# Cascading Supply Chain Attacks: What Threat Intel & AppSec Teams Can Learn From The Next Generation Of Supply Chain Attacks

// Ali N. Khan
Field CISO - ReversingLabs

10/19/2023

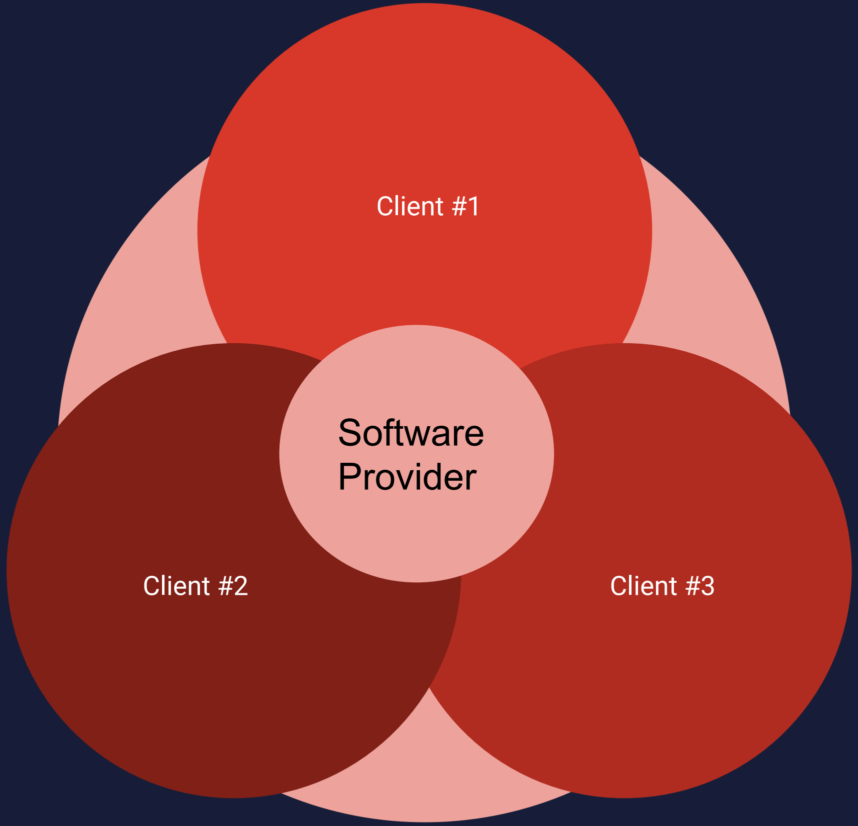
REVERSINGLABS
GitHub
CircleCI
Twilio
3CX
MoveIT
SolarWinds

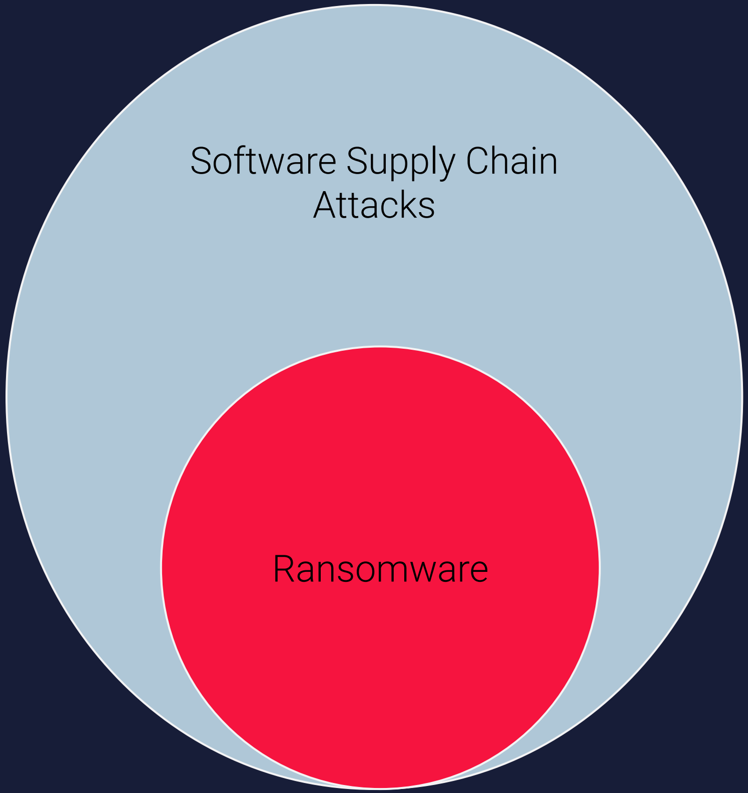
# What Is A Software Supply Chain Attack ?

A software supply chain attack occurs when a cyber threat actor infiltrates a software vendor's network and employs malicious code to compromise the software before the vendor sends it to their customers. The compromised software then compromises the customer's data or system. *- CISA*

Compromising software code through cyber attacks, insider threats, other close access activities at any phase of the supply chain to infect an unsuspecting customer.
*- DNI*

# Blast Radius





# Building Pillars Of Trust In Software

## CISO

- Budgeting
- Program Development
- Skills Gap
- Fusion Center
- CIO/CTO Alignment
- ASPM / TPRM

## Threat Intel

- Strategic Intelligence
- Tactical Intelligence
- Operational Intelligence
- Nation State
- Liability / Legal Risk
- Vuln Management

## AppSec

- Collaborate w/ Dev
- DevSecOps
- CI/CD
- Post-Build
- Pre-Deploy
- Modernize to Product Security

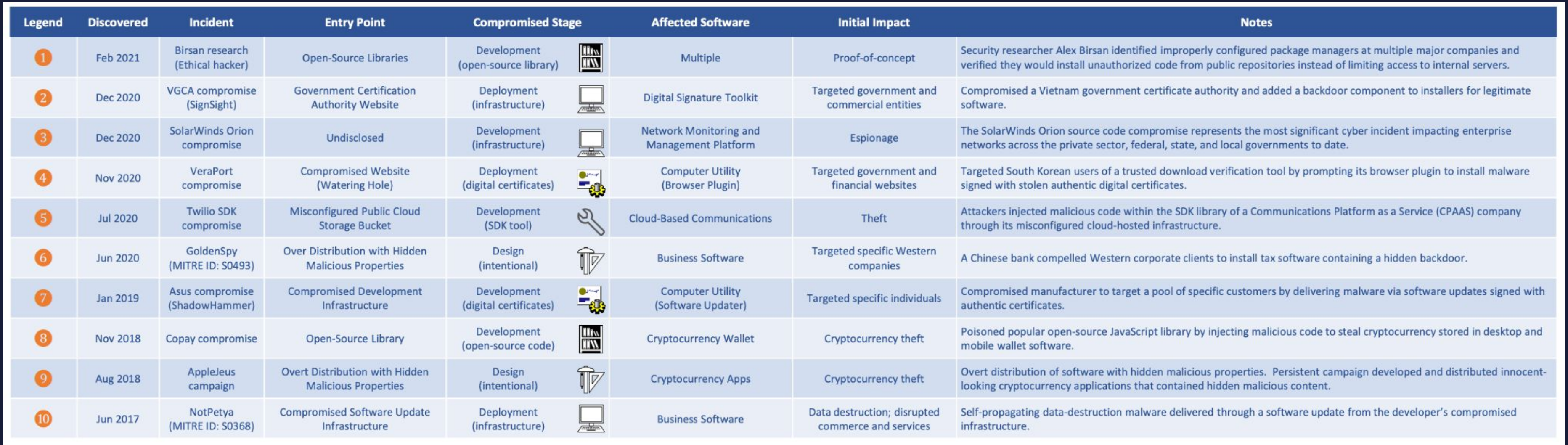
## TPRM

- Automate SRA
- Vendor Onboarding
- SBOM
- Inventory
- Integrity
- Assurance

## SOC / PSIRT / Fusion Center

# Threat Intel

# Third Party Risk Management Intel



| Legend | Discovered | Incident | Entry Point | Compromised Stage | Affected Software | Initial Impact | Notes |
|---|---|---|---|---|---|---|---|
| 1 | Feb 2021 | Birsan research (Ethical hacker) | Open-Source Libraries | Development (open-source library) | Multiple | Proof-of-concept | Security researcher Alex Birsan identified improperly configured package managers at multiple major companies and verified they would install unauthorized code from public repositories instead of limiting access to internal servers. |
| 2 | Dec 2020 | VGCA compromise (SignSight) | Government Certification Authority Website | Deployment (infrastructure) | Digital Signature Toolkit | Targeted government and commercial entities | Compromised a Vietnam government certificate authority and added a backdoor component to installers for legitimate software. |
| 3 | Dec 2020 | SolarWinds Orion compromise | Undisclosed | Development (infrastructure) | Network Monitoring and Management Platform | Espionage | The SolarWinds Orion source code compromise represents the most significant cyber incident impacting enterprise networks across the private sector, federal, state, and local governments to date. |
| 4 | Nov 2020 | VeraPort compromise | Compromised Website (Watering Hole) | Deployment (digital certificates) | Computer Utility (Browser Plugin) | Targeted government and financial websites | Targeted South Korean users of a trusted download verification tool by prompting its browser plugin to install malware signed with stolen authentic digital certificates. |
| 5 | Jul 2020 | Twilio SDK compromise | Misconfigured Public Cloud Storage Bucket | Development (SDK tool) | Cloud-Based Communications | Theft | Attackers injected malicious code within the SDK library of a Communications Platform as a Service (CPAAS) company through its misconfigured cloud-hosted infrastructure. |
| 6 | Jun 2020 | GoldenSpy (MITRE ID: S0493) | Over Distribution with Hidden Malicious Properties | Design (intentional) | Business Software | Targeted specific Western companies | A Chinese bank compelled Western corporate clients to install tax software containing a hidden backdoor. |
| 7 | Jan 2019 | Asus compromise (ShadowHammer) | Compromised Development Infrastructure | Development (digital certificates) | Computer Utility (Software Updater) | Targeted specific individuals | Compromised manufacturer to target a pool of specific customers by delivering malware via software updates signed with authentic certificates. |
| 8 | Nov 2018 | Copay compromise | Open-Source Library | Development (open-source code) | Cryptocurrency Wallet | Cryptocurrency theft | Poisoned popular open-source JavaScript library by injecting malicious code to steal cryptocurrency stored in desktop and mobile wallet software. |
| 9 | Aug 2018 | AppleJeus campaign | Overt Distribution with Hidden Malicious Properties | Design (intentional) | Cryptocurrency Apps | Cryptocurrency theft | Overt distribution of software with hidden malicious properties. Persistent campaign developed and distributed innocent-looking cryptocurrency applications that contained hidden malicious content. |
| 10 | Jun 2017 | NotPetya (MITRE ID: S0368) | Compromised Software Update Infrastructure | Deployment (infrastructure) | Business Software | Data destruction; disrupted commerce and services | Self-propagating data-destruction malware delivered through a software update from the developer's compromised infrastructure. |

