(Advanced) Computer Security!

Introduction to Software Security
(part 2)

Prof. Travis Peters

Montana State University

CS 476/594 – Computer Security

Spring 2021

https://www.traviswpeters.com/cs476

Reminders

- Please update your **Slack** profile
  - first name, last name, nice photo
- Please update your **GitHub** profile
  - first name, last name, nice photo
- Please update your **Zoom** profile
  - first name, last name, nice photo
  - **No unprofessional/distracting backgrounds please!**

  -> Let's take a minute now to do this for Zoom...

Today

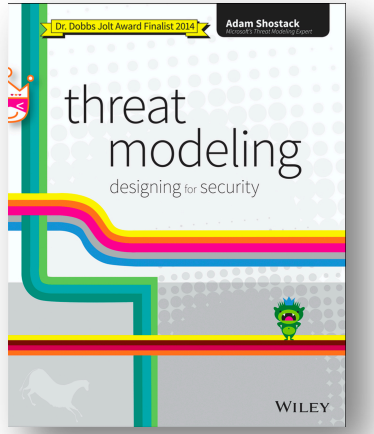
- Announcements
  - "Your First Threat Model" posted – due next Tuesday

- Learning Objectives
  - ~~Basic ideas from security~~
    - ~~CIA Triad (plus…)~~
    - ~~Common threats & defenses~~
    - Threat Modeling 101
  - Review some basics
    - Models/layout of a computer & a program
    - Linux & Basic Linux Security
    - (Basic C programming and command line usage)

How do we come up with all of this?

What's the "right" approach?

Threat Modeling

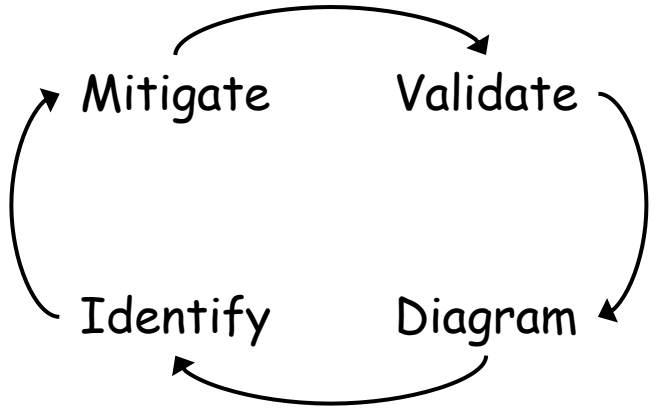


- **Threat Modeling**
  - a consistent & structured approach to defense
  - identify and prioritize threats and feasible countermeasures

All models are wrong, some models are useful.
— George Box

- "What's Your Threat Model?"
  -> *A simple question to reveal **who**/**what** you're worried about!*

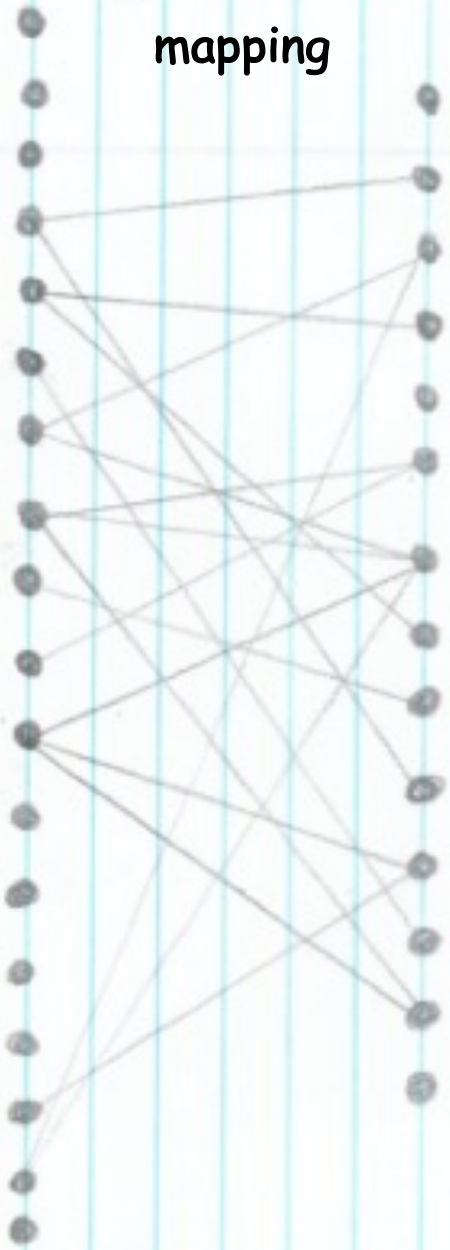
- As you begin threat modeling, focus on four key questions:
  - What are you building?
  - What can go wrong?
  - What should you do about those things that can go wrong?
  - Did you do a decent job of analysis?



