Introduction

CMSC 414: Computer and Network Security
Fall 2016, Section 0201
Michelle Mazurek

Some material from Prof. Dave Levin
• Normally, we care about **correctness**
  • Does software achieve desired behavior?

• Security is a kind of correctness
  • Does software prevent **undesired** behavior?

*The key difference is the adversary!*
What are undesired behaviors?

- Reveals info that users want to hide
  - Corporate secrets, private data, PII
  - Confidentiality
- Modifies info or functionality
  - Destroy records, change data mid-processing, install unwanted software
  - Integrity
- Deny access to data or service
  - Crash website, DoS, fairness
After Illinois hack, FBI warns of more attacks on state election board systems
Concern about more attacks mounting as presidential elections approach.
SEAN GALLAGHER – 8/29/2016, 11:55 AM

Actively exploited iOS flaws that hijack iPhones patched by Apple
Jailbreak vulnerabilities allowed attackers to tap encrypted chat messages.
DAN GOODIN – 8/25/2016, 3:10 PM

Trading in stock of medical device paused after hackers team with short seller
St. Jude Medical declares claim of vulnerability "false and misleading."
SEAN GALLAGHER – 8/26/2016, 5:22 PM

HTTPS and OpenVPN face new attack that can decrypt secret cookies
More than 600 sites found to be vulnerable to demanding exploit called Sweet32.
DAN GOODIN – 8/24/2016, 11:45 AM
Why are attacks so common?

• Systems are complex, people are limited

• Many attacks exploit a \textit{vulnerability}
  • A \textit{software defect} that can be manipulated to yield an undesired behavior

• Software defects come from:
  • Flaws in \textit{design}
  • Bugs in \textit{implementation}
Case study: Heartbleed

• SSL is the main protocol for secure (encrypted) online communication

• Heartbleed was a vulnerability in the most popular SSL server
HOW THE HEARTBLEED BUG WORKS:

Server, are you still there? If so, reply "POTATO" (6 letters).

User Meg wants these 6 letters: POTATO. User Ada wants pages about "irl games". Unlocking secure records with master key 5130985733435.

https://xkcd.com/1354/
SERVER, ARE YOU STILL THERE? IF SO, REPLY "BIRD" (4 LETTERS).

HMM...

User Olivia from London wants pages about "Hans in car why". Note: Files for IP 375.381.283.17 are in /tmp/files-3843. User Meg wants these 4 letters: BIRD. There are currently 345 connections open. User Brendan uploaded the file selfie.jpg (contents: 334ba9e2c8eb9ff59be3b6f).

BIRD
Server, are you still there? If so, reply “HAT” (500 letters).

User Meg wants these 500 letters: HAT. Lucas requests the "missed connections" page. Eve (administrator) wants to set server's master key to "14835038534". Isabel wants pages about "snakes but not too long". User Karen wants to change account password to "CoKoBaSt".
Case study: Heartbleed

- SSL is the main protocol for secure (encrypted) online communication
- Malformed packet allows you to see server memory
  - Passwords, keys, emails, visitor logs …..
- Fix: Don’t let the user tell you how much data to send back!
  - This is a design flaw
RSA breach, 2011

1. **Flash exploit**: When run by vulnerable Flash player version, allows arbitrary code exec.

2. **Excel embed**: Runs automatically when spreadsheet is opened.

3. **Spear phishing**: Spreadsheet attached to email claiming to be from trusted party, about relevant content
   - Any “From” address can be forged
Why are attacks so common?

- Attacks via design flaws, implementation bugs
- All software has bugs
- Normal users don’t see most bugs
  - Post-deployment, usually rare corner cases
- Too expensive to fix every bug
  - Fix what will affect normal users
Why are attacks so common?

- Normal users avoid bugs
- Adversaries look for them to exploit

*Attacks are possible even with perfect software*
Why are attacks so common?

• Because it’s profitable
  • (Or attackers think it is)
• Because systems have weak links
Steps toward more security....

