CSE 484 / CSE M 584: Computer Security and Privacy

Winter 2022

Tadayoshi (Yoshi) Kohno
yoshi@cs

UW Instruction Team: David Kohlbrenner, Yoshi Kohno, Franziska Roesner. Thanks to Dan Boneh, Dieter Gollmann, Dan Halperin, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

Announcements

- Things Due:
  - Ethics Form: Friday
  - Homework #1: Due next Thursday
  - Research Readings (CSE M 584): Due next Thursday (and every Thursday thereafter)
- Discussion Board:
  - Ideally set up soon

Final Project

- **No midterm or final exam!**
- Instead: **12-15 min video** about a security/privacy topic of your choice
  - Groups of up to 3 people (groups strongly encouraged)
  - Security is a broad field, and this class can't remotely cover everything – this is your chance to explore a security or privacy topic in more detail!
  - Multiple checkpoint deadlines throughout quarter
- Details linked from website's Assignments page

Prerequisites (CSE 484)

- Required: Data Abstractions (CSE 332)
- Required: Hardware/Software Interface (CSE 351)
- Assume: Working knowledge of C and assembly
  - One of the labs will involve writing buffer overflow attacks in C
  - You must have detailed understanding of x86 architecture, stack layout, calling conventions, etc.
- Assume: Working knowledge of software engineering tools for Unix environments (gdb, etc)
- Assume: Working knowledge of Java and JavaScript
- **Assume: Ability to learn new programming languages / skills easily**

Prerequisites (CSE 484)

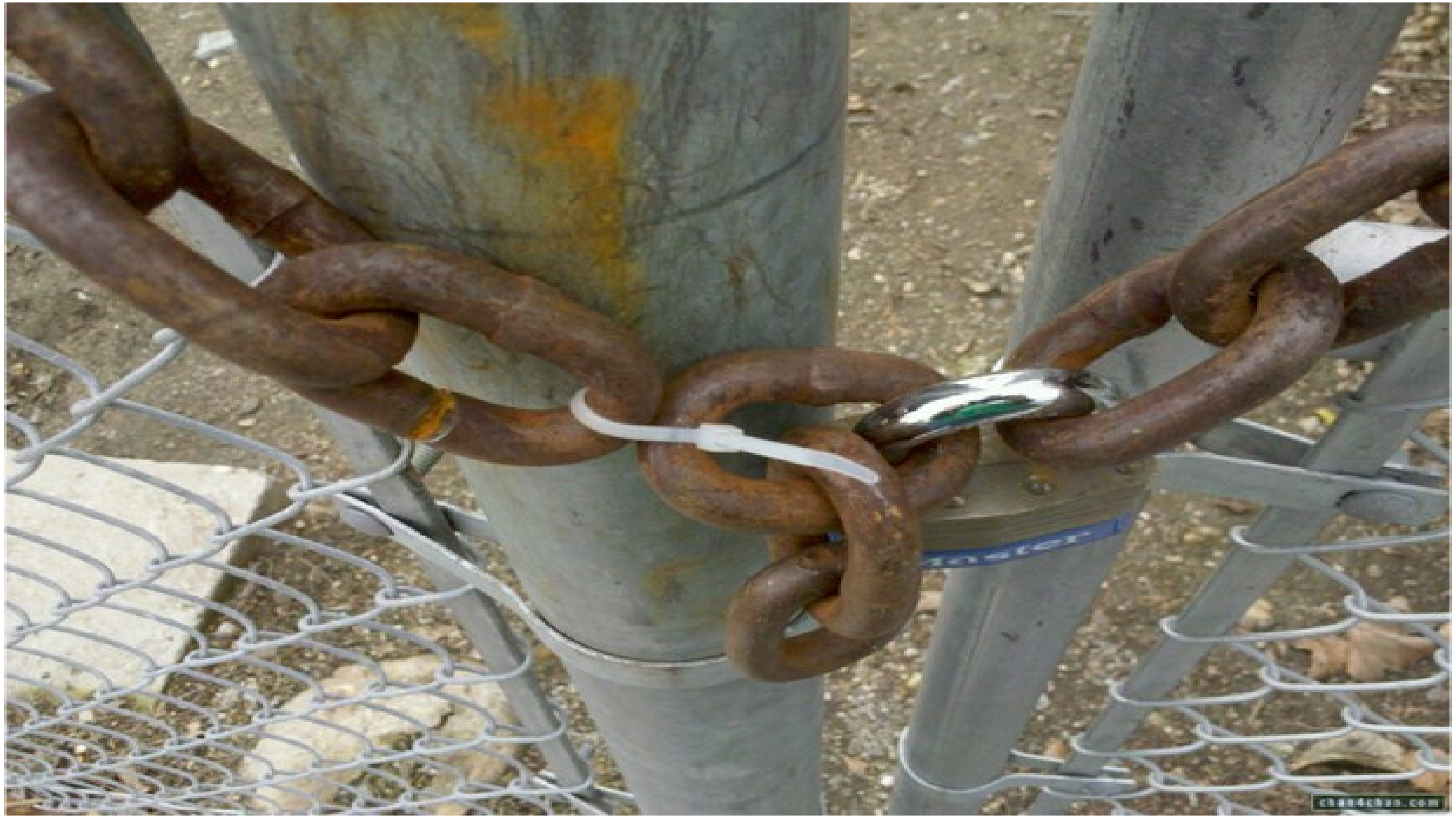
- Useful (not required): Computer Networks; Operating Systems
  - Will help provide deeper understanding of security mechanisms and where they fit in the big picture
- Useful (not required): Complexity Theory; Discrete Math; Algorithms
  - Will help with the more theoretical aspects of this course.

Prerequisites (CSE 484)

- Most of all: Eagerness to learn!
  - This is a 400 level course.
  - We expect you to push yourself to learn as much as possible.
  - We expect you to be a strong, independent learner capable of learning new concepts from the lectures, the readings, and on your own.

  - **Of course, this quarter is different than usual. Take care of yourselves and communicate with us!**

Another Example



THREAT MODELING

Threat Modeling (Security Reviews)

- Assets: What are we trying to protect? How valuable are those assets?
- Adversaries: Who might try to attack, and why?
- Vulnerabilities: How might the system be weak?
- Threats: What actions might an adversary take to exploit vulnerabilities?
- Risk: How important are assets? How likely is exploit?
- Possible Defenses
- Not "traditional" threat modeling, but important (both in general, and to help better understand the system prior to threat modeling):
  - Benefits: Who might the system benefit, and how?
  - Harms: Who might the system harm, and how?

