EQUILIBRIUM

THE SECURITY BY DESIGN CONFERENCE

Securing the Software Supply Chain: Transparency in the Age of the Software Driven Society

Chris Hughes

President and Co-Founder at Aquia

Chris Hughes

President and Co-Founder, Aquia

EQUILIBRIUM

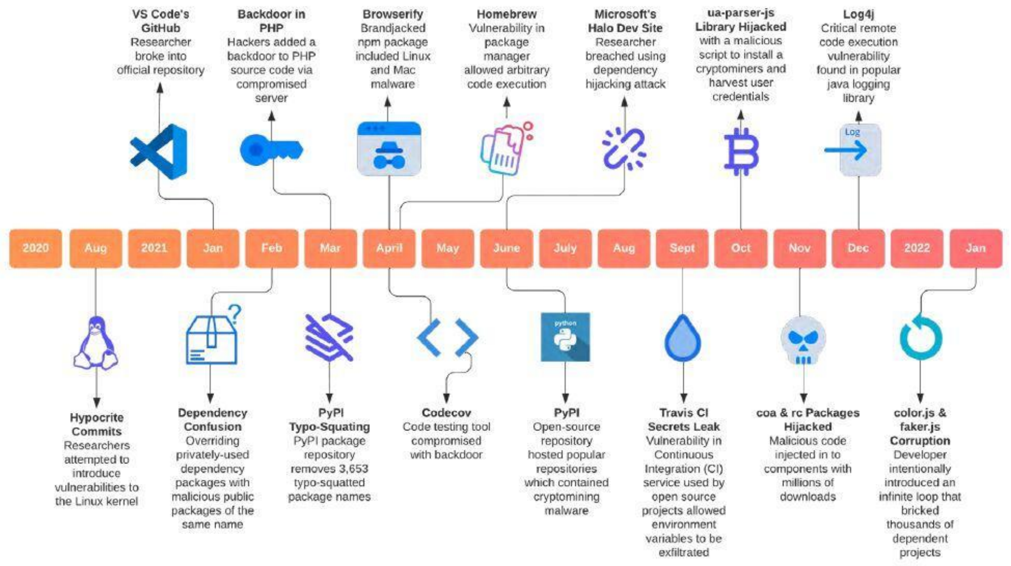
Open-Source Adoption – The <u>Good</u>

- Open Source expedites innovation
- Creates a robust community and ecosystem
- Enables cross-organizational collaboration
- Metrics:
  - 97% of organizations are using OSS
  - 77% of organizations have increased OSS use
  - 79% of organizations sponsor OSS organizations
  - Highest increases involve OSS DevOps and Cloud-native CI/CD Tools

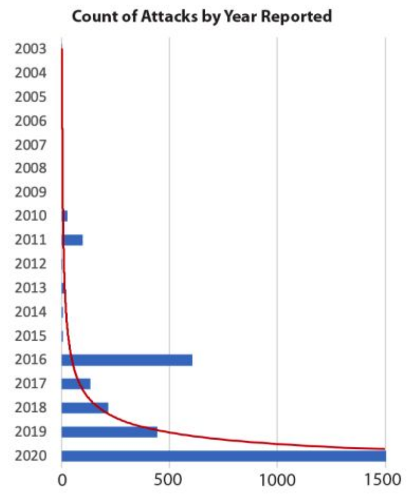
Linux Distributions & Virtualization
Red Hat
ubuntu
SUSE
KVM
Community Versions
debian
openSUSE
fedora
ClefOS
alpine
Networking & Monitoring
NGINX
HAPROXY
MESOS
Prometheus
ZABBIX
ZooKeeper
etcd
Cloud & Container Services
docker
LXD
openstack
kubernetes
minikube
HELM
Istio
okd
OPENSHIFT
APACHE
docker Compose
MARATHON
Sysdig
Terraform
kata containers
podman
Languages & Runtimes
Java
JS
R
python
Open Liberty
Ruby
Scala
GO
node
pypy
ERLANG
php
RAILS
OCaml
OpenJ9
OpenJDK
Apache Tomcat
Groovy
Clojure
HYPERLEDGER FABRIC
WildFly
TensorFlow
DevOps/Automation
CHEF
ANSIBLE
puppet
Jenkins
JMeter
Travis CI
ANTLR
Maven
sonarqube
Gradle
Zowe
SALTSTACK
Middleware
ACTIVEMQ
Camel
muleESB
RabbitMQ
APACHE GEODE
mosquitto
HIBERNATE
Big Data, Observability, Analytics
Flink
splunk>
Solr
Spark
kafka
Ignite
fluentd
Grafana
logstash
elasticsearch
kibana
Databases
CouchDB
mongoDB
MariaDB
RethinkDB
redis
PostgreSQL
MySQL
cassandra
SCYLLA
Couchbase

Open-Source Adoption – The Bad

- Experts estimate 60-80% of modern software is comprised of OSS (Linux Foundation)
- Software supply chain attacks on the rise
- Many projects supported by unpaid volunteers
- Incidents such as Log4j send organizations scrambling – lack of visibility at the component level



