



SOFTWARE TRANSPARENCY

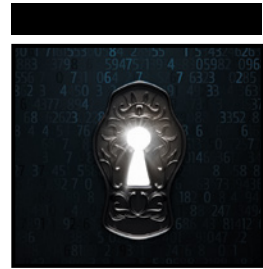
Supply Chain Security in an
Era of a Software-Driven Society

Chris Hughes and Tony Turner

Foreword by **Allan Friedman, PhD**
Technical Editor, **Steve Springett**

WILEY

Software Transparency



Software Transparency

Supply Chain Security in an Era of a
Software-Driven Society

Chris Hughes
Tony Turner

WILEY

Copyright © 2023 by John Wiley & Sons, Inc. All rights reserved.

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada and the United Kingdom.

ISBN: 978-1-394-15848-5

ISBN: 978-1-394-15850-8 (ebk)

ISBN: 978-1-394-15849-2 (ebk)

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning, or otherwise, except as permitted under Section 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 750-4470, or on the web at www.copyright.com. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at www.wiley.com/go/permission.

Trademarks: WILEY and the Wiley logo are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Further, readers should be aware that websites listed in this work may have changed or disappeared between when this work was written and when it is read. Neither the publisher nor authors shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic formats. For more information about Wiley products, visit our web site at www.wiley.com.

Library of Congress Control Number: 2022951423

Cover images: © Dmytro Yelisiiev/Getty Images;

© fergregory/Adobe Stock

Cover design: Wiley

This is dedicated to my children, Carolina, Calvin, Callie, and Clayton, as well as my wife, Kathleen, for their unwavering love and support which serves as motivation to always strive to be my best. I also would like to dedicate it to my mother Dawn and grandfather Bill, who always taught me the importance of hard work.

— Chris Hughes

This book marks a transition point of so many things in my life, and throughout it all, my wife Becki, and my two sons, Alex and Gavin, have patiently supported me. I'd like to thank them from the bottom of my heart for always being there to remind me of what is most important in life.

— Tony Turner



About the Authors



Chris Hughes currently serves as the co-founder and CISO of Aquia. Chris has nearly 20 years of IT/cybersecurity experience, ranging from active duty time with the U.S. Air Force, as a civil servant with the U.S. Navy and General Services Administration (GSA)/FedRAMP, as well as time as a consultant in the private sector. In addition, he is an adjunct professor for M.S. Cybersecurity programs at Capitol Technology University and University of Maryland Global Campus. Chris also participates in industry working groups such as the Cloud Security Alliances Incident Response and SaaS Security Working Group and serves as the Membership Chair for Cloud Security Alliance D.C. Chris also co-hosts the Resilient Cyber podcast.

Chris holds various industry certifications such as the CISSP/CCSP from (ISC)², as well as holding both the AWS and Azure security certifications and Cloud Security Alliance's Certificate of Cloud Auditing Knowledge (CCAK). He holds a B.S. in Information Systems, an M.S. in Cybersecurity, and an MBA. He regularly consults with IT and cybersecurity leaders from various industries to assist their organizations with their digital transformation journeys, while keeping security a core component of that transformation. Chris has authored many articles and thought pieces on software supply chain security, as well as presenting on the topic at industry events such as Carnegie Mellon's DevSecOps Days Washington, D.C. among others.



Tony Turner is the founder and CEO of Opswright, a software company focused on security engineering for critical infrastructure. With over 25 years' experience in cybersecurity, Tony has worked as an internal security engineer, developer, network and systems administrator, security consultant, architect, and managed professional services for security vendors. Most recently he functioned as the VP of Research and Development for the Fortress Labs

team at Fortress Information Security, where he spearheaded their software transparency and vulnerability management product research and roadmap. Tony also founded and leads the OWASP Orlando chapter since 2011, focused on improving the visibility of application security; is the project leader for WAFEC, a web application firewall documentation project; and co-founded the Security BSides Orlando conference. He sits on multiple trade organizations and industry groups from MITRE, ISA, OWASP, CISA, GlobalPlatform, and others, largely focused on the topics of software transparency, and software and product security within critical infrastructure.

Tony graduated with a B.S. from Hodges University in Naples, Florida, and holds several industry certifications, including the CISSP; CISA; six credentials from SANS/GIAC, OPSE, CSTP, and CSWAE; and multiple vendor certifications from F5, Imperva, and others.