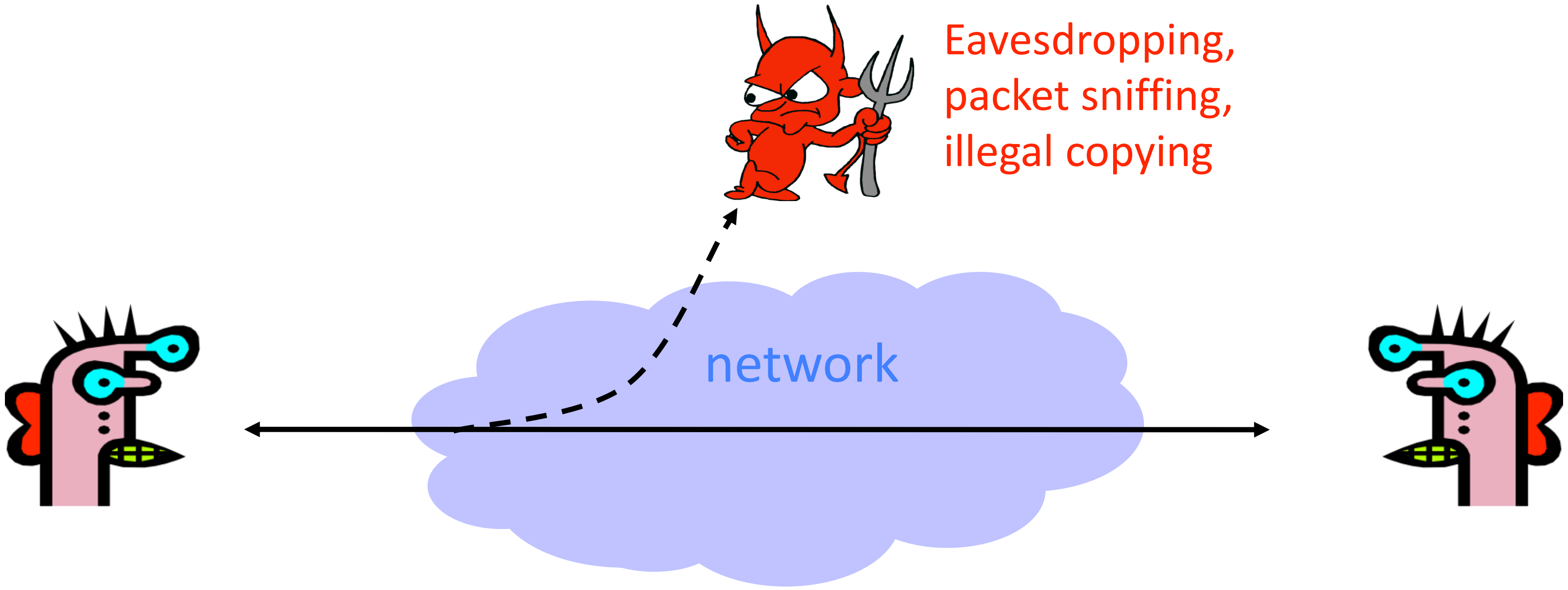
What’s *Security*, Anyway?

- Common general security goals: “CIA”
  - Confidentiality
  - Integrity
  - Availability

- Or the extension: CPIAAU (Parkerian Hexad)
  - Confidentiality
  - Possession or Control
  - Integrity
  - Authenticity
  - Availability
  - Utility

Confidentiality (Privacy)

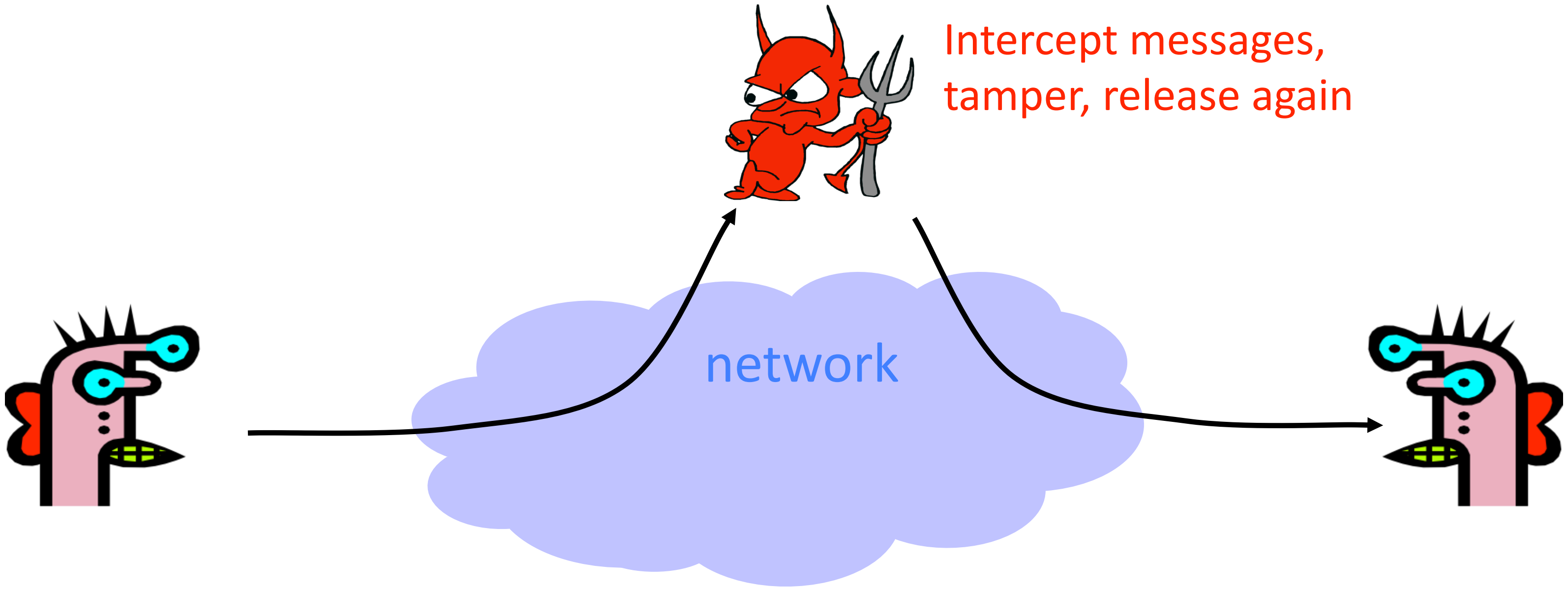
- Confidentiality is concealment of information.



Eavesdropping, packet sniffing, illegal copying

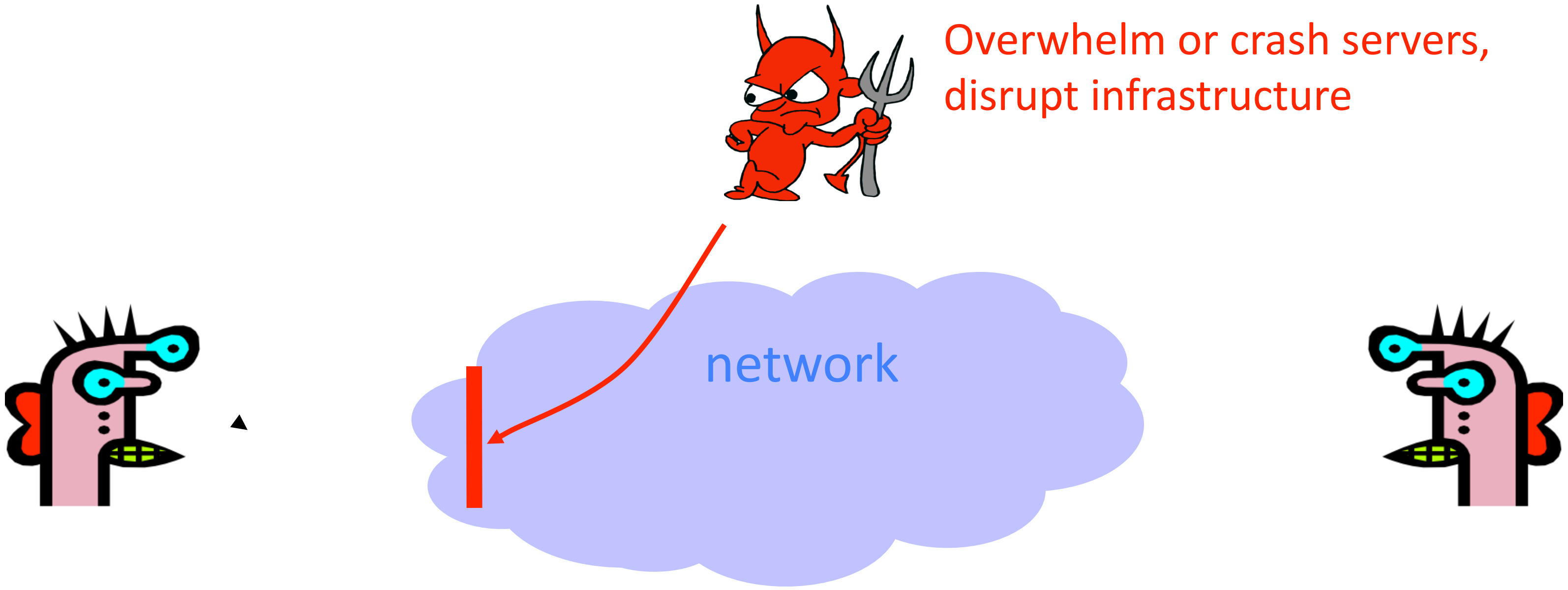
Integrity

- Integrity is prevention of unauthorized changes.



