Secure Computer Systems

Mustaque Ahamad, Ph.D.

Professor & Associate Director: Educational Outreach, Institute for Information Security & Privacy

Georgia Tech - College of Computing

Getting Started: Course Introduction, Necessary Background & Course Roadmap

Before We Begin: Getting to Know Your Instructional Team

- Dr. Ahamad has taught CS 6238 since it was first developed
- A team of several teaching assistants (TAs) will assist him in delivery of the course
- TA team will be managed by Dr. Ahamad but will have primary responsibility for programming projects and discussion fourm responses

Necessary Background, Assessment and Instructor Expectations

Necessary Background

- Undergraduate OS/architecture/systems programming course(s)
- Strong programming skills (not vulnerability exploitation but implementation of protection/security mechanisms)
- Yes, you can acquire the needed background as we go but it is your responsibility

Assessment

1. No textbook but you will read covered topics in research papers
   - Must read them as we go
2. Weekly quizzes will test that you do keep with with course readings
3. Programming projects will reinforce covered concepts with hands-on implementations
4. Two exams (mid-term & Final)
5. Final grade based on curve

Instructor Expectations

- Always be eager to learn something new or to share new ideas with others
- Active participation in discussion forums expected
  - Piazza is used for online discussions
- We will not answer questions about projects at the last minute (get started early)
- Get to know me and the TAs via virtual office hours (large class; but we can make it work)
  - Please attend as many as possible

Administrivia (But Really Important)

GT Honor Code, Plagiarism etc.

- Your work should be your work
    - Collaboration is great and encouraged but submit your own work
- Quizzes may cover material not discussed in lectures (but from assigned readings)
- Exams will cover material discussed in lectures
- Final exam will include topics covered after midterm
- Quizzes will be administered via Canvas, so please make sure to check technology requirements for an online class

Revisiting the Security Mindset

Revisiting the Security Mindset (in the OS context)

Threats

- Cyber criminals to nation-states
- Complex ecosystems
  - Vulnerability discovery and their exploitation
  - Underground economy and marketplace for exploit-kits, compromised resources, fulfillment etc.

Vulnerabilities

- Complexity of OS

Attacks

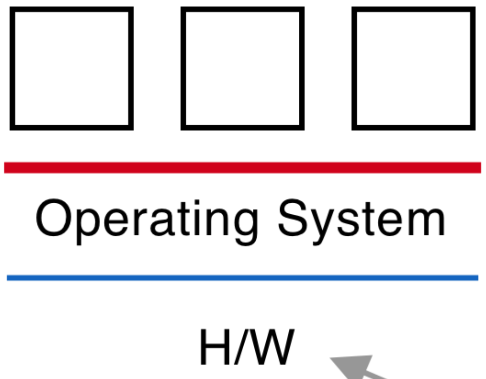
- Why attacks against the OS are attractive?
- Unauthorized access to sensitive information (Confidentiality, Integrity and Availability)

What does an Operating System Do?

Why Do We Have an OS?

- Makes it easier to use/share physical resources
- Manages/controls physical resources to efficiently utilize them
- Must have access to all physical resources

A Closer Look



Apps and Services

Use virtualized resource (Files, address, spaces etc)

(Real resources)
(Memory, CPU, disk)

TCB as a Reference Monitor

TCB as a Reference Monitor

- Untrusted apps need to access/reference protected resources
- TCB must “**monitor**” such references
- No reference should be able to bypass the TCB