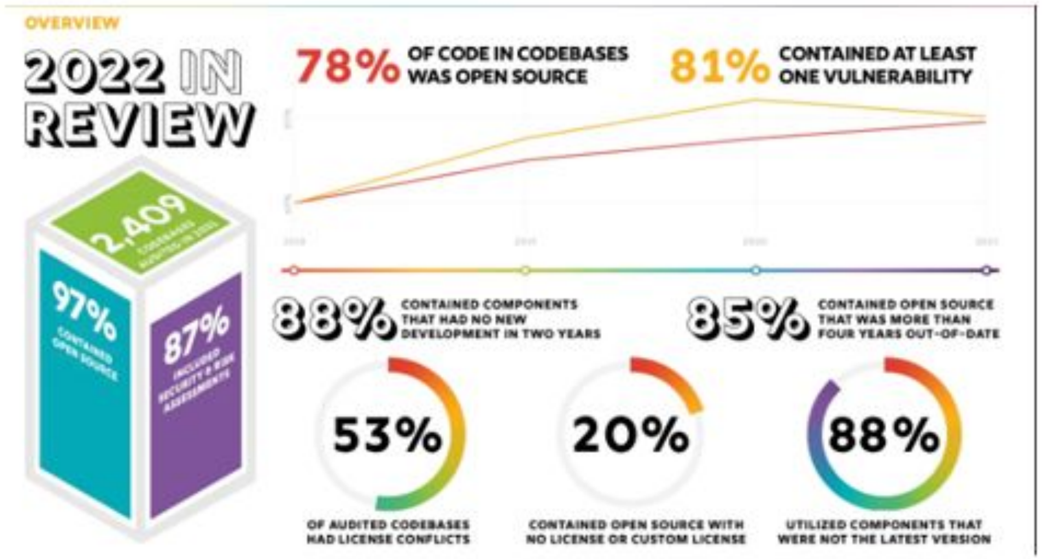
VS Code's GitHub Researcher broke into official repository
Backdoor in PHP Hackers added a backdoor to PHP source code via compromised server
Browserify Brandjacked npm package included Linux and Mac malware
Homebrew Vulnerability in package manager allowed arbitrary code execution
Microsoft's Halo Dev Site Researcher breached using dependency hijacking attack
ua-parser-js Library Hijacked with a malicious script to install a cryptominers and harvest user credentials
Log4j Critical remote code execution vulnerability found in popular java logging library
Log
2020
Aug
2021
Jan
Feb
Mar
April
May
June
July
Aug
Sept
Oct
Nov
Dec
2022
Jan
python
Hypocrite Commits Researchers attempted to introduce vulnerabilities to the Linux kernel
Dependency Confusion Overriding privately-used dependency packages with malicious public packages of the same name
PyPI Typo-Squating PyPI package repository removes 3,653 typo-squatted package names
Codecov Code testing tool compromised with backdoor
PyPI Open-source repository hosted popular repositories which contained cryptomining malware
Travis CI Secrets Leak Vulnerability in Continuous Integration (CI) service used by open source projects allowed environment variables to be exfiltrated
coa & rc Packages Hijacked Malicious code injected in to components with millions of downloads
color.js & faker.js Corruption Developer intentionally introduced an infinite loop that bricked thousands of dependent projects



Expansive Growth of Open Source Software (OSS)

We've tremendous growth of OSS adoption/use:

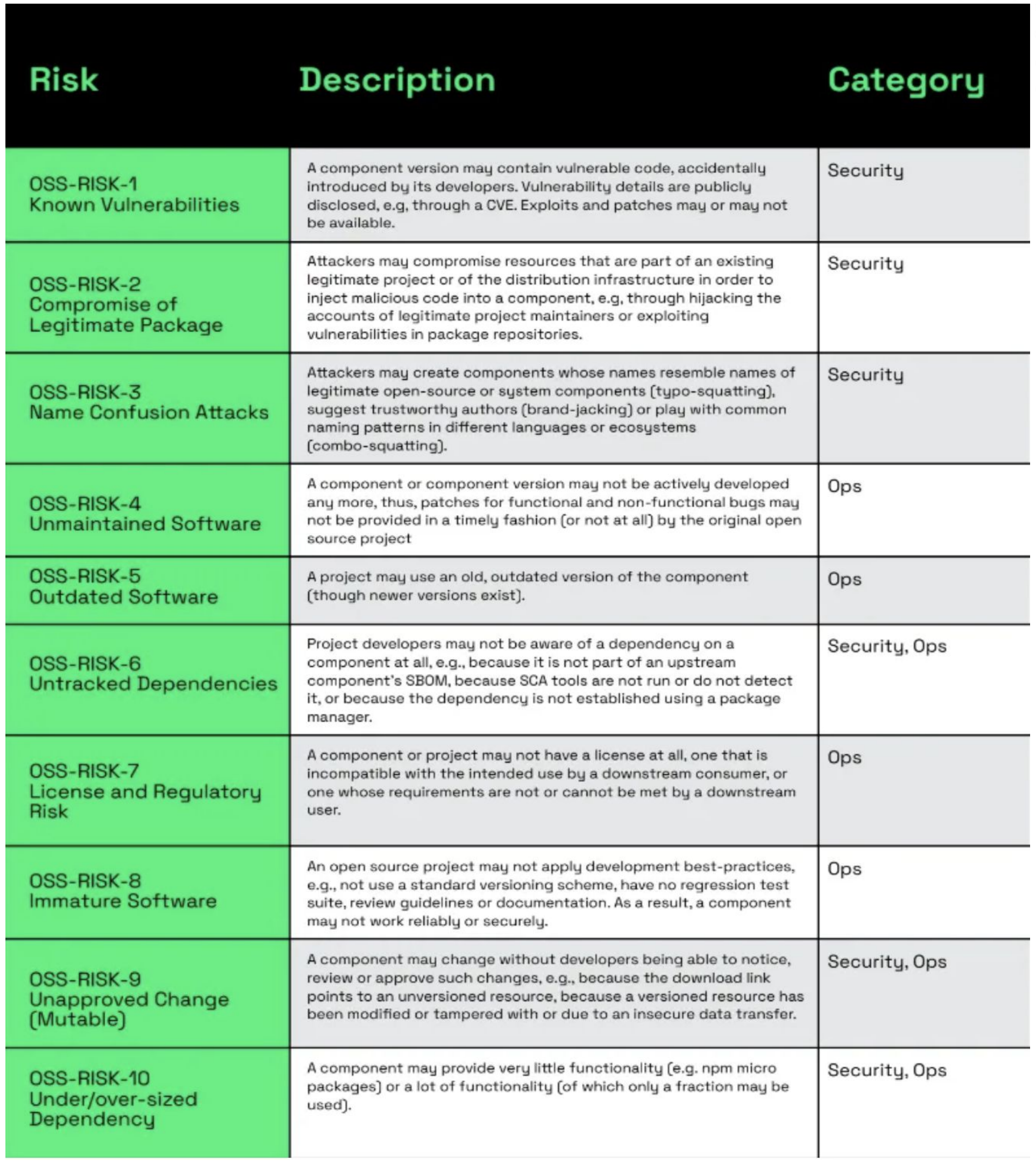
- 60-80% of modern codebases contain OSS
- 91% of those codebases are comprised of OSS
- Accelerates time
- Saves cost
- Fosters a thriving ecosystem and community
- High Bus Factor:
  - 25% of projects have ONE developer contributing code
  - 94% have 10 or fewer



Source: Synopsys Open Source Security and Risk Analysis Report 2022

Open Source Risk - Not Just Known Vulnerabilities

- CVE's are a problem, but not the full picture
- Need to look at *leading* indicators of risk
- OpenSSF Scorecard is a great resource
- Better selection of OSS components can save significant time later in the SDLC