Availability

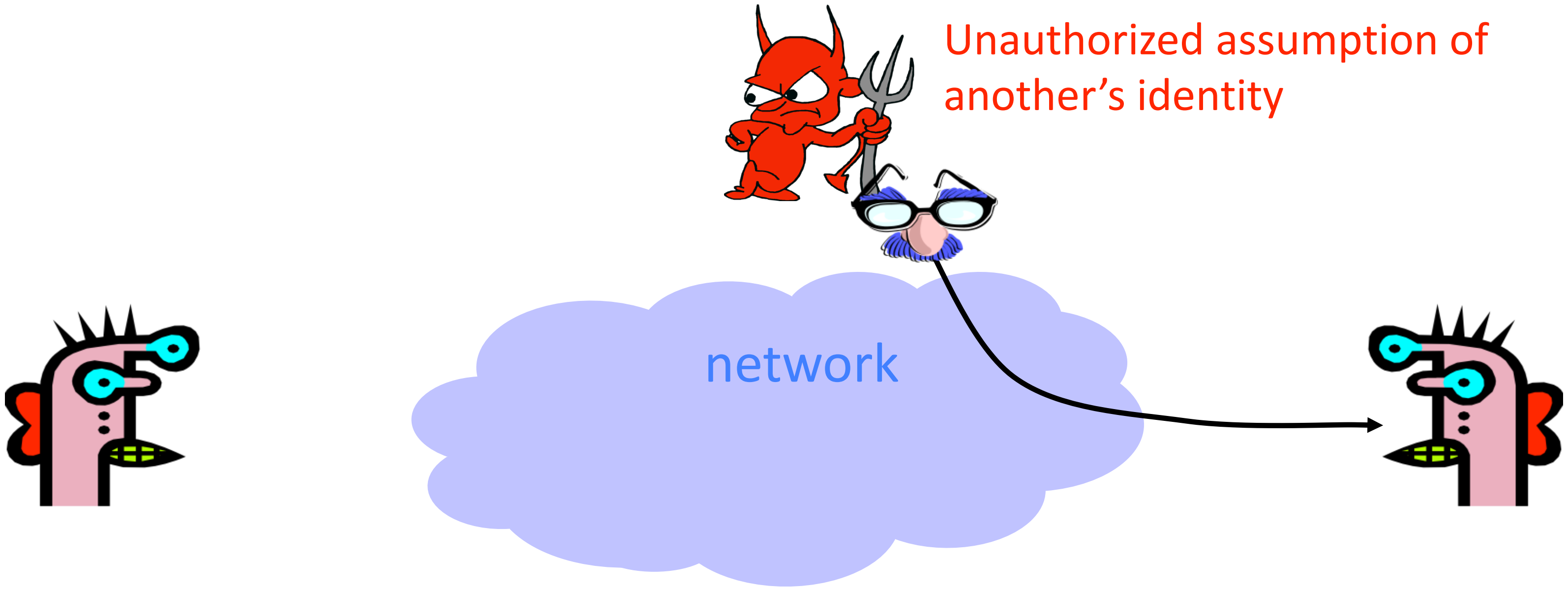
- Availability is ability to use information or resources.



Overwhelm or crash servers, disrupt infrastructure

Authenticity

- Authenticity is knowing who you're talking to.



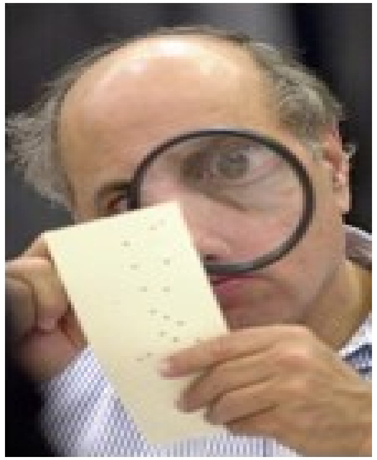
Unauthorized assumption of another's identity

Threat Modeling

- There’s no such thing as perfect security
  - But, attackers have limited resources
  - **Make them pay unacceptable costs / take on unacceptable risks to succeed!**
- Defining security per context: identify assets, adversaries, motivations, threats, vulnerabilities, risk, possible defenses

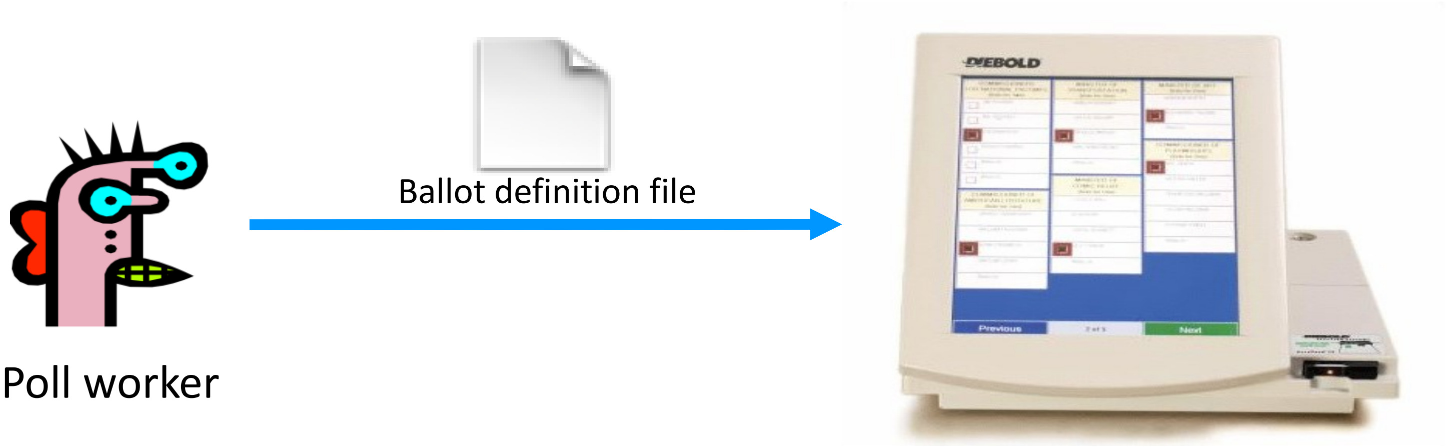
Threat Modeling Example: Electronic Voting