Source: ODNI

REVERSINGLABS

# Software Intelligence

## SOLARWINDS - BUILD ENVIRONMENT COMPROMISE

**SolarWinds Build infrastructure**

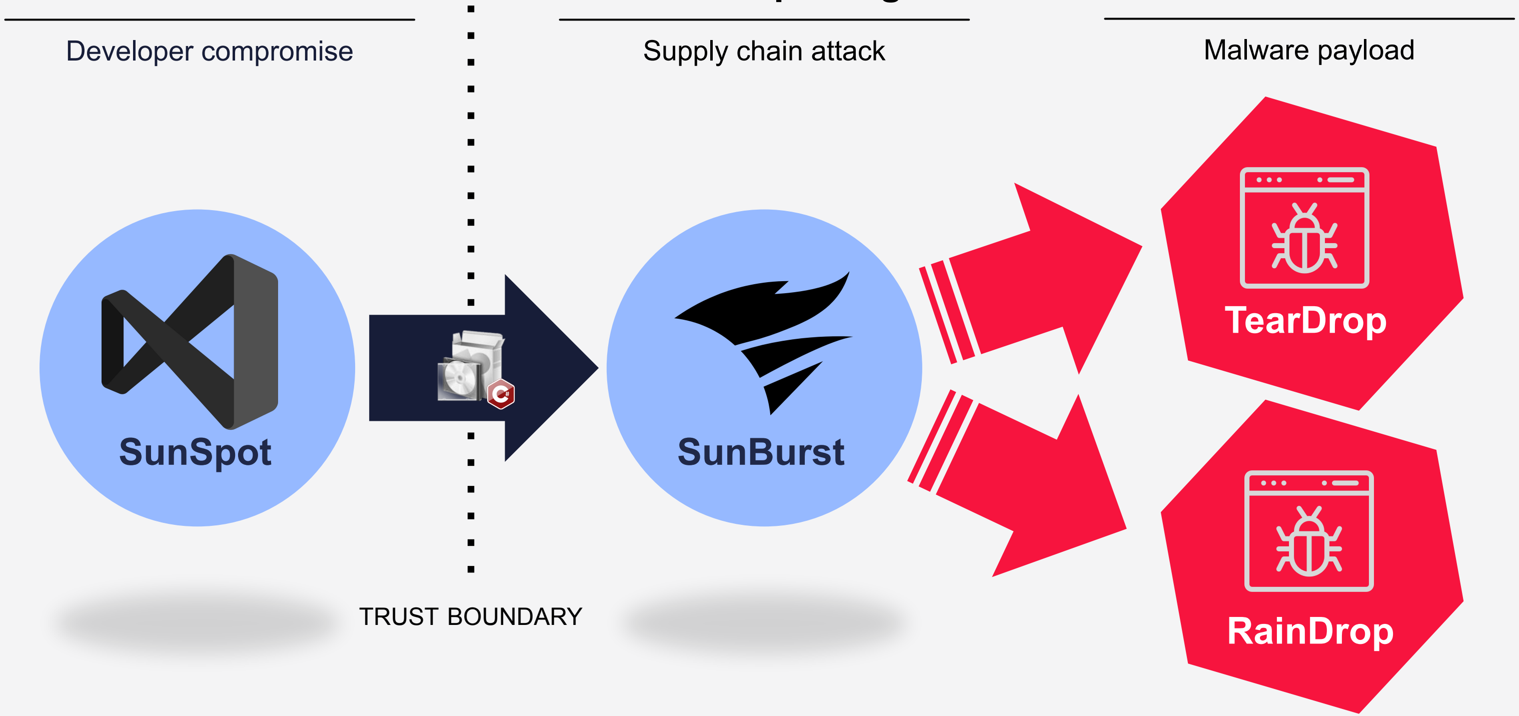
Developer compromise

**Orion Software package**

Supply chain attack

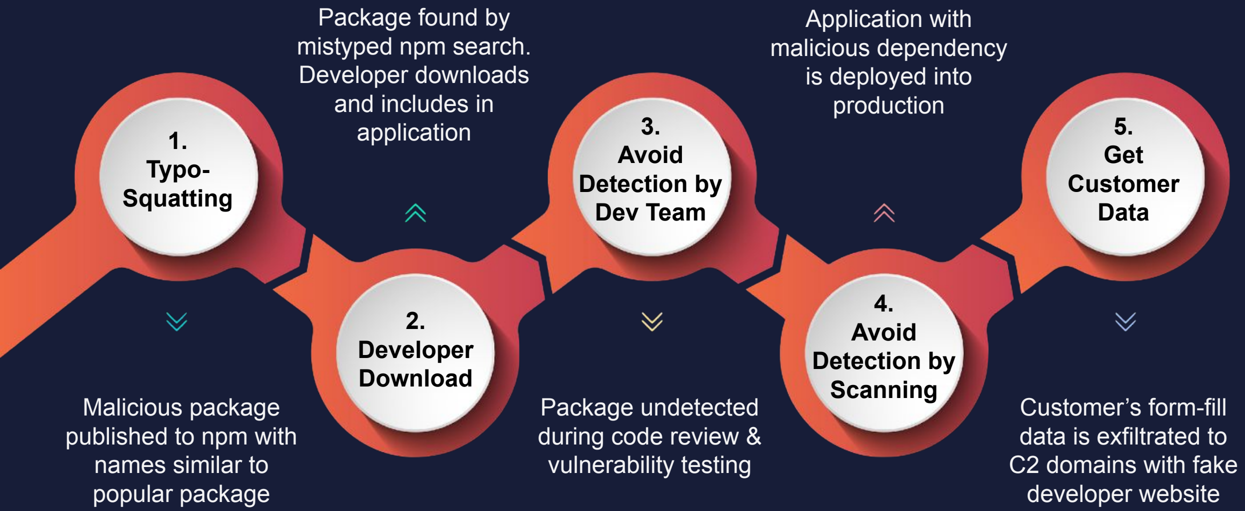
**Targeted customers**

Malware payload



# Evolving Threat Landscape

## IconBurst - Attack Path Explained



1. Typo-Squatting

Malicious package published to npm with names similar to popular package

2. Developer Download

Package found by mistyped npm search. Developer downloads and includes in application

3. Avoid Detection by Dev Team

Package undetected during code review & vulnerability testing

4. Avoid Detection by Scanning

Application with malicious dependency is deployed into production

5. Get Customer Data

Customer's form-fill data is exfiltrated to C2 domains with fake developer website

### Executive Summary

**What:** 100+ malicious Javascript packages

- Steals **users'** form-fill data
- Identified by finding obfuscation in open source

