CSCI 1800 Cybersecurity and International Relations

Engineering for Security

John E. Savage

Brown University
Outline

- Security modeling including access control
- Federal security regulations and standards
- Software vulnerability assessments
- Microsoft’s Security Development Lifecycle
- Introduction to Threat Analysis
- Security can be violated even if code is perfect
Policy, Models and Trust

• To have secure systems, engineers need
  – Clear security goals
  – Effective implementation strategy

• A security policy puts constraints on actions that can taken by actors on objects in the system in order to achieve security goals.
Components of a Security Policy

• **Actors**
  – Individual or group agents interacting with a system.

• **Objects**
  – Informational/computational resources affected by policy.

• **Actions**
  – Possible modifications to objects, e.g. read, edit, copy, remove

• **Permissions**
  – Rules constraining actions that actors may take on objects.

• **Protections**
  – Policy features, e.g. confidentiality, integrity, availability (CIA)
What is a Security Model?

• A **security model** is an abstraction providing **conceptual language** to specify **security policies**.
  – E.g. Unclassified (U), Confidential (C), Secret (S), Top Secret (TS)
  – **Compartments** for sensitive compartmentalized information (SCI), such as human intelligence (HUMINT), satellite observations (GEOINT), signals intelligence (SIGINT)
  – Why are these compartments sensitive?
Two Models of Access Control

• **Discretionary access control**
  – Owner may specify permissions on files
  – A more relaxed form of control

• **Mandatory access control**
  – Administrator fixes permissions in advance.
  – More strict control

• **Rules** have **subjects** (parties requesting access) and **objects** (those things being accessed).
Bell-LaPadula (BLP) Access Control Model

• Next slide describes this slide graphically
• Applies to confidentiality – dates from the 1970s
• Object x and user u have security levels $L(x)$ & $L(u)$
  – Some security levels: Unclass, Class, Secret, TopSecret
• For users u and v, v has higher clearance than u if $L(u) \leq L(v)$. u can pass info to v but not vice versa.

• No “read up” (user can’t see more secure data)
  – User u can read x only if $L(x) \leq L(u)$
• No “write down” (user can’t use more secure data)
  – User u can write to object x only if $L(u) \leq L(x)$. 

Lect 21 4/15/20 © JE Savage
BLP Access Control Model

\[ L(u) \leq L(v) \]

BLP model weakness: only handles confidentiality
Ken Biba (‘77) Access Control Model

• Goal of Biba’s model is to maintain data integrity:
  – i.e. data accuracy & consistency of data over its life-cycle.

• Let $I(x)$ and $I(u)$ be the integrity of user $u$ & object $x$
  – The larger is $I(u)$ or $I(x)$ the more trustworthy is the user $u$ or accurate the object $x$.

• Don’t read from lower integrity level
  – User $u$ can read object $x$ only if $I(u) \leq I(x)$.

• Don’t write to higher integrity level.
  – User $u$ can write to object $x$ only if $I(x) \leq I(u)$.
What is Role-Based Access Control?

• Components: users, roles, permissions, sessions
  – A role is a collection of users.
  – A session is an interaction for a period of time.

• Role hierarchy is defined, as in a corporation.
  – President IsA manager IsA employee
  – Higher role user inherits permissions of lower one
  – When is this not a good idea?
    Should the CEO be allowed to fix to an IT problem?

• Role constraints may be imposed
  – Example: avoid conflicts of interest.
Early USG Security Standards

  – Division A: system has a formal process for verification of security
  – Division B: mandatory access control
  – Division C: discretionary access control
  – Division D: minimal protection criteria
Newer USG Security Standards

- **Common Criteria** for Information Technology Security Evaluation – an **ISO standard**
  - It subsumes the Orange Book
  - Defines key concepts related to security evaluations
  - Framework for documenting security goals
  - Not a certification vouching for product security.
USG Regulations

• HIPAA (1996)
  – Sets privacy standards on patient records for healthcare providers and employers.