- Popular replacement to traditional paper ballots



AP

Pre-Election

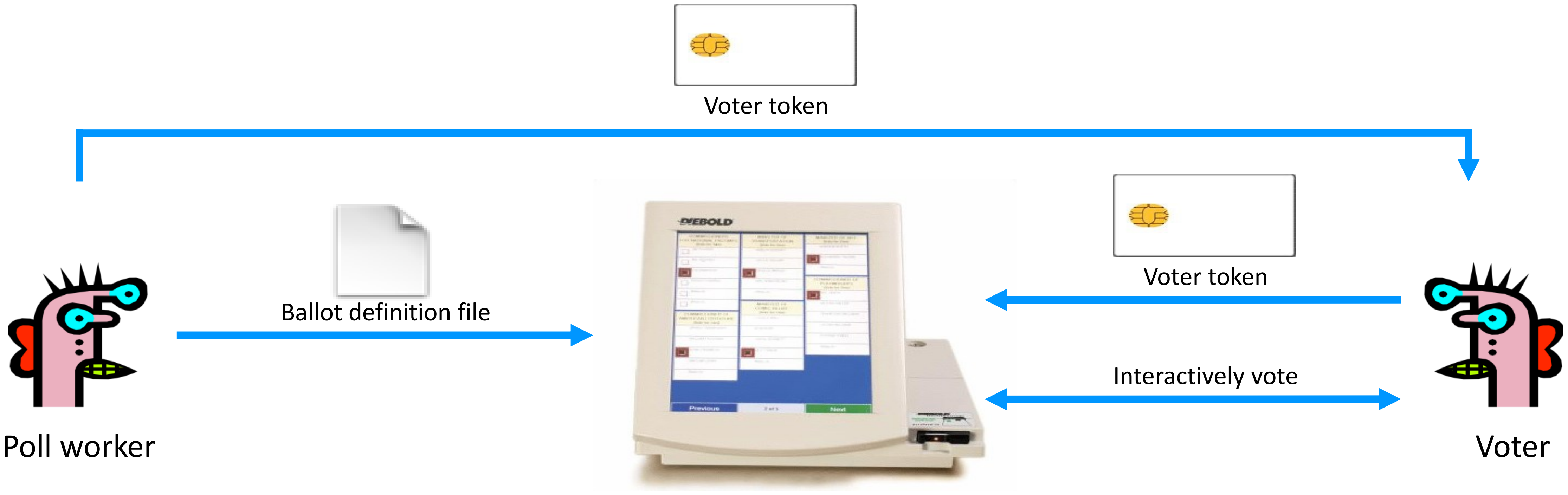


Ballot definition file

Poll worker

Pre-election: Poll workers load “ballot definition files” on voting machine.

Active Voting



Voter token

Ballot definition file

Voter token

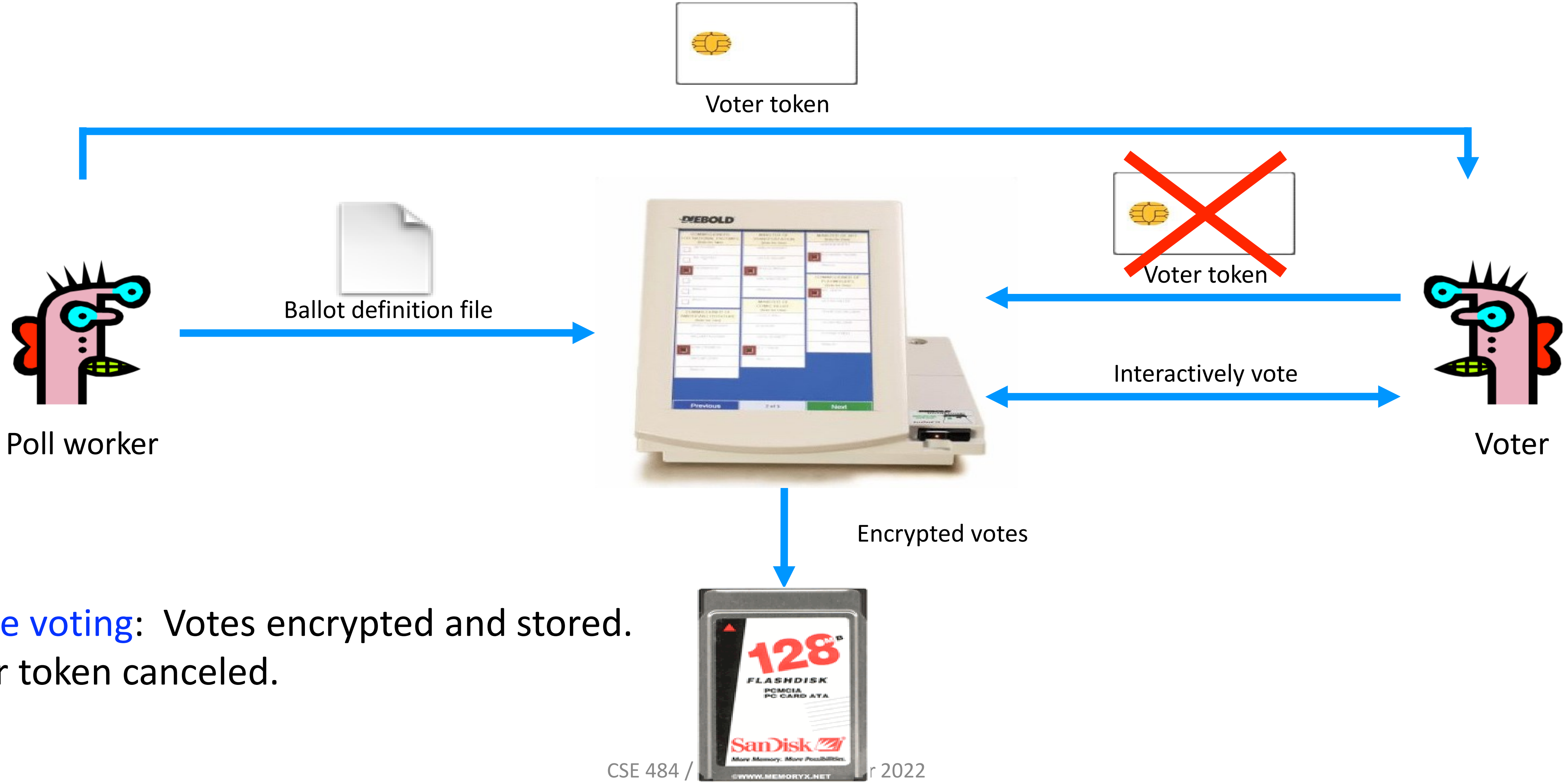
Interactively vote

Poll worker

Voter

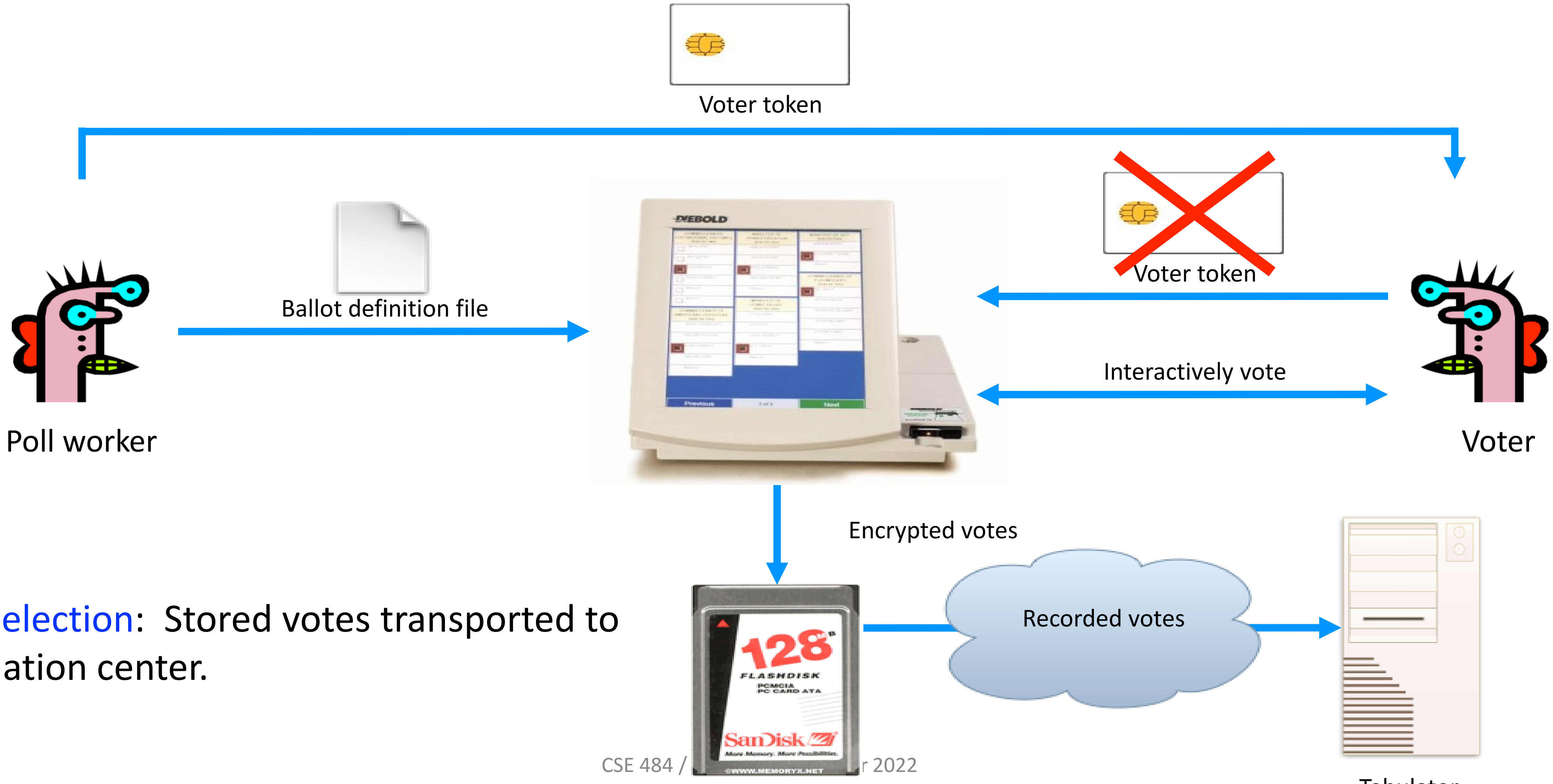
Active voting: Voters obtain single-use tokens from poll workers. Voters use tokens to activate machines and vote.