• Eliminate bugs or design flaws, or make them harder to exploit
  – Think like an attacker!
• Deeply understand systems we build
• Be mindful of user-controlled inputs
• Never roll your own
Today’s agenda

• What is security
• Administrivia
• Case study
• Trusting trust
People

• Me: Michelle Mazurek (mmazurek@cs)

• TAs: Michael Cassano, Jake Hammontree, Nishant Rodrigues
Resources

• Everything is on the class website:
  • https://sites.umiacs.umd.edu/mmazurek/414-f16
  • Or will be soon

• We will also use Piazza
  • You should have been added
  • https://piazza.com/class/iqlca72yle4j

• Office hours on website
  • Occasionally also by request
Reading

• Mostly papers
• Recommended: textbooks, outside resources
  • Listed on website
  • Share your recommendations on Piazza
Prerequisite knowledge

• Reasonably proficient in C and Unix
  • Not terrified of x86 assembly
  • See self-test on resources section of website

• Creative and resourceful

• No prior knowledge in networking, crypto
Grading

- Projects: 50%
  - 12, 12, 12, 14
- Midterms: 15%, 15%
  - Schedule and excused absences on website
- Final: 20%
Ethics and legality

• You will learn about, implement attacks

• *Do not use them without explicit written consent from everyone involved!*
  • Make sure you know who is involved

• If you want to try something, tell me and I will try to help set up a test environment

• Don’t violate: Ethics, UMD policies, state and national laws
Read the syllabus

- Late policy
- Good-faith effort requirement
- Excused absences
- Academic integrity
What’s in this course?

• Software security
• Crypto
• Network security
• Human behavior
Software security

Memory safety

Malware

Web security

Static analysis

Design principles
What’s in this course?

- Software security
- Crypto
- Network security
- Human behavior
Applied crypto

- What it is (medium-high level)
- How to use it responsibly

**Black-box approach**

- **Authentication**
- **Designing protocols that use crypto**

**Anonymity**
What’s in this course?

• Software security
• Crypto
• Network security
• Human behavior
Network security

• How to build secure networked systems

**Attacks on TCP, DNS, BGP**

**Anonymity**
What’s in this course?

• Software security
• Crypto
• Network security
• Human behavior
Human behavior

- Security is limited by humans’ incentives, capabilities, and preferences

**Security as secondary task**

**Warnings and habituation**

**Passwords**

**Design principles**
Plugs

• AWC @ UMD
  • Twitter: umdawc; awc-umcp.squarespace.com
  • First GBM is Thursday

• BIBIFI competition
  • https://builditbreakit.org/
case study

E-VOTING ANALYSIS

Kohno et al., IEEE S&P 2004
Halderman, 2016
“Security mindset”

• Consider a complex system:
  • Potential security threats?
  • Hidden and explicit assumptions
  • How to mitigate the risks?
  • What are different players’ incentives?
1. Summarize the system

1. Pre-election: Poll worker loads “ballot definition” via e.g. USB

2. Voting: Voter obtains single-use smartcard, votes, vote stored encrypted, card canceled

3. Post-election: Votes decrypted and sent to tabulator, who counts
2. Identify goals/requirements

- **Confidentiality**: Can’t find out who I voted for
- **Integrity**: Can’t alter votes
- **Availability**: Can’t deny opportunity to vote
- **Usability**: General public can vote correctly without undue burden

What if the attacker can violate these, but you catch him/her?
3. Identify adversaries/threats

- Poll worker, voter, outsider
- Display one vote / count a different vote
- Vote multiple times
- End election early (DOS)
- Tamper with stored data
- Reveal who voted for whom
Diebold Accuvote TS

• Used in 37 states! (in 2004)
• No cryptography protects smartcards, ballot definition file
• “Protected counter” in single, mutable file
• Pose as voting machine, send to tabulator
• Homebrew crypto protects vote logs
  • Hardcoded key since at least 1998
• Read the paper for more
Diebold Accuvote TS

- Used in 37 states (in 2004)
- No cryptography protects smartcards, ballot definition file
- “Protected counter” in single, mutable file
- Pose as voting machine, send to tabulator
- Homebrew crypto protects vote logs
- Hardcoded key since at least 1998
- Read the paper for more

https://crayfisher.files.wordpress.com/2012/07/double_facepalm_tng1.jpg
Follow-up

• More researchers confirmed these bugs and found others (got real hardware)
• State investigations: MD, CA, OH
  • Similar problems from other manufacturers
  • Sequoia AVC: designed 1980, used in NJ 2009
• “By the 2014 general election, 70% of American voters were casting ballots on paper”
Takeaways

- Adversarial thinking
- Whole-systems view
  - Hardware, software, network, users, economics
- Only as strong as weakest link
  - Break into building vs. sniff unencrypted traffic
  - You have to be right always, adversary once
- Never homebrew crypto!
- Security through obscurity DOESN’T WORK!
Is anything really secure?

- Security requires context
  - Threat model
  - Definition of security / protection
  - Who/what do you trust? (Assumptions)
- Trust no one!
  - How did you compile?
  - Who built your OS?
  - Required reading: Thompson