Growing Risks in the Software Supply Chain

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Mark Sherman, Ph.D.
Technical Director, CERT

Platform Security Summit 2019
Oct 3, 2019

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM19-0965

The SEI is a DoD R&D Federally Funded Research and Development Center

Established in 1984 at Carnegie Mellon University

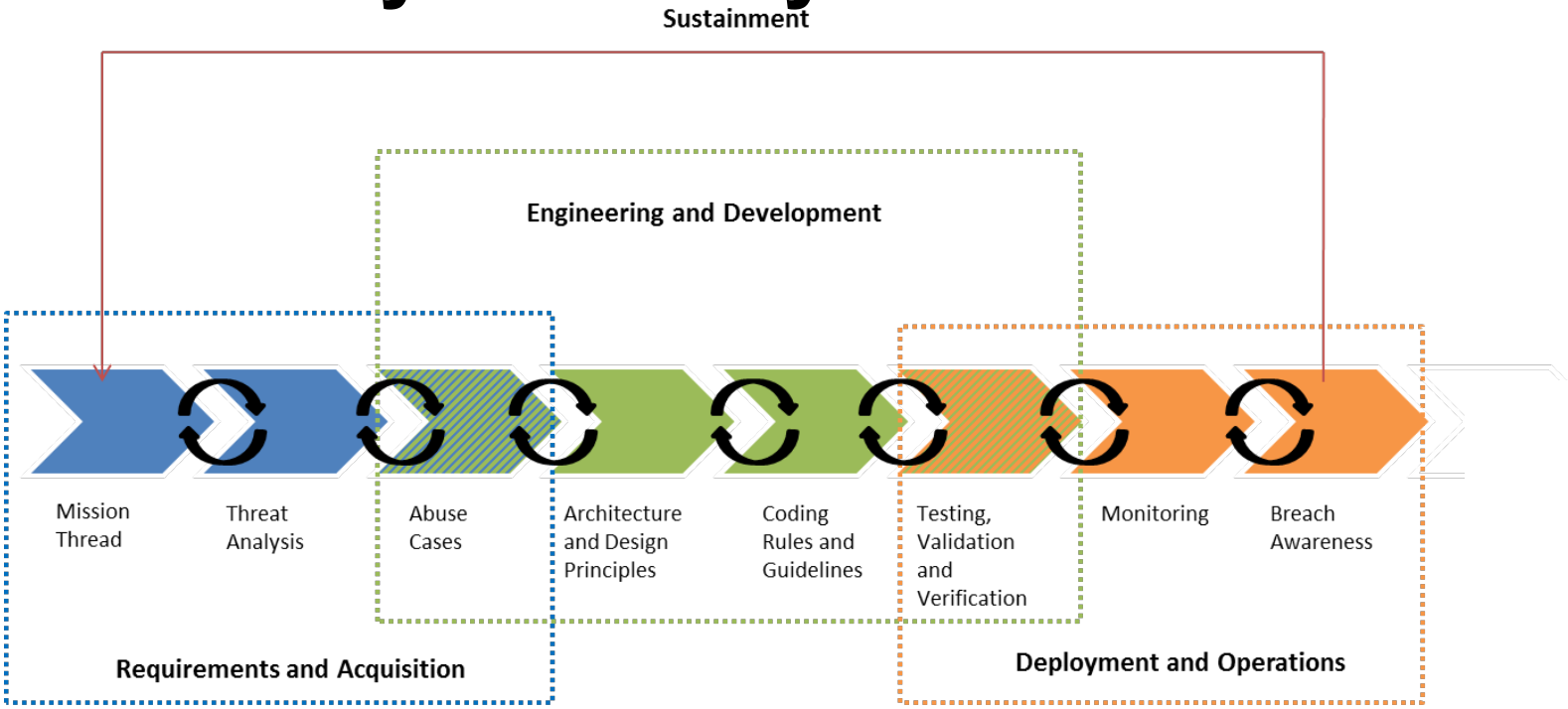
~700 employees (ft + pt), of whom about 70% are engaged in technical work

Initiated CERT cybersecurity program in 1988

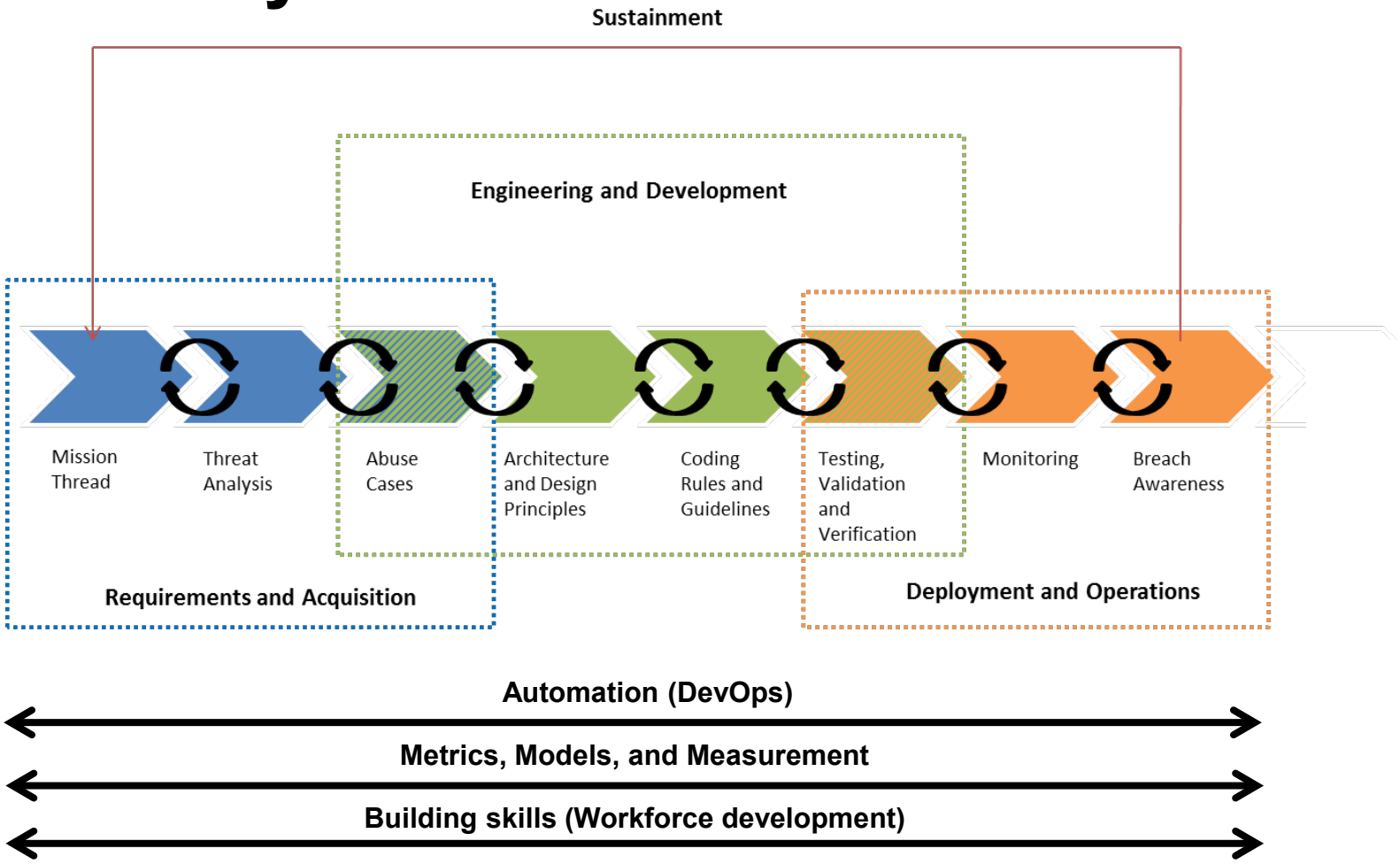
Offices in Pittsburgh and DC, with several locations near customer facilities

