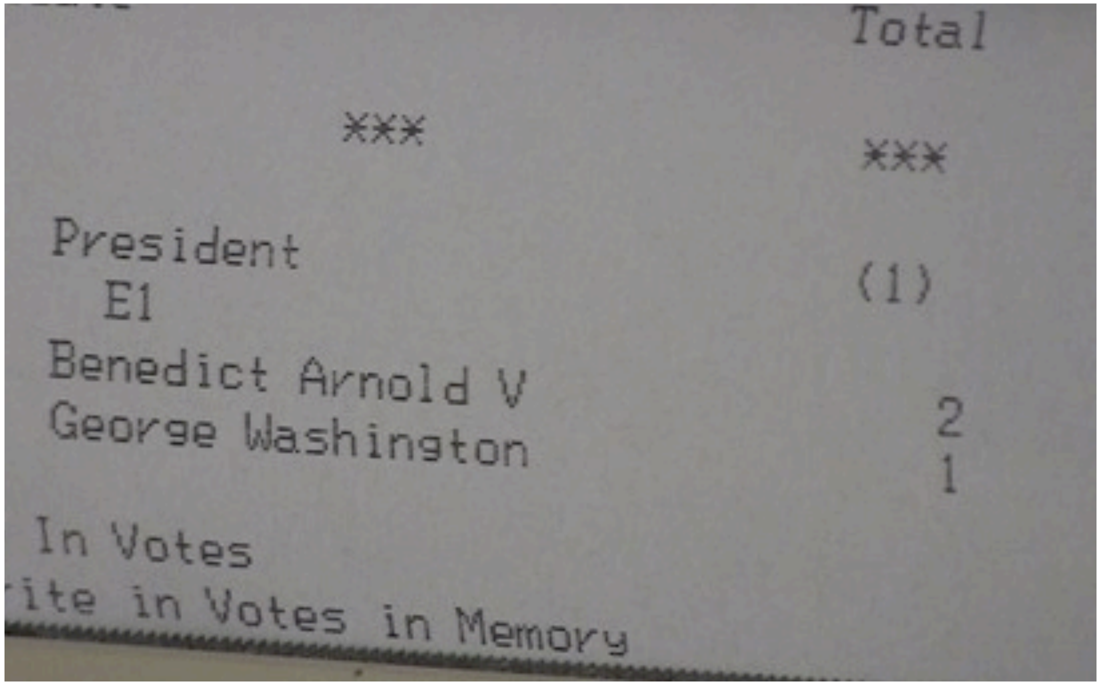




```
Remove sploit cart
Turn power off
```