Additionally, Tony has been involved with the certification exam writing process for GIAC and F5 ASM web application firewall certification and is a course author for SANS on Defending Product Supply Chains, to be released in 2023.

Tony lives in Indialantic, Florida, with his wife, Becki; two sons, Alex and Gavin; five cats; three lizards; two guinea pigs; and an army of squirrels in his backyard that demand their morning tribute.



About the Technical Editor

Steven Springett has over 25 years leading product development teams and has spent over 14 years focused on supply chain security. Currently, Steve is the director of secure development at ServiceNow, where he leads application security architecture, threat modeling, security champions, and developer enablement across the organization. Steve is passionate about helping organizations identify and reduce risk from the use of third-party and open source components. He is an open source advocate and leads the OWASP Dependency-Track project, is a coauthor of the OWASP Software Component Verification Standard (SCVS), and is the chair of the OWASP CycloneDX Core Working Group, a bill of materials standard that provides advanced supply chain capabilities for cyber-risk reduction. Steve holds a CSSLP and CCSK, among other industry certifications. Steve lives in Chicago's North Shore with his wife, Vera, and daughter, Aryana.



Acknowledgments

We would like to acknowledge the countless industry professionals who we have worked with, for, and learned with over our several decades of combined cybersecurity experience.

This includes industry thought leaders on the topic of software supply chain security, such as Allan Friedman, Joshua Corman, Robert Wood, Virginia Wright, Jason Weiss, and, of course, Steve Springett, who, in addition to serving as an industry leader on the topic of software supply chain security, also served as our technical editor and sounding board. We would also like to thank the countless public and private sector leaders who have participated in working groups and efforts associated with software supply chain security through groups such as OWASP, the Linux Foundation, OpenSSF, Department of Defense, NTIA, CISA, Carnegie Mellon's Software Engineering Institute (SEI), and others. Without the community's collective expertise, knowledge sharing, and collaboration, we would not be able to produce this body of knowledge to give back to the industry. Software is critical to nearly every aspect of our society, and we're in this fight together.



Contents at a Glance

Foreword	xxi	
Introduction	xxv	
Chapter 1	Background on Software Supply Chain Threats	1
Chapter 2	Existing Approaches—Traditional Vendor Risk Management	25
Chapter 3	Vulnerability Databases and Scoring Methodologies	41
Chapter 4	Rise of Software Bill of Materials	71
Chapter 5	Challenges in Software Transparency	99
Chapter 6	Cloud and Containerization	111
Chapter 7	Existing and Emerging Commercial Guidance	137
Chapter 8	Existing and Emerging Government Guidance	179
Chapter 9	Software Transparency in Operational Technology	219
Chapter 10	Practical Guidance for Suppliers	229
Chapter 11	Practical Guidance for Consumers	245
Chapter 12	Software Transparency Predictions	261
Index	283	



Contents

Foreword	xxi
Introduction	xxv
Chapter 1 Background on Software Supply Chain Threats	1
Incentives for the Attacker	1
Threat Models	2
Threat Modeling Methodologies	3
Stride	3
Stride-LM	4
Open Worldwide Application Security Project (OWASP)	
Risk-Rating Methodology	4
DREAD	5
Using Attack Trees	5
Threat Modeling Process	6
Landmark Case 1: SolarWinds	14
Landmark Case 2: Log4j	18
Landmark Case 3: Kaseya	21
What Can We Learn from These Cases?	23
Summary	24
Chapter 2 Existing Approaches—Traditional Vendor Risk Management	25
Assessments	25
SDL Assessments	28
Application Security Maturity Models	29
Governance	30
Design	30
Implementation	31
Verification	31
Operations	32