Active Voting



Active voting: Votes encrypted and stored. Voter token canceled.

Post-Election



Post-election: Stored votes transported to tabulation center.

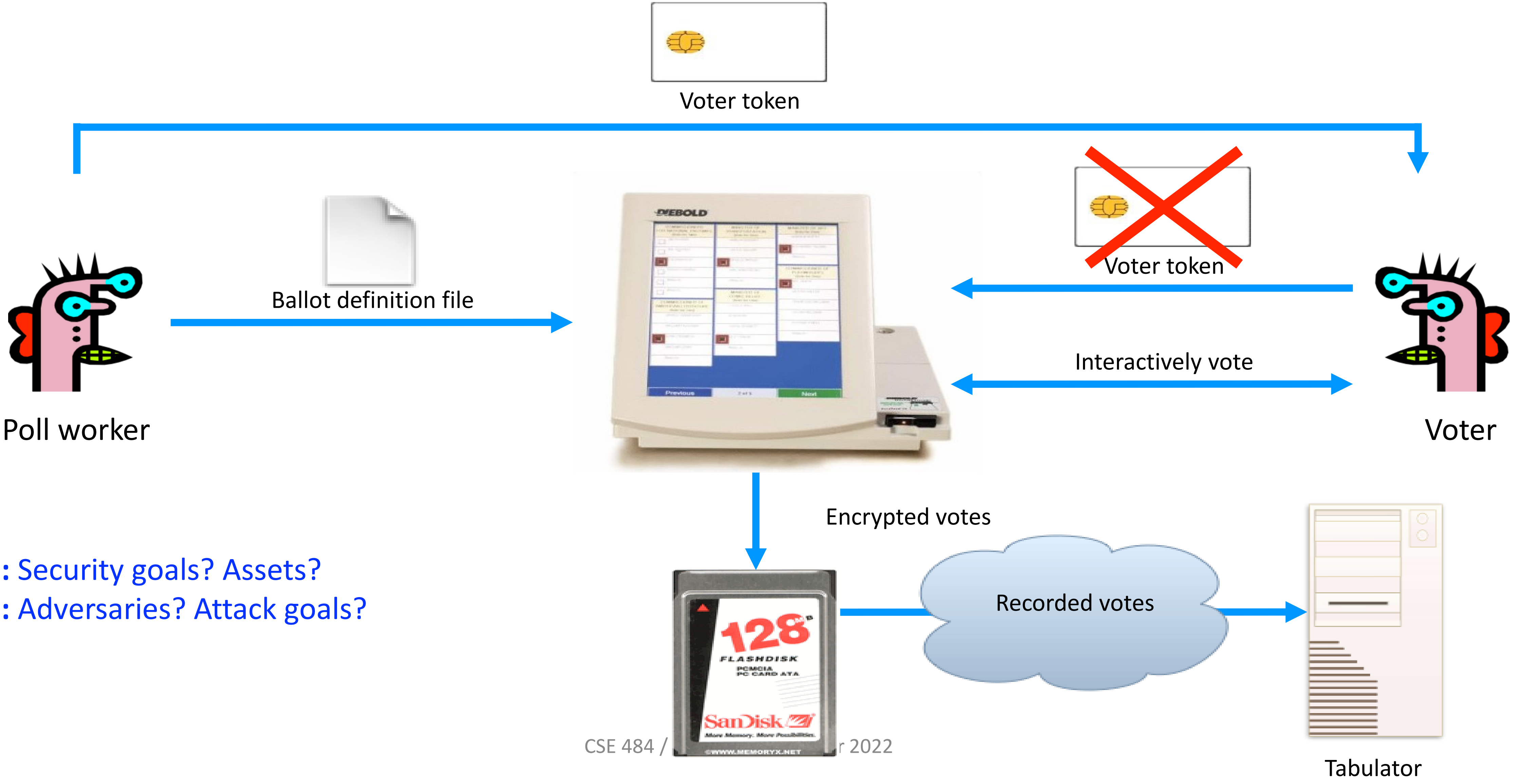
In-Class “Worksheet”

- Go to Canvas -> Quizzes -> “In-Class Activity – January 5”
- Fill out the questions while discussing with your breakout group
  - Everyone should submit their own
  - **No need for polish or complete sentences** – jot things down as you would on a piece of paper while chatting in class
- Q1: What do you think are the **security goals** of the electronic voting system described in class and shown above? What would be some of the **assets** that must be protected?
- Q2: Who are the **adversaries** who might try to attack this electronic voting system? What might be the **attacker’s goals**? What potential **threats or vulnerabilities** do you see?

Security and E-Voting (Simplified)

- Functionality goals:
  - Easy to use, reduce mistakes/confusion, make voting more accessible
- Security goals:

Can You Spot Any Potential Issues?



Q1: Security goals? Assets?
Q2: Adversaries? Attack goals?

What Software is Running?

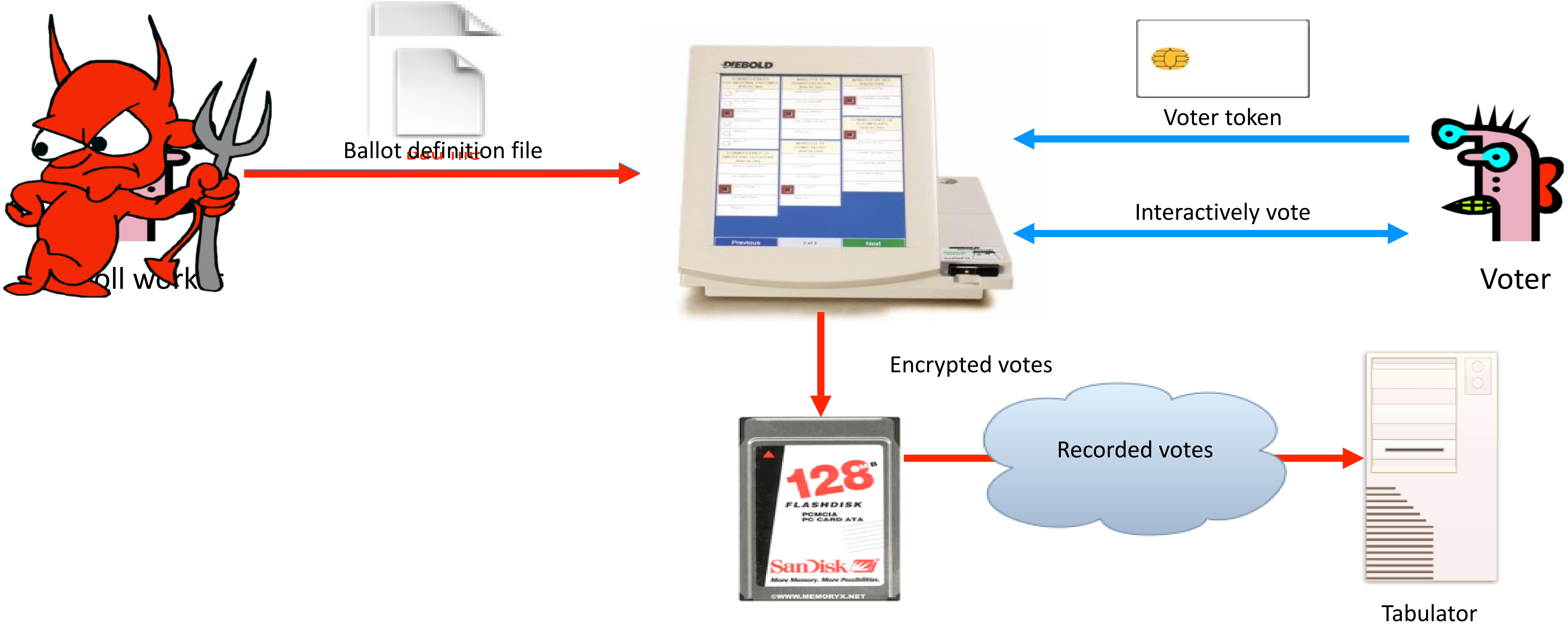
Problem: An adversary (e.g., a poll worker, software developer, or company representative) able to control the software or the underlying hardware could do whatever they wanted.