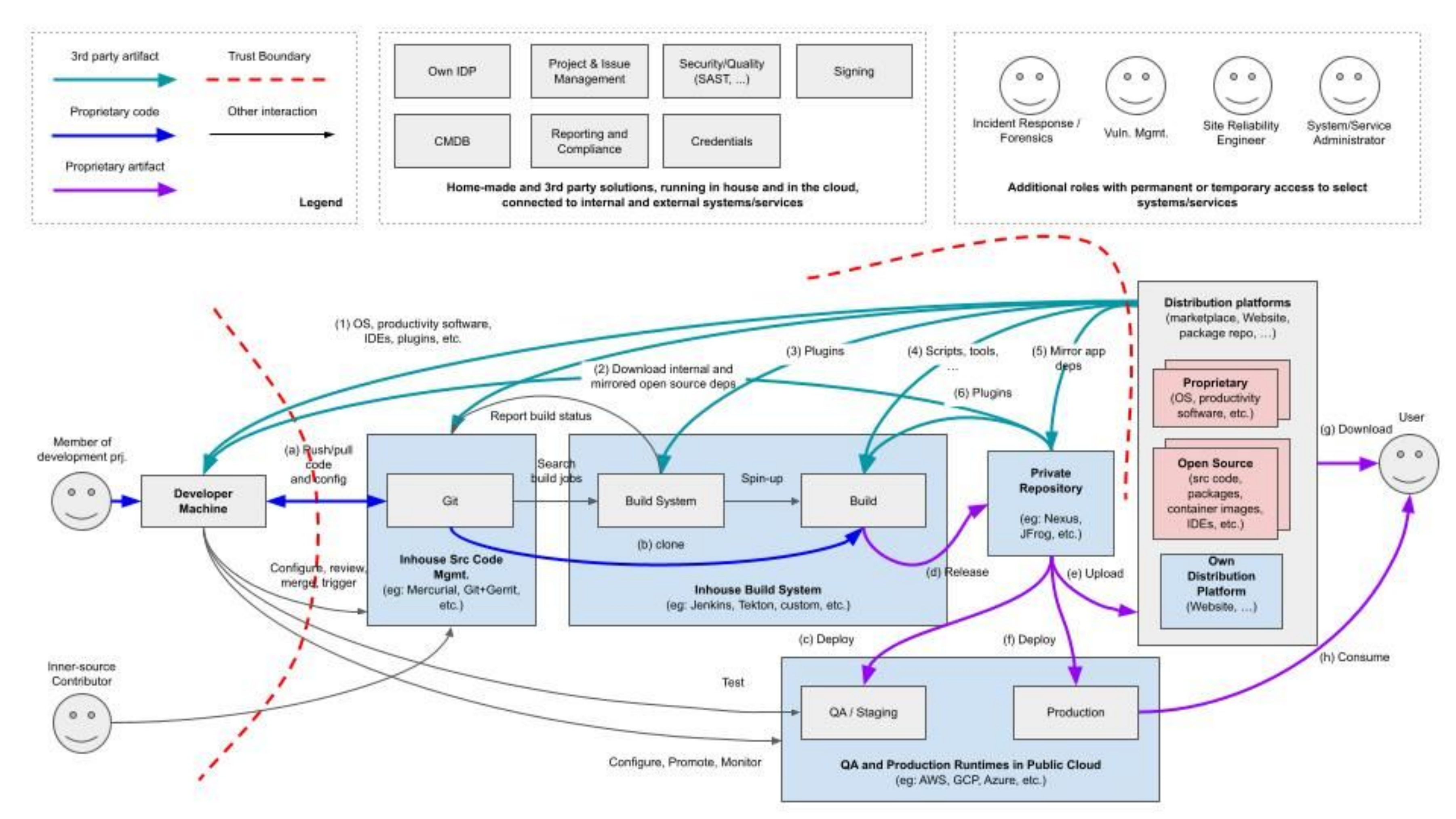
**Where:** npmjs.com

**When:** On-going – **17,000+** downloads

- Began Feb 17 2022

**Why**: Data can be used for:

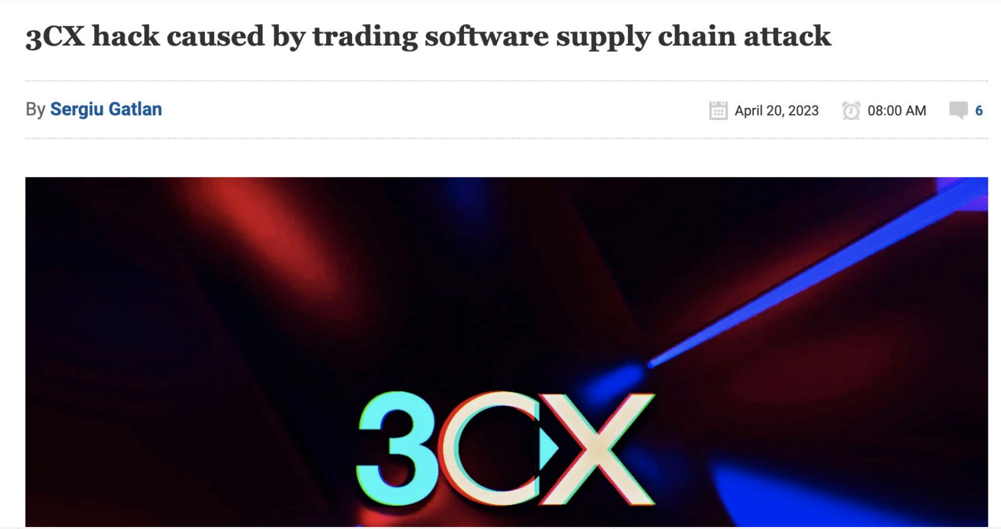
- Identity theft
- Recon for future attacks


3rd party artifact
Trust Boundary
Proprietary code
Other interaction
Proprietary artifact
Legend
Own IDP
Project & Issue Management
Security/Quality (SAST, ...)
Signing
CMDB
Reporting and Compliance
Credentials
Home-made and 3rd party solutions, running in house and in the cloud, connected to internal and external systems/services
Incident Response / Forensics
Vuln. Mgmt.
Site Reliability Engineer
System/Service Administrator
Additional roles with permanent or temporary access to select systems/services
(1) OS, productivity software, IDEs, plugins, etc.
(2) Download internal and mirrored open source deps
(3) Plugins
(4) Scripts, tools, ...
(5) Mirror app deps
(6) Plugins
Distribution platforms (marketplace, Website, package repo, ...)
Proprietary (OS, productivity software, etc.)
Open Source (src code, packages, container images, IDEs, etc.)
Own Distribution Platform (Website, ...)
Report build status
Member of development prj.
Developer Machine
(a) Push/pull code and config
Git
Search build jobs
Build System
Spin-up
Build
(b) clone
Inhouse Src Code Mgmt. (eg: Mercurial, Git+Gerrit, etc.)
Inhouse Build System (eg: Jenkins, Tekton, custom, etc.)
Private Repository (eg: Nexus, JFrog, etc.)
(d) Release
(e) Upload
(g) Download
User
Configure, review, merge, trigger
Inner-source Contributor
(c) Deploy
(f) Deploy
Test
QA / Staging
Production
(h) Consume
Configure, Promote, Monitor
QA and Production Runtimes in Public Cloud (eg: AWS, GCP, Azure, etc.)

*Source: OpenSSF*

# Cascading Supply Chain Attack

## Proposed Attack Scenario Chain - Confirmed By Mandiant



### 3CX hack caused by trading software supply chain attack

By **Sergiu Gatlan**

April 20, 2023 08:00 AM 6

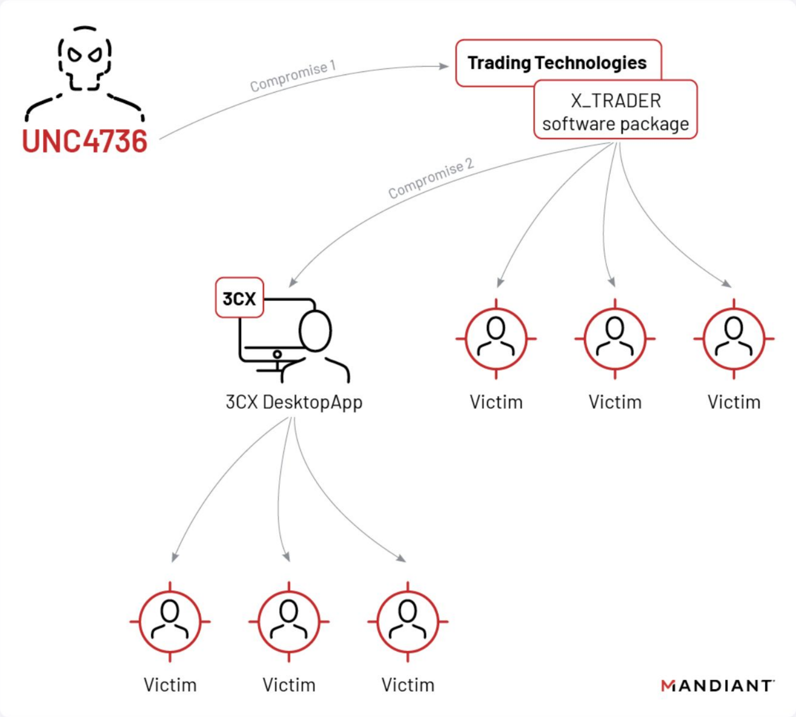


*Figure 1: 3CX software supply chain compromise linked to Trading Technologies software supply chain compromise*