Threat models can take various forms...

Security Controls

Sc1 Account Timeouts
Sc2 Account lockouts
Sc3 Two-factor authentication
Sc4 Password complexity guidelines
Sc5 Configure device ports
Sc6 Patch management
Sc7 Intrusion Detection/Prevention
Sc8 Data encryption
Sc9 Air-Gap from network
Sc10 Physical security
Sc11 DOS Protection
Sc12 Event Reporting
Sc13 offsite Data Backups
Sc14 Antivirus protection
Sc15 Penetration Testing
Sc16 Input sanitation
Sc17 Code Signing
Sc18 Runtime Application Self-protect



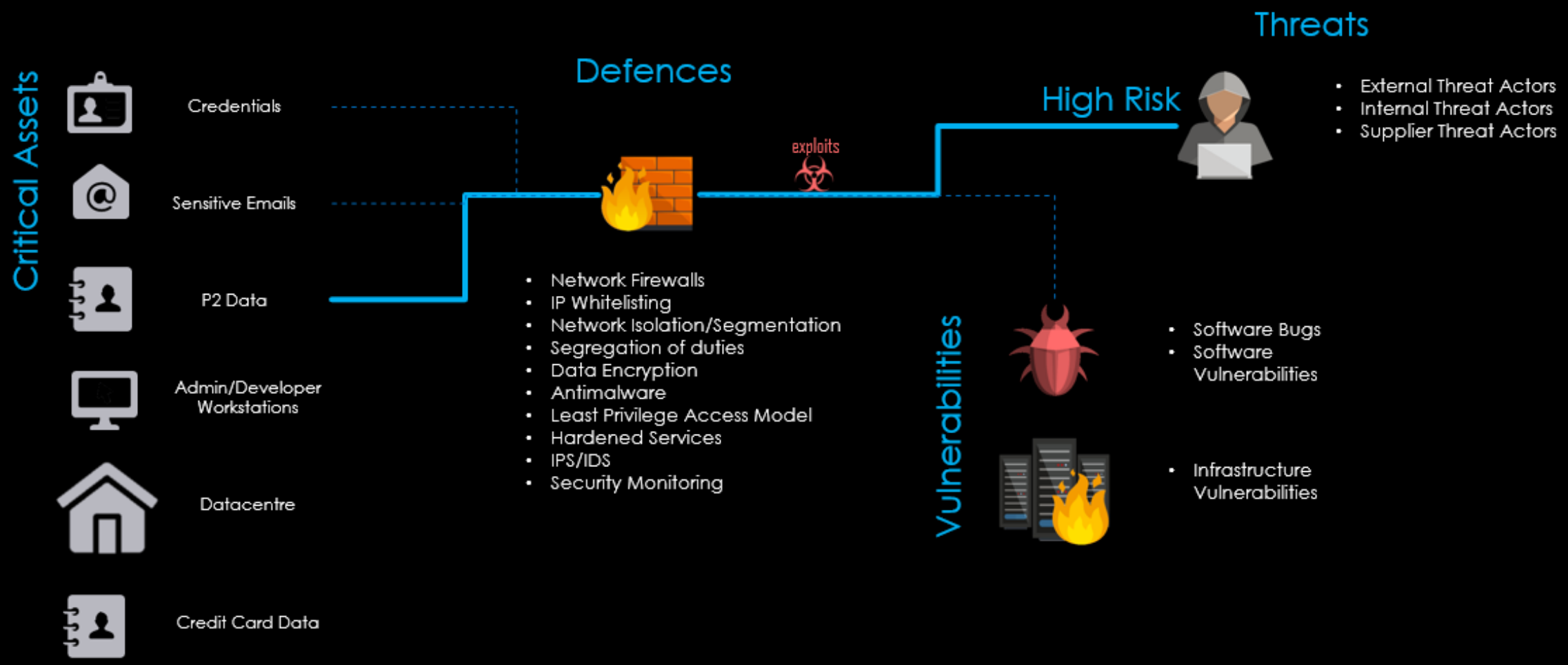
V1 Hardcoded PWDs
V2 Weak PWDs
V3 Command Injected flaw
V4 Open ports
V5 No account lockout
V6 Unencrypted services
V7 insecure web applications
V8 insecure network services
V9 insecure cloud interface
V10 Account enumeration
V11 Cross-site scripting
V12 Buffer overflow
V13 removal of physical security
V14 missing authorization