KEYS TO THE KINGDOM

Photo taken from Diebold's online store. The keys that open every Diebold touch-screen voting machine. Working copies have been made from the photo.

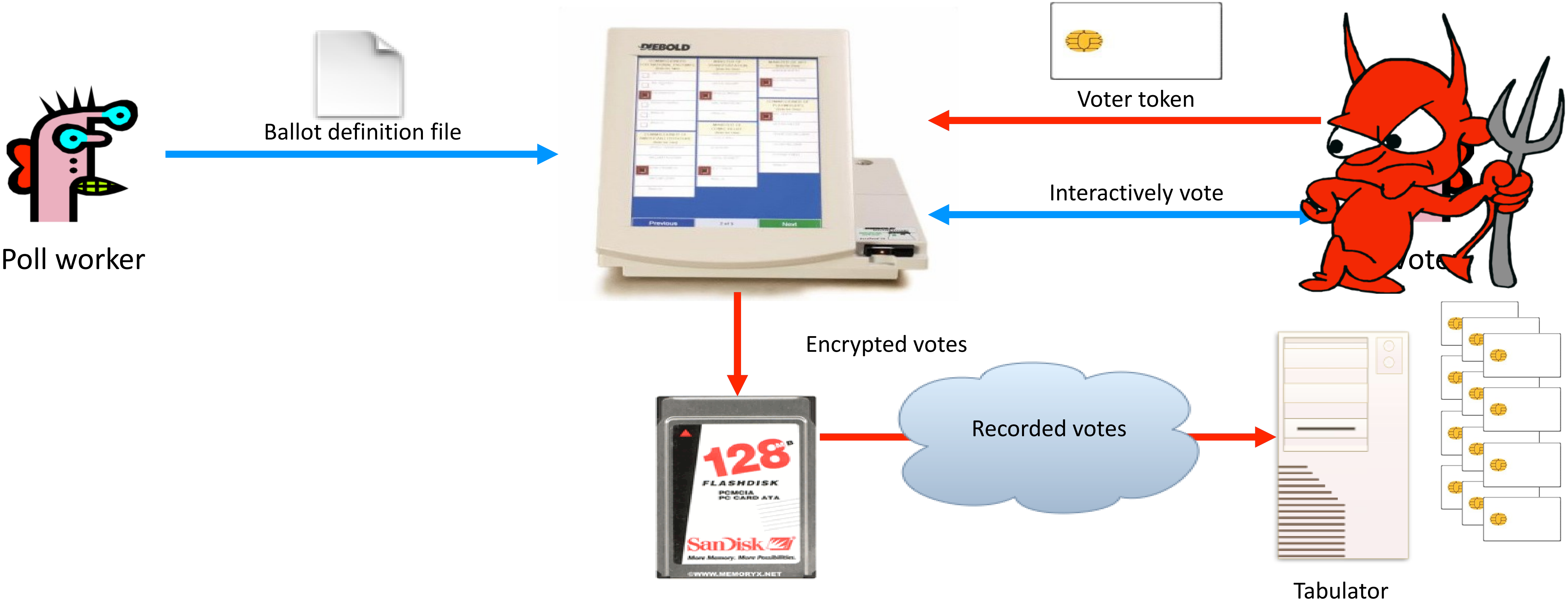
Problem: Ballot definition files are not authenticated.

Example attack: A malicious poll worker could modify ballot definition files so that votes cast for "Mickey Mouse" are recorded for "Donald Duck."



Problem: Smartcards can perform cryptographic operations. But there is no authentication from voter token to terminal.

Example attack: A regular voter could make their own voter token and vote multiple times.



