Challenges in Developing Secure Software

Aspen Olmsted, College of Charleston, USA
Stefan Schauer, AIT Austrian Institute of Technology GmbH, Austria
Lidia Prudente Tixteco, Instituto Politecnico Nacional, Mexico
George Yee, Carleton University, Canada
Hans-Joachim Hof, Technical University of Ingolstadt, Germany
What is SSD

The practice used when developing software aimed at minimizing the application’s vulnerabilities to threats.
Types of threats

- Malicious User Penetration
  - Avoiding authorization rules
  - Gaining access to trusted resources
  - Avoiding licensing

- Denial of Service

- Unavailability due to application crash

- Unavailability due to application partition

- Data integrity violation
Do we have the correct SDLC for secure software?
Do we have the correct programming paradigms for secure software development?

- Procedural
- Object Oriented
- Functional
- Declarative
Do we value security in the requirements phase?

- Are non-functional requirements 1st class citizens?
- Do we have the ability to model non-functional requirements?
Do we have the correct social norms for secure software development?

- Teach me to Trust
- Trust No-One
Panel Discussion

Challenges in Developing Secure Software

Hans-Joachim Hof
hof@insi.science
http://insi.science

INSicherheit – Ingolstadt Research Group Applied IT Security,
CARISSMA – Center of Automotive Research
Technical University of Ingolstadt
The Internet of Things

- Connecting millions of embedded devices to the Internet to gain new insights, save costs

- However:
  - High cost pressure on consumer devices ➔ security often ignored
  - Slow innovation cycles on many other devices ➔ security not included yet, takes long to change things

- Observation: history repeats itself, vulnerabilities from the 90ies have a renaissance, perimeter protection is back, ...
The Internet of Malicious Things

- 2016 Dyn Cyberattack by Mirai Botnet (>620 Gbit/s)

The Internet of Malicious Things

- 2016 Dyn Cyberattack by Mirai Botnet (>620 Gbit/s)

The Internet of Malicious Things

- 2016 Dyn Cyberattack by Mirai Botnet (>620 Gbit/s)

The Internet of Malicious Things

- 2016 Dyn Cyberattack by Mirai Botnet (>620 Gbit/s)

- IoT devices in a botnet
  - admin/123456
  - root/anko
  - root/S4321
  - root/Zte521

- Provider of core services (DNS)
- Dyn not avail.
- Dyn not avail.
- Dyn not avail.

- Users of core services
  - Amazon
  - Twitter
  - Spotify

The Internet of Malicious Things

- 2016 Dyn Cyberattack by Mirai Botnet (>620 Gbit/s)

Challenges in Embedded Systems Software Security

- Non-functional requirement on device with limited resources/budget
  → security things get “optimized”:
  - Use of weak crypto (encryption algorithm, block mode, initialization vector)
  - Key management weaknesses
  - Disabled security checks (e.g., signature check of firmware update)
  - Treating not-protected (integrity/confidentiality) information as security information (e.g., version control by filename, treating a public ID as secret)

- Not enough reuse of security solutions (“you are not THAT special”)
  - Individual hardware for special use case hinders reuse
  - A trend to build own security functions (e.g., to avoid TLS)

- You need to get it as right as possible in the first try
  - Deploying updates is hard/impossible with some embedded devices
Challenges in Embedded Systems Software Security

- Special design issues
  - Safety vs. Security
  - Keeping secrets used in M2M secret ("there are no secrets in hardware")
  - Administrative access for field engineers
  - Firmware updates

- Security still not understood
  - Security as an afterthought (...see automotive penetration testing)
  - Default passwords, secret keys in firmware, secret keys from SDL documentation, ...

Biggest issues: security education/guidance + encourage having respect for security issues
Topic: Challenges in Developing Secure Software

Discussion:

• Is it possible to develop secure software?
• Does the IoT promote or hinder the development of secure software?
Is It Possible to Develop Secure Software?

• **What do we understand by “secure software”?**

  - Attackers believe that they have something to gain by attacking the software, e.g. data, notoriety
  - Attackers identify an associated vulnerability and attack the vulnerability
  - Secure SW is able to defend against these attacks and still function as intended
Is It Possible to Develop Secure Software?