Vulnerabilities

Former CSCI 476 Student
(Thanks, Maddi Tandberg!)

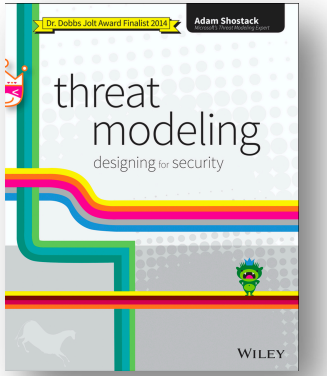
Non-invasive attack	Remote	Local	Potential damage	Probability of attack	Mitigation
Communication protocol	✓		Access to all data on network; escalation of privilege on device	High	Secure communication protocol (TLS) with a True Random Number Generator
Authentication	✓	✓	Impersonation of device on network, access to all assets	High	Follow proper authentication guidelines, fuzz testing
Code vulnerabilities	✓	✓	Control of device	High	Isolate application memory in CPU
JTAG (debug) access		✓	Full control of device; access to all assets	High	Disable JTAG; re-enable only with secure protocol
GPIO control		✓	Change device state	Medium	Monitor GPIO for unexpected behavior, fuzz testing
Side channel (DPA/TA)		✓	Access to keys	Medium	Side channel resistant cryptographic algorithm implementations
External bus monitoring		✓	Access to keys and application data	Low	Encrypt external bus interfaces

Threat Model



Threat Modeling

"What's Your Threat Model?"



Brainstorming Variants...

- Literature review – study systems that are similar to yours
- Free-form brainstorming – Gather around a whiteboard; enumerate threats/possible defenses
- Scenario Analysis - Propose a scenario and ask "what might go wrong?"
- Pre-Mortem - Given a project/deadline, assume failure. What do you do next?
- Movie Plotting – Pick outrageous ideas; what happens next? (Think: Ocean's Eleven, The Italian Job, etc.)

Try it! (Scenario Analysis)

Threat Model *handing your phone to a cute person in a bar...*

Physically steal device
- Germs... (COVID)
- theft
- spend your money!
  ↳ steal account info

- download apps/ malicious sw
- blackmail (e.g., photos)
- nfc (perhaps disable by default)

assume
- you are distracted

social media accounts
↳ they could post to accounts

phone case may carry ID or credit/debit cards

2FA

Are there any problems w/ relying solely on brainstorming?

Some scenarios are more likely then others

Need structure — we might miss something important!

We have biases

We don't know what we don't know... / there may be resources that we need

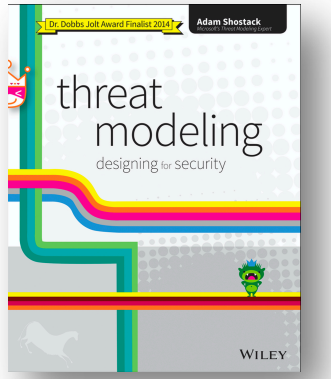
Limited info from stakeholders

Threat Modeling

Structured Approaches to Threat Modeling

TL; DR – There is no one, right answer. The "right" approach is the one that works for you and your team!