Ballot definition file

Poll worker

Voter token

Interactively vote

ote

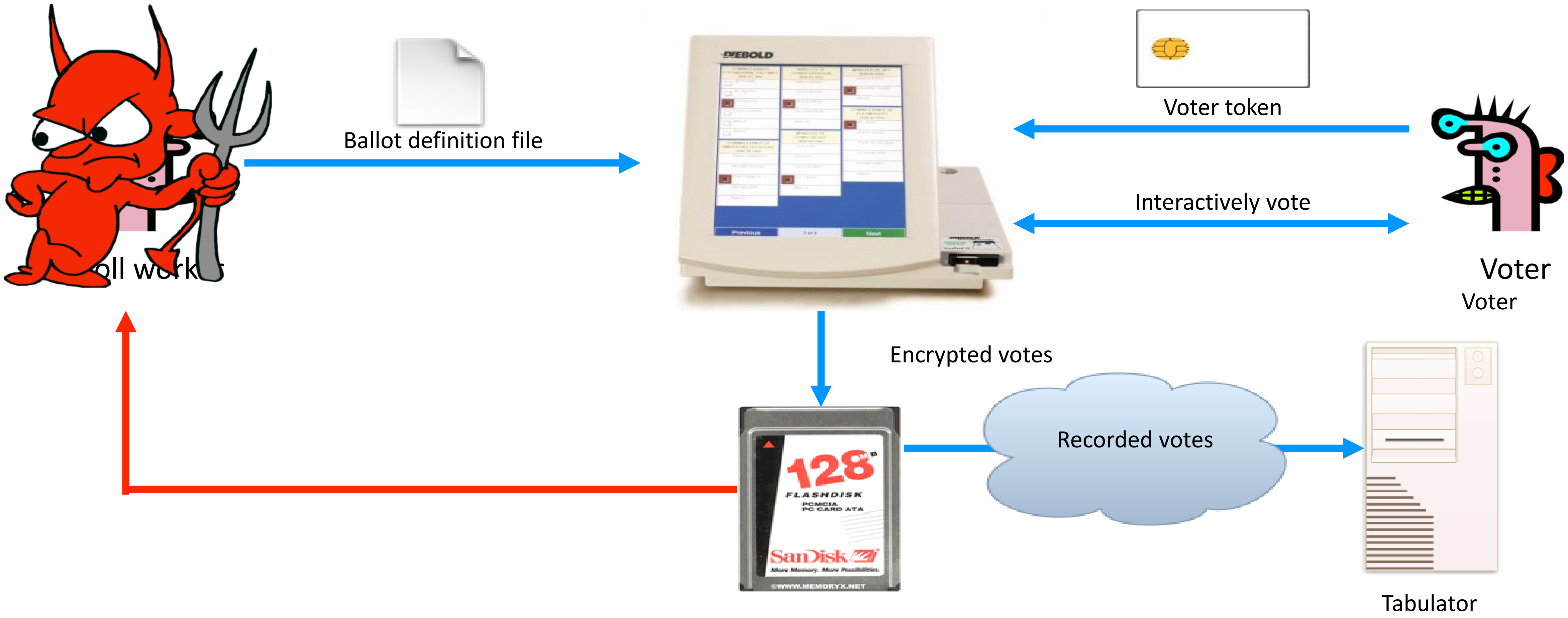
Encrypted votes

Recorded votes

Tabulator

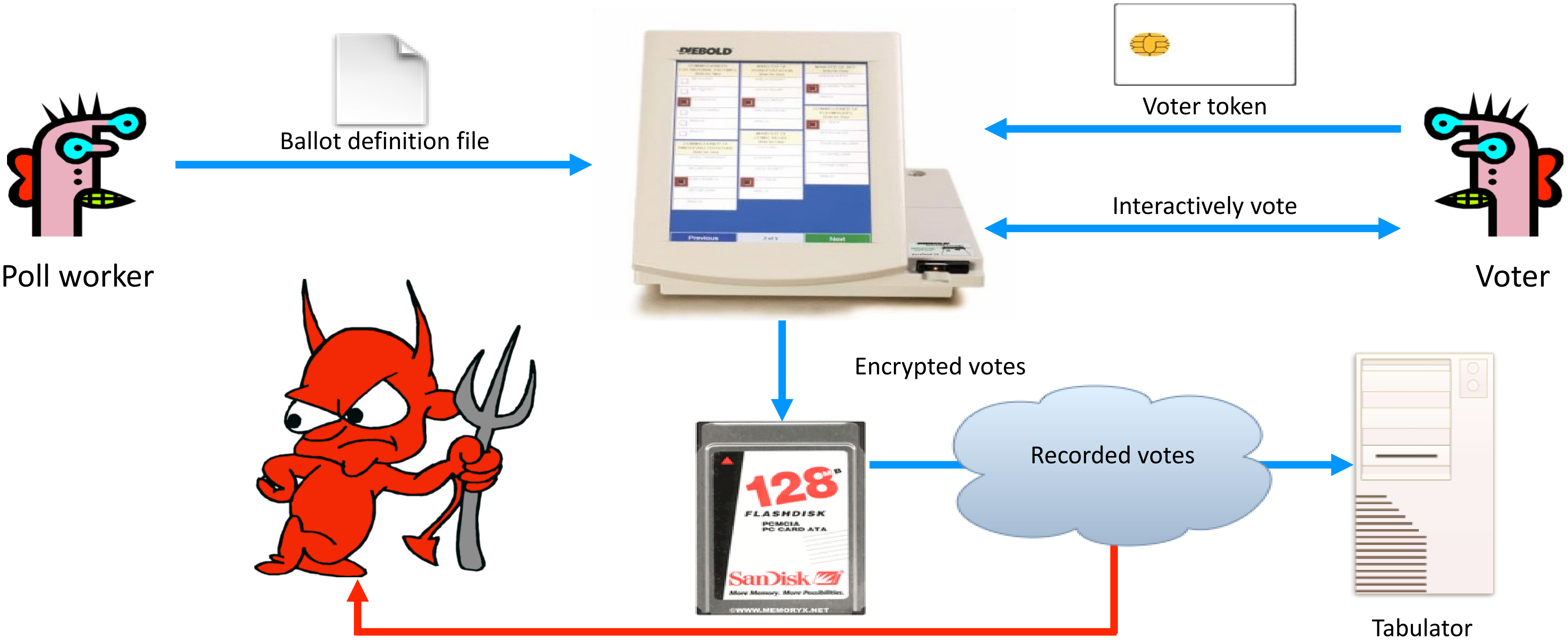
Problem: Encryption key (“F2654hD4”) hard-coded into the software since (at least) 1998. Votes stored in the order cast.

Example attack: A poll worker could determine how voters vote.



Problem: When votes transmitted to tabulator over the Internet or a dialup connection, they are decrypted first; the cleartext results are sent the the tabulator.

Example attack: A sophisticated outsider could determine how voters vote.



TOWARDS DEFENSES

Approaches to Security

- Prevention
  - Stop an attack
- Detection
  - Detect an ongoing or past attack
- Response and Resilience
  - Respond to / recover from attacks

- The threat of a response may be enough to deter some attackers

Whole System is Critical

- Securing a system involves a whole-system view
  - Cryptography
  - Implementation
  - People
  - Physical security
  - Everything in between

- This is because “security is only as strong as the weakest link,” and security can fail in many places
  - No reason to attack the strongest part of a system if you can walk right around it.

Whole System is Critical

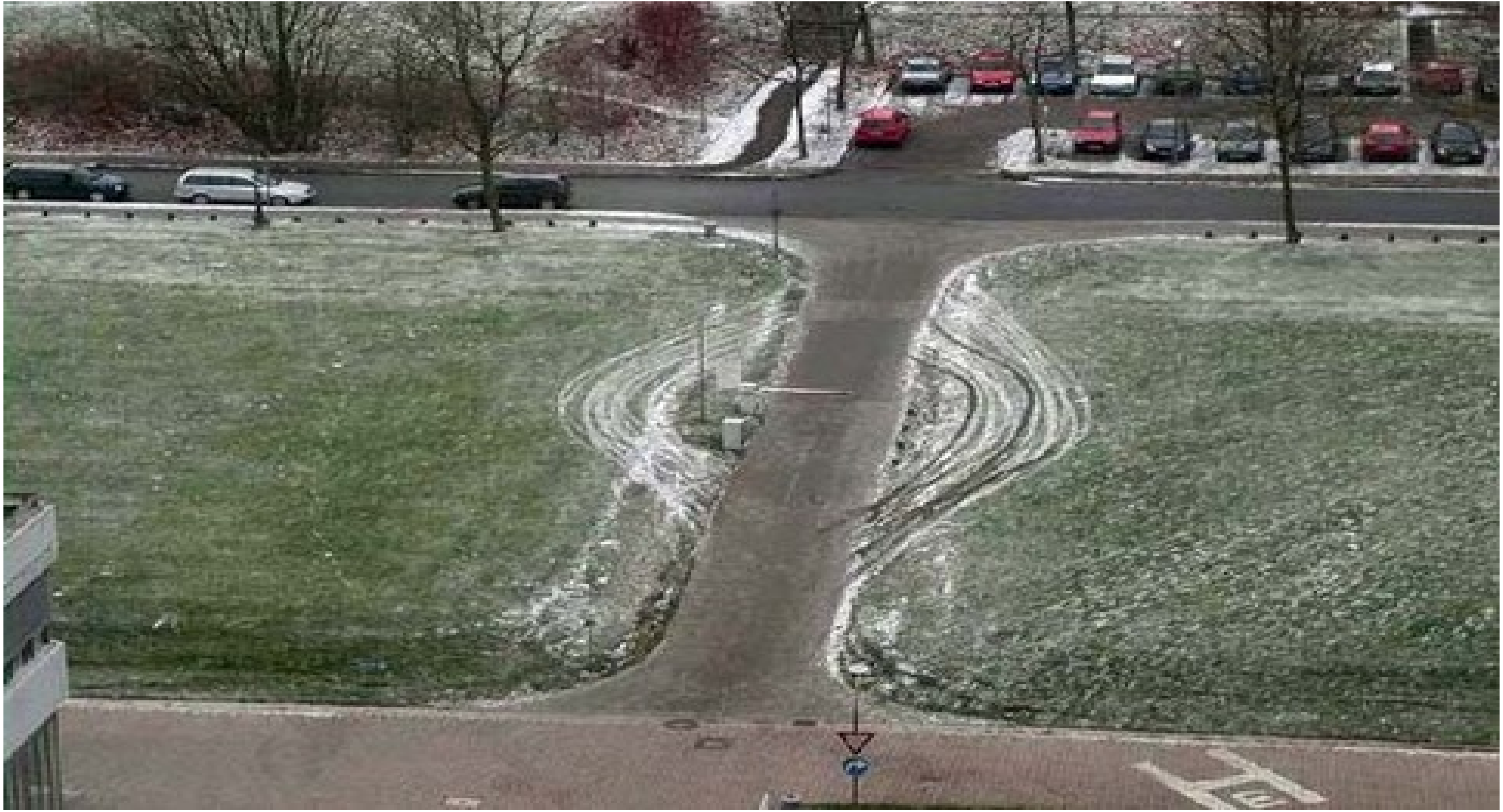
- Securing a system involves a whole-system view
  - Cryptography
  - Implementation
  - People
  - Physical security
  - Everything in between
- This is because “security is only as strong as the weakest link,” and security can fail in many places
  - No reason to attack the strongest part of a system if you can walk right around it.