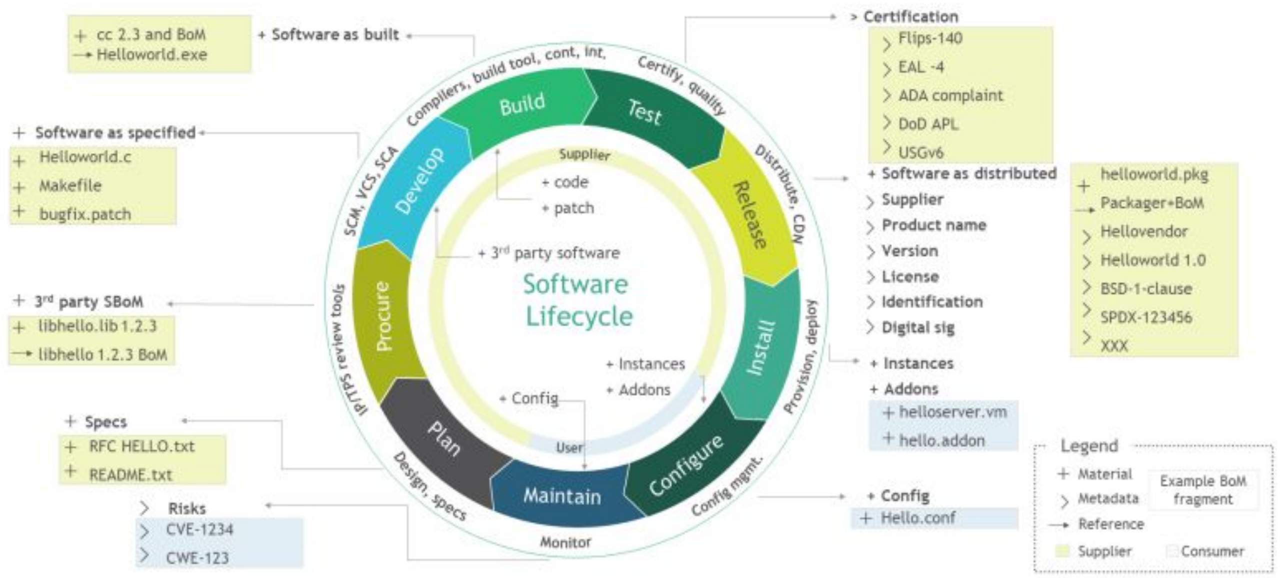


Risk	Description	Category
OSS-RISK-1 Known Vulnerabilities	A component version may contain vulnerable code, accidentally introduced by its developers. Vulnerability details are publicly disclosed, e.g, through a CVE. Exploits and patches may or may not be available.	Security
OSS-RISK-2 Compromise of Legitimate Package	Attackers may compromise resources that are part of an existing legitimate project or of the distribution infrastructure in order to inject malicious code into a component, e.g, through hijacking the accounts of legitimate project maintainers or exploiting vulnerabilities in package repositories.	Security
OSS-RISK-3 Name Confusion Attacks	Attackers may create components whose names resemble names of legitimate open-source or system components (typo-squatting), suggest trustworthy authors (brand-jacking) or play with common naming patterns in different languages or ecosystems (combo-squatting).	Security
OSS-RISK-4 Unmaintained Software	A component or component version may not be actively developed any more, thus, patches for functional and non-functional bugs may not be provided in a timely fashion (or not at all) by the original open source project	Ops
OSS-RISK-5 Outdated Software	A project may use an old, outdated version of the component (though newer versions exist).	Ops
OSS-RISK-6 Untracked Dependencies	Project developers may not be aware of a dependency on a component at all, e.g., because it is not part of an upstream component's SBOM, because SCA tools are not run or do not detect it, or because the dependency is not established using a package manager.	Security, Ops
OSS-RISK-7 License and Regulatory Risk	A component or project may not have a license at all, one that is incompatible with the intended use by a downstream consumer, or one whose requirements are not or cannot be met by a downstream user.	Ops
OSS-RISK-8 Immature Software	An open source project may not apply development best-practices, e.g., not use a standard versioning scheme, have no regression test suite, review guidelines or documentation. As a result, a component may not work reliably or securely.	Ops
OSS-RISK-9 Unapproved Change (Mutable)	A component may change without developers being able to notice, review or approve such changes, e.g., because the download link points to an unversioned resource, because a versioned resource has been modified or tampered with or due to an insecure data transfer.	Security, Ops
OSS-RISK-10 Under/over-sized Dependency	A component may provide very little functionality (e.g. npm micro packages) or a lot of functionality (of which only a fraction may be used).	Security, Ops

EQUILIBRIUM

OSS Adoption – What To Do?

- Establishing a robust Cybersecurity Supply Chain Risk Management (C-SCRM) program is a great step forward
- Engage with orgs such as OpenSSF, Linux Foundation and others
- Crowdsourcing is catching on NIST 800-161r1 – Cyber Supply Chain Risk Management Practices for Systems and Organizations – Appendix F
  - Foundational, Sustaining and Enhancing Capabilities
  - SCA/SBOM/VEX, Centralized Hardened Internal Repos of OSS etc.



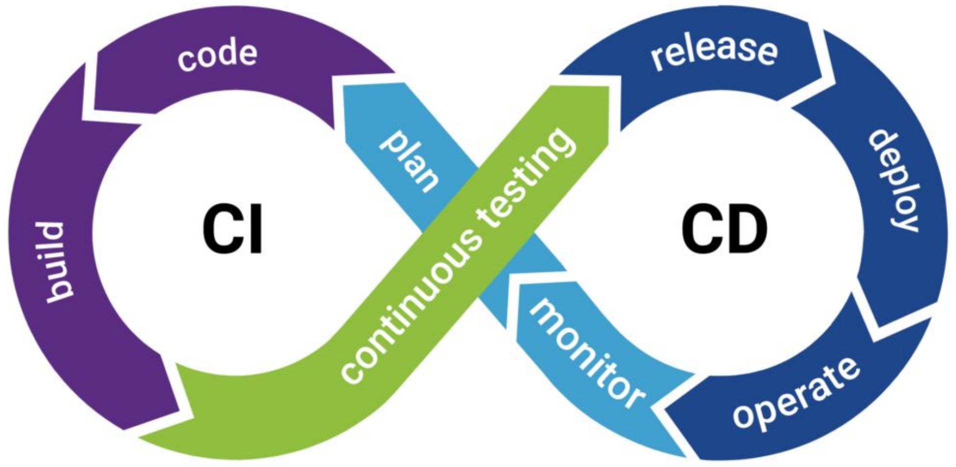
NIST Software Security in Supply Chains: Open-Source Software Controls

Published capabilities across levels of maturity

- **Foundational**
  - Utilize SSDF Protect/Response guidance
  - Ensure OS components are acquired via secure channels from trustworthy repos
- **Sustaining**
  - Utilize SCA on in-house codebases to look for vulnerable components
  - Create/maintain internal repos or libraries of known/good OSS components for developers to use
- **Enhancing**
  - Prioritize the use of more secure programming languages
  - Automate the pipeline of collecting, storing and scanning OSS components for internal repos prior to introduction to the dev environments
- **OMB Memo M-22-18, 23-16 and CISA Self-Attestation Form**
  - Agencies MUST obtain self-attestation to conformity with secure software development practices for all third-party software used by the agency (e.g. SSDF and NIST Cyber EO Software Supply Chain Guidance)
  - Agencies may determine a third-party assessment/3PAO is required
  - SBOM's may be required by agencies in solicitation requirements (must be in formats as defined by NTIA)