Application Security Assurance	32
Static Application Security Testing	33
Dynamic Application Security Testing	34
Interactive Application Security Testing	35
Mobile Application Security Testing	36
Software Composition Analysis	36
Hashing and Code Signing	37
Summary	39
Chapter 3 Vulnerability Databases and Scoring Methodologies	41
Common Vulnerabilities and Exposures	41
National Vulnerability Database	44
Software Identity Formats	46
CPE	46
Software Identification Tagging	47
PURL	49
Sonatype OSS Index	50
Open Source Vulnerability Database	51
Global Security Database	52
Common Vulnerability Scoring System	54
Base Metrics	55
Temporal Metrics	57
Environmental Metrics	58
CVSS Rating Scale	58
Critiques	59
Exploit Prediction Scoring System	59
EPSS Model	60
EPSS Critiques	62
CISA’s Take	63
Common Security Advisory Framework	63
Vulnerability Exploitability eXchange	64
Stakeholder-Specific Vulnerability Categorization and Known Exploited Vulnerabilities	65
Moving Forward	69
Summary	70
Chapter 4 Rise of Software Bill of Materials	71
SBOM in Regulations: Failures and Successes	71
NTIA: Evangelizing the Need for SBOM	72
Industry Efforts: National Labs	77
SBOM Formats	78
Software Identification (SWID) Tags	79
CycloneDX	80
Software Package Data Exchange (SPDX)	81
Vulnerability Exploitability eXchange (VEX) and Vulnerability Disclosures	82
VEX Enters the Conversation	83

	VEX: Adding Context and Clarity	84
	VEX vs. VDR	85
	Moving Forward	88
	Using SBOM with Other Attestations	89
	Source Authenticity	89
	Build Attestations	90
	Dependency Management and Verification	90
	Sigstore	92
	Adoption	93
	Sigstore Components	93
	Commit Signing	95
	SBOM Critiques and Concerns	95
	Visibility for the Attacker	96
	Intellectual Property	97
	Tooling and Operationalization	97
	Summary	98
Chapter 5	Challenges in Software Transparency	99
	Firmware and Embedded Software	99
	Linux Firmware	99
	Real-Time Operating System Firmware	100
	Embedded Systems	100
	Device-Specific SBOM	100
	Open Source Software and Proprietary Code	101
	User Software	105
	Legacy Software	106
	Secure Transport	107
	Summary	108
Chapter 6	Cloud and Containerization	111
	Shared Responsibility Model	112
	Breakdown of the Shared Responsibility Model	112
	Duties of the Shared Responsibility Model	112
	The 4 Cs of Cloud Native Security	116
	Containers	118
	Kubernetes	123
	Serverless Model	128
	SaaS SBOM and the Complexity of APIs	129
	CycloneDX SaaS SBOM	130
	Tooling and Emerging Discussions	132
	Usage in DevOps and DevSecOps	132
	Summary	135
Chapter 7	Existing and Emerging Commercial Guidance	137
	Supply Chain Levels for Software Artifacts	137
	Google Graph for Understanding Artifact Composition	141
	CIS Software Supply Chain Security Guide	144
	Source Code	145

Build Pipelines	146
Dependencies	148
Artifacts	148
Deployment	149
CNCF's Software Supply Chain Best Practices	150
Securing the Source Code	152
Securing Materials	154
Securing Build Pipelines	155
Securing Artifacts	157
Securing Deployments	157
CNCF's Secure Software Factory Reference Architecture	157
The Secure Software Factory Reference Architecture	158
Core Components	159
Management Components	160
Distribution Components	160
Variables and Functionality	160
Wrapping It Up	161
Microsoft's Secure Supply Chain Consumption Framework	161
S2C2F Practices	163
S2C2F Implementation Guide	166
OWASP Software Component Verification Standard	167
SCVS Levels	168
Level 1	168
Level 2	169
Level 3	169
Inventory	169
Software Bill of Materials	170
Build Environment	171
Package Management	171
Component Analysis	173
Pedigree and Provenance	173
Open Source Policy	174
OpenSSF Scorecard	175
Security Scorecards for Open Source Projects	175
How Can Organizations Make Use of the Scorecards Project?	177
The Path Ahead	178
Summary	178
Chapter 8 Existing and Emerging Government Guidance	179
Cybersecurity Supply Chain Risk Management Practices for Systems and Organizations	179
Critical Software	181
Security Measures for Critical Software	182
Software Verification	186
Threat Modeling	187
Automated Testing	187
Code-Based or Static Analysis and Dynamic Testing	188

