CS 591: Introduction to Computer Security

Lecture 2: Voting Machine Study Access Control

James Hook

Objectives:

- Review/Discuss Analysis of Diebold machine
- Introduce Access Control

Discussion

- Thompson, Can You Coun on Voting Machines?, NY Times, January, 2008
  - Reaction?
  - Conflicts of interest in design and deployment?
  - Conflicts of interest in Testing? How independent?
  - Role of the vendor in operations?
  - How to prove the presence of a transient bug? The absence?

Discussion

- If we can make a good ATM why is it hard to make a good voting machine?
- “You have to be able to convince the loser they lost” … “Not only must the losing candidate believe in the loss; the public has to believe in it, too.”

Discussion

- Feldman, Halderman, and Felten, Security Analysis of the Diebold AccuVote-TS Voting Machine, September 2006
  - Reaction to the paper?

Discussion Questions

- What was the basic architecture of the voting machine?
- How did FHF steal votes?
- What other attacks did FHF consider?
- How did the viral propagation mechanism work?

Discussion Questions

- Is the analysis credible?
- Is the threat model credible?
- Is this representative of commercial systems today?
- Did Diebold follow best practices?
- Are the FHF results reproducible?
- Did Felton's lab follow a sound methodology in analyzing the machine?

Discussion Questions

- Having read the analysis of the Diebold machine, are you surprised that Sequoia used a threat of law suit to prevent Felten's lab from analyzing their machine?
- Having seen this analysis of a fielded commercial system, are you more or less concerned about the discrepencies observed in Union County elections?

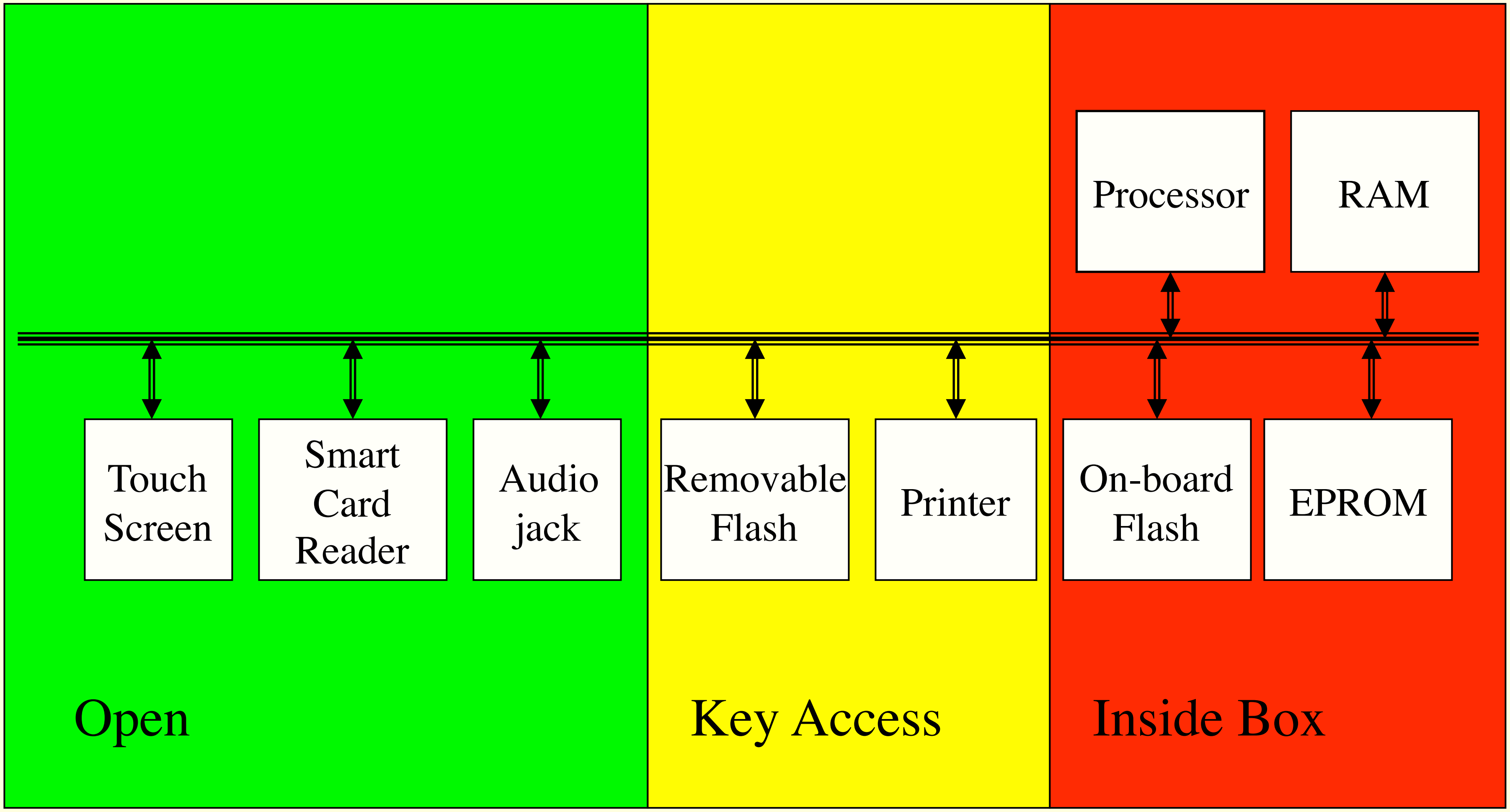
Discussion Questions

- Do you like Oregon’s vote by mail system?
- Are Appel’s comments on Minnesota relevant to voting in Oregon?