- **Structured Brainstorming Models:**
  - Asset-centric – focus on things of value; things attacker want; things you want to protect
  - Attacker-centric – focus on attackers/archetypes/personas and their capabilities
  - Software-centric – focus on SW; most SW is backed by structured models
    - Data Flow Diagrams (DFG), UML, Swim Lane Diagrams, State Diagrams, etc.



- **Threat Modeling Methodologies**
  - **STRIDE**
    = Spoofing, Tampering, Repudiation, Info Disclosure, Denial of Service, and Elevation of Privilege
    https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)?redirectedfrom=MSDN
  - **Attack Trees**
    = Represent attacks/threats against a system in a tree structure; goal = root node, approaches to achieving goal = leaf nodes
  - **Attack Lists & Libraries (Barnard's List, OWASP Top 10, CAPEC, etc.)**
    = An enumeration of common attackers, threats, vulnerabilities, etc. – *a great place to start!*

-> help developers identify types of attacks that software tends to experience

Summary

- **Security** is the application & enforcement of policies through defense mechanisms over data and resources.
  - Security Policy = WHAT
  - Defense Mechanism = HOW

- Continuously revisit your **threat model** - the security field is always evolving
  - new **assets**, **threats**, **capabilities**, **technologies**

- Security is hard! **You** need to be ready to think through new situations that arise, leveraging what you've already learned (here, past experiences, case studies, reports, etc.) to find S&P solutions for those new situations.

(Advanced) Computer Security!

Systems & Software

(A Brief Review)

Prof. Travis Peters

Montana State University

CS 476/594 – Computer Security

Spring 2021

https://www.traviswpeters.com/cs476

Review

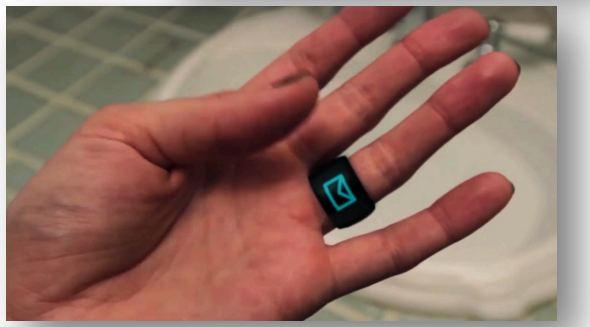
— https://media.tenor.co

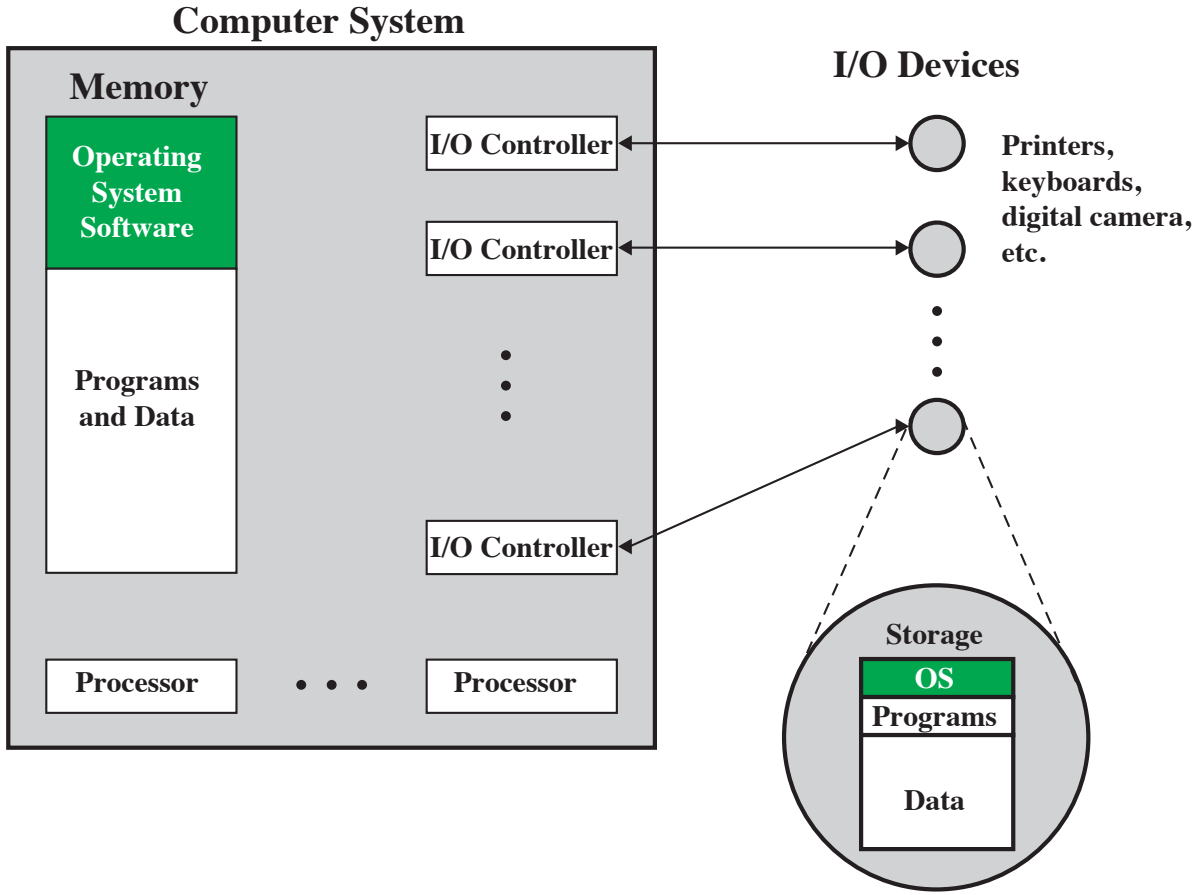
Many of the following concepts are specific to the Linux, which is most relevant for this course. Many of the ideas, however, are universal.