~$145M in annual funding

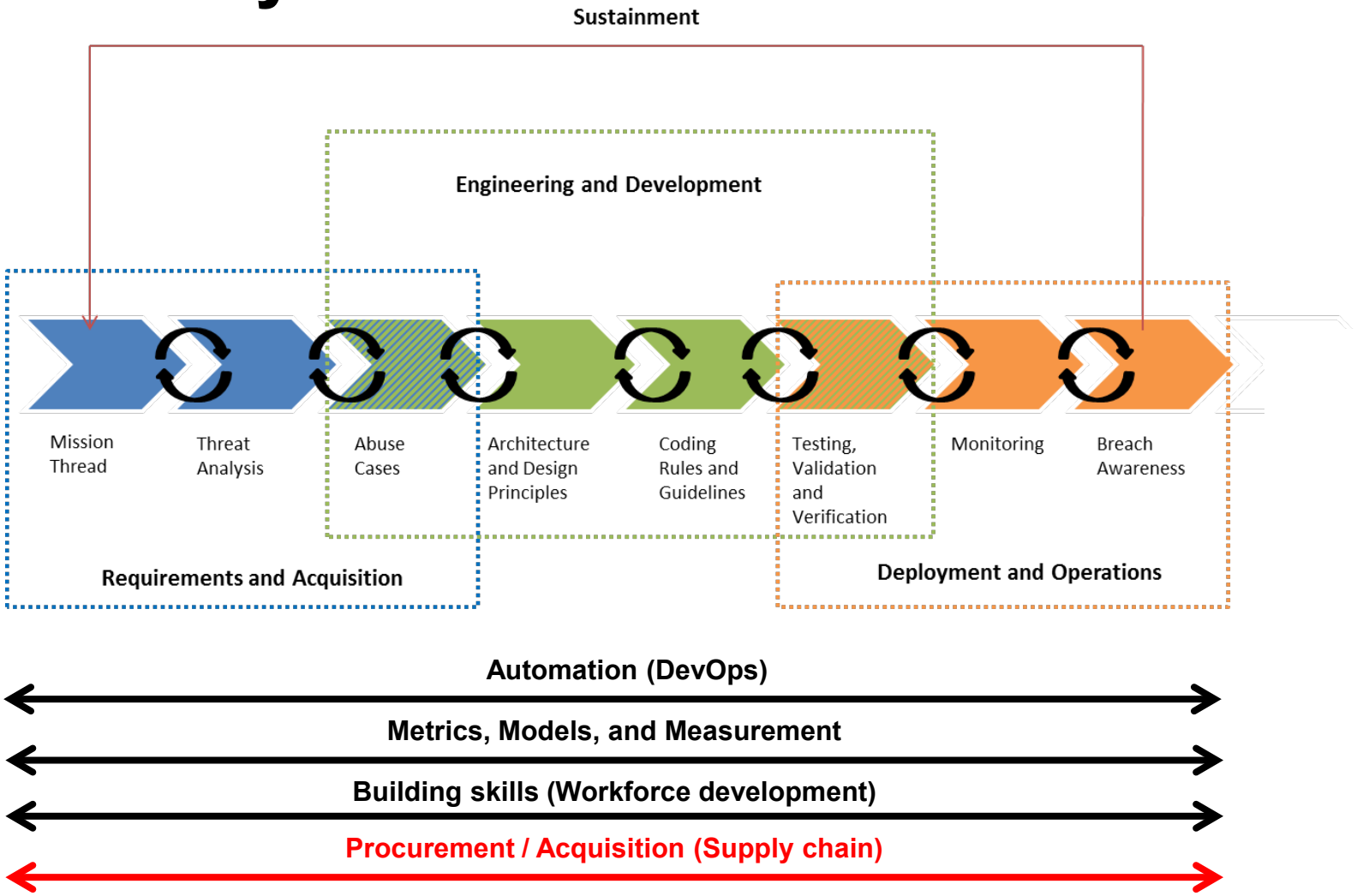
Cybersecurity is a lifecycle issue



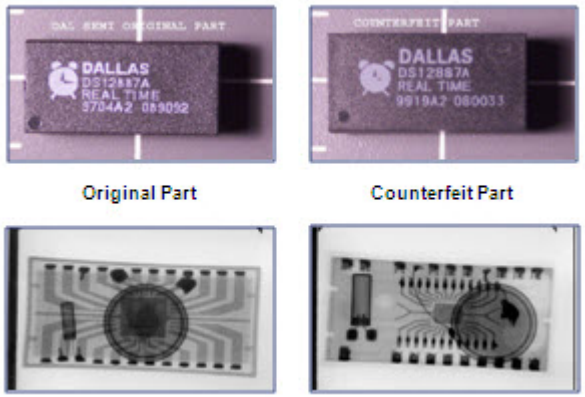
Cross lifecycle issues

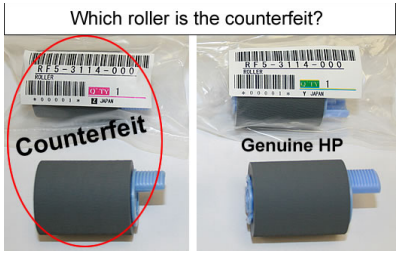


Cross lifecycle issues



Conventional view of supply chain risk





Sources: http://www.nytix.com/NewYorkCity/articles/handbags.html; http://www.laserwisetech.co.nz/secret.php; http://www.muscatdaily.com/Archive/Oman/Fake-car-parts-contribute-to-rise-in-road-accidents-Experts; http://www.andovercg.com/services/cisco-counterfeit-wic-1dsu-t1.shtml; http://unites-systems.com/l.php?id=191

Supply chains 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.

Source: Wikipedia, https://en.wikipedia.org/wiki/Cold_chain

Value chains add value at each step

Value chain

The idea of the value chain is based on the process view … seeing a manufacturing (or service) organization … made up of subsystems each with inputs, transformation processes and outputs.

Source: https://en.wikipedia.org/wiki/Value_chain

Evolution of software development

Custom development – context:

- Software was limited
  - Size
  - Function
  - Audience
- Each organization employed developers
- Each organization created their own software

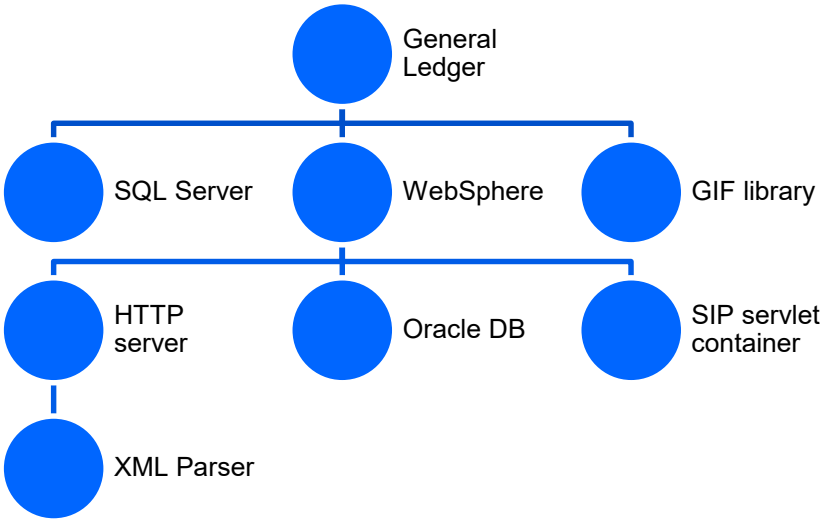
Supply chain: practically none

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

Development is now assembly



Collective development – context:

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

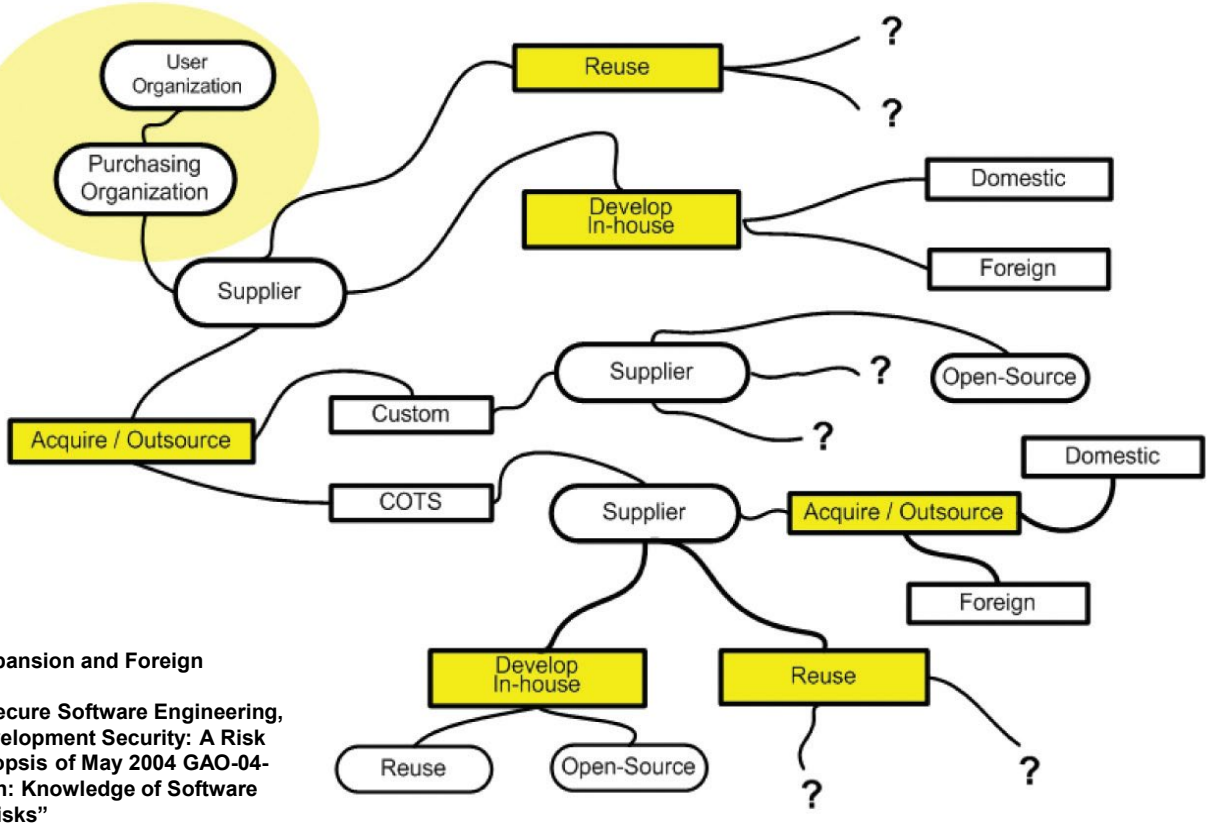
Supply chain: long

Note: hypothetical application composition

Software supply (value) chain (tree) for assembled software

Expanding the scope and complexity of acquisition and deployment

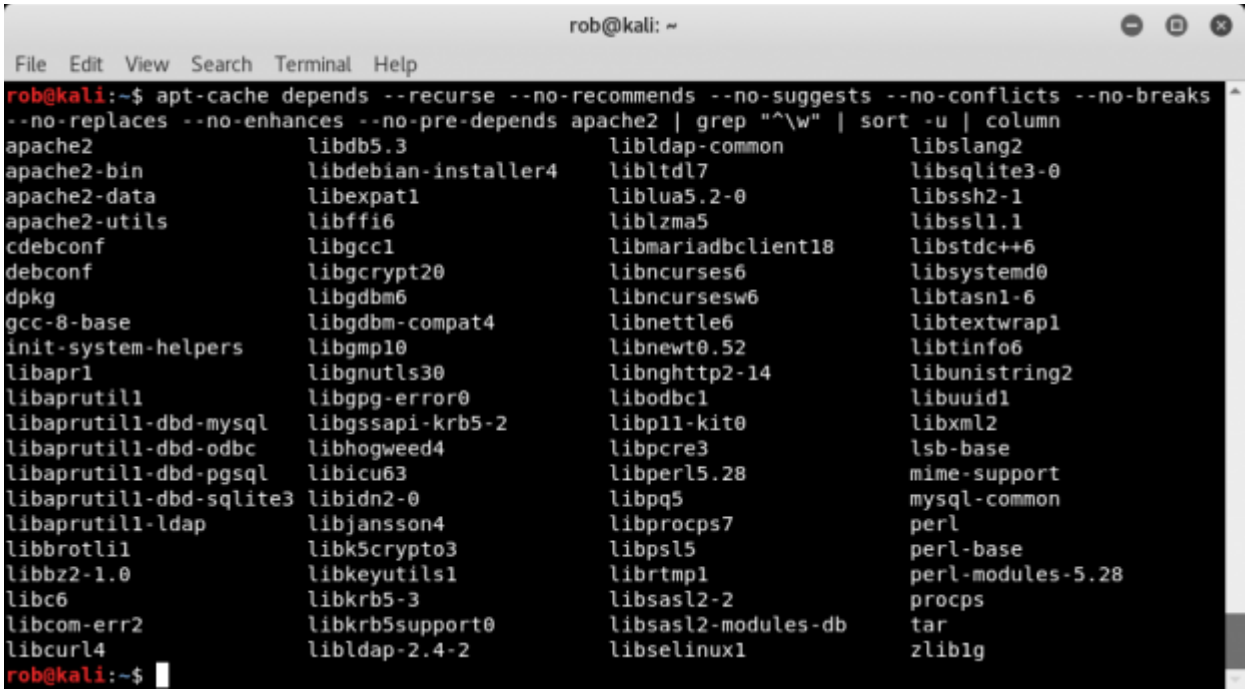
Visibility and direct controls are limited (only in shaded area)



Source: "Scope of Supplier Expansion and Foreign Involvement" graphic in DACS www.softwaretechnews.com Secure Software Engineering, July 2005 article "Software Development Security: A Risk Management Perspective" synopsis of May 2004 GAO-04-678 report "Defense Acquisition: Knowledge of Software Suppliers Needed to Manage Risks"

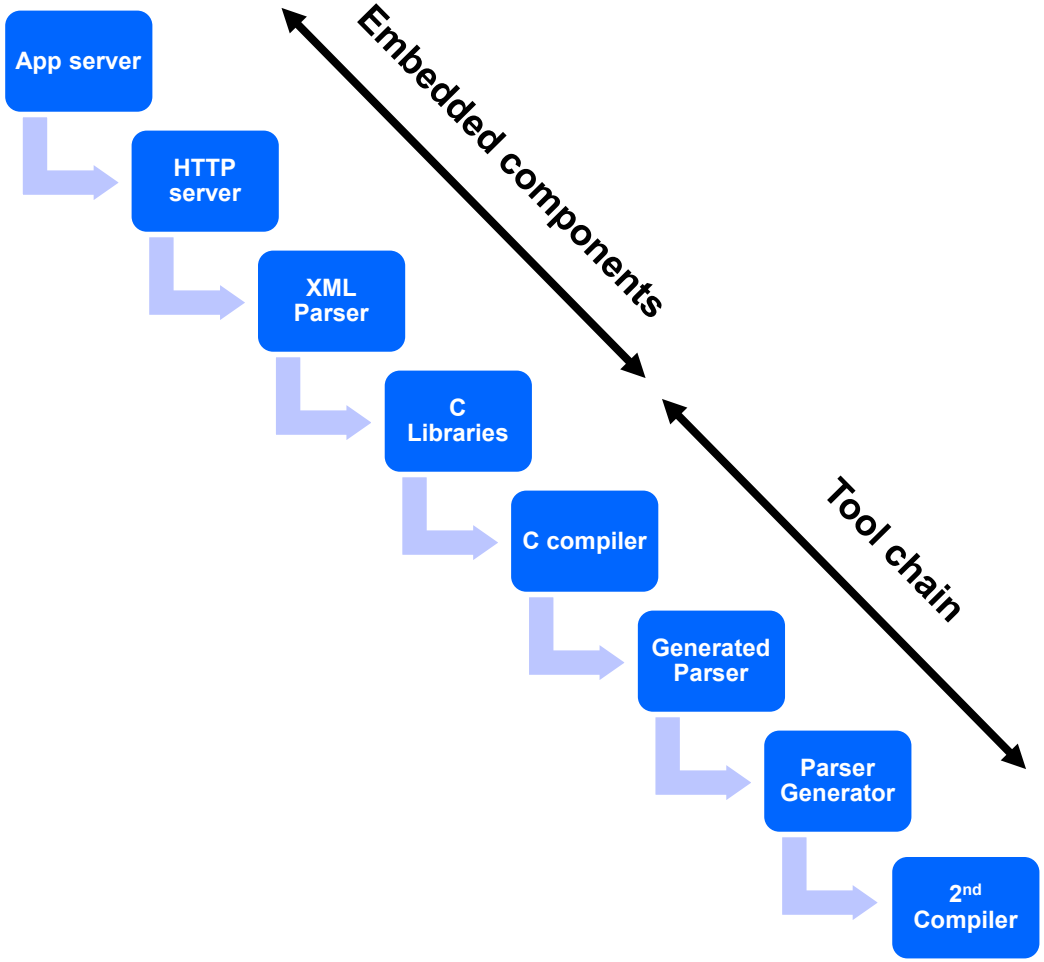
Supply chain breadth: Assembly – Apache Example

“First level dependencies of the Apache web server”

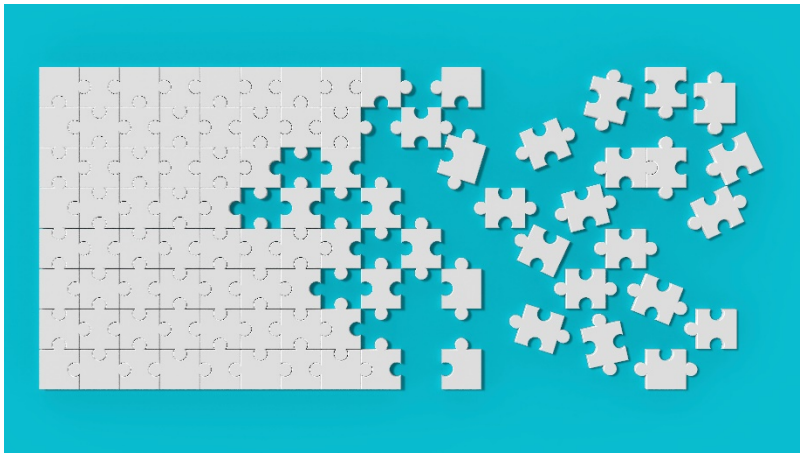


From: Rob Graham, Software Bill of Materials (SBoM) - Does It Work for DevSecOps?, Jan 14, 2019, https://www.alienvault.com/blogs/security-essentials/software-bill-of-materials-sbom-does-it-work-for-devsecops