Review for Hard-Coded Secrets	188
Run with Language-Provided Checks and Protection	189
Black-Box Test Cases	189
Code-Based Test Cases	189
Historical Test Cases	189
Fuzzing	190
Web Application Scanning	190
Check Included Software Components	190
NIST’s Secure Software Development Framework	191
SSDF Details	192
Prepare the Organization (PO)	193
Protect the Software (PS)	194
Produce Well-Secured Software (PW)	194
Respond to Vulnerabilities (RV)	196
NSAs: Securing the Software Supply Chain Guidance Series	197
Security Guidance for Software Developers	197
Secure Product Criteria and Management	199
Develop Secure Code	202
Verify Third-Party Components	204
Harden the Build Environment	206
Deliver the Code	207
NSA Appendices	207
Recommended Practices Guide for Suppliers	209
Prepare the Organization	209
Protect the Software	210
Produce Well-Secured Software	211
Respond to Vulnerabilities	213
Recommended Practices Guide for Customers	214
Summary	218
Chapter 9 Software Transparency in Operational Technology	219
The Kinetic Effect of Software	220
Legacy Software Risks	222
Ladder Logic and Setpoints in Control Systems	223
ICS Attack Surface	225
Smart Grid	227
Summary	228
Chapter 10 Practical Guidance for Suppliers	229
Vulnerability Disclosure and Response PSIRT	229
Product Security Incident Response Team (PSIRT)	231
To Share or Not to Share and How Much Is Too Much?	236
Copyleft, Licensing Concerns, and “As-Is” Code	238
Open Source Program Offices	240
Consistency Across Product Teams	242
Manual Effort vs. Automation and Accuracy	243
Summary	244

Chapter 11	Practical Guidance for Consumers	245
	Thinking Broad and Deep	245
	Do I Really Need an SBOM?	246
	What Do I Do with It?	250
	Receiving and Managing SBOMs at Scale	251
	Reducing the Noise	253
	The Divergent Workflow—I Can't Just Apply a Patch?	254
	Preparation	256
	Identification	256
	Analysis	257
	Virtual Patch Creation	257
	Implementation and Testing	258
	Recovery and Follow-up	258
	Long-Term Thinking	259
	Summary	259
Chapter 12	Software Transparency Predictions	261
	Emerging Efforts, Regulations, and Requirements	261
	The Power of the U.S. Government Supply Chains to Affect Markets	267
	Acceleration of Supply Chain Attacks	270
	The Increasing Connectedness of Our Digital World	272
	What Comes Next?	275
Index		283



Foreword

Many of us will remember December 2021 as a time spent hunched over small travel laptops in relatives' guest rooms, dealing with the Log4j crisis. That vulnerability in an open source Java-logging framework developed by The Apache Software Foundation was scored as severe by both official metrics and software experts. It was not the hardest vulnerability to address, either through patches or other inline mitigations. Yet, the real challenge most organizations faced was location: Where is this darn thing? Buried deep in countless modern applications' supply chains, both producers and users of software simply didn't have a usable roadmap of where to focus.

The hard part of security should be in identifying vulnerabilities and discovering attackers sneaking into our supply chains. Instead, we've discovered that understanding what's in the software we make has required a nontrivial amount of work.

The idea of tracking what goes into software isn't new. Academics have been talking about it since the 1990s. Early idea discussions were happening in disparate corners of the software world in the 2000s. Failure to account for the open source licenses got a number of large companies into serious legal trouble. Collecting and leveraging supplier data formed an integral part of the revolution in heavy industry quality that dates back to the late 1940s, with the Deming and the "Toyota revolution" that inspired the DevSecOps and modern software revolution decades later.

Indeed, it's quite remarkable that we don't have better transparency in our software supply chains. I often use the example of a Twinkie (first advanced by Audra Hatch and Josh Corman) to raise the question of why we have a better understanding of what is in a nonbiodegradable snack than what is in the software that runs in our companies, governments, and critical infrastructure.

As we embark on our journey to understand how to implement software bill of materials (SBOMs), it's useful to take a moment to understand why we didn't have this capacity at the beginning, how we're beginning to make progress, and what the value of this transparency is. There are some less-flattering reasons why few organizations wanted to share, including not wanting to be exposed to the previously mentioned open source license compliance risks. Frankly, many organizations did not want to admit to the scale of technical debt or incur the costs of having to set up the basic internal infrastructure and processes to track their dependency data. Moreover, it should be acknowledged that starting this SBOM journey hasn't been easy—it has required bringing together technical expertise, an understanding of the diverse software ecosystem, and an appreciation of incentives. But a massive amount of progress has been made in the last five years.

The first thing we needed was a shared vision of what an SBOM is. Many experts had a general idea; these were debated and refined from 2018 to 2020 in the open, international, “multistakeholder process” that the National Telecommunications and Information Administration (NTIA) convened. The community defined the basics of an SBOM and laid out its core use cases. We then needed a machine-readable means to convey this information across the supply chain. Fortunately, some across the software world were ahead of us, so we were able to align the Linux Foundation's Software Package Data Exchange (SPDX) and Open Worldwide Application Security Project's (OWASP's) CycloneDX to meet these models.