Case Study

- We will use the FHF paper as a case study
- As we encounter concepts we will attempt to instantiate them in the context of the voting machine domain

Voting Machine Architecture



3/31/09 08:29

Boot Process

- Boot device specified by hardware jumpers (inside box)
  - EPROM
  - on-board flash (default)
  - ext flash
- On Boot:
  - Copy bootloader into RAM; init hardware
  - Scan Removable flash for special files
    - "fboot.nb0" => replace bootloader in on-board flash
    - "nk.bin" => replace OS in on-board flash
    - "EraseFFX.bsq" => erase file system on on-board flash
  - If no special files uncompress OS image
  - Jump to entry point of OS

Boot (continued)

- On OS start up:
  - run Filesys.exe
    - unpacks registry
    - runs programs in HKEY_LOCAL_MACHINE\Init
      - shell.exe (debug shell)
      - device.exe (Device manager)
      - gwes.exe (graphics and event)
      - taskman.exe (Task Manager)
  - Device.exe mounts file systems
    - \ (root): RAM only
    - \FFX: mount point for on-board flash
    - \Storage Card: mount point for removable flash

Boot (continued)

- Customized taskman.exe
  - Check removable flash
    - explorer.glb => launch windows explorer
    - *.ins => run proprietary scripts
      - (script language has buffer overflow vulnerabilities)
      - used to configure election data
    - default => launch “BallotStation”
      - \FFX\Bin\BallotStation.exe

BallotStation

- Four modes: pre-download, pre-election testing, election, post-election
- Mode recorded in election results file
  - \Storage Card\CurrentElection\election.brs

Stealing Votes

- Malicious processes runs in parallel with BallotStation
- Polls election results file every 15 seconds
  - If election mode and new results
    - temporarily suspend Ballot Station
    - steal votes
    - resume Ballot Station

Viral propagation

- Malicious bootloader
  - Infects host by replacing existing bootloader in on-board flash
  - subsequent bootloader updates print appropriate messages but do nothing
- fboot.nb0
  - package contains malicious boot loader
  - and vote stealing software

Access Control Model

Objectives

- Introduce the concept of Access Control
- Relate mechanism to Confidentiality, Integrity and Availability

Articulating Policy

- How do we articulate a security policy?
- How do we provide mechanisms to enforce policy?
- Voting
  - Different individuals in different roles
    - Voter, Poll worker, …
  - Different actions
    - Vote, define ballot, start and stop election, …
  - Logical and physical entities
    - Ballot, stored tally, final tally, voting machine, removable flash, on-board flash, …

Ad hoc policies

- Discus
  - Only voters should vote
  - Only poll workers should start and start elections

Access Control Matrix Model

- Lampson ‘71, refined by Graham and Denning (‘71, ‘72)
- Concepts
  - **Objects**, the protected entities, O
  - **Subjects**, the active entities acting on the objects, S
  - **Rights**, the controlled operations subjects can perform on objects, R

  - **Access Control Matrix**, A, maps Objects and Subjects to sets of Rights
  - State: (S, O, A)

Voting: Subjects, Objects, Rights

- Subjects: (Roles)
  - Voter, Poll worker, …
- Rights: (Actions)
  - Vote, define ballot, start and stop election, …
- Objects: (Logical and physical entities)
  - Ballot, stored tally, final tally, voting machine, removable flash, on-board flash, …

- Question: Is every voter a subject? Or is the role of voter a subject? One-person-one-vote?

Exercise

- Sketch Access Control Matrix (ACM) for Voting

	Ballot	Stored Tally	Final Tally	…
Voter	read	increment		
Poll Worker			print	
…				

Questions

- What about modes?
  - Once the election starts the ballot should not change
  - Voters should only vote when the election is happening

Questions

- Levels of abstraction
  - Some objects are physical, some are logical
  - When considering the programming model you now have processes and files (and possibly modes of operation)
- Exercise:
  - Sketch ACMs with processes as subjects and files as objects for voting and post-election modes

Exercise

- Compare the ACMs for files and processes with the original ACM
- Is every operation specified in the original feasible in the refined ACMs?
- Is every feasible operation in the refined ACMs allowed in the original?

Mechanisms

- Policy specifies abstract goals
- Mechanisms are concrete devices, algorithms, or processes that assist in implementing a policy
- For example, passwords are a mechanism that can support an authentication policy
  - Mechanisms are not always perfect!

Access Control Mechanisms

- Most operating systems provide some mechanisms for supporting access control
- Typically:
  - Processes are associated with users (or user identification numbers), which are the subjects
  - Files are objects
  - Rights are: read, write, append, execute, search, ...

Applying the Mechanism

- Can a generic Access Control mechanism help make the Voting machine more trustworthy?
- What about modes?
  - Mode is not part of typical AC mechanisms
  - However rights can be changed
    - A typical right is “own” which in discretionary access control generally allows the subject to change rights
  - Analysis of systems that actively change rights is potentially difficult

Limitations on Mechanisms

- Simple mechanisms are preferred
- All computational mechanisms must be decidable
- In general, useful mechanisms must be computationally cheap

Access Control

- Is Access Control biased to
  - Confidentiality
  - Integrity
  - Availability
- Exercise
  - Develop scenarios in which a confidentiality (integrity, availability) property is expressed using an access control matrix

Model vs. Mechanism

- Earlier I presented the model of the AC Matrix
- Does UNIX implement the full AC Matrix?
  - What key simplifications does UNIX adopt?
  - Why?
- Is the full ACM mechanism a good idea?
  - Is it a good model?

A Good Model

- ACM is a good model because any mechanism of compatible granularity can be described in terms of how it approximates the ACM model

Next Lecture

- Discussion
  - Tibet
    - NY Times article http://www.nytimes.com/2009/03/29/technology/29spy.html?emc=eta1
    - Nagaraja and Anderson tech report http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-746.html