Supply chain depth: supply chain has a long path



Large number of components in assembled software



Sontaype:

- 85% of modern applications are assembled from open source components; can be as high as 97% for web applications
- Average has 460 components; some applications had 2,000-4,000 OSS

Gonzalez, et al:

- Applications contain over 80% of common code; Unique code only represents 5% of all code

Sources: Sonatype, “2019 State of the Software Supplu Chain”, https://www.sonatype.com/hubfs/SSC/2019%20SSC/SON_SSSC-Report-2019_jun16-DRAFT.pdf; H. Gonzalez, N. Stakhanova, A. Ghorbani, “Measuring code reused in Android apps,” 2016 14th Annual Conference on Privacy, Security and Trust (PST), Dec 12-14, 2016, https://ieeexplore.ieee.org/document/7906925

Cloning represents additional, hidden components in the supply chain

Source: B. Baker, “On Finding Duplication and Near-Duplication in Large Software Systems,” Proceedings of 2nd Working Conference on Reverse Engineering, Jul 14-16, 1995, https://ieeexplore.ieee.org/abstract/document/514697 ;

L. Jiang, G. Misherghi, Z.Su, S. Glondu, “DECKARD: Scalable and Accurate Tree-based Detection of Code Clones,” 29th International Conference on Software Engineering (ICSE’07), May 20-26, 2007, https://web.cs.ucdavis.edu/~su/publications/icse07.pdf

Z. Li, S. Lu, S. Myagmar, Y. Zhou, “CP-Miner: Finding Copy-Paste and Related Bugs in Large-Scale Software Code,” IEEE Transactions on Software Engineering, Vol 32, No. 3, Mar 2006, https://people.cs.uchicago.edu/~shanlu/paper/TSE-CPMiner.pdf

c. Kapser, “Toward an Understanding of Software Code Cloning as a Development Practice,” PhD Thesis, U. Waterloo, 2009, https://pdfs.semanticscholar.org/bdae/5ede2999eae51645b5c91004706485a53af0.pdf

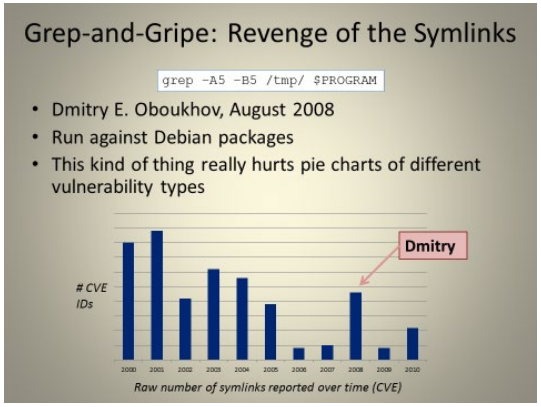
Cloning: cutting and pasting code – “microcomponents”

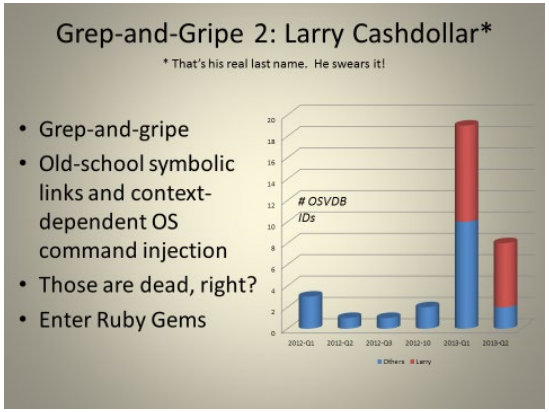
- Typically10–15% of the source code in large software systems is part of one or more code clones [Kapser]
- 19% of X Windows System [Baker]
- 20% of other large programs (>1M LOC) [Baker]
- Throughout Linux
  - 22.7% of Linux kernel [Jang]
  - 190,000 copy-pasted segments in Linux [Li]
  - 150,000 copy-pasted segments in FreeBSD. [Li]
- 29% of JDK [Jangh]

Open source is not secure

Heartbleed and Shellshock were found by exploitation

Other open source software illustrates vulnerabilities from cursory inspection

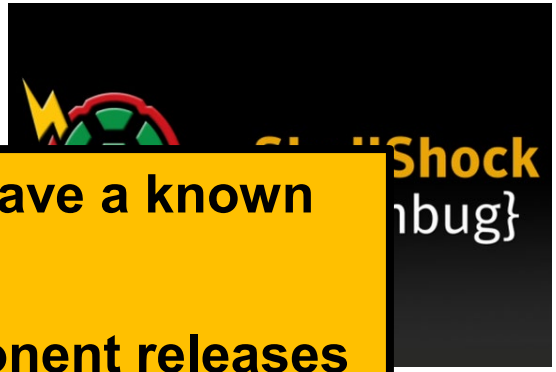




Sources: Steve Christey (MITRE) & Brian Martin (OSF), Buying Into the Bias: Why Vulnerability Statistics Suck, https://media.blackhat.com/us-13/US-13-Martin-Buying-Into-The-Bias-Why-Vulnerability-Statistics-Suck-Slides.pdf; Sonatype, Sonatype Open Source Development and Application Security Survey; Sonatype, "2019 State of the Software Supplu Chain", https://www.sonatype.com/hubfs/SSC/2019%20SSC/SON_SSSC-Report-2019_jun16-DRAFT.pdf

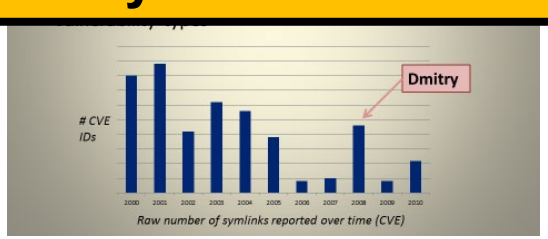
Open source is not secure

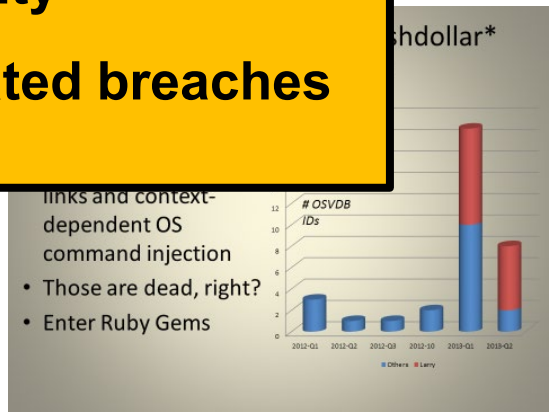
Heartbleed and Shellshock were found by exploitation



- **51% of JavaScript components have a known security vulnerability**
- **1 in 10 downloads of Java component releases have a known security vulnerability**
- **71% increase in open source related breaches over the last 5 years**

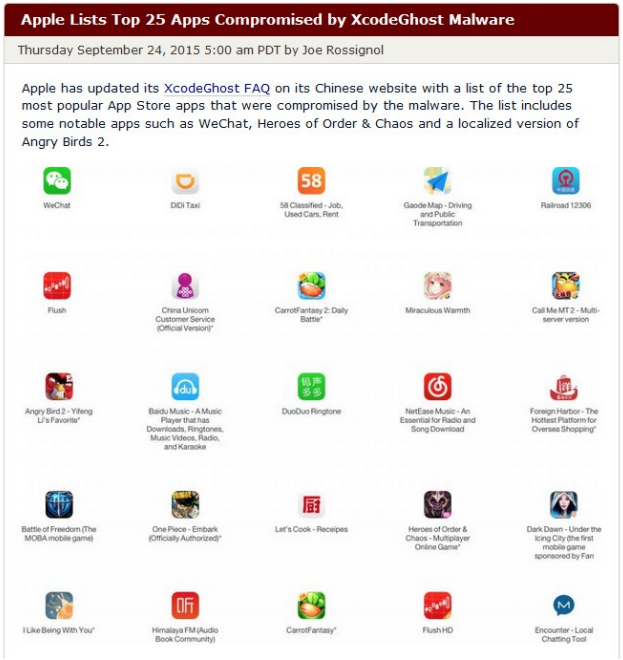
Other open sou
software illustra
vulnerabilities f
inspection





Sources: Steve Christey (MITRE) & Brian Martin (OSF), Buying Into the Bias: Why Vulnerability Statistics Suck, https://media.blackhat.com/us-13/US-13-Martin-Buying-Into-The-Bias-Why-Vulnerability-Statistics-Suck-Slides.pdf; Sonatype, Sonatype Open Source Development and Application Security Survey; Sonatype, "2019 State of the Software Supplu Chain", https://www.sonatype.com/hubfs/SSC/2019%20SSC/SON_SSSC-Report-2019_jun16-DRAFT.pdf

Corruption in the tool chain already exists



Apple Lists Top 25 Apps Compromised by XcodeGhost Malware

Thursday September 24, 2015 5:00 am PDT by Joe Rossignol

Apple has updated its XcodeGhost FAQ on its Chinese website with a list of the top 25 most popular App Store apps that were compromised by the malware. The list includes some notable apps such as WeChat, Heroes of Order & Chaos and a localized version of Angry Birds 2.