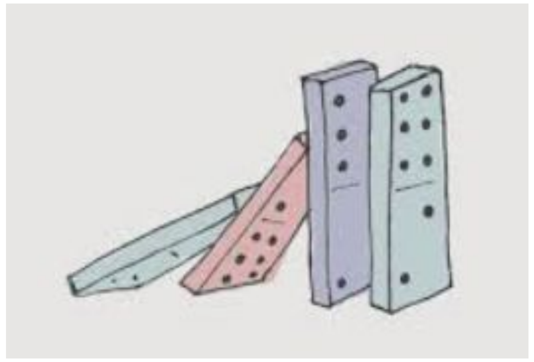
CI/CD Pipelines – The Good

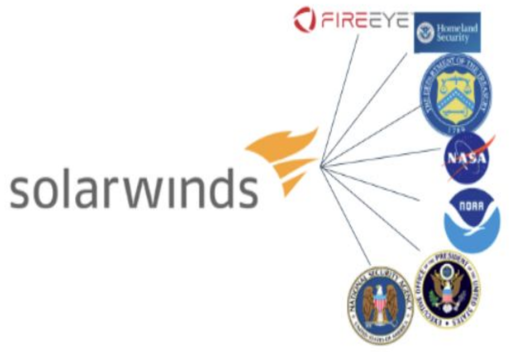
- CI/CD adoption has changed the way developers deliver software
- Has enabled security tooling automation and integration – e.g. “shifting security left”
- Enables robust toolchains to achieve full CI/CD capabilities and security requirements



CI/CD Pipelines – The Bad

- Many organizations haven't adopted unified CI platforms, leading to a myriad of integrations and complexity
- While the Pipeline(s) facilitate secure delivery, they are part of your attack surface – organizations must address this
- A compromise of the pipeline leads to massive supply chain security concerns and cascading impacts
- Malicious actors are even compromising signing systems and releasing signed malicious payloads

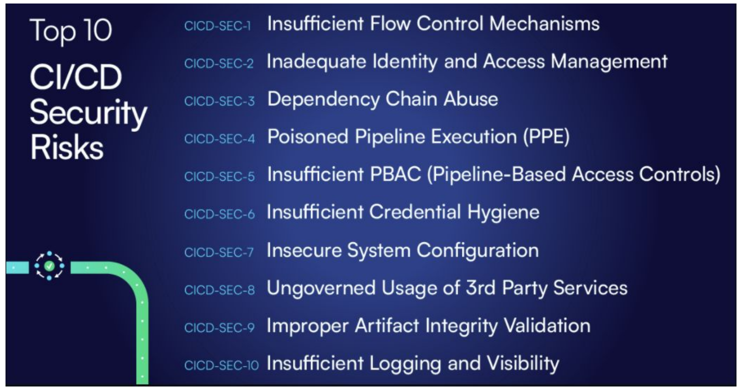




CI/CD Pipelines – What To Do

- Your CI/CD pipeline enables value delivery but also can be a threat vector
- OWASP has released an excellent CI/CD Risk Lists and Best Practices Guide
- Threat Model/Adversary Emulation
- Supply chain Levels for Software Artifacts (SLSA) - security framework
  - Prevent tampering
  - Improve Integrity
  - Secure Packages

SLSA



Top 10 CI/CD Security Risks

CICD-SEC-1	Insufficient Flow Control Mechanisms
CICD-SEC-2	Inadequate Identity and Access Management
CICD-SEC-3	Dependency Chain Abuse
CICD-SEC-4	Poisoned Pipeline Execution (PPE)
CICD-SEC-5	Insufficient PBAC (Pipeline-Based Access Controls)
CICD-SEC-6	Insufficient Credential Hygiene
CICD-SEC-7	Insecure System Configuration
CICD-SEC-8	Ungoverned Usage of 3rd Party Services
CICD-SEC-9	Improper Artifact Integrity Validation
CICD-SEC-10	Insufficient Logging and Visibility

Kubernetes & Containers

- Kubernetes and Containers are closely linked with Cloud-native architecture and DevSecOps Adoption
- Up to 75% of global organizations have adopted Containers
- Kubernetes is the de-facto Container Orchestration tool of choice
- Reduced development timelines, cost optimization and improved scalability