The next step was creating tools to generate and use this machine-readable data. Great progress has been made across the software world in implementing SBOM generally, with new tools emerging across the ecosystem. We're seeing more sector- and technology-specific tools, because the needs of the industrial control systems (ICSs) and operational technology (OT) firmware world are somewhat different from traditional enterprise software, which, in turn, have unique features and integrations compared to the cloud-native world. Generation tools have begun to mature and new tools emerge all the time to help organizations use this data, both operationally and strategically. (It's been one of the perks of my job, first at the NTIA and now at the Cybersecurity and Infrastructure Security Administration [CISA], to meet with start-ups and open source innovators to find ways to meet real needs across the software marketplace.)

Of course, the third leg of the tripod, along with technology and markets, is government. The U.S. government has focused on securing the supply chain in recent years, but there has been strong interest as well as policy innovations by our partners around the world. SBOMs moved to the main stage of cybersecurity policy in May 2021, when the President's Executive Order (EO) on Improving the Nation's Cybersecurity declared, among other things, that suppliers to the U.S. government will provide the purchaser with an SBOM. As this book goes

to press, the final details of those regulations are expected. This goes along with regulations of medical devices in the United States by the Food and Drug Administration (FDA), activity from other U.S. regulators, and proposed regulations emerging around the world.

Because you have this book in your hands or on your screen, you probably don't need to be convinced of the value of transparency across the software supply chain. But you may have to convince others, evangelize across your organization or corner of the software ecosystem, argue for a budget, or convince your suppliers to share data. Despite the usual infosec vendors' hyperbole, the "SB" in SBOM doesn't stand for "silver bullet." SBOM is a data layer. We are still building the tools and processes to the turn that data into intelligence and, ultimately, into action. Just like a Common Vulnerabilities and Exposures (CVE) identifier doesn't actually fix a vulnerability on your network—and CVEs are now an indispensable part of our global vulnerability ecosystem—so, too, will SBOMs become an integral part of the software security and quality world.

Transparency in the software supply chain aids across the life cycle of software. For those of us who build software, SBOMs are a powerful tool to understand our processes and to ensure that we're not building things that are not secure-by-design or shipping with known risks. For those of us who choose or buy software, why would we use a supplier who doesn't understand what they are delivering? Why would we potentially adopt software that was outdated before its use? For those of us who operate software, without an SBOM, how can we respond to newly identified risks or make plans for systems that are built on end-of-life or end-of-support software? Of course, many of us occupy all three roles. It seems inevitable that more uses of this data will be identified and built as SBOMs become more ubiquitous.

There have been a lot of incredible contributions to the software transparency movement, championing the idea of SBOMs, building tools, and debating vital edge cases. Authors Hughes and Turner do a great job of capturing these advances—including some of their own—and explaining the details and nuances as we go from the idea of SBOM to the practice of SBOM. While I expect practitioners around the world to pick up and use this volume to incredible success for their organizations and their customers, it is my paradoxical wish that this book's critical value is actually short-lived. As more of us start to build and manage our software with the types of interoperable automation the authors so helpfully envision and follow the course they lay out, SBOMs will cease to be new and shiny and become a natural, automated part of how all software is made, sold, and used.

It's then we can turn to the next challenge.

Dr. Allan Friedman

Allan Friedman is Senior Advisor and Strategist at the Cybersecurity and Infrastructure Security Agency. He coordinates the global cross-sector community

efforts around software bill of materials (SBOM). He was previously the Director of Cybersecurity Initiatives at NTIA, leading pioneering work on vulnerability disclosure, SBOM, and other security topics. Prior to joining the federal government, Friedman spent over a decade as a noted information security and technology policy scholar at Harvard's Computer Science department, the Brookings Institution, and George Washington University's Engineering School. He is the co-author of the popular text *Cybersecurity and Cyberwar: What Everyone Needs to Know*, has a C.S. degree from Swarthmore College and a PhD from Harvard University.



Introduction

We are living in a time where software touches every aspect of our society. Software is involved in everything from critical infrastructure, digital commerce, and national security. In fact, as of this writing, the World Economic Forum (WEF) predicts that by the end of 2022, 60 percent of global gross domestic product (GDP) will be tied to digital systems (www3.weforum.org/docs/WEF_Responsible_Digital_Transformation.pdf). However, that same WEF report found that only 45 percent of people trust the technology that powers our modern economies and society. Part of that lack of trust can be traced to many years of notable digital data breaches and a long-standing issue with transparency when it comes to the software supply chain.