*Source: Arstechnica*

# 3CX Tampering

Digital Signature Tampering

**X Trader Software Package**

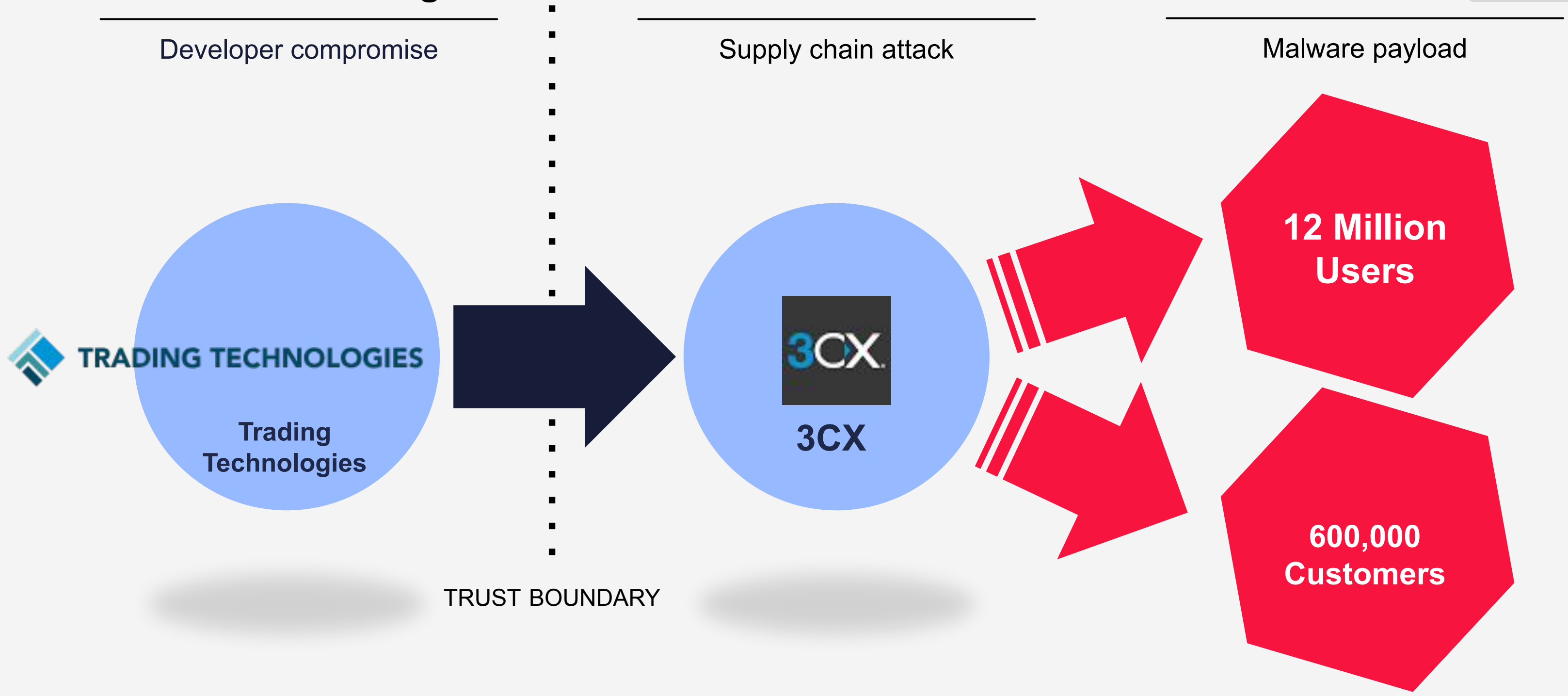
Developer compromise

**3CX Desktop Application (Windows & Apple)**

Supply chain attack

**Targeted customers**

Malware payload



TRADING TECHNOLOGIES

Trading Technologies

3CX

3CX

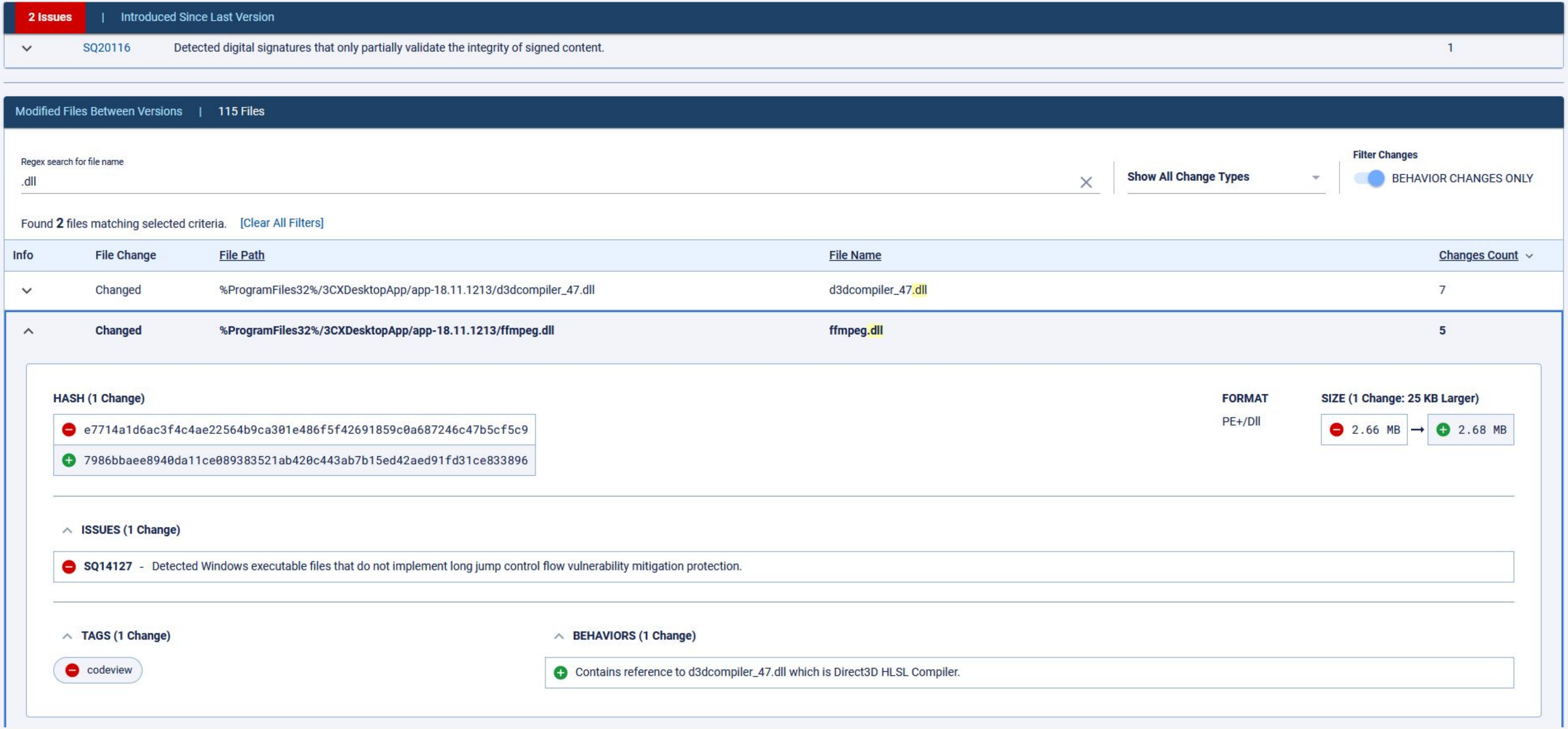
TRUST BOUNDARY

12 Million Users

600,000 Customers

ЯEVERSINGLABS

# 3CX Build Environment Targeted



**2 Issues** | Introduced Since Last Version

SQ20116 Detected digital signatures that only partially validate the integrity of signed content. 1

Modified Files Between Versions | 115 Files

Regex search for file name: .dll

Show All Change Types

Filter Changes: BEHAVIOR CHANGES ONLY

Found **2** files matching selected criteria. [Clear All Filters]

| Info | File Change | File Path | File Name | Changes Count |
|---|---|---|---|---|
|  | Changed | %ProgramFiles32%/3CXDesktopApp/app-18.11.1213/d3dcompiler_47.dll | d3dcompiler_47.dll | 7 |
|  | **Changed** | **%ProgramFiles32%/3CXDesktopApp/app-18.11.1213/ffmpeg.dll** | **ffmpeg.dll** | **5** |

**HASH (1 Change)**

```
⊖ e7714a1d6ac3f4c4ae22564b9ca301e486f5f42691859c0a687246c47b5cf5c9
⊕ 7986bbaee8940da11ce089383521ab420c443ab7b15ed42aed91fd31ce833896
```

**FORMAT**

PE+/Dll

**SIZE (1 Change: 25 KB Larger)**

⊖ 2.66 MB → ⊕ 2.68 MB

**ISSUES (1 Change)**

⊖ **SQ14127** - Detected Windows executable files that do not implement long jump control flow vulnerability mitigation protection.

**TAGS (1 Change)**

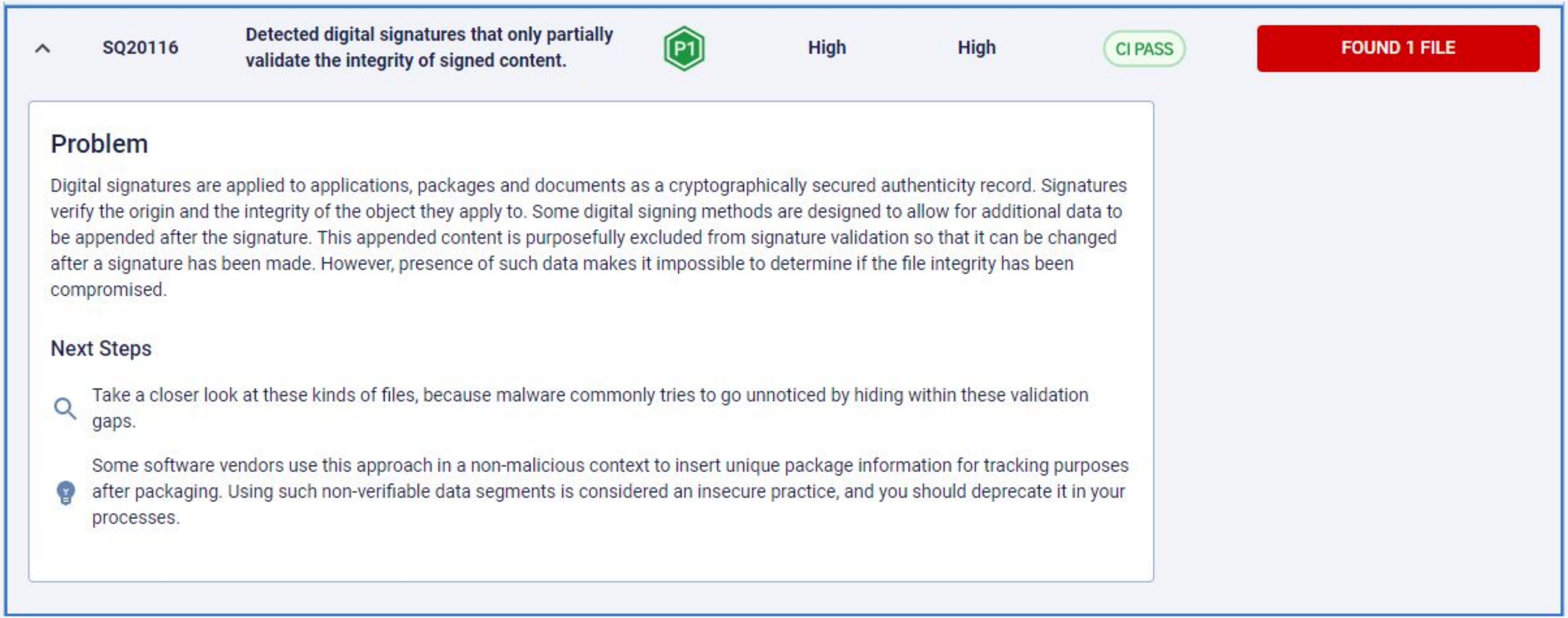
⊖ codeview

**BEHAVIORS (1 Change)**

⊕ Contains reference to d3dcompiler_47.dll which is Direct3D HLSL Compiler.

# 3CX - Trustable Verdict

- Embracing stricter security rules inside their business environment
- Security review of the release artifacts -> behavioral differences between the versions



SQ20116 | Detected digital signatures that only partially validate the integrity of signed content. | P1 | High | High | CI PASS | FOUND 1 FILE

**Problem**

Digital signatures are applied to applications, packages and documents as a cryptographically secured authenticity record. Signatures verify the origin and the integrity of the object they apply to. Some digital signing methods are designed to allow for additional data to be appended after the signature. This appended content is purposefully excluded from signature validation so that it can be changed after a signature has been made. However, presence of such data makes it impossible to determine if the file integrity has been compromised.

