Cybersecurity is a lifecycle issue

Sustainment

Engineering and Development

Requirements and Acquisition

Mission Thread
Threat Analysis
Abuse Cases
Architecture and Design Principles
Coding Rules and Guidelines
Testing, Validation and Verification
Monitoring
Breach Awareness

Deployment and Operations
Cross lifecycle issues

- Mission Thread
- Threat Analysis
- Abuse Cases
- Architecture and Design Principles
- Coding Rules and Guidelines
- Testing, Validation and Verification
- Monitoring
- Breach Awareness

Sustainment

Engineering and Development

Requirements and Acquisition

Automation (DevOps)

Deployment and Operations

Metrics, Models, and Measurement

Building skills (Workforce development)
Cross lifecycle issues

- Mission Thread
- Threat Analysis
- Abuse Cases
- Architecture and Design Principles
- Coding Rules and Guidelines
- Testing, Validation and Verification
- Monitoring
- Breach Awareness

- Automation (DevOps)
- Metrics, Models, and Measurement
- Building skills (Workforce development)
- Procurement / Acquisition (Supply chain)
Conventional view of supply chain risk

Supply chains also maintain product properties

Cold Chain

A cold chain is a temperature-controlled supply chain. An unbroken cold chain is an uninterrupted series of storage and distribution activities which maintain a given temperature range.

Software is the new hardware – IT

IT moving from specialized hardware to software, virtualized as

- Servers: virtual CPUs
- Storage: SANs
- Switches: Soft switches
- Networks: Software defined networks
- Communications: Software defined radios
Software is the new hardware – cyber physical

- **Cellular**
  - Main processor
  - Graphics processor
  - Base band processor (SDR)
  - Secure element (SIM)

- **Automotive**
  - Autonomous vehicles
  - Vehicle to infrastructure (V2I)
  - Vehicle to vehicle (V2V)

- **Industrial and home automation**
  - 3D printing (additive manufacturing)
  - Autonomous robots
  - Interconnected SCADA

- **Aviation**
  - Next Gen air traffic control
  - Fly by wire

- **Smart grid**
  - Smart electric meters
  - Smart metering infrastructure

- **Embedded medical devices**
Mission function is increasingly delivered in software

“The [F-35] aircraft relies on more than 20 million lines of code to "fuze" information from the JSF's radar, infrared cameras, jamming gear, and even other planes and ground stations to help it hunt down and hide from opponents, as well as break through enemy lines to blow up targets on the ground. ... But if the computer doesn't work, the F-35's greatest advertised advantages over existing rivals and future threats would suddenly become moot.”

The Week, 2016

Vehicle technology following the same path

2010 Jeep Cherokee
(12 ECUs)

2014 Jeep Cherokee
(32 ECUs)

Common assertion that modern high end vehicles have
- Over 100M lines of code
- Over 50 antennas
- Over 100 ECUs

Software is the new hardware – everything

90 percent of [Samsung’s] products -- which includes everything from smartphones to refrigerator-- would be able to connect to the Web by 2017. In five years, every product in the company's entire catalog would be Internet connected.

B.K. Yoon, Samsung co-CEO

CNET
Jan 5, 2015

Software is eating the world.
Marc Andreessen, WSJ, Aug 20, 2011

http://www.wsj.com/articles/SB10001424053111903480904576512250915629460
Evolution of software development – 1960s

Custom development – context:
- Software was limited
  - Size
  - Function
  - Audience
- Each organization employed developers
- Each organization created their own software

Supply chain: practically none
Evolution of software development – 1970s

Shared development – ISVs (COTS) – context:
- Function largely understood
  - Automating existing processes
- Grown beyond ability for using organization to develop economically
- Outside of core competitiveness by acquirers

Supply chain: software supplier
Evolution of software development – 1990s

Development is now assembly using collective development

- Too large for single organization
- Too much specialization
- Too little value in individual components

Supply chain: long

Note: hypothetical application composition
Evolution of software development – the rise of open source