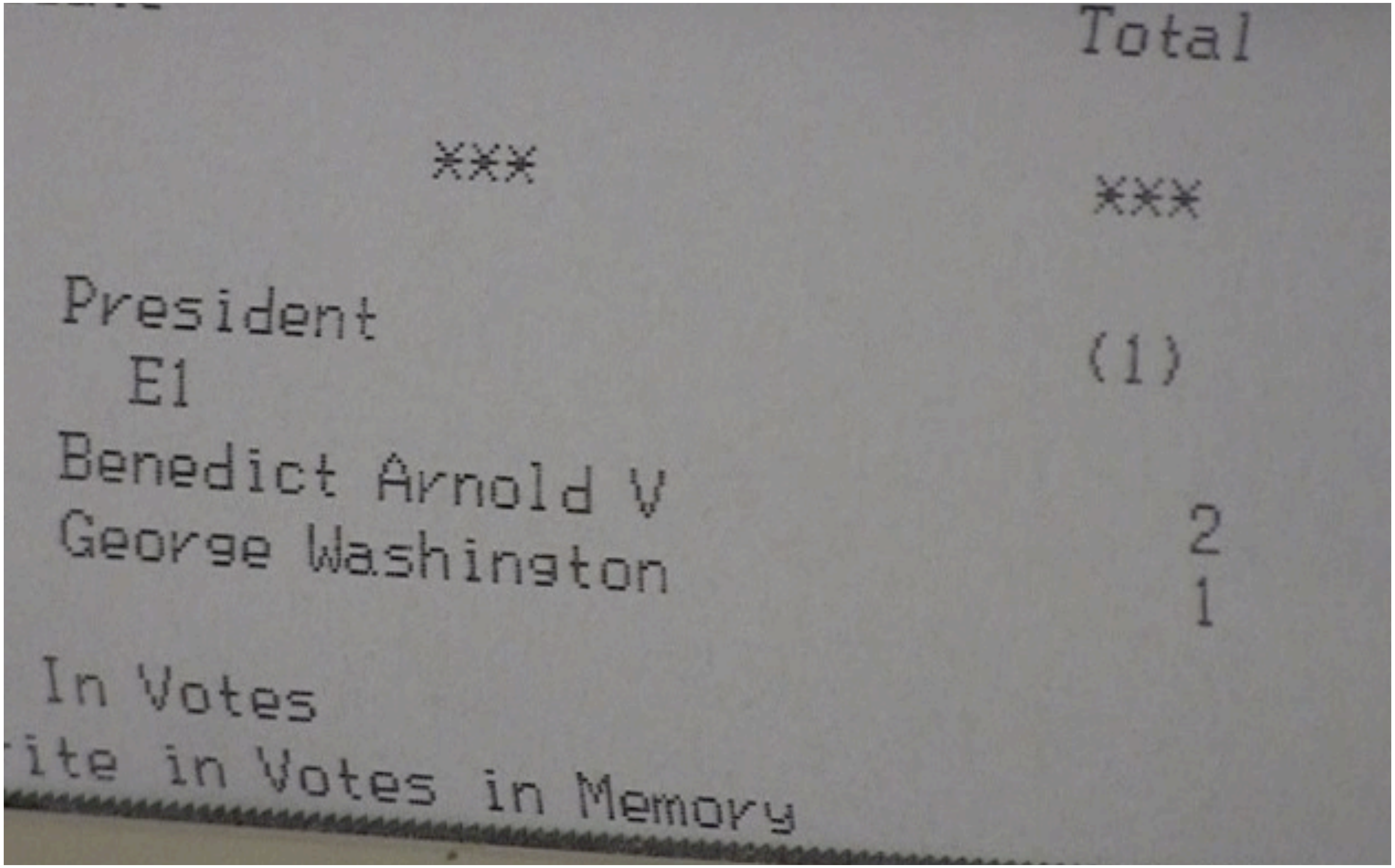
Vote-Stealing Program

- Election seems to run as normal
- Three votes cast for Washington
- Silently shifts votes



```
                                         Total

                  ***                     ***
President
  E1                                      (1)
Benedict Arnold V                           2
George Washington                           1

In Votes
ite in Votes in Memory
```