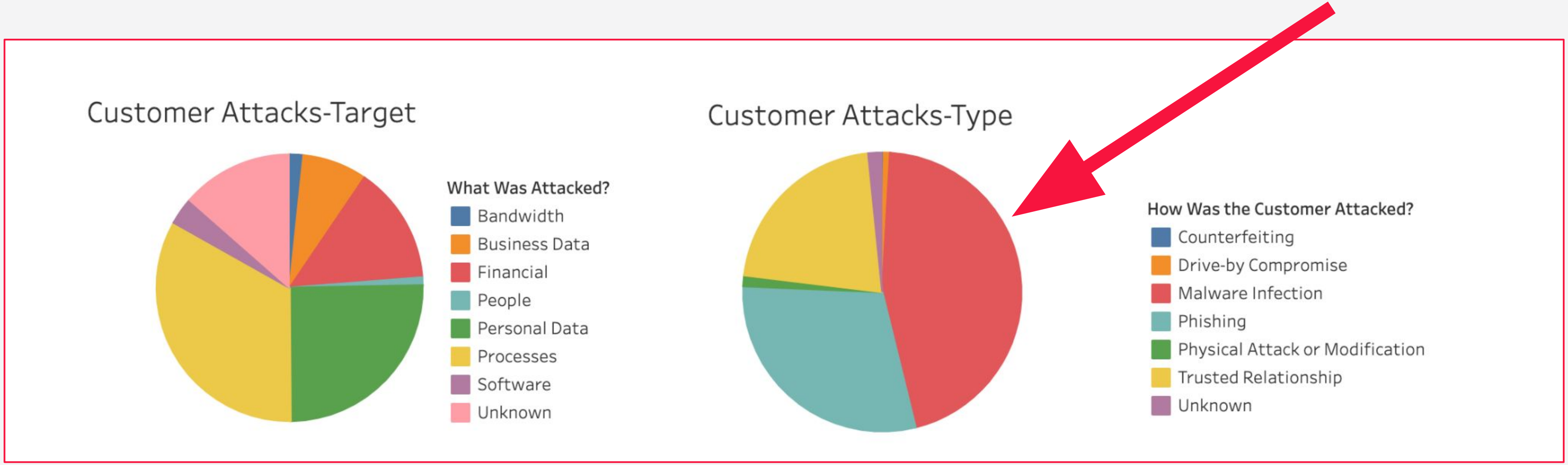
**Next Steps**

Take a closer look at these kinds of files, because malware commonly tries to go unnoticed by hiding within these validation gaps.

Some software vendors use this approach in a non-malicious context to insert unique package information for tracking purposes after packaging. Using such non-verifiable data segments is considered an insecure practice, and you should deprecate it in your processes.

# Malware Intelligence

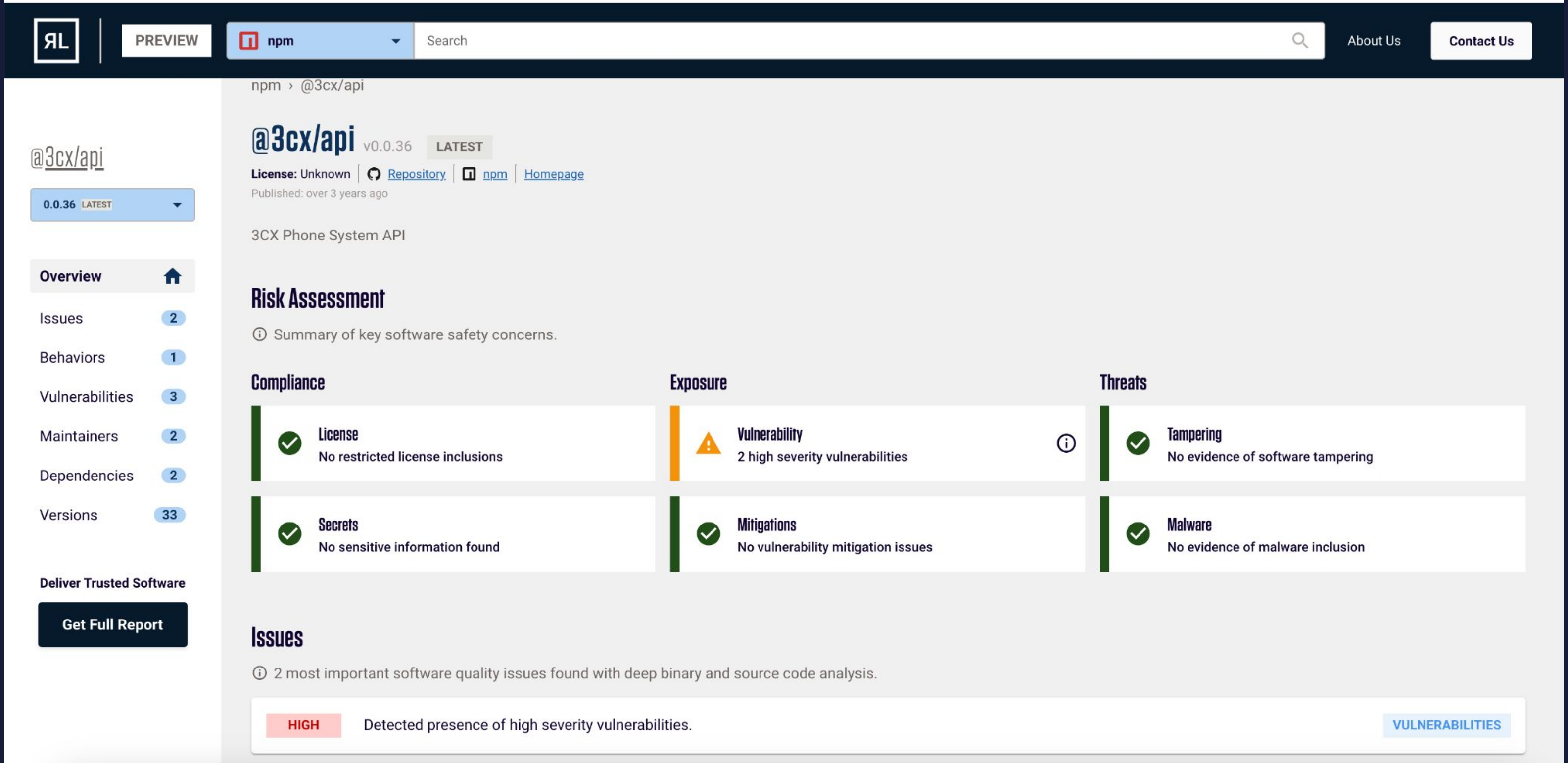
Malware Hijacks Operational Processes Enabling Tampering, Privilege Escalation, etc.



*Source: https://www.comparitech.com/software-supply-chain-attacks/*

# Finding Secure Software

https://find.secure.software/npm/packages/@3cx/api



npm › @3cx/api

## @3cx/api v0.0.36 LATEST

**License:** Unknown | Repository | npm | Homepage

Published: over 3 years ago

3CX Phone System API

### Risk Assessment

ⓘ Summary of key software safety concerns.

**Compliance**

✓ **License** No restricted license inclusions

✓ **Secrets** No sensitive information found

**Exposure**

⚠ **Vulnerability** 2 high severity vulnerabilities

✓ **Mitigations** No vulnerability mitigation issues

**Threats**

✓ **Tampering** No evidence of software tampering

✓ **Malware** No evidence of malware inclusion

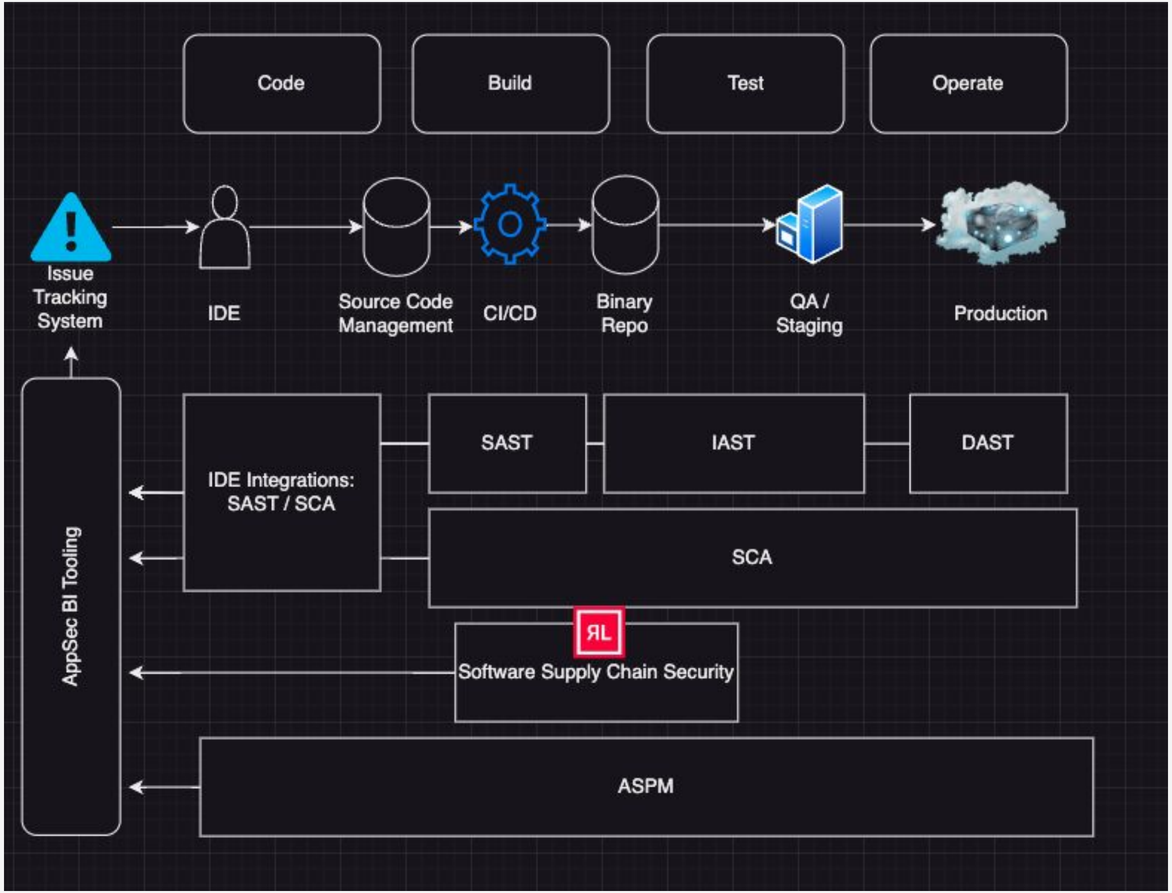
### Issues

ⓘ 2 most important software quality issues found with deep binary and source code analysis.