TCB Requirements

Tamper-proof

- Untrusted code cannot alter it

Complete mediation

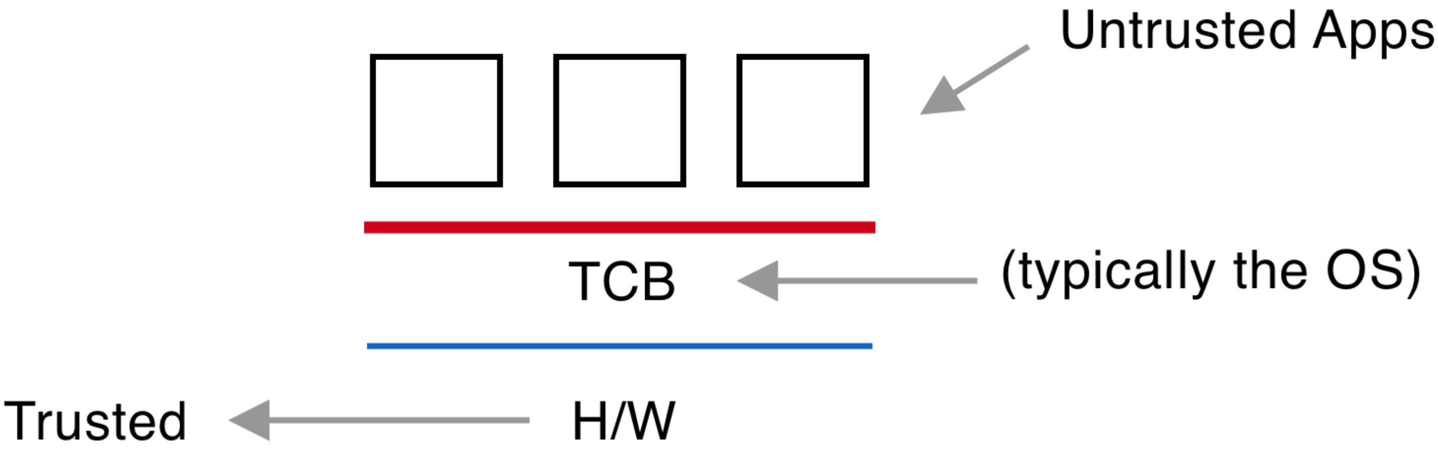
- Cannot be bypassed

Correctness

- No vulnerabilities (ideally but we will talk more)

Trusted Computing Base

Trusted Computing Base



Untrusted Apps

TCB

(typically the OS)

Trusted

H/W

What Does the Reference Monitor Do?

1. Who is making the request?
    - Authentication
2. Is the source of the request authorized to access the resource?
    - Authorization (a.k.a access control)
3. Gold standard of security (Authentication, Authorization and Audit)

Role of an Operating System in Protecting Resources

What Happens if an Attack Compromises the OS?

- Attacker will have access to all resources
- No security for any user/service!!

↓

OS MUST BE TRUSTED SO IT IS NOT COMPROMISED

Would OS Compromise Allow the Following Attacks?

- User impersonation?
- Data destruction and exfiltration?
- Making security tools ineffective?
- Malicious services (e.g., spam delivery)?

What is Needed for Trustworthiness?

Getting Back to Trusted in TCB

Trust (Merriam-Webster):

- **a:** assured reliance on the character, ability, strength or truth of someone or something
- **b:** one in which confidence is placed

Trust comes from:

(i) What TCB does

(ii) How well it does what it is supposed to do

(i) is somewhat easier to answer but how about **(ii)** ?

Where Does Trust Come From?

- **What the TCB does?**
  - What core functions must the TCB include?
- **How well it does it?**
  - Structuring, testing, formal models/verification
- **Who develops the TCB? – Really?**
  - Reflections paper

The "Reflections on Trusting Trust"Paper

- Turing award lecture by Ken Thompson, available at https://dl.acm.org/ft_gateway.cfm?id=358210&ftid=801607&dwn=1&CFID=35714467&CFTOKEN=2a4d8469bfc91e57-B7741A5F-A9D1-03D4-97981B9980DA96DA
- Figure out C programming and code in the paper
- Login Trojan
- Paper shows how finding the Trojan can be made hard even with source code
- Can you trust code that you can review?

Reflections on Trusting…

- **Step 0**: Produce $C_{N,S}$ and $C_{N,B}$.
- **Step 1**: Modify $C_{N,S}$ to produce $C_{M,S}$.
- **Step 2**: Compile $C_{M,S}$ with $C_{N,B}$ to create $C_{M,B}$.
- **Step 3**: Remove $C_{M,S}$. Compile $C_{N,S}$ with $C_{M,B}$ to generate C'.
- **Step 4**: Install $C_{N,S}$ and C'.