System Overview

A Computer In A Nutshell

A computer, is a computer, is a computer, …





A Computer In A Nutshell

A computer, is a computer, is a computer, …

Q: What are the "resources"? / assets

Compute power

Memory / contents — programs/code

Value of the HW itself

Network / IP address

Battery

User(s)

OS (manager of resources)

I/O

process → apps

files / filesystems

Q: What "threats" might you anticipate?

Accessing webcam (spyware)

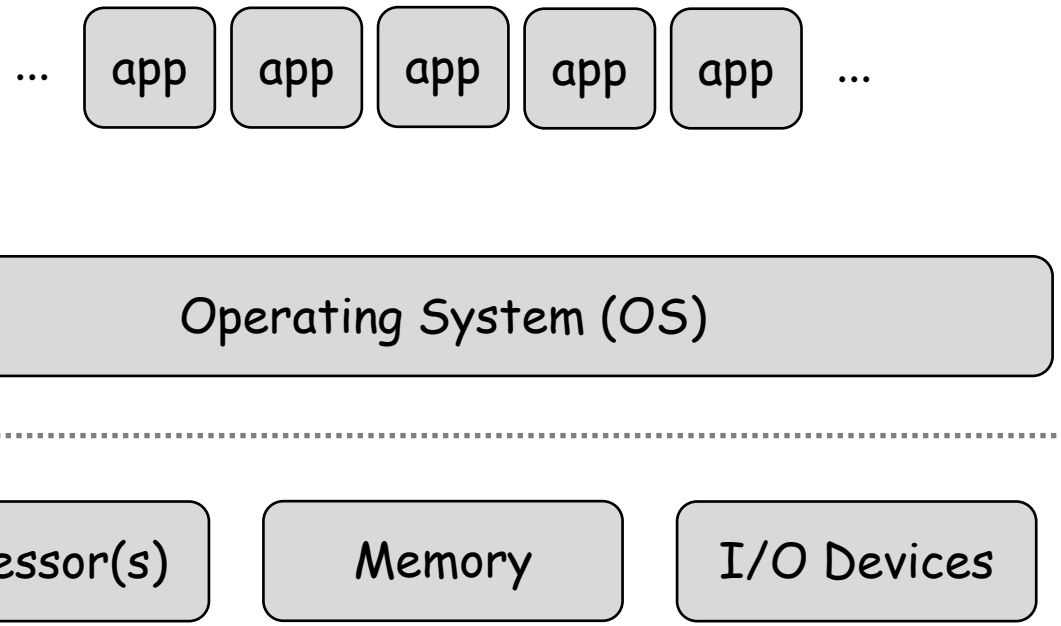
Access your network

Accessing your files

using your compute (crypto mining)