Don't Neglect Security

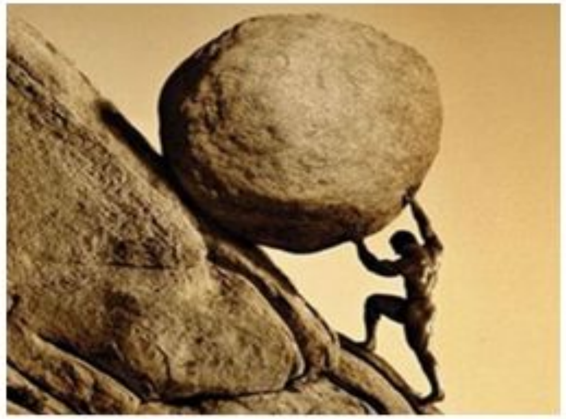
- Palo Alto's Unit 42 discovered 99% of Kubernetes Helm charts in Artifact Hub have insecure configurations
- Public Container Registries such as Docker Hub, Quay and Google Container Registry containers include critical findings in up to 91% of images
- Recommendations:
    - Utilize Container/Manifest Scanning
    - Pre-Hardened Images
    - Image Signing/Hashing
    - Leverage Guidance such as CIS, CNCF, DoD Container Hardening Guide and Kubernetes STIG
    - Scan Containers throughout lifecycle
    - Update IR Plans and Playbooks to account for Kubernetes and Containers
    - These insecure configurations and vulnerabilities exist in IaC too

SaaS Security - The Overlooked Software Supply Source

- Organizations are increasingly consuming applications and software in the form of SaaS
- Large enterprises are consuming upwards of 300~ SaaS applications, adding up to 10 new SaaS apps a month
- IT/Security control roughly 20% of SaaS usage
- SaaS consumers should implement SaaS Governance/Security, including SBOM's
- Twilio incident involved 130 other SaaS providers
- Incidents demonstrate even the most capable (e.g. MSFT, Okta etc.) are vulnerable

State of Vulnerability Backlogs

- 2022 saw a record 26,558 CVE's reported in NVD
- "Critical" vulnerabilities up 59% from 2021
- Report from Rezilion/Ponemon
  - 66% have a backlog of more than 100,000 vulnerabilities
  - Average number of vulnerabilities in backlog is *1.1 million*
- Cyentia Institute found:
  - Organizations typically have the capacity to remediate 1 out of 10 vulnerabilities in their environment in a given month



4

Vulnerability Scoring and Prioritization Struggles

- CVE Growth in NVD
  - 200,000+
  - 20,000+ in 2023 alone
  - Almost 15% YoY growth
- Historically, organizations have used CVSS Severity Scores to prioritize vulnerabilities
- This is problematic, because less than 5% of *all* known CVE's are ever exploited
- Organizations are wasting tremendous time, effort and energy prioritizing vulnerabilities that are unlikely to ever be exploited and present little risk

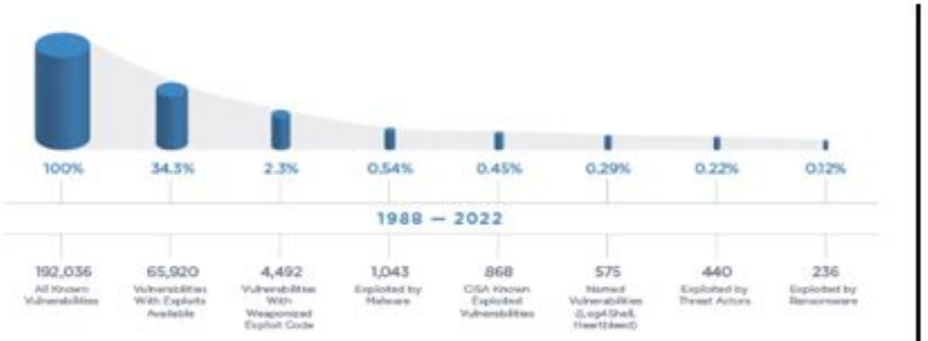


Figure 2: Evolution of vulnerability threat landscape, 1988 – 2022

Source: Qualys TruRisk Report 2022

8

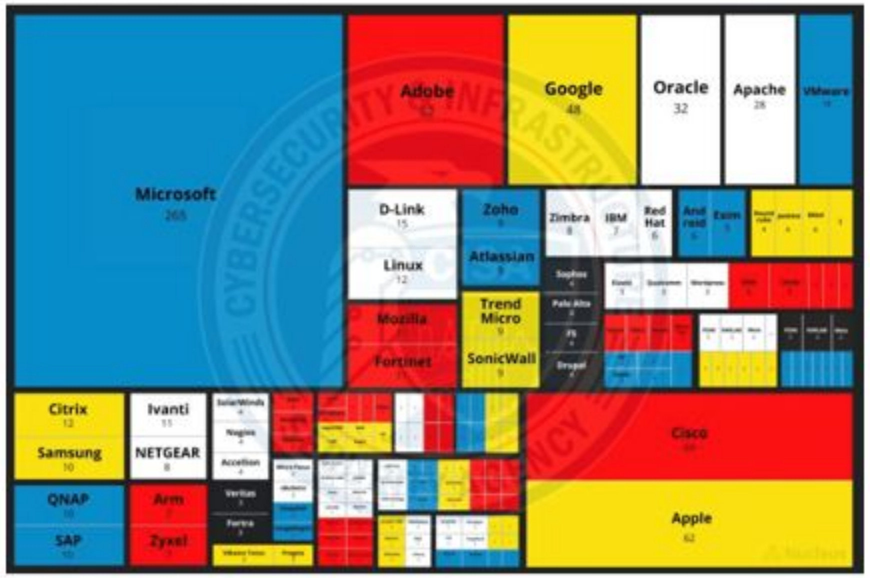
Emerging Vulnerability Scoring and Prioritization Systems