HIGH Detected presence of high severity vulnerabilities. VULNERABILITIES

# AppSec Teams

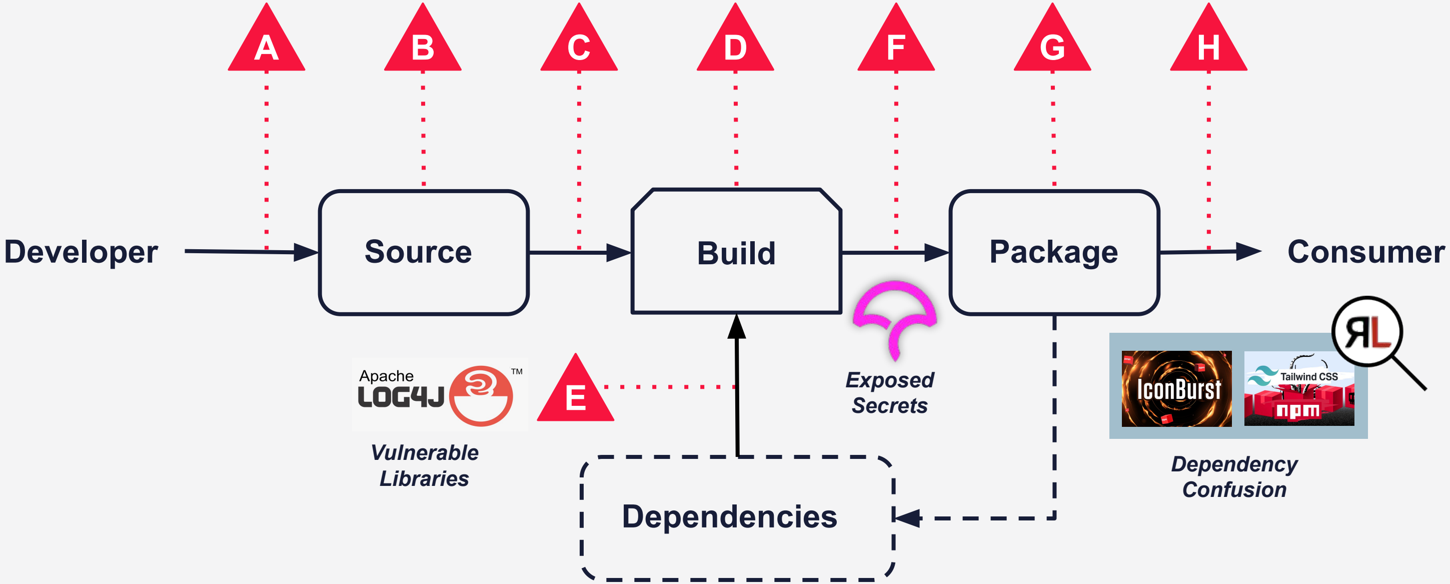
# Post-Build // Pre-Deploy



- Post-Build Opportunity
- Pre-Deploy Test

# Each Layer Has Risk

Considering the breath of attack vectors available



A B C D F G H

Developer

Source

Build

Package

Consumer

E

Dependencies

*Vulnerable Libraries*

*Exposed Secrets*

*Dependency Confusion*

Key

| | |
|---|---|
| A | Submit unauthorized change |
| B | Compromise source repo |
| C | Build from modified source |
| D | Compromise build process |
| E | Use compromised dependency |
| F | Upload a modified package |
| G | Compromise package repo |
| H | Use compromised package |

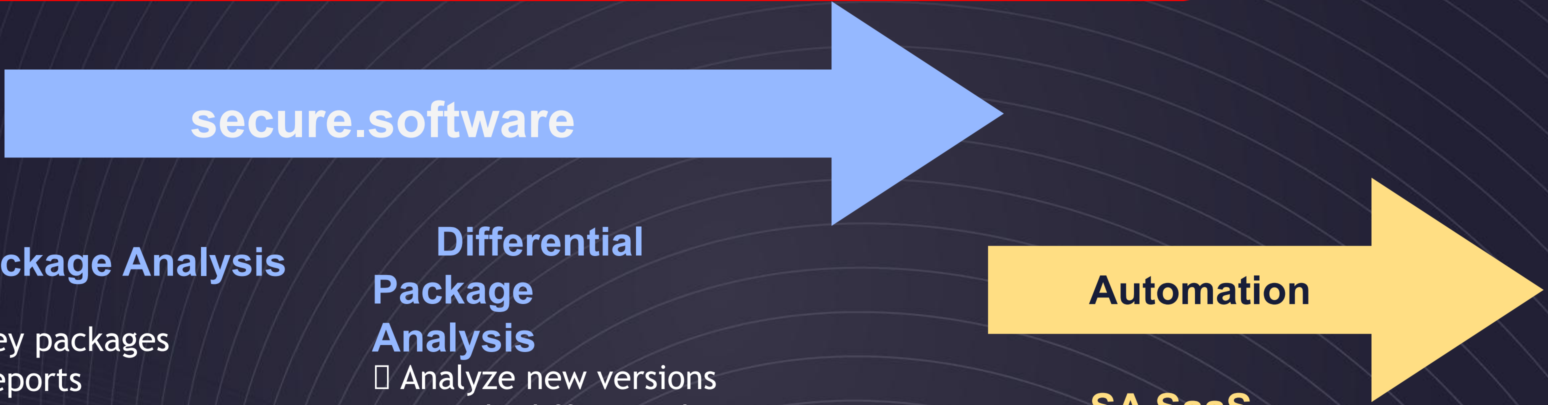
# Common Software Supply Chain Use Cases

AppSec | DevSecOps | SOC | CTI | TPRM | PSIRT

- Ability to detect anomalous added functionality within a software package
- Identify risky application behaviors
- Large application artifacts can be analyzed for security relevant issues
- Ability to generate industry-standard formatted Software Bill of Materials (SBOM)
- Ensure applications are not shipping to production with embedded malware or digital signatures issues
- Ability to process/analyze DMG, EXE, ESD, and MSI file types written in C/C++
- Query application portfolio in response to known malicious file, package, OSINT (e.g. log4j)
- Ability to compare software package risk posture from release to release
- Validate a software package as a final security check prior to production

REVERSINGLABS

# Differential Package Analysis



secure.software

Automation

## Initial Package Analysis

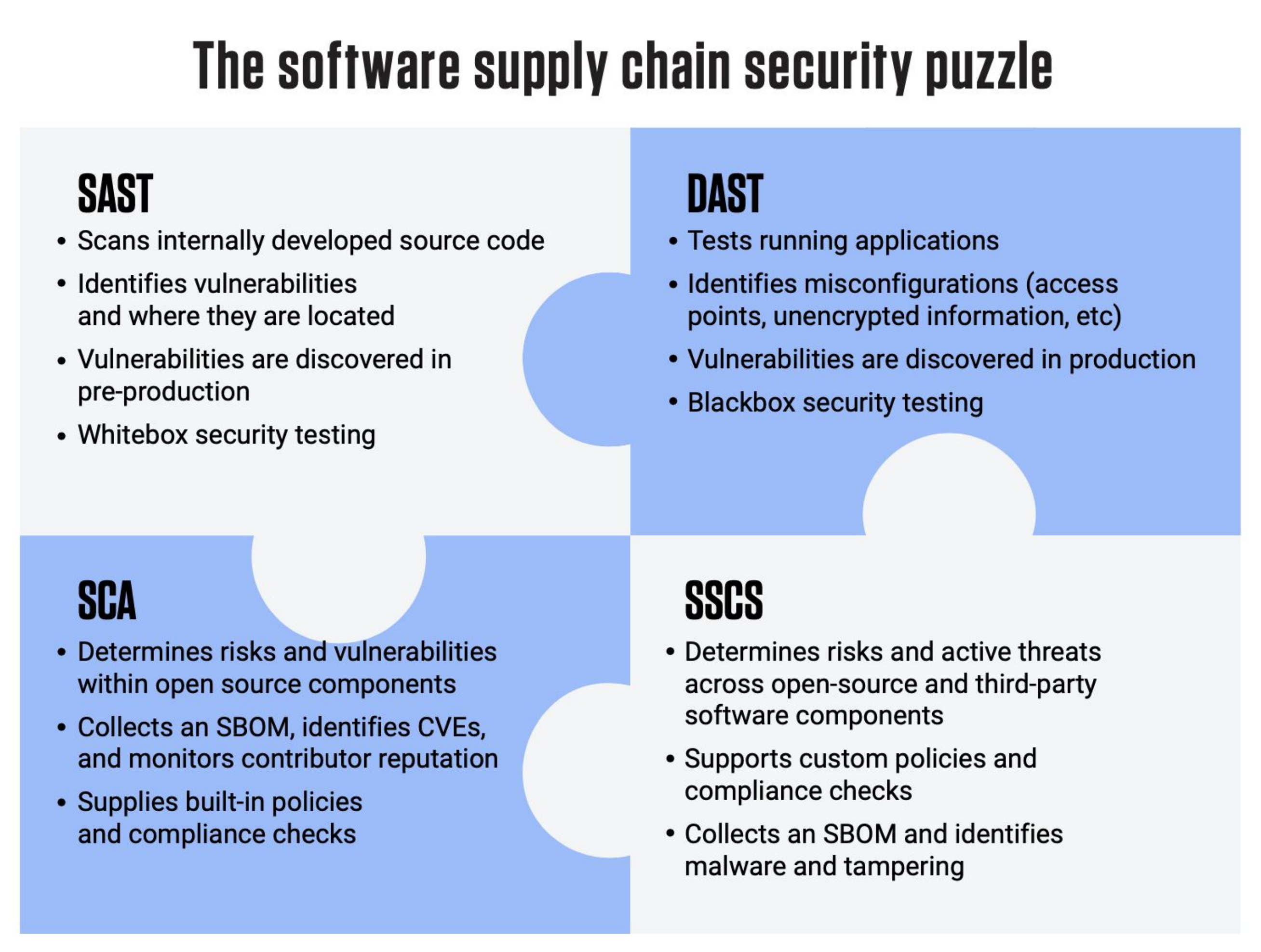
- Analyze key packages
- Provide reports
- Review and prioritize issues
- Evaluate component risks
- mitigation strategies