Whole System is Critical

- Securing a system involves a whole-s
  - Cryptography
  - Implementation
  - People
  - Physical security
  - Everything in between

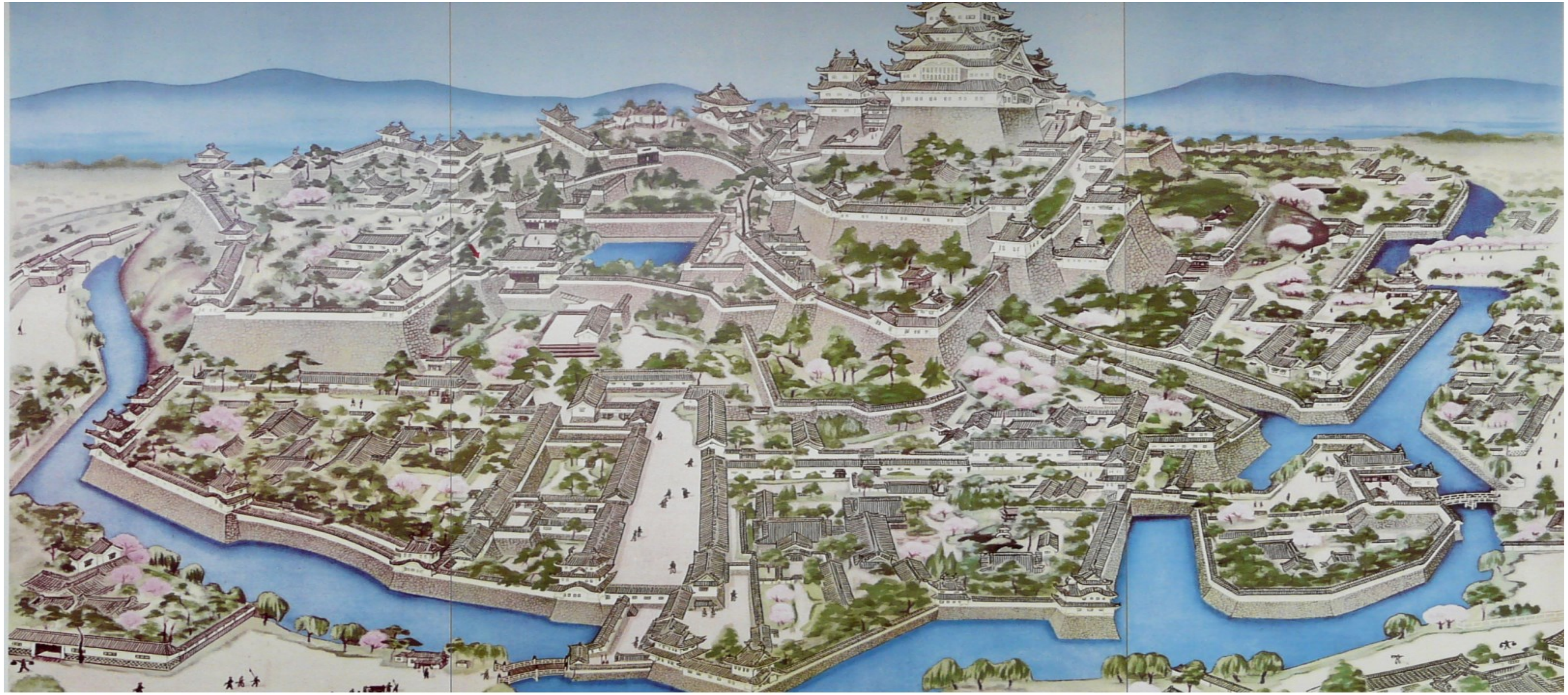
- This is because "security is only as strong as the weakest link," and security can fail in many places
  - No reason to attack the strongest part of a system if you can walk right around it.

Whole System is Critical

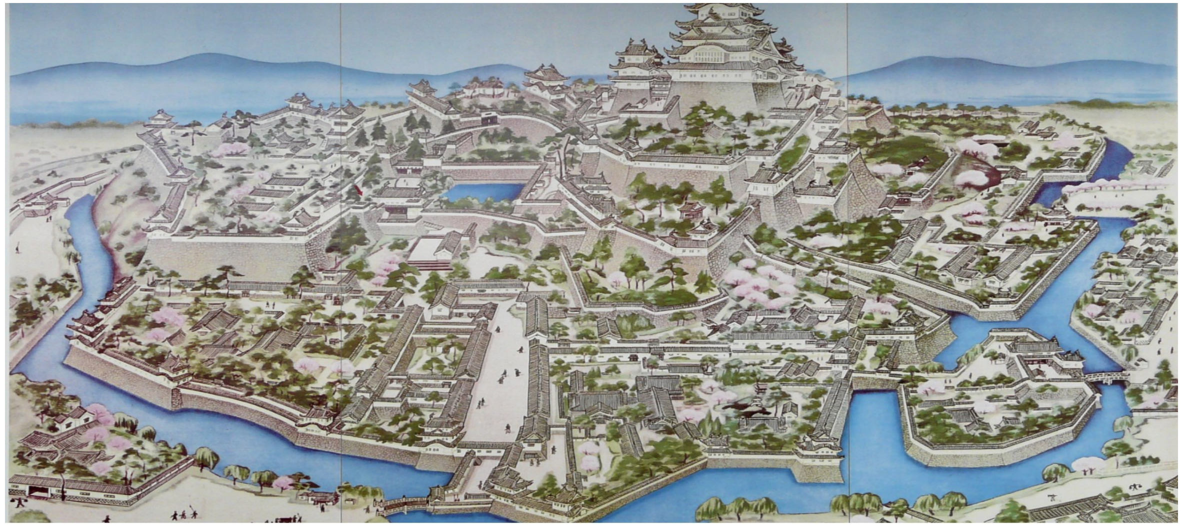


Whole System is Critical

Attacker’s Asymmetric Advantage



Attacker’s Asymmetric Advantage



- Attacker only needs to win in one place
- Defender’s response: Defense in depth

From Policy to Implementation

- After you've figured out what security means to your application, there are still challenges:
  - Requirements bugs and oversights
    - Incorrect or problematic goals
  - Design bugs and oversights
    - Poor use of cryptography
    - Poor sources of randomness
    - ...
  - Implementation bugs and oversights
    - Buffer overflow attacks
    - ...
  - Is the system **usable**?

Many Participants

- Many parties involved
  - System developers
  - Companies deploying the system
  - The end users
  - The adversaries (possibly one of the above)
- Different parties have different goals
  - System developers and companies may wish to optimize cost
  - End users may desire security, privacy, and usability
    - Related question: Do system developers / companies really understand the needs and values of all their users? Or all stakeholders who might be impacted by the system?
  - But the relationship between these goals is quite complex (e.g., will customers choose features or security?)

Better News

- There are a lot of defense mechanisms
  - We'll study some, but by no means all, in this course
- It's important to understand their limitations
  - "If you think cryptography will solve your problem, then you don't understand cryptography… and you don't understand your problem" -- Bruce Schneier