- As we now realize the challenges of legacy/traditional scoring and prioritization, several others have emerged:
  - CISA Known Exploited Vulnerability (KEV) catalogue
  - Exploit Prediction Scoring System (EPSS)
  - Stakeholder Specific Vulnerability Categorization (SSCV)

Emerging Vulnerability Scoring and Prioritization Systems – CISA KEV

- Launched November 2021 as part of Binding Operational Directive (BOD) 22-01
- Helps Federal agencies (and commercial entities) prioritize ***known exploited*** vulnerabilities
- Recently hit 1,000 vulnerabilities listed
- To appear on the KEV, must:
  - Be assigned a CVE identifier
  - Be under active or attempted success exploitation
  - Has *clear* remediation guidance (e.g. patches/mitigations)

CYBERSECURITY & INFRASTRUCTURE SECURITY AGENCY

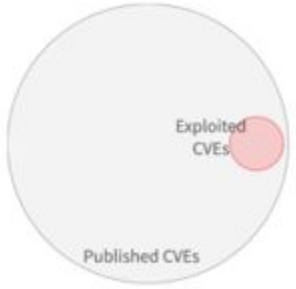


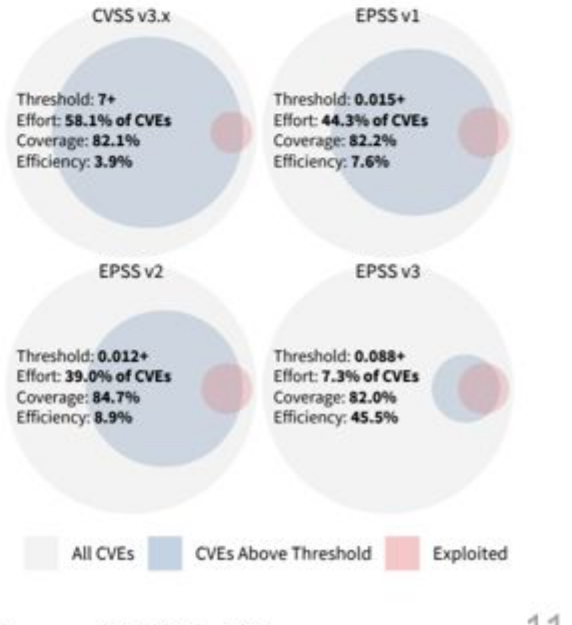
Source: Patrick Garrity @ Nucleus Security

10

Emerging Vulnerability Scoring and Prioritization Systems – Exploit Prediction Scoring System (EPSS)

- As discussed, only 2-7% of vulnerabilities are ***ever*** seen to be exploited in the wild
- EPSS produces a probability score between 0 and 1 (0% and 100%) that a vulnerability will be exploited in the next 30 days
- Uses a variety of data sources, such as:
  - Published CVE's
  - Published exploit code
  - Exploitation-in-the-wild activity
  - And more

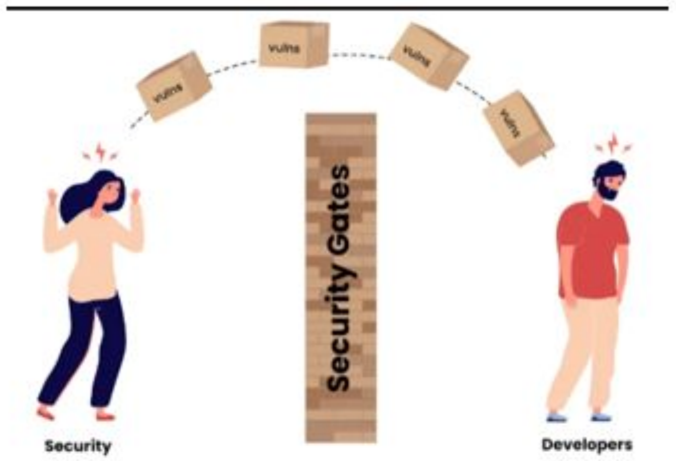


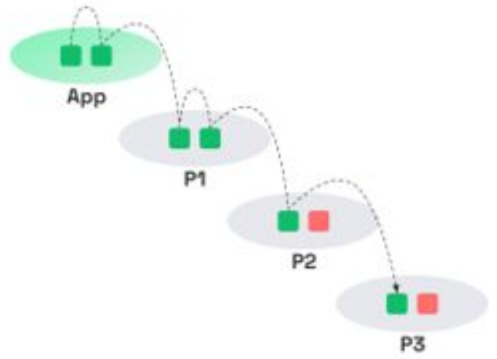


Source: EPSS 3.0 whitepaper

Throwing Toil and Building Silos

- Despite all of the talk of "breaking down silos" and DevSecOps, we (Security) are generally throwing toil over the fence
- Vulnerability with little to no context
- "guilty until proven innocent" mindset
- Erecting "gates" with little context into key things we've discussed
- We've shifted toil left (e.g. SAST, DAST, IaC, SCA et al)
- Many organizations still primarily use CVSS for prioritization, without taking into consideration:
  - Known Exploitation (e.g. CISA KEV)
  - Exploitation Probability (e.g. EPSS)
  - Exploitability (e.g. Reachability Analysis, Architecture etc.)
  - Business Context/Criticality (e.g. data sensitivity, mission essentiality)