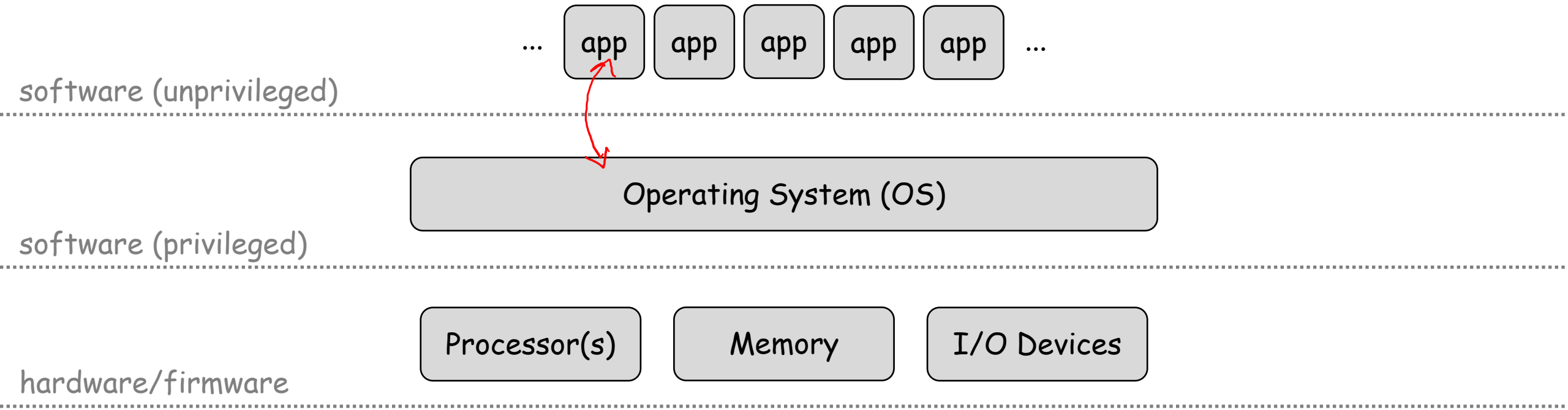
recruited to botnet

Typical Layers of a Computer



Typical Layers of a Computer

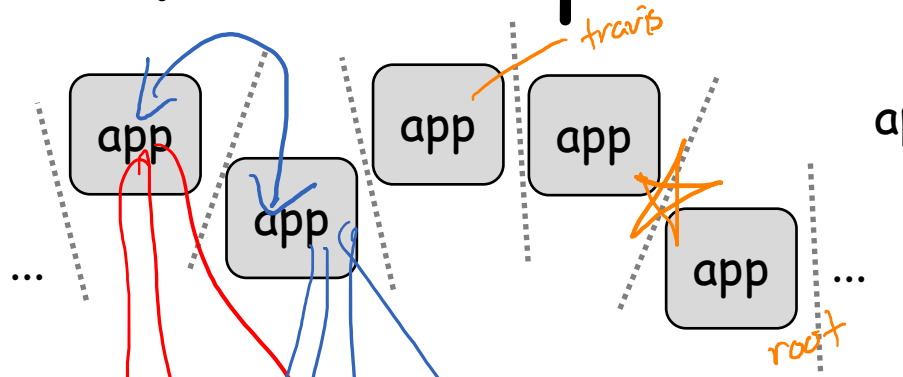
Thinking about trust boundaries…



Threats tend to cluster around **trust boundaries** and **complex parsing**,
but may appear anywhere that **information is under the control of an attacker**.