- 90% of modern applications are assembled from 3rd party components
- Most applications are now assembled from hundreds of open source components, often reflecting as much as 90% of an application
- At least 75% of organizations rely on open source as the foundation of their applications

Distributed development – context:
- Amortize expense
- Outsource non-differential features
- Lower acquisition (CapEx) expense

Supply chain: opaque

Sources: Geer and Corman, “Almost Too Big To Fail,” ;login: (Usenix), Aug 2014; Sonatype, 2014 open source development and application security survey
Evolution of software development – the rise of open source

- 90% of modern applications are assembled from 3rd party components
- At least 75% of organizations rely on open source as the foundation of their applications
- Most applications are now assembled from hundreds of open source components, often reflecting as much as 90% of an application

Sources: Geer and Corman, “Almost Too Big To Fail,” ;login: (Usenix), Aug 2014; Sonatype, 2014 open source development and application security survey

“Developers are gorging themselves on an ever-expanding supply of open source components”

Sonatype, “2016 State of the Software Supply Chain”

Supply chain: opaque

Distributed development – context:
- Amortize expense
- Outsource non-differential features
- Lower acquisition (CapEx) expense

Sources: Geer and Corman, “Almost Too Big To Fail,” ;login: (Usenix), Aug 2014; Sonatype, 2014 open source development and application security survey
Open source supply chain has a long path
Today: Software supply chain for assembled software

Expanding the scope and complexity of acquisition and deployment

Visibility and direct controls are limited (only in shaded area)

Corruption along the supply chain is easy

Unexpected or unintended behaviors in components

Knowledgeable analysts can convert packaged binary into malware in minutes

Sources: Pedro Candel, Deloitte CyberSOC Academy, Deloitte
http://www.8enise.webcastlive.es/webcast.htm?video=08; http://www.microsoft.com/Products/Games/FSInsider/freeflight/PublishingImages/scene.jpg;
Corruption in the tool chain already exists

- XcodeGhost corrupted Apple’s development environment
  - Major programs affected
    - WeChat
    - Badu Music
    - Angry Birds 2
    - Heroes of Order & Chaos
    - iOBD2

Sources:
Versions of Android illustrate open source fragmentation

| Device model | GT-I9100 | GT-I9003 | HTC Incredible S | GT-S5360 | GT-S5830 | LG-P500 | LG-P9900 | LG-P9 | ST18i | MT15i | MT DR | MT S | LG-P3 | LG-P9 | MB200 | HTC | GT-I | LG-P9 | R80 | Dell | G | LG-P9 | LG-P9 | LG-P3 | LG | U8 | LG | U20i | M8 | LG | Dell \\
|             |         |         |                  |          |          |         |          |       |       |       |       |       |       |       |       |       |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |
|             |         |         |                  |          |          |         |          |       |       |       |       |       |       |       |       |       |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |      |

Source: http://opensignal.com/reports/fragmentation.php
Open source is not secure

Heartbleed and Shellshock were found by exploitation

Other open source software illustrates vulnerabilities from cursory inspection

Open source is not secure

Heartbleed and Shellshock were found by exploitation.

Other open source software illustrates vulnerabilities from cursory inspection.

1.8 billion vulnerable open source components downloaded in 2015

26% of the most common open source components have high risk vulnerabilities

On average, applications have 22.5 open source vulnerabilities

Reducing software supply chain risk factors

Software supply chain risk for a product needs to be reduced to an acceptable level.

- **Supplier Capability**: Supplier follows practices that reduce supply chain risks.
- **Product Security**: Delivered or updated product is acceptably secure.
- **Product Distribution**: Methods of transmitting the product to the purchaser guard against tampering.
- **Operational Product Control**: Product is used in a secure manner.
Supplier security commitment evidence

Supplier employees are educated as to security engineering practices

• Documentation for each engineer of training and when trained/retrained
• Revision dates for training materials
• Lists of acceptable credentials for instructors
• Names of instructors and their credentials

Supplier follows suitable security design and development practices

• Documented design guidelines
• Has analyzed attack patterns appropriate to the design such as those that are included in Common Attack Pattern Enumeration and Classification (CAPEC)
• Protection against insider (developer) threat
Evaluate a product’s threat resistance

What product characteristics minimize opportunities to enter and change the product’s security characteristics?