Vote-Stealing Program

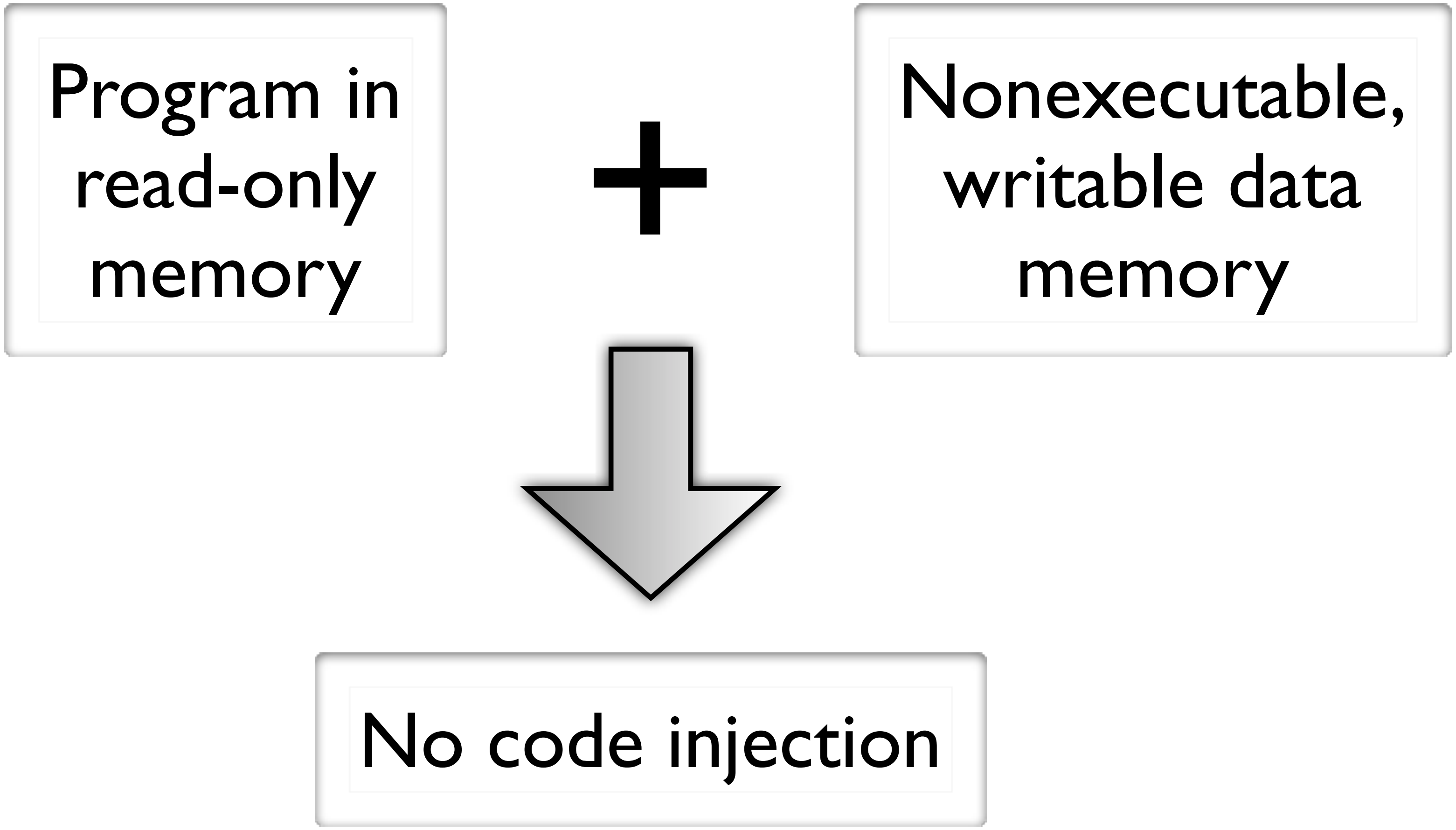


```
                                        Total

              ***
                                         ***

 President
                                         (1)
   E1
 Benedict Arnold V
 George Washington                          2
                                            1

In Votes
ite in Votes in Memory
```

Code Injection?

- Earlier, we assumed we could inject code
- Hardware interlock prevents fetching instructions from RAM
- Program code in read-only memory

Harvard Architecture



Program in read-only memory

+

Nonexecutable, writable data memory

No code injection

Solution: Return-oriented programming

Computer Voting

Voting procedures require testing

- Pre-election Logic and Accuracy Testing (LAT)
- Parallel testing
- Post-election LAT

How can these tests be overcome?

Computer Voting

Other security challenges

- Presentation attacks (miscalibrated touch screens)
- “Stuff” ballots with positive/negative votes
- Malicious program in memory cartridge
- Operating system compromise
- Attack central tabulator
- And many others

What Next?

- CSE 30 Computer Organization and Systems Programming – learn low-level programming
- CSE 141 Computer Architecture – learn how computer hardware really works
- CSE 127 Computer Security – learn the basics of using cryptography, malware (e.g., viruses), botnets, etc.
- CSE 227 Grad. Computer Security – learn about the latest developments in the field

Great Security Faculty at UCSD

Stefan Savage

Geoff Voelker

Hovav Shacham