- ***Is C' malicious?***
- ***Would recompiling the source fix it?***

Can We Know if There’s a Trojan?

- Review, update and recompile the compiler source
  - The Trojan will stay forever
- Compiler was generated with other tools
  - Tool chain?
- Cannot trust a software system unless the people behind the system and the entire tool chain can be trusted
- In the context of an OS, this is pretty sobering
- With this in mind, let us try anyway

TCSEC: Revisiting the Orange Book

How Much to Trust?

- We will really focus on TCSEC or the Orange Book
- The “If A1 is the answer….” paper was a trip down memory lane or the distant past of secure computer systems
- Paper talks about why and how we got there.

- **High Level Goal:** What questions can I ask (and check the answers) to determine how much to trust a system?

TCSEC Divisions/Classes

- **Class D**
- Not in other classes (fails to meet any requirements, including isolation of TCB from untrusted applications)
- **Class C1**
- Isolation of TCB
- User authentication
- Access control (discretionary)
- **Class C2**
- Add accountability/audit requirements.
- Logs

TCSEC Classes (cont.)

- **Class B1**
- Mandatory access control
- Well-defined TCB
- Penetration testing
- **Class B2**
- Confinement and covert channels
- TCB structuring (e.g., modularity)
- **Class B3**
- Defined security model
- Separation of security code from non-security functions
- Least privilege (a design principle we will revisit)
-

TCSEC Classes (cont.)

- **Class A1**
- Verified design (formal model for TCB design)
- **Class A2**
- Formal verification of TCB implementation
- **Increasing Assurance**
- As we move up, more security functionality and greater level of assurance about correctness
- **Common criterion**
- Vendors want to get highest rating to claim trustworthiness

Secure Boot and Trust Policy Module (TPM)

What is Trust Policy Module or TPM?

- What is the basic idea of **“trust”**?
- Root of **“trust”** is hardware (the TPM chip)
- Platform measurements and attestation
- Would you “**trust”** because you have a TPM?
- Who benefits from such **“trust”** ?

- **For more information:**
- Ross Anderson FAQ at https://www.cl.cam.ac.uk/~rja14/tcpa-faq.html

An Example

ZerodiumVerified account @Zerodium Jan 7

Announcement:
“We are increasing our bounties for almost every product. We're now paying $2,000,000 for remote iOS jailbreaks, $1,000,000 for WhatsApp/iMessage/SMS/MMS RCEs, and $500,000 for Chrome RCEs. More information at: https://zerodium.com/program.html#changelog …”
–
Why is someone paying millions of dollars for jailbreak exploits?

Course Roadmap

Course Roadmap

Design Principles for Secure Systems

Hardware Support for Protection of Resources

- First two TCB requirements
  - Tamper-proof and complete mediation
  - Need hardware help
  - Memory protection, privilege rings, privileged instructions
  - Virtualization

Authentication

- How do we know if an authentication method is good?
- Secure implementation of an interesting authentication method

Course Roadmap (cont.)

- **Discretionary Access Control** (DAC)
  - Access controx matrix
  - ACLs and C-lists
  - Examples of how systems implement them
  - Limitations of DAC
- **Mandatory Access Control (MAC)**
  - Bell Padula, Biba and other MAC models
  - Information flow
  - SELinux

Course Roadmap (cont.)

- **Covert/side Channels**
- **Distributed Systems Security**
- Authentication, secure network communication, trust, secure boot, delegation
- End-to-end security
- **Database Security**
- **DB Security, Inference Attacks, Privacy, MAC in DB**

Summary

Trusted Computing Base

- Operating system (OS) plays a critical role in protecting resources
- OS serves as the trusted computing base (TCB) and reference monitor
  - Tamper-proof
  - Complete mediation
  - Correctness
- Trust comes from what the TCB does and how well it does it
- Trust is also enhanced by following design principles for secure systems
- Read assigned papers for more details