Software supply chain attacks are far from a new phenomenon, and concerns about trusting code have origins dating back to Ken Thompson's famous "Reflections on Trusting Trust" paper in 1984, where Thompson discussed the inability to trust code you did not create yourself. While the idea that code consumed from external sources could be untrustworthy or downright malicious, the manifestations of that statement have only been exacerbated in recent years as software supply chain attacks accelerate. Malicious actors, driven by a variety of motives, have realized that rather than targeting a single entity, they can compromise widely used software (whether proprietary or open source) and have a cascading impact across an entire ecosystem of consumers.

As these incidents have accelerated, organizations have increased their efforts to both understand software supply chain challenges, complexities, and incidents, and have implemented security measures to mitigate their associated risks. The Cloud Native Computing Foundation (CNCF) has compiled a catalog (<http://github.com/cncf/tag-security/tree/main/supply-chain-security/compromises>) of supply chain compromises dating back to 2003. This catalog

captures supply chain compromises from a variety of methods, such as exploited developer tooling, developer negligence, malicious maintainers, or even attack chaining (i.e., several compromises chained together to enable an attack).

Before some of the landmark cases discussed in this text, such as SolarWinds, Log4j, and others, the Office of the Director of National Intelligence (ODNI) published a paper (<http://dni.gov/files/NCSC/documents/supplychain/20190327-Software-Supply-Chain-Attacks02.pdf>) in 2017, calling it a watershed year for software supply chain attacks. The document laid out several significant software supply chain attacks in 2017, which impacted organizations supporting the U.S. government in addition to commercial leaders, several of which originated from nation-state actors. The ODNI paper points out that many software development and distribution channels lack proper security. ODNI also described some malicious actors going upstream due to better cyber hygiene among some organizations. It is often more efficient for malicious actors to go upstream and compromise downstream software consumers at scale, rather than targeting one individual organization. Some software supply chain attacks may be indiscriminate, targeting any consumers, while others may seek out specific upstream software producers knowing who some of their downstream consumers are.

There is no denying that digitally enabled systems have led to unprecedented efficiency, productivity, and innovation. That said, these same digitally connected software-empowered systems have now created levels of systemic risk that are only beginning to be fully understood. It is said that complex interdependencies can heighten systemic risk, and it would be hard to argue that the current state of the software ecosystem and modern digital systems are anything but complex. Malicious actors are realizing the value of targeting the software supply chain as well, with Gartner, an industry leader in technology research, predicting that by 2025, some 45 percent of organizations will experience a software supply chain attack. Some argue that estimate may be low, with organizations such as Sonatype producing a 2022 report (<http://sonatype.com/state-of-the-software-supply-chain/introduction>) showing a 742 percent annual increase in software supply chain attacks over the past three years.

As notable software supply chain attacks have increased, we have now seen ambitious efforts on behalf of governments as well as private sector organizations. In the United States, the White House issued Executive Order 14028, “Improving the Nation’s Cybersecurity” (<http://whitehouse.gov/briefing-room/presidential-actions/2021/05/12/executive-order-on-improving-the-nations-cybersecurity>), which has a section dedicated to enhancing software supply chain security. This order includes a section about a common lack of transparency as it relates to software being consumed by organizations, including the federal government.

In this book, we will discuss in more detail relevant software supply chain incidents, industry activities, emerging solutions, as well as the significant challenges that remain when it comes to securing the software supply chain.

Why Is This Critical?

Before we dive into the specifics of software supply chain threats and emerging frameworks and guidance, we should initially discuss why this is such a critical issue that impacts every aspect of modern society. As mentioned previously, digital platforms are quickly on pace to touch over half of the global economic output. Powering those digital platforms and systems is software, much of which is open source software (OSS) components. OSS use is ubiquitous across much of the modern software ecosystem. A recent study (<http://synopsys.com/content/dam/synopsys/sig-assets/reports/rep-ossra-2022.pdf>) found that 97 percent of modern software codebases contain OSS, and not only do they contain OSS, but over half of the codebases are OSS.