• Adversary Model
  • Resources
  • Access
  • Risk tolerance
  • Objectives

• Successful Attack
  • Diagnose system to identify an attack
  • Gain necessary access
  • Execute the attack

---

Is It Possible to Develop Secure Software?

• Vulnerabilities
  • Allow confidentiality, integrity, or availability to be compromised
  • Allow attacks to be identified
  • Allow access
  • Estimates the level of security
    • More vulnerabilities means lower security
    • More secured vulnerabilities means higher security
Is It Possible to Develop Secure Software?

- Critical phases of software development for baking in security

To develop secure software, it is essential to find vulnerabilities during design and coding, secure those vulnerabilities, and test to ensure that the vulnerabilities have truly been secured.
Is It Possible to Develop Secure Software?

- First, the bad news: developing truly secure software is very difficult and maybe impossible
  - Existence of unknown vulnerabilities
  - New side effect vulnerabilities
  - Ineffectiveness of securing vulnerabilities, e.g. attackers finding ways to defeat the security
  - Lack of tools that allow developers to better code for security, e.g. a programming language with security constructs
  - Lack of OS support for security
  - No software warranty, i.e. no accountability
  - Insufficient financial resources and time
  - ...
Is It Possible to Develop Secure Software?

- A little good news: tools and technologies are improving, security more in the public mind
  - Static code analyzers
  - Secure coding practices
  - Research on finding vulnerabilities
  - Companies get sued for privacy breaches, e.g. Equifax (unpatched flaw in open source SW)
  - Greater public realization of the need for security

No, for truly secure; maybe, for acceptable risks
Does the IoT promote or hinder the development of secure software?

- Network picture of IoT in a building

  Links can be wireless or physical cable

- IoT devices characterized as
  - Large variety of devices
  - Connected to the Internet
  - Some with relatively lower computation power, e.g. wearables
  - Some with low electrical energy requirements, e.g. sensors
Does the IoT promote or hinder the development of secure software?

- **The case for “hinder”**
  - Lower processing power may mean that some current security measures are not usable, e.g. encryption and decryption
  - Large variety of devices in a local area may invite applications involving inter-device communication, which lead to more vulnerabilities, e.g. home management
  - Large number of devices in a given area will lead to a larger number of vulnerabilities in the area, i.e. software will really need to be secure making it harder to develop
  - The newness of the technology may call for new software that have vulnerabilities not imagined before
  - The newness of the technology means that security is even lower in the mindset than in traditional areas of software development, i.e. devices have no consideration for security
Does the IoT promote or hinder the development of secure software?

- The case for “promote”
  - Devices with relatively lower computation power may mean relatively simpler software needed, which should make it easier to identify vulnerabilities in the software
  - Popular IoT applications involve daily living (e.g. home management, health monitoring) for which security breaches would be “in your face”. This may increase pressure for having secure software

**Hinder**
Panel

Challenges in Developing Secure Software

Lidia Prudente Tixteco
lprudente@ipn.mx
México
Ideas

Although the awareness of development of secure software is growing, many developments do not include security principles.

There is a false belief that Firewalls, IDS, and VPNs protect applications.

Software manufacturers are more concerned with releasing new systems than with ensuring their security.
## Security Checks Along SDLC

<table>
<thead>
<tr>
<th>Vulnerability Checks</th>
<th>SDLC Phases</th>
<th>Maturity of Tools, Practices</th>
<th>Injected Vulnerabilities (Not Necessarily Security)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vulnerabilities in requirements, business processes flow, algorithms</td>
<td>Analysis</td>
<td>Embryonic</td>
<td>15%</td>
</tr>
<tr>
<td>Vulnerabilities caused by interrelations of modules and (Web) services, logic and data flow</td>
<td>Design</td>
<td>Embryonic</td>
<td>40%</td>
</tr>
<tr>
<td>Vulnerabilities in language instructions, implementation of logic and data flow</td>
<td>Construct</td>
<td>Low</td>
<td>35%</td>
</tr>
<tr>
<td>Vulnerabilities in executables, UI. Assembly of secure services could be insecure</td>
<td>Testing</td>
<td>Low</td>
<td>10%</td>
</tr>
<tr>
<td>Missing patches, administrative errors, misconfiguration. If vulnerability found — back to analysis</td>
<td>Operations</td>
<td>Low-Medium</td>
<td></td>
</tr>
</tbody>
</table>
How is software development taught?