Source: Endor Labs State of Dependency Management 2022

13

Notable Industry Efforts

- White House held Software Security Summit in early 2022
- OpenSSF Published OSS Security Mobilization Plan
- 3 High Level Goals
  - Securing OSS Production
  - Improving Vulnerability Discovery & Remediation
  - Shorten Ecosystem Patching Response Time
- Key Focus Areas:
  - Developer Education/Certification
  - Digital Signatures
  - OpenSSF IR Team
  - SBOM Everywhere
  - Risk Assessment Dashboard – 10k OSS Projects

OpenSSF
OPEN SOURCE SECURITY FOUNDATION

THE LINUX FOUNDATION

WHITEPAPER
The Open Source Software Security Mobilization Plan

CISA Secure-by-Design Initiative

- Looking to build on momentum from the latest National Cybersecurity Strategy
- Shift burden from consumers/customers to suppliers
- Issued v2 of their Secure-by-Design/Default principles
- Series of Secure-by-Design Alerts/Blogs (Default Passwords, SQLi etc)

Guidance Galore

- NIST Secure Software Development Framework (SSDF)
- Supply Chain Levels for Software Artifacts (SLSA)
- NSA/CISA - Securing the Software Supply Chain for Developers
- OWASP Software Component Verification Standard (SCVS)
- Cloud Native Computing Foundation (CNCF) - Software Supply Chain Best Practices
- CISA Secure-by-Design Publications

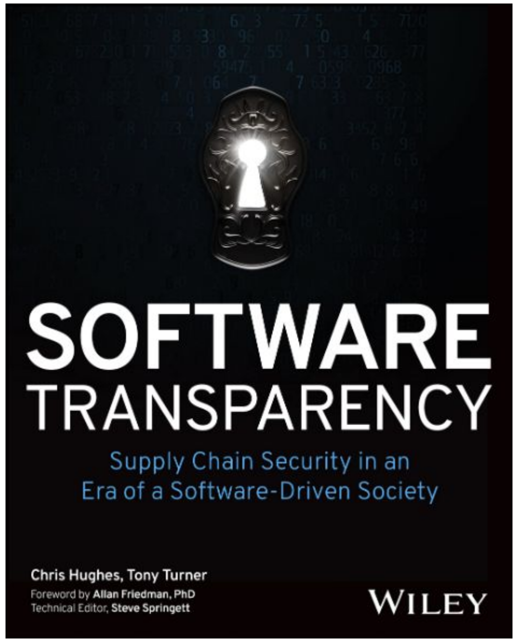
Long story short, we have no shortage of guidance and emerging best practices but we need to bridge the divide from theory to practice.

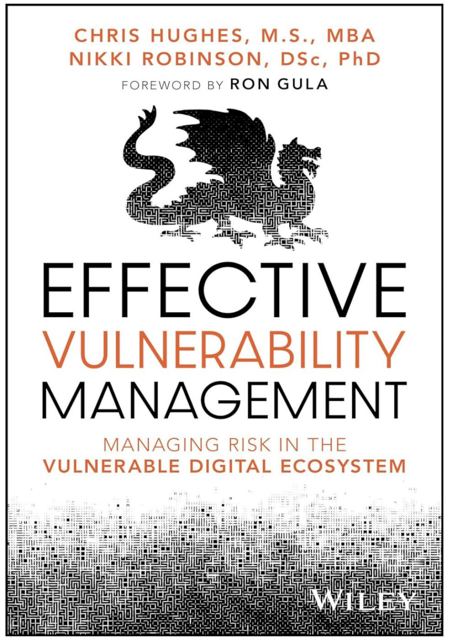
We're best practices rich and implementation poor

Getting Started

Organizations can get started by:

- Implementing the Guidance from NIST and others
  - Establishing Known Trusted Repositories of Software Components
  - Understand where components exist in your enterprise and among vendors and be prepared to respond *when* something happens
  - Creating capabilities for ingesting, generating, enriching and utilizing SBOM's in industry standardized formats
  - Focus on mature vulnerability intelligence and prioritization
  - Implement mature governance around OSS consumption and usage
  - Establish SaaS Security & Governance programs and efforts
- Questions? Chris@aquia.us


SOFTWARE TRANSPARENCY
Supply Chain Security in an Era of a Software-Driven Society
Chris Hughes, Tony Turner
Foreword by Allan Friedman, PhD
Technical Editor, Steve Springett
WILEY


CHRIS HUGHES, M.S., MBA
NIKKI ROBINSON, DSc, PhD
FOREWORD BY RON GULA
EFFECTIVE VULNERABILITY MANAGEMENT
MANAGING RISK IN THE VULNERABLE DIGITAL ECOSYSTEM
WILEY

Thank You

For more information, contact me at chris@aquia.us.

Aq Aquia