• Family Educational Rights and Privacy Act (FERPA) (‘74)
  – Requires protection of privacy of educational records in US

• Federal Information Security Management Act (FISMA)
  – Revised in 2014 – regulates government information security.
  – It requires federal agencies to implement processes and controls designed to ensure the confidentiality, integrity, and availability of system-related information.
  – Must follow FISMA and NIST standards, and legislative requirements, such as the Privacy Act of 1974.
Software Vulnerability Assessment

• The problem: software can be enormous
  — Mac OS X 10.4 has > 86 million lines of code!
  — Code can have both performance & security bugs

• “A vulnerability is a security exposure that results from a product weakness ... the product developer did not intend to introduce and should fix once it is discovered.” — Microsoft definition
How Many Errors are Tolerable?

• How many errors per 1,000 lines of code (KLOC)?
  – Estimates vary from 15-50 defects per KLOC
  – MSFT gets bug density of ½ bug/KLOC in production*

* https://labs.sogeti.com/how-many-defects-are-too-many/
Types of Vulnerability Assessment

• **Black-box analysis**
  – Penetration test (pentest) done without knowledge of innards.
  – Pentests look for security vulnerabilities

• **White-box analysis**
  – Same but with full knowledge of hardware/software, network environment, etc.
Code Analysis for Privacy/Security

• Goal: **Find and remove privacy/security hazards.**

• Good analysis requires training and investment
  – Software engineers generally need education on this.
  – Microsoft’s Security Development Lifecycle (SDL) represents a big step forward.

• Benefits: Improved security, privacy and reliability.
Two Approaches to Code Analysis

• **Static code analysis** studies source code, that is, the text of programs.
  – This is an example of white-box testing.

• **Dynamic analysis** examines running programs.
Components of Static Code Analysis*

• **Data Flow Analysis**
  – Does analysis of basic blocks *(next)*, sections of code in which control stays within a block during execution

• **Control flow graph (CFG)**
  – Shows all possible control paths, i.e. paths through code

• **Taint analysis**
  – A variables touched by a user is “tainted.”
  – How do tainted variables affect the CFG and actions?

* [https://www.owasp.org/index.php/Static_Code_Analysis](https://www.owasp.org/index.php/Static_Code_Analysis)
Example of a Basic Block

```php
$a = 0;
b = 1;

if ($a == $b)
{
    # start of block
    echo "a and b are the same";
}
# end of block
else
{
    # start of block
    echo "a and b are different";
}
# end of block
```

Note: # starts a comment
Modeling System Threats

- *Data flow analysis* is preferable to focusing on assets or studying motivations of attackers.
- Group components by trust boundaries, 3 below.
Control Flow Graph

node 1

node 2

node 3

node 4

node 5

node 6

node 7

node 8

node 9

© JE Savage
Dynamic Code Analysis

• Good analysis explores all important paths
  – Good choice of test data will exercise more paths
  – Incomplete testing can result in catastrophic failure
• Fuzzing can reveal hidden errors. What is it?
  – Run code on virtual machine – no damage from crash
  – Try inputs of length 1, 2, 3, … until crash, maybe
  – Note: malware may detect it is being run in a virtual machine and not exhibit its malicious behavior
2018 Verizon Data Breach Report*

Source: 444 \( \times 10^6 \) malware detections
Organizations: 130,000
22 = Median number malwares/org/year

Information Security Attributes
The CIA Triad

- **Confidentiality**: Access to information is limited to those with proper authorization.

- **Integrity**: Maintaining the consistency, accuracy and trustworthiness of data during its life cycle.

- **Availability**: Reliable access is maintained to resources by authorized parties.
Microsoft’s Security Development Lifecycle (SDL)

• **Bill Gates** inaugurated Microsoft’s **Trustworthy Computing Initiative** in **2002**.
  – Success with major new corporate initiatives often requires support from top management.