- XcodeGhost corrupted Apple's development environment
- Major programs affected
  - WeChat
  - Badu Music
  - Angry Birds 2
  - Heroes of Order & Chaos
  - iOBD2
- Not alone
  - Expensive Wall (2017)
  - HackTask (2017)

Sources: http://www.macrumors.com/2015/09/24/xcodeghost-top-25-apps-apple-list/
http://www.itntoday.com/2015/09/the-85-ios-apps-affected-by-xcodeghost.html

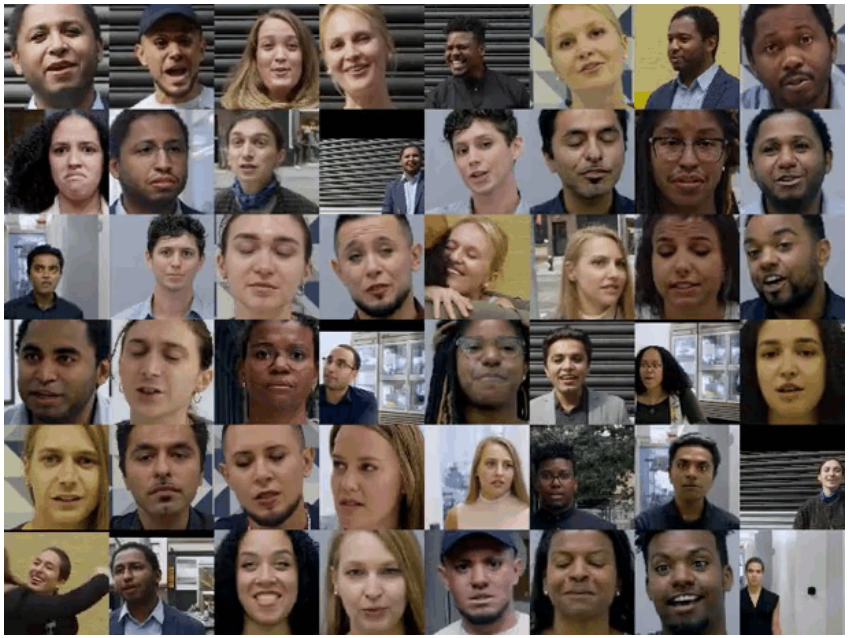
AI and Data Make Supply Chain Issues Worse

Newer, advanced software depends on these additional "supplies"

Relatively less is known about the security of these "supplies"

Machine Learning Frameworks	Data Sources
• Pandas • Numpy • Scikit-learn • Matplotlib • TensorFlow • Keras • Seaborn • Pytorch & Torch	• Kaggle • UCI Machine Learning Repository • Find Datasets • Data.gov • xView • ImageNet • Google's Open Images

Machine learning system face training data supply challenges



Rich supplies of “deep fakes” are readily accessible

Source: https://ai.googleblog.com/2019/09/contributing-data-to-deepfake-detection.html

Poor detection of deep fakes

FaceForensics Benchmark — Benchmarks ▾ | Data and Documentation | About | Submit

FaceForensics Benchmark

This table lists the benchmark results for the Binary Classification scenario.

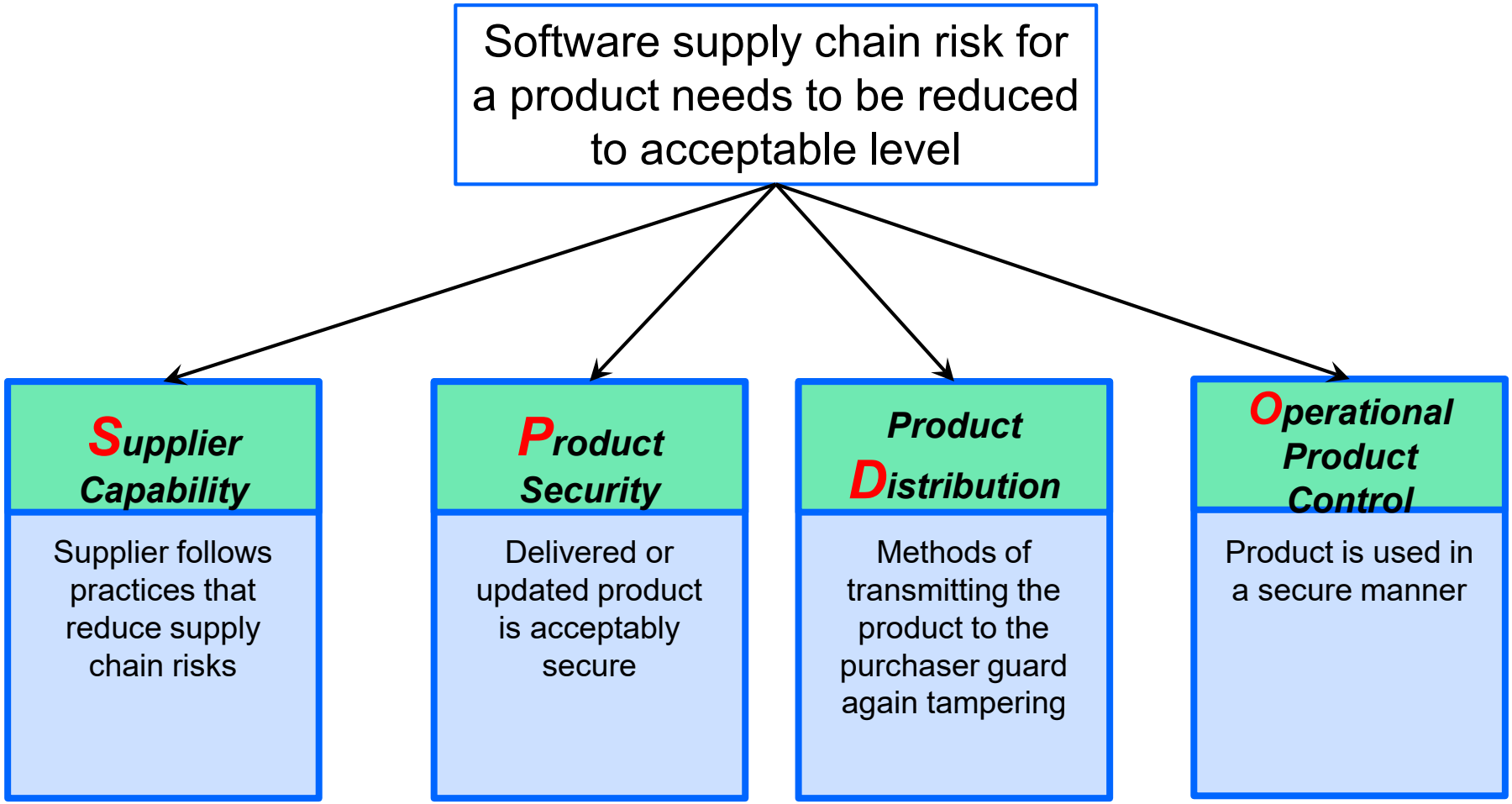
Method	Info	Deepfakes	Face2Face	FaceSwap	NeuralTextures	Pristine	Total
Xception	P	**0.964**	**0.869**	**0.903**	**0.807**	0.524	**0.710**
Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, Matthias Nießner: FaceForensics++: Learning to Detect Manipulated Facial Images. ICCV 2019							
MesoNet		0.873	0.562	0.612	0.407	**0.726**	0.660
Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen: Mesonet: a compact facial video forgery detection network. arXiv							
XceptionNet Full Image	P	0.745	0.759	0.709	0.733	0.510	0.624
Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, Matthias Nießner: FaceForensics++: Learning to Detect Manipulated Facial Images. ICCV 2019							
Bayar and Stamm		0.845	0.737	0.825	0.707	0.462	0.616
Belhassen Bayar and Matthew C. Stamm: A deep learning approach to universal image manipulation detection using a new convolutional layer. ACM Workshop on Information Hiding and Multimedia Security							
Rahmouni		0.855	0.642	0.563	0.607	0.500	0.581
Nicolas Rahmouni, Vincent Nozick, Junichi Yamagishi, and Isao Echizen: Distinguishing computer graphics from natural images using convolution neural networks. IEEE Workshop on Information Forensics and Security.							
Recasting		0.855	0.679	0.738	0.780	0.344	0.552
Davide Cozzolino, Giovanni Poggi, and Luisa Verdoliva: Recasting residual-based local descriptors as convolutional neural networks: an application to image forgery detection. ACM Workshop on Information Hiding and Multimedia Security							
Steganalysis Features		0.736	0.737	0.689	0.633	0.340	0.518
Jessica Fridrich and Jan Kodovsky: Rich Models for Steganalysis of Digital Images. IEEE Transactions on Information Forensics and Security							