OSS is now fundamentally integrated into some of society's most critical infrastructure and systems. Research has shown that over 90 percent of codebases for industries such as transportation, financial services, and manufacturing contain OSS. The same trends exist across industries such as telecommunications and health care as well. In the United States, the Department of Defense (DoD) released a memo titled "Software Development and Open Source Software" (<http://dodcio.defense.gov/Portals/0/Documents/Library/SoftwareDev-OpenSource.pdf>). The memo, which is part of their broader Software Modernization Strategy (<https://dodcio.defense.gov/portals/0/documents/library/softwaredev-opensource.pdf>), calls OSS "the bedrock of the software-defined world and is critical in delivering software faster, resiliently, as is key to their software modernization efforts." The Software Modernization Strategy states that "the ability to securely and rapidly deliver resilient software capability is a competitive advantage that will define future conflicts."

Innovative use of software is not just critical for national security purposes, as emphasized by the DoD, but is pivotal for society at large. In a hearing as part of the U.S. House of Representatives Committee on Science, Space and Technology titled "Securing the Digital Commons: Improving the Health of the Open Source Software Ecosystem" (www.congress.gov/event/117th-congress/house-event/114727), several elected officials as well as industry experts testified to the importance of a resilient OSS ecosystem. Congressman Bill Foster called OSS the "hidden workforce of the digital ecosystem." Congresswoman Haley Stevens stated that "a vibrant open-source ecosystem is an engine for U.S. competitiveness and growth." In an Open Source Software Security Summit hosted by the White House in 2022, it was emphasized that most major software packages include OSS, including software that is used by the national security community (www.whitehouse.gov/briefing-room/statements-releases/2022/01/13/readout-of-white-house-meeting-on-software-security).

Many are beginning to make the case that OSS is such a vital aspect of society that it should be considered critical infrastructure as well, comparing it to interstate highways, power grids, water treatment, and other fundamental aspects of

our society (<http://hbr.org/2021/09/the-digital-economy-runs-on-open-source-heres-how-to-protect-it>). The argument also claims that designating OSS critical infrastructure would lead to additional resources, cross-sector coordination, public awareness, and dialogue.

Despite the ubiquity of OSS and its importance to national security, commercial industry, and society, it can be argued that its security concerns have been neglected. In an article titled “Open-Source Security: How Digital Infrastructure Is Built on a House of Cards” (<http://lawfareblog.com/open-source-security-how-digital-infrastructure-built-house-cards>), researcher Chinmayi Sharma makes the case that OSS and its associated vulnerabilities are pervasive across all critical infrastructure sectors and, due to a lack of resources and incentives, pose a systemic risk to society.

This lack of attention is not lost on malicious actors either, as some studies state we have seen software supply chain attacks experience as much as a 650 percent year-over-year (YoY) increase in 2021. This is not a unique phenomenon, with 2020 seeing an over 400 percent increase. These stark increases are also reflected in sources such as the Cloud Native Computing Foundation’s (CNCF) Catalog of Supply Chain Compromises.

This uptick in software supply chain attacks is also supported by research from organizations such as the Atlantic Council, in their “Breaking Trust: Shades of Crisis Across an Insecure Software Supply Chain” white paper (<http://atlanticcouncil.org/wp-content/uploads/2020/07/Breaking-trust-Shades-of-crisis-across-an-insecure-software-supply-chain.pdf>). Their research shows a sharp increase in software supply chain attacks, with third-party applications among the primary attack vectors, along with OSS. The report documents more than 100 software supply chain attacks over the last decade, through a myriad of vectors such as hijacked updates, malicious dependencies, compromised software development platforms, and account compromises. This demonstrates not just the consistent and increasing use of software supply chain attacks by malicious actors, but also the diversity of attack vectors that malicious actors can use to compromise downstream software consumers due to the complexity of the modern software ecosystem. As demonstrated by the report, not only are these attacks impacting millions of users, but they are also quickly becoming a standardized method of nation-state conflict and engagement in the modern digital society. The attention from malicious nation-state actors was also emphasized by ODNI in its 2017 publication (<http://dni.gov/files/NCSC/documents/supplychain/20190327-Software-Supply-Chain-Attacks02.pdf>).

It is not just OSS components that are being targeted. Malicious actors are also targeting managed service providers (MSPs), cloud service providers (CSPs), and several other entities, all of which play various roles in the modern software ecosystem.

Malicious actors have realized the fruitfulness of targeting software supply chain components or suppliers and causing a massive downstream impact,