## Differential Package Analysis

- Analyze new versions
- Provide differential reports
- Identify high-risk changes
- Evaluate component changes
- Modify mitigations strategies

## SA SaaS Integration

- Automatically analyze new versions
- Analysis history
- Re-analysis on emergent threats
- Alerting on risks



# The software supply chain security puzzle

## SAST

- Scans internally developed source code
- Identifies vulnerabilities and where they are located
- Vulnerabilities are discovered in pre-production
- Whitebox security testing

## DAST

- Tests running applications
- Identifies misconfigurations (access points, unencrypted information, etc)
- Vulnerabilities are discovered in production
- Blackbox security testing

## SCA

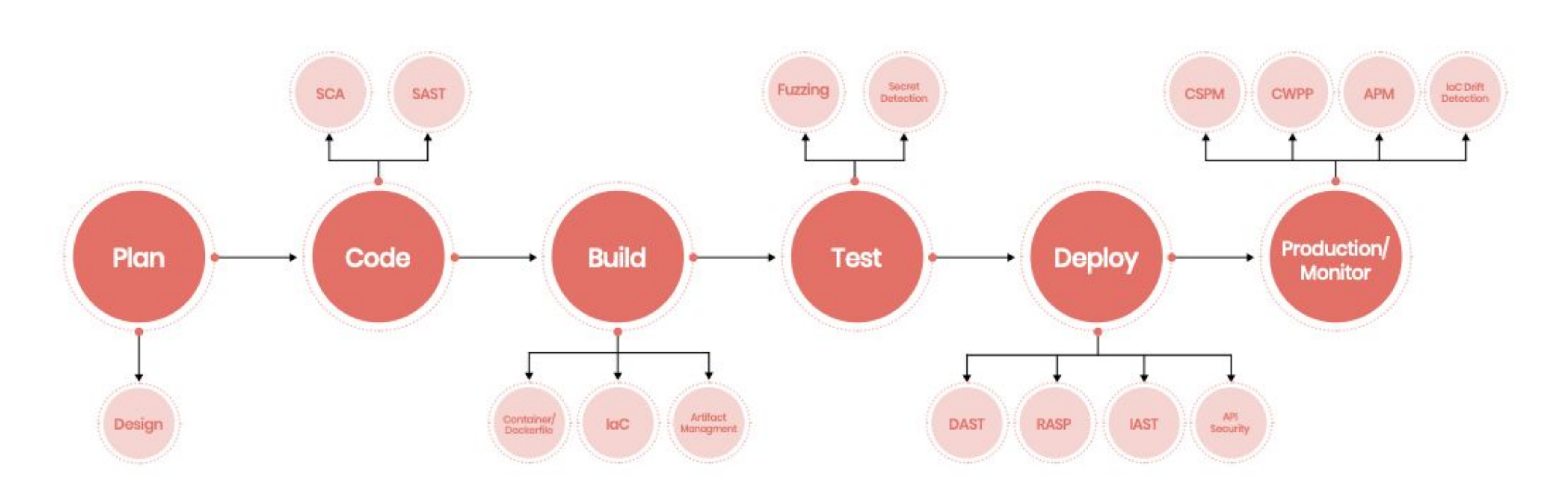
- Determines risks and vulnerabilities within open source components
- Collects an SBOM, identifies CVEs, and monitors contributor reputation
- Supplies built-in policies and compliance checks

## SSCS

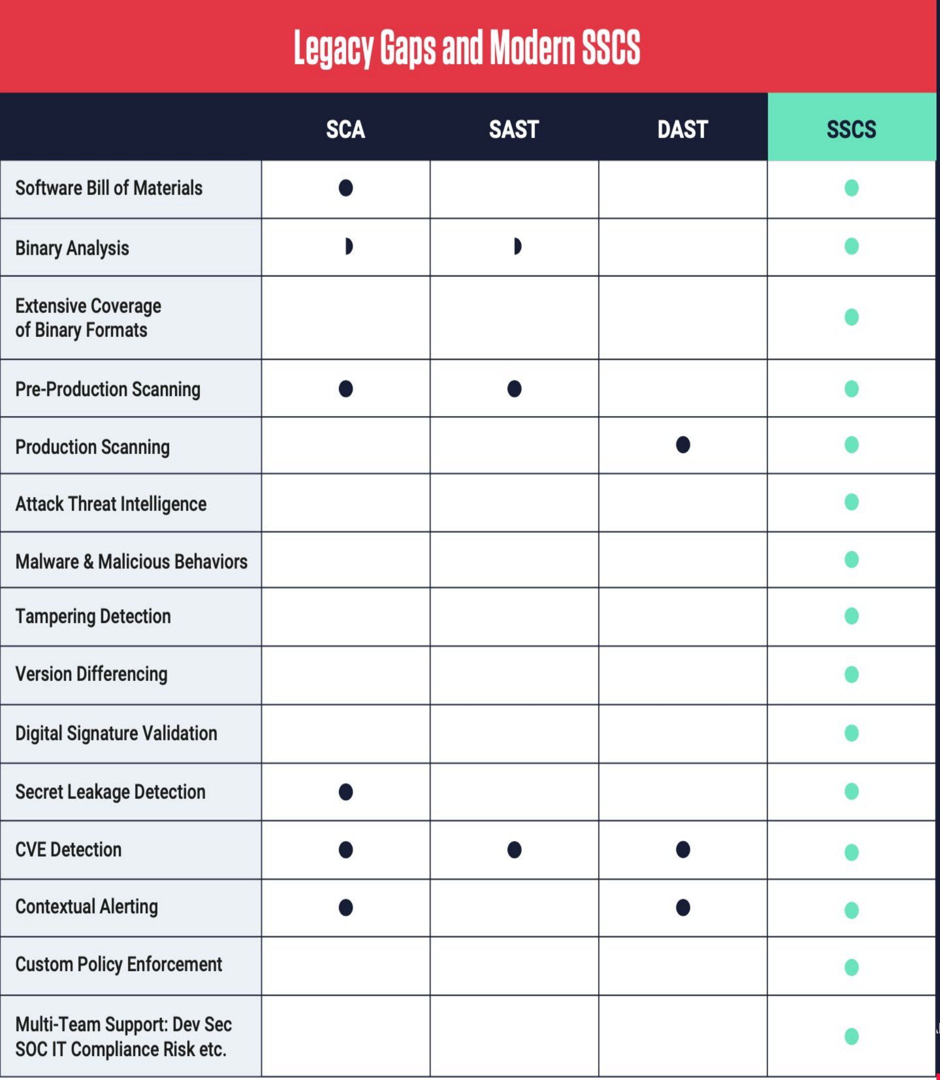
- Determines risks and active threats across open-source and third-party software components
- Supports custom policies and compliance checks
- Collects an SBOM and identifies malware and tampering

# Tool Sprawl

Lower Risk | Reduce Costs | Build Trustable Verdicts



*Source: Dazz Guide to building secure SDLC*



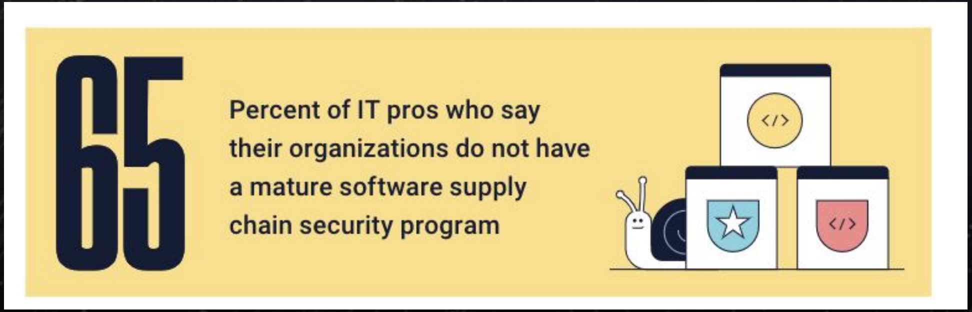
## Legacy Gaps and Modern SSCS

| | SCA | SAST | DAST | SSCS |
|---|---|---|---|---|
| Software Bill of Materials | ● | | | ● |
| Binary Analysis | ◗ | ◗ | | ● |
| Extensive Coverage of Binary Formats | | | | ● |
| Pre-Production Scanning | ● | ● | | ● |
| Production Scanning | | | ● | ● |
| Attack Threat Intelligence | | | | ● |
| Malware & Malicious Behaviors | | | | ● |
| Tampering Detection | | | | ● |
| Version Differencing | | | | ● |
| Digital Signature Validation | | | | ● |
| Secret Leakage Detection | ● | | | ● |
| CVE Detection | ● | ● | ● | ● |
| Contextual Alerting | ● | | ● | ● |
| Custom Policy Enforcement | | | | ● |
| Multi-Team Support: Dev Sec SOC IT Compliance Risk etc. | | | | ● |

# SSCS Use Cases

- Typosquatting
- Bypassing commit controls
- Software Distribution Networks
- Functionality Vulnerabilities
- CI/CD Platform Attacks

# CISOs



65

Percent of IT pros who say their organizations do not have a mature software supply chain security program