Cannot reliable verify that training data obtained through a supply chain

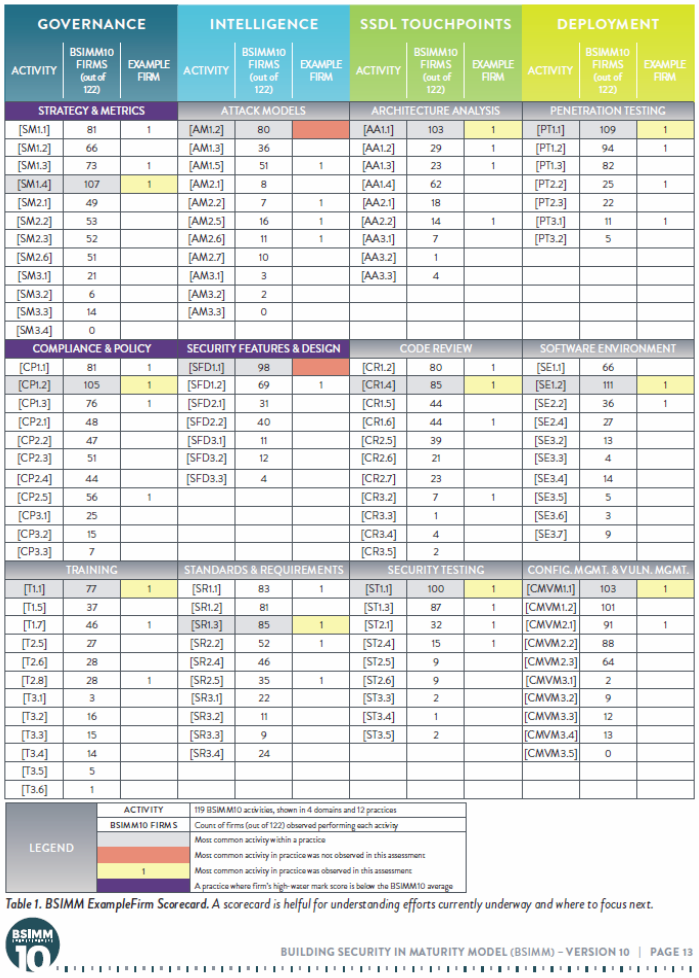
Preconfigured machine learning systems provide a vehicle to distribute bad training data

Source: http://kaldir.vc.in.tum.de/faceforensics_benchmark/index.php **(as of 9/25/19)**

Reducing software supply chain risk factors



Supplier capability: security commitment evidence



GOVERNANCE			INTELLIGENCE			SSDL TOUCHPOINTS			DEPLOYMENT		
ACTIVITY	BSIMM10 FIRMS (out of 122)	EXAMPLE FIRM	ACTIVITY	BSIMM10 FIRMS (out of 122)	EXAMPLE FIRM	ACTIVITY	BSIMM10 FIRMS (out of 122)	EXAMPLE FIRM	ACTIVITY	BSIMM10 FIRMS (out of 122)	EXAMPLE FIRM
STRATEGY & METRICS			**ATTACK MODELS**			**ARCHITECTURE ANALYSIS**			**PENETRATION TESTING**		
[SM1.1]	81	1	[AM1.2]	80		[AA1.1]	103	1	[PT1.1]	109	1
[SM1.2]	66		[AM1.3]	36		[AA1.2]	29	1	[PT1.2]	94	1
[SM1.3]	73	1	[AM1.5]	51	1	[AA1.3]	23	1	[PT1.3]	82	
[SM1.4]	107	1	[AM2.1]	8		[AA1.4]	62		[PT2.2]	25	1
[SM2.1]	49		[AM2.2]	7	1	[AA2.1]	18		[PT2.3]	22	
[SM2.2]	53		[AM2.5]	16	1	[AA2.2]	14	1	[PT3.1]	11	1
[SM2.3]	52		[AM2.6]	11	1	[AA3.1]	7		[PT3.2]	5	
[SM2.6]	51		[AM2.7]	10		[AA3.2]	1				
[SM3.1]	21		[AM3.1]	3		[AA3.3]	4				
[SM3.2]	6		[AM3.2]	2							
[SM3.3]	14		[AM3.3]	0							
[SM3.4]	0										
COMPLIANCE & POLICY			**SECURITY FEATURES & DESIGN**			**CODE REVIEW**			**SOFTWARE ENVIRONMENT**		
[CP1.1]	81	1	[SFD1.1]	98		[CR1.2]	80	1	[SE1.1]	66	
[CP1.2]	105	1	[SFD1.2]	69	1	[CR1.4]	85	1	[SE1.2]	111	1
[CP1.3]	76	1	[SFD2.1]	31		[CR1.5]	44		[SE2.2]	36	1
[CP2.1]	48		[SFD2.2]	40		[CR1.6]	44	1	[SE2.4]	27	
[CP2.2]	47		[SFD3.1]	11		[CR2.5]	39		[SE3.2]	13	
[CP2.3]	51		[SFD3.2]	12		[CR2.6]	21		[SE3.3]	4	
[CP2.4]	44		[SFD3.3]	4		[CR2.7]	23		[SE3.4]	14	
[CP2.5]	56	1				[CR3.2]	7	1	[SE3.5]	5	
[CP3.1]	25					[CR3.3]	1		[SE3.6]	3	
[CP3.2]	15					[CR3.4]	4		[SE3.7]	9	
[CP3.3]	7					[CR3.5]	2				
TRAINING			**STANDARDS & REQUIREMENTS**			**SECURITY TESTING**			**CONFIG. MGMT. & VULN. MGMT.**		
[T1.1]	77	1	[SR1.1]	83	1	[ST1.1]	100	1	[CMVM1.1]	103	1
[T1.5]	37		[SR1.2]	81		[ST1.3]	87	1	[CMVM1.2]	101	
[T1.7]	46	1	[SR1.3]	85	1	[ST2.1]	32	1	[CMVM2.1]	91	1
[T2.5]	27		[SR2.2]	52	1	[ST2.4]	15	1	[CMVM2.2]	88	
[T2.6]	28		[SR2.4]	46		[ST2.5]	9		[CMVM2.3]	64	
[T2.8]	28	1	[SR2.5]	35	1	[ST2.6]	9		[CMVM3.1]	2	
[T3.1]	3		[SR3.1]	22		[ST3.3]	2		[CMVM3.2]	9	
[T3.2]	16		[SR3.2]	11		[ST3.4]	1		[CMVM3.3]	12	
[T3.3]	15		[SR3.3]	9		[ST3.5]	2		[CMVM3.4]	13	
[T3.4]	14		[SR3.4]	24					[CMVM3.5]	0	
[T3.5]	5										
[T3.6]	1										

LEGEND		
	ACTIVITY	119 BSIMM10 activities, shown in 4 domains and 12 practices
	BSIMM10 FIRMS	Count of firms (out of 122) observed performing each activity
		Most common activity within a practice
		Most common activity in practice was not observed in this assessment
	1	Most common activity in practice was observed in this assessment
		A practice where firm's high-water mark score is below the BSIMM10 average

Table 1. BSIMM ExampleFirm Scorecard. A scorecard is helpful for understanding efforts currently underway and where to focus next.

Supplier institutionalizes secure development practices

"Building Security In Maturity Model" scorecard is one way to gauge practice adoption

Source: S. Migues, J. Steven, M. Ware, BSIMM10, https://www.bsimm.com/content/dam/bsimm/reports/bsimm10.pdf

Open source components could use a proxy process for supplier commitments

Develop an alternative evaluation method for open source component acceptance, e.g.,

- History of project
- Length of existence
- Frequency of updates and fixed
- Composition of committers
- Popularity

ATOS' QSOS is one example with four steps:

- Define
- Evaluate
- Select
- Qualify

Sources: http://www.qsos.org/method; http://dist.qsos.org/qsos-2.0_en.pdf

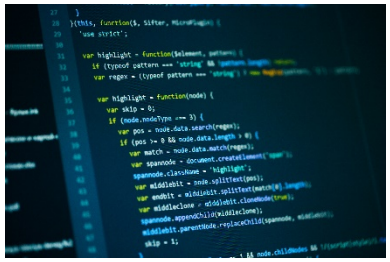
Product security: Evaluate a product's threat resistance

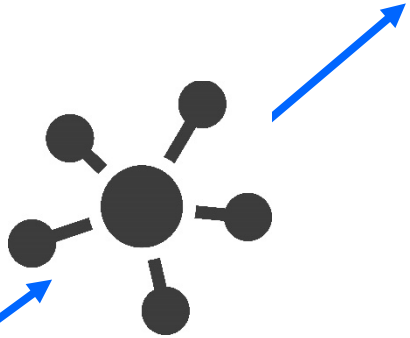
What product characteristics minimize opportunities to enter and change the product's security characteristics?