• **Every product** that impacts privacy or may be used by children needs security analysis.
  – This means almost all hardware/software products

• **Microsoft code** is now among the most secure!
Less Costly to Fix Defects Early*

* https://www.researchgate.net/figure/255965523_fig1_Figure-3-IBM-System-Science-Institute-Relative-Cost-of-Fixing-Defects
Microsoft’s SDL

- Personnel must be trained.
- Security requirements, risk assessment needed.
- Threat modeling (STRIDE) will reduce attack surface.
  - Spoofing, Tampering, Repudiation, Information Disclosure, Denial of service, Elevation of privilege.
- Implementation requires good tools to protect against attacks.
- Plan to handle errors found after release of code.
- Verification needed via dynamic analysis including fuzzing.
STRIDE Threats*

- S – Spoofing
- T – Tampering
- R – Repudiation
- I – Information Disclosure
- D – Denial of Access or Service
- E – Elevation of Privilege

* Microsoft’s mnemonic for types of software threats
STRIDE Explained

- **S** – pretending to be another person or thing
- **T** – modifying something one should not
- **R** – falsely claiming not to have taken an action
- **I** – exposing information to those unauthorized
- **D** – denying users access to a service
- **E** – acquiring access at an elevated level
# STRIDE Elaborated

<table>
<thead>
<tr>
<th>Threats</th>
<th>Objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Spoofing</strong></td>
<td>Authenticity</td>
</tr>
<tr>
<td><strong>Tampering</strong></td>
<td>Integrity</td>
</tr>
<tr>
<td><strong>Repudiation</strong></td>
<td>Non-Repudiation</td>
</tr>
<tr>
<td><strong>Information Disclosure</strong></td>
<td>Confidentiality</td>
</tr>
<tr>
<td><strong>Denial of Service</strong></td>
<td>Availability</td>
</tr>
<tr>
<td><strong>Elevation of Privilege</strong></td>
<td>Proper Authorization</td>
</tr>
</tbody>
</table>
Four Ways to Address Threats (META)

• Mitigate it
  – Increase the work to exploit it
• Eliminate it
  – Usually requires elimination of features
• Transfer it
  – Let some other system element cope with it
• Accept it
  – Risk acceptance may be less costly than other steps
Source Material

Is Open Source Software a Panacea?

• Software is available for modification under liberal copyright policy.
• Do many eyeballs on the code make it secure?
  – “... in reality that doesn’t happen” Cowan 2002.
• Russia believes it – avoids US software.
  – Putin orders Russian government to move to Open Source Software by 2015. (12/28/2010)
• Problems: No incentive to find bugs. Coders not trained to find them. It is hard!

Bug-Free Software Can Be Exploited!* 

• Talk: **Software Exploitation: Hardware is the New Black** by Cristiano Giuggrida, MIT, 11/5/08
  
  – “Verified bug-free software can be exploited by a relatively low-effort attacker.”
  
  – “[S]tate-of-the-art security defenses, which have proven useful to raise the bar against traditional software exploitation techniques, are completely ineffective against such attacks.”

• Surprisingly, some operating systems vulnerable

* [https://www.csail.mit.edu/event/software-exploitation-hardware-new-black](https://www.csail.mit.edu/event/software-exploitation-hardware-new-black)
Exploitation of Bug-Free Software

• “Dedup Est Machina: Memory Deduplication as an Advanced Exploitation Vector” by Cristiano Giuffrida †
  – Uses Rowhammer to changes bits in memory
    • See next slide
  – Exploits de-duplication to obtain side information
  – Allows user to obtain gain arbitrary read/write access to memory

† https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=7546546
Row-Hammer Attack

• De-duplication
  – Space saved by holding only one copy of a file
  – If one copy changed, a new copy created, which takes time
  – This provides one bit of side information!
  – Which is enough to violate security (too hard to explain)
Review

- Security modeling including access control
- Federal security regulations and standards
- Software vulnerability assessments
- Microsoft’s Security Development Lifecycle
- Introduction to Threat Analysis
- Security can be violated even if code is perfect