# Addressing Software Supply Chain Threats

## Risk & Compliance

- Executive Order 14028
- NIST
- GRC

## AppSec/Dev Control

- Open-Source Repos (NPM, PyPi, GitHub)
- DevOps Tool Exposures (Build & Binary Compilation)
- Software Release Composition & Dependency Analysis
- Automated SSCS scanning as part of your CI/CD process

## TPRM / ITSM / Procurement

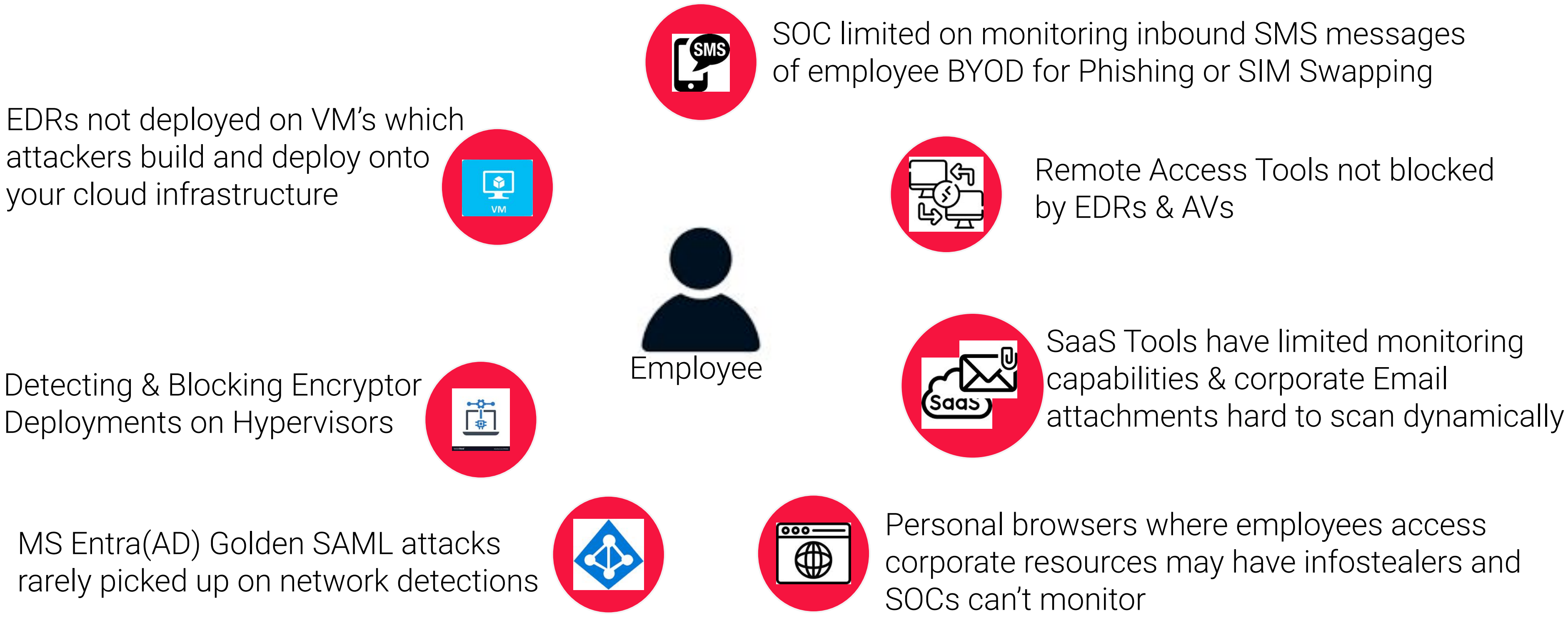
- 3rd Party COTS Software Selection
- Recurring Automatic Software Updates
- ITSM Security Validation

## SOC

- Manual Malware Analysis
- Triage, Investigation & Remediation
- Threat Hunting

# Modern Day Challenges For IT Leaders

Traditional Definitions of Assets in NIST just won’t cut it for proper risk assessments



SOC limited on monitoring inbound SMS messages of employee BYOD for Phishing or SIM Swapping

EDRs not deployed on VM’s which attackers build and deploy onto your cloud infrastructure

Remote Access Tools not blocked by EDRs & AVs

Employee

SaaS Tools have limited monitoring capabilities & corporate Email attachments hard to scan dynamically

Detecting & Blocking Encryptor Deployments on Hypervisors

MS Entra(AD) Golden SAML attacks rarely picked up on network detections

Personal browsers where employees access corporate resources may have infostealers and SOCs can’t monitor

REVERSINGLABS

# Building Trust In Software

*Answering some foundational discovery questions, to understand how we can help partner better with InfraGard Community*

**Do you know what makes up the software that you entrust sensitive business data with?**

- *Generate a comprehensive SBOM including commercial & OSS components/dependencies*
- *Extract embedded files which may be hiding malware or sensitive information (i.e. secrets)*

**Are "pen & paper" security questionnaires a bottleneck in quickly onboarding new vendors?**

- *Automate testing at scale, analyzing COTS software packages in seconds*
- *Independently test software, don't rely on vendor self attestation or evidence*

**Can your security tooling (e.g. anti-virus) scan large and complex 3rd party software (> 5 GB)?**

- *Analyze large (10GB binaries) and complex files (support of 4k+ file types) at the speed of business*

**Can you identify if software you purchased has been tampered with?**

- *Detect digital signatures that have been maliciously manipulated*
- *Pinpointing suspicious behaviors within any component across release versions*

**Do you struggle to assess the security risk presented by software vendors pre-contract?**

- *Assess 3rd party COTS software, using only the binary package (no source code required)*
- *Leverage analysis results to make informed business (e.g. procurement) decisions, considering security risk*

**Do you analyze software releases (patches, hotfixes, etc.) before proceeding with updates?**

- *Perform differential analysis to identify suspicious changes introduced between releases*

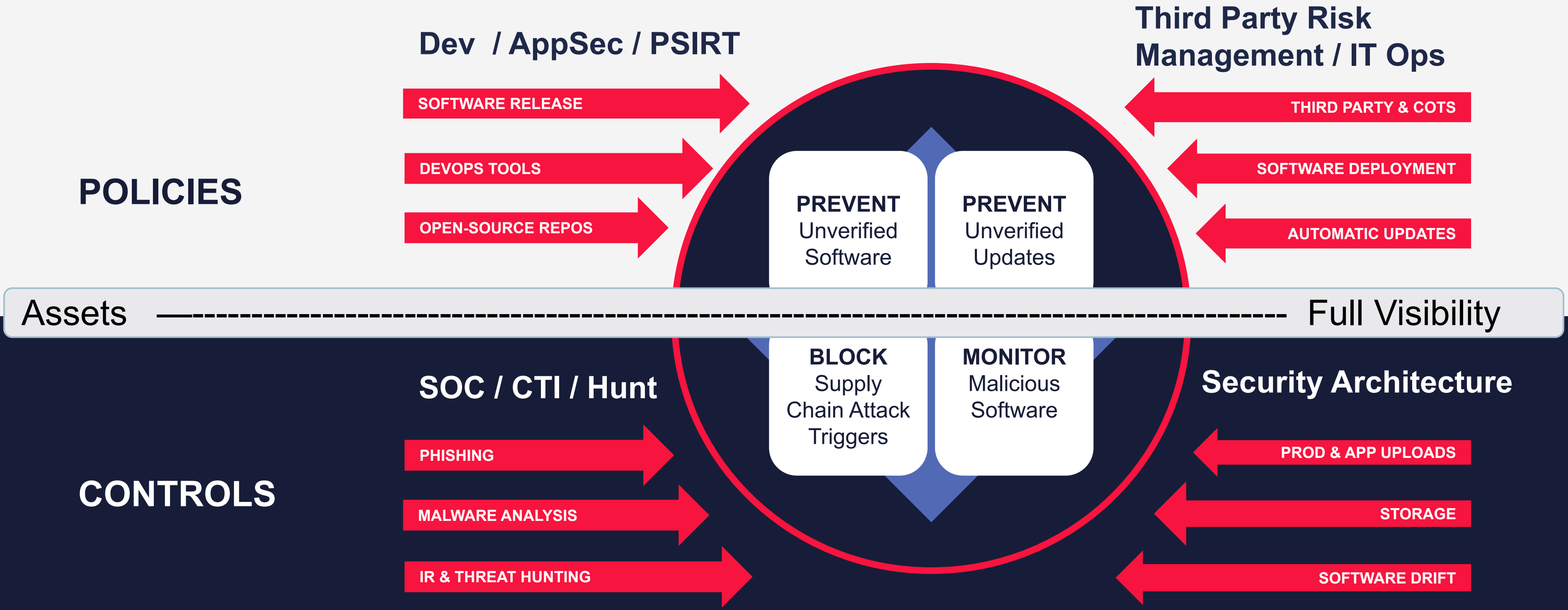
# Security assessment

- Detailed control over tools used in organizations development environment
- Like with firewalls, forbid everything and make exclusions after security assessment
- Central repositories for tooling used across your organization
- Software scheduled for automatic updating should first go through security review to prevent automated proliferation
- Keep in mind that plugins and extensions can be as equally dangerous
- Perform security assessment of third party modules used in your code base to prevent inclusion of compromised modules into your product
- Perform security assessment of release artifacts to prevent distribution of the product to your customers in case it gets compromised

# Collective Defense Fusion Strategy

REVERSINGLABS

## Building a SSCS Program In Your Organization



POLICIES

Dev / AppSec / PSIRT

SOFTWARE RELEASE

DEVOPS TOOLS

OPEN-SOURCE REPOS

Third Party Risk Management / IT Ops

THIRD PARTY & COTS

SOFTWARE DEPLOYMENT

AUTOMATIC UPDATES

**PREVENT** Unverified Software

**PREVENT** Unverified Updates

Assets —------------------------------------------------------------------------------------- Full Visibility

**BLOCK** Supply Chain Attack Triggers

**MONITOR** Malicious Software

CONTROLS

SOC / CTI / Hunt

PHISHING

MALWARE ANALYSIS

IR & THREAT HUNTING

Security Architecture

PROD & APP UPLOADS

STORAGE

SOFTWARE DRIFT

# Thank You

ali.khan@reversinglabs.com