• Attack surface evaluation: Exploitable features have been identified and eliminated where possible
  - Access controls
  - Input/output channels
  - Attack enabling applications – email, Web

• Design and coding weaknesses associated with exploitable features have been identified and mitigated (CWE)

• Independent validation and verification of threat resistance

• Dynamic, Static, Interactive Application Security Testing (DAST, SAST, IAST)

• Delivery in or compatibility with Runtime Application Self Protection (RASP) containers
Establishing good product distribution practices

Recognize that supply chain risks are accumulated
• Establish provenance procedures
  - Subcontractor/COTS-product supply chain risk is inherited by those that use that software, tool, system, etc.

Apply to the acquiring organizations and their suppliers
• Require good security practices by their suppliers
• Assess the security of delivered products
• Address the additional risks associated with using the product in their context

Minimize internal suppliers
• Single point of distribution to development community

Ideally open source is built with a compiler you trust
Maintain operational attack resistance

Who assumes responsibility for preserving product attack resistance with product deployment?

- Maintaining inventory of components
- Patching and version upgrades (component lifecycle management)
- Expanded distribution of usage
- Expanded integration

Usage changes the attack surface and potential attacks for the product

- Change in feature usage or risks
- Are supplier risk mitigations adequate for desired usage?
- Effects of vendor upgrades/patches and local configuration changes
- Effects of integration into operations (system of systems)
Steel furnaces have been successfully attacked

“Steelworks compromise causes massive damage to furnace.

One of the most concerning was a targeted APT attack on a German steelworks which ended in the attackers gaining access to the business systems and through them to the production network (including SCADA). The effect was that the attackers gained control of a steel furnace and this caused massive damages to the plant.”

Connecting automotive systems to internet opens system to attack

Extending systems opens vulnerabilities not anticipated
- Optimizations performed assuming one attack method
- Assumptions no longer hold with additional integrations

Studies suggest that new operational environments are a leading cause for introducing new vulnerabilities in existing systems.

What about open source?

Establish a supplier for open source components
Establish a process for tracking open source vulnerabilities
Restrict open source components that can be used
Establish an internal open source component distribution process
Maintain a registry of where open source components are used
Institute an update policy to remediate discovered and patched vulnerabilities

Source: [http://opensource.org/](http://opensource.org/)
Business decisions are about risk

There are many risks to a business process or mission thread

- Within a system
- Collection of systems

Supply chain is one of many risk components

Evaluate software supply chain risk in the larger context of

- Supply chain risk
- System risk
- System of systems risk

SERA: Security Engineering Risk Analysis
Where to start

Anywhere

- No meaningful controls over what components are applications (76%)
- No coordination of security practices in various stages of the development life cycle (81%)
- No acceptance tests for third-party code (47%)

Plenty of models to choose from

- **BSIMM**: Building Security in Maturity Model
- **CMMI**: Capability Maturity Model Integration for Acquisitions
- **PRM**: SwA Forum Processes and Practices Group Process Reference Model
- **RMM**: CERT Resilience Management Model
- **SAF**: Software Assurance Framework
- **SAMM**: OWASP Open Software Assurance Maturity Model
- **O-TTPS**: Open Group Open Trusted Technology Provider™ Standard, Version 1.1

Further reading


Contact Information

Mark Sherman
Technical Director
Cyber Security Foundations
Telephone: +1 412-268-9223
Email: mssherman@sei.cmu.edu

U.S. Mail
Software Engineering Institute
Customer Relations
4500 Fifth Avenue
Pittsburgh, PA 15213-2612
USA

Web
www.sei.cmu.edu
www.sei.cmu.edu/contact.cfm

Customer Relations
Email: info@sei.cmu.edu
Telephone: +1 412-268-5800
SEI Phone: +1 412-268-5800
SEI Fax: +1 412-268-6257