- Attack surface evaluation: Exploitable features have been identified and eliminated where possible
- Design and coding weaknesses associated with exploitable features have been identified and mitigated (CWE)
  - Dynamic, Static, Interactive Application Security Testing (DAST, SAST, IAST)
  - Independent validation and verification of threat resistance
- Delivery in or compatibility with Runtime Application Self Protection (RASP) containers

There is a growing body of 3rd parties who perform some of this analysis for open source components

Open source components could use a proxy process for product threat resistance



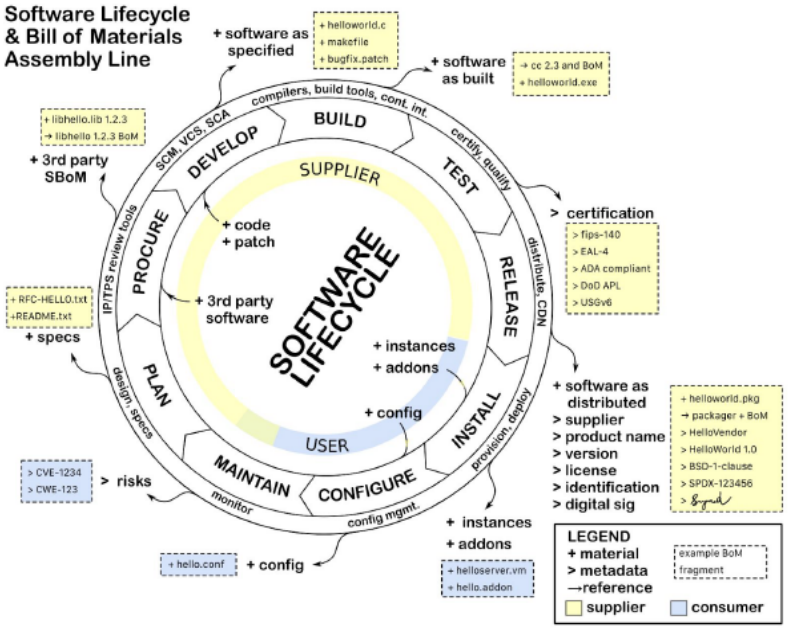


3rd party vendors have defined processes for evaluating open source components, such as

- CVE/NVD checking
- Local scanning
- Centralized distribution

Source: https://guides.sonatype.com/iqserver/technical-guides/lifecycle-scanning/; https://www.whitesourcesoftware.com/open-source-security/

Establish a Software Bill of Materials

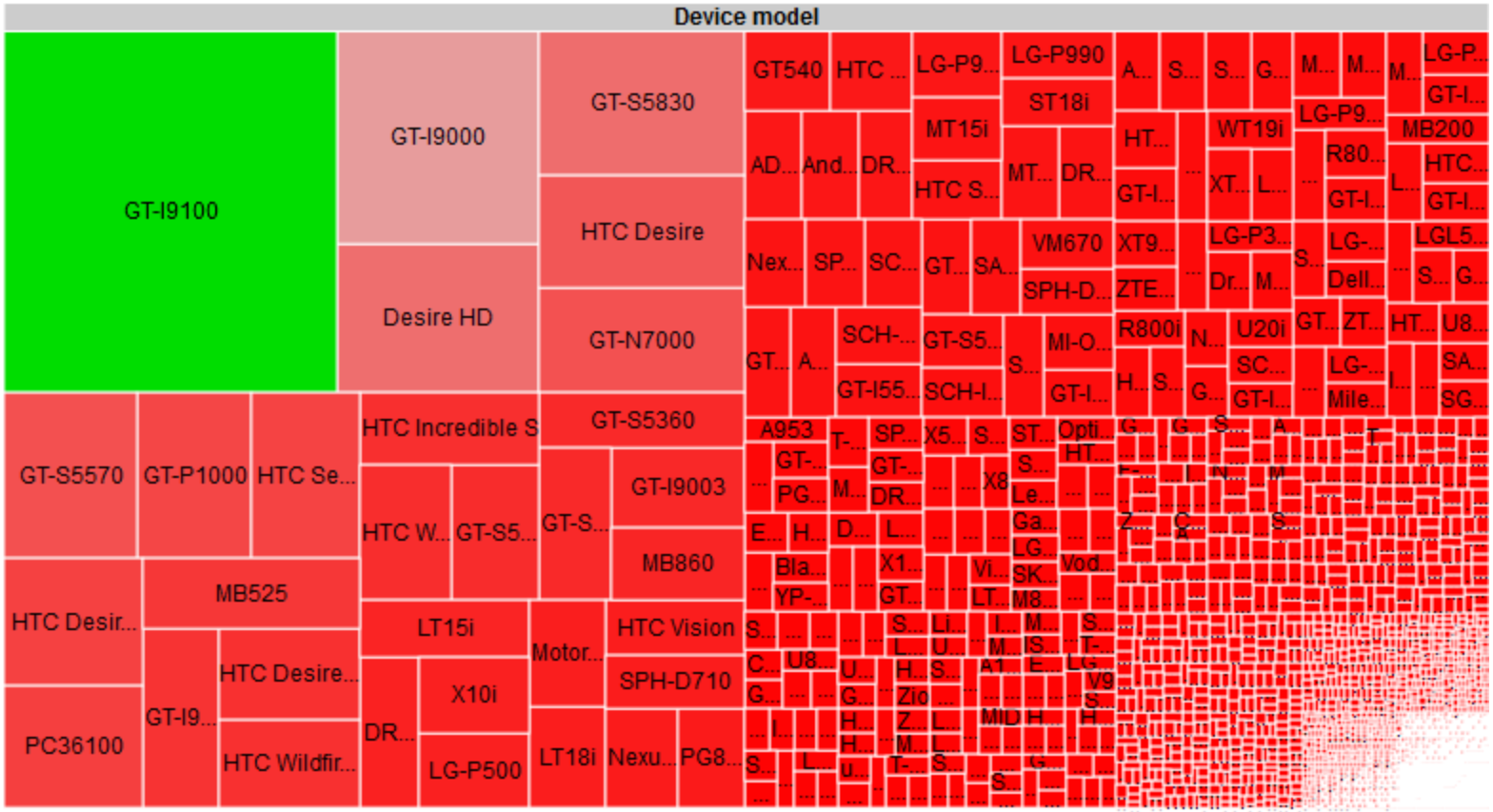


- Need to know the individual sub-components of a piece of software. Third-party components which could transitively inject vulnerabilities.
- Example formats and specifications:
  - Software Identification (SWID) Tags
  - Common Platform Enumeration (CPE)
  - Software Package Data Exchange (SPDX)
- Participation in NTIA Software Component Transparency (Dept of Commerce)
- Challenges
  - Large aggregations/granularity
  - Component removal
  - Fragmentation of components

Source: Home page https://www.ntia.doc.gov/SoftwareTransparency;

"Survey of Existing SBOM Formats and Standards," NTIA, Sept 3, 2019, https://www.ntia.doc.gov/files/ntia/publications/ntia_sbom_formats_and_standards_whitepaper_2019_0904.pdf

Avoid fragmentation: Versions of Android illustrate open source fragmentation



Source: http://opensignal.com/reports/fragmentation.php

(https://web.archive.org/web/20150326232333/http://opensignal.com/reports/fragmentation-2013/fragmentation-2013.pdf)