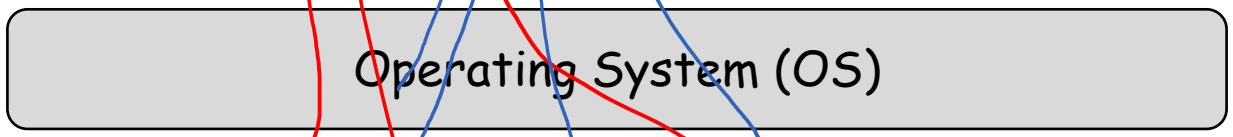
Typical Layers of a Computer

Thinking about trust boundaries…

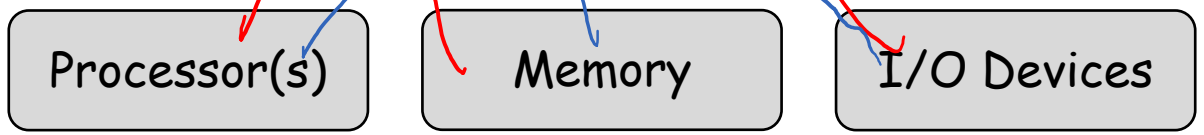


apps/users can have varying levels of privileges too…

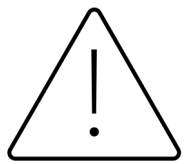
software (unprivileged)



software (privileged)



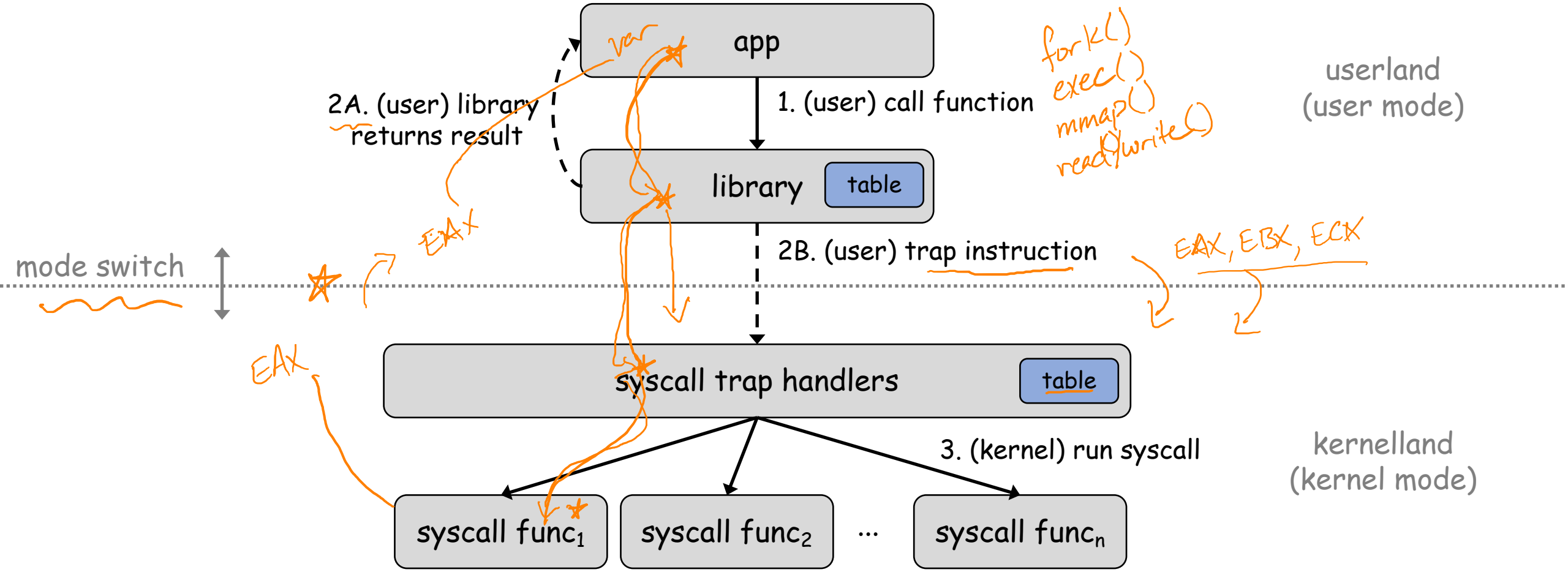
hardware/firmware



Threats tend to cluster around **trust boundaries** and **complex parsing**, but may appear anywhere that **information is under the control of an attacker**.

How Apps Use System Resources

A computer, is a computer, is a computer, …



How Apps Use System Resources (Example!)

A computer, is a computer, is a computer, …