1. Functional Requirements
2. Non-Functional Requirements
3. Project Requirements
4. Stakeholders Requirements
5. Security Requirements
6. Security Functional Requirements
7. Security Guaranty Requirements
## NIST SP 800-64

<table>
<thead>
<tr>
<th>SDLC</th>
<th>Initiation</th>
<th>Acquisition / Development</th>
<th>Implementation</th>
<th>Operations / Maintenance</th>
<th>Disposition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Needs Determination:</td>
<td>- Functional Statement of Need</td>
<td>- Installation</td>
<td>- Performance measurement</td>
<td>- Appropriateness of disposal</td>
</tr>
<tr>
<td></td>
<td>- Perception of a Need</td>
<td>- Market Research</td>
<td>- Inspection</td>
<td>- Contract modifications</td>
<td>- Exchange and sale</td>
</tr>
<tr>
<td></td>
<td>- Linkage of Need to Mission and Performance Objectives</td>
<td>- Feasibility Study</td>
<td>- Acceptance testing</td>
<td>- Operations</td>
<td>- Internal organization screening</td>
</tr>
<tr>
<td></td>
<td>- Assessment of Alternatives to Capital Assets</td>
<td>- Requirements Analysis</td>
<td>- Initial user training</td>
<td>- Maintenance</td>
<td>- Transfer and donation</td>
</tr>
<tr>
<td></td>
<td>- Preparing for investment review and budgeting</td>
<td>- Alternatives Analysis</td>
<td>- Documentation</td>
<td></td>
<td>- Contract closeout</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Cost-Benefit Analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Software Conversion Study</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Cost Analysis</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Risk Management Plan</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Acquisition Planning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SECURITY CONSIDERATIONS</td>
<td>- Security Categorization</td>
<td>- Risk Assessment</td>
<td>- Inspection and Acceptance</td>
<td>- Configuration Management</td>
<td>- Information Preservation</td>
</tr>
<tr>
<td></td>
<td>- Preliminary Risk Assessment</td>
<td>- Security Functional Requirements Analysis</td>
<td>- System Integration</td>
<td>- and Control</td>
<td>- Media Sanitization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Security Assurance Requirements Analysis</td>
<td>- Security Certification</td>
<td>- Continuous Monitoring</td>
<td>- Hardware and Software Disposal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Cost Considerations and Reporting</td>
<td>- Security Accreditation</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Security Planning</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Security Control Development</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Developmental Security Test and Evaluation</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>- Other Planning Components</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CHALLENGES IN DEVELOPING SECURE SOFTWARE

Panel Discussion, SECURWARE 2017
Rome, September 12th 2017

Stefan Schauer, AIT
PERSONAL INTRODUCTION

• **Affiliation**
  • AIT Austrian Institute of Technology
  • Center for Digital Safety & Security
  • Secure Communication Technologies Group

• **Scientific Background**
  • Master in computer science (IT security)
  • PhD in theoretical physics (quantum cryptography)

• **Current Research**
  • Risk and security management for critical infrastructures (CIs)
  • CI interdependencies and assessment of cascading effects
  • Game theoretic approaches for risk management
IMPACT OF INSECURE SOFTWARE

• Security needs to be an integral part of software development
  • IT systems (and software) influences our life in multiple different ways (communication, transport, government, personal data, …)
  • In many fields security is only a by-product or add-on to the developed IT systems
  • Several approaches towards Security by Design are present and need to be integrated from the start

• Flaws and errors in software open doors for attacks
  • Software vulnerabilities are mostly due to error-prone implementation
  • Flaws in software can be used to create unexpected and malicious behavior
IMPACT OF INSECURE SOFTWARE

• IT systems are misused by malicious parties
  • Botnets are created and DDoS attacks are using thousands of IoT devices

• Systems get hacked and encrypted
  • Crypto ransomware like WannaCry and Petya creates data loss and stops the operation of several important services

• Attackers get control of highly-relevant systems or information
  • Electrical power system gets shut down by attackers in Ukraine

  The price for developing secure software might be small, the potential impacts of error-prone systems can be severe!
LET’S START THE DISCUSSION

Dr. Stefan Schauer
Center for Digital Safety & Security
Austrian Institute of Technology
Klagenfurt, Austria
stefan.schauer@ait.ac.at