Product distribution: 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

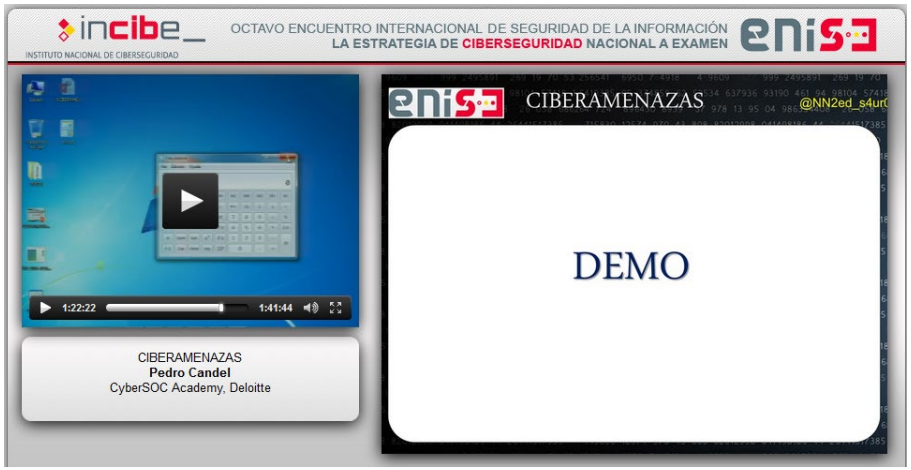
- Require good security practices by their suppliers
- Assess the security of delivered products
- Address the additional risks associated with using the product in their context

Minimize internal suppliers

- Single point of distribution to internal development community
- No cloning

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; https://www.withfriendship.com/user/mithunss/easter-eggs-in-microsoft-products.php

Distribution Environment Attacks

Types of supply chain attacks that leveraged compromised code and the development environment:

Download site attacks

- Havex/Dragonfly (2014), KingSlayer (2015), Fioxif/CCleaner (2017), Expensive Wall (2017), Shadowpad (2017)
- Repackaged applications with malware
  - Up to 50% of Android applications on some download sites are repackaged applications with malware

Patch site attacks

- NotPetya/MeDoc (2017) paralyzed networks worldwide

Sources: H. Gonzalez, N. Stakhanova, A. Ghorbani, “Measuring code reused in Android apps,” **2016 14th Annual Conference on Privacy, Security and Trust (PST), Dec 12-14, 2016, https://ieeexplore.ieee.org/document/7906925**

Maintain operational attack resistance

Usage changes the attack surface and potential attacks for the product

- Change in feature usage or risks
- Supplier risk mitigations adequate for desired usage
- Effects of vendor upgrades/patches and local configuration changes
- Effects of integration into operations (system of systems)

Preserving product attack resistance with product deployment

- Maintaining inventory of components
- Patching and version upgrades (component lifecycle management)

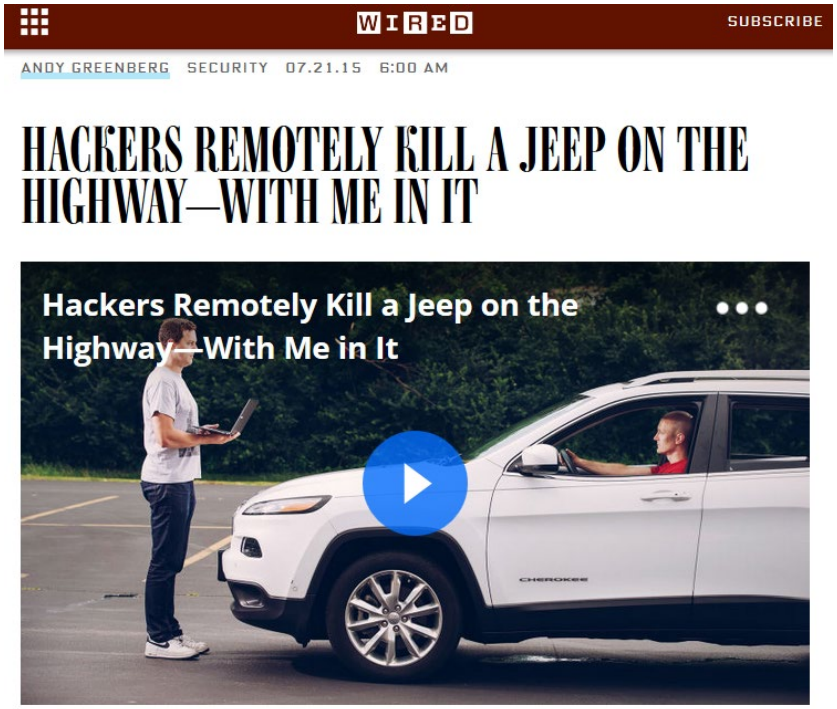
Steel furnaces have been successfully attacked through changed operational assumptions

"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."

Source: Sources: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Publikationen/Lageberichte/Lagebericht2014.pdf?__blob=publicationFile;
http://www.resilienceoutcomes.com/state-ict-security/

Connecting automotive systems to internet opens system to attack thru changed operational environment



Extending systems opens vulnerabilities not anticipated

- Optimizations performed assuming one attack method
- Assumptions no longer hold with additional integrations

Source: http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/

Vulnerabilities emerge in existing code

Defects in functionality found early and in new code

Vulnerabilities found in legacy code and late ("honeymoon effect")

New operating environments are a major cause of vulnerabilities

Carefully weigh benefits (risk reduction) vs cost (time, space) of implementing defense in depth.

Clark, Frei, Blaze, Smith, "Familiarity Breeds Contempt: The Honeymoon Effect and the Role of Legacy Code in Zero-Day Vulnerabilities," ACSAC '10 Dec. 6-10, 2010, p. 251-260."

Staying current with software supply chain issues

Government, industry and standards organizations are working together to improve the software supply chain

- DHS' CISA ICT Supply Chain Risk Management Task Force
- Dept of Commerce's NTIA Software Component Transparency
- NIST Supply Chain Risk Management Practices for Federal Information Systems and Organizations

OBJECTIVE 6

To integrate supply chain risk management (SCRM) concepts into the RMF to protect against untrustworthy suppliers, insertion of counterfeits, tampering, unauthorized production, theft, insertion of malicious code, and poor manufacturing and development practices throughout the SDLC.

And more is being worked on

(Ron Ross, RMF 2.0 presentation, chart 20, https://csrc.nist.gov/CSRC/media/Presentations/RMF-2-0-Risk-Management-Framework-Simplify-Inno/images-media/sp800-37r2-ipd-rollout-DOJ-20180509.pdf)

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

Further reading

Alberts, Christopher, et al., “Introduction to the Security Engineering Risk Analysis (SERA) Framework,” Software Engineering Institute, Nov 2014, http://resources.sei.cmu.edu/asset_files/TechnicalNote/2014_004_001_427329.pdf

Alberts, Christopher, John Haller, Charles M. Wallen and Carol Woody, “Assessing DoD System Acquisition Supply Chain Risk Management,” CrosssTalk - The Journal of Defense Software Engineering, May/June 2017, http://www.crosstalkonline.org/storage/issue-archives/2017/201705/201705-albert.pdf

Axelrod, C. Warren, “Mitigating Software Supply Chain Risk,” ISCA Journal Online, Vol 4., 2013, http://www.isaca.org/Journal/Past-Issues/2013/Volume-4/Pages/JOnline-Mitigating-Software-Supply-Chain-Risk.aspx

Axelrod, C. Warren, “Malware, Weakware and the Security of Software Supply Chains,” Cross-Talk, March/April 2014, p. 20, http://www.crosstalkonline.org/storage/issue-archives/2014/201403/201403-Axelrod.pdf

Ellison, Robert, et al, “Software Supply Chain Risk Management: From Products to Systems of Systems,” Software Engineering Institute, Dec 2010, https://resources.sei.cmu.edu/asset_files/technicalnote/2010_004_001_15194.pdf

Ellison, Robert, et al. “Evaluating and Mitigating Software Supply Chain Security Risks,” Software Engineering Institute, May 2010, http://resources.sei.cmu.edu/asset_files/technicalnote/2010_004_001_15176.pdf

Ellison, Robert and Woody, Carol, “Supply-Chain Risk Management: Incorporating Security into Software Development,” Proceedings of the 43rd Hawaii International Conference on System Sciences, 2010, http://resources.sei.cmu.edu/asset_files/WhitePaper/2013_019_001_297341.pdf

Jarzombek, Joe, “Collaboratively Advancing Strategies to Mitigate Software Supply Chain Risks,” July 30, 2009, http://csrc.nist.gov/groups/SMA/ispab/documents/minutes/2009-07/ispab_july09-jarzombek_swa-supply-chain.pdf

Software Assurance Forum, Processes and Practices Working Group, “Software Assurance Checklist for Software Supply Chain Risk Management,” https://buildsecurityin.us-cert.gov/sites/default/files/20101208-SwAChecklist.pdf

“Software Supply Chain Risk Management & Due-Diligence,” Software Assurance Pocket Guide Series: Acquisition & Outsourcing, Vol II, Version 1.2, June 16, 2009, https://buildsecurityin.us-cert.gov/sites/default/files/DueDiligenceMWV12_01AM090909.pdf

Third Party Software Security Working Group, “Appropriate Software Security Control Types for Third Party Service and Product Providers,” Financial Services Information Sharing and Analysis Center, 2013 http://docs.ismgcorp.com/files/external/WP_FSISAC_Third_Party_Software_Security_Working_Group.pdf