

Software Supply Chain Security



All the software you use matters.

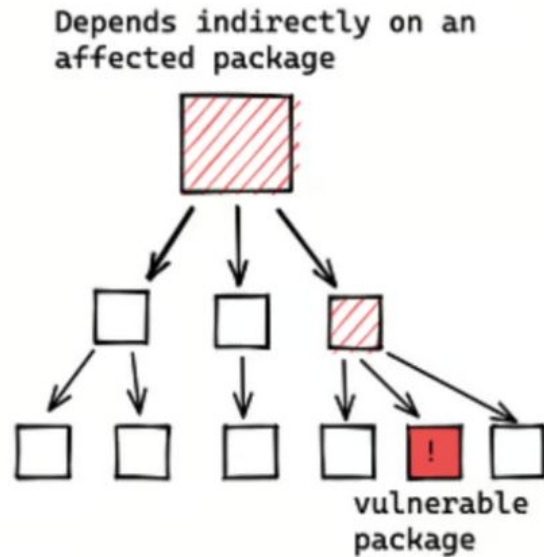
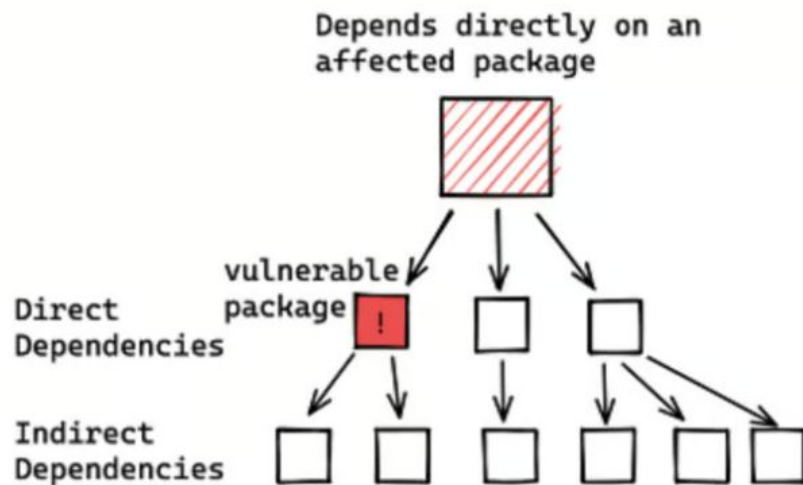
~ Why this talk?



What is a dependency?

- External Software Components
- Types
 - Direct Dependencies
 - Transitive Dependencies
- Package Managers
 - npm
 - yum
 - pip
- Eg. - *npm install git* | *pip install <pkg_name>*


Relationships



US govt, FireEye breached after SolarWinds supply-chain attack

By [Sergiu Gatlan](#)

 December 14, 2020

 10:04 AM

 3



Hackers start pushing malware in worldwide Log4Shell attacks

By [Lawrence Abrams](#)

 December 12, 2021

 06:07 PM



Hackers compromise 3CX desktop app in a supply chain attack

By [Sergiu Gatlan](#)

 March 29, 2023

 06:46 PM

 0

PyTorch discloses malicious dependency chain compromise over holidays

By [Ax Sharma](#)



January 1, 2023



01:26 AM



1

Dependency Confusion: How I Hacked Into Apple, Microsoft and Dozens of Other Companies

The Story of a Novel Supply Chain Attack



Alex Birsan · [Follow](#)

11 min read · Feb 9, 2021



19K



52

Dependency Confusion Supply-Chain Attack Hit Over 35 High-Profile Companies

Feb 10, 2021



Ravie Lakshmanan

```
ENGINEERING - Apple Inc. | 17.171.1 | @idms/idms-pmrpc | c6269b74ec56  
ENGINEERING - Apple Inc. | 17.122. | @idms/idms-pmrpc | 580f8f68bad3  
ENGINEERING - Apple Inc. | 17.171.1 | @idms/idms-pmrpc | d7bea26b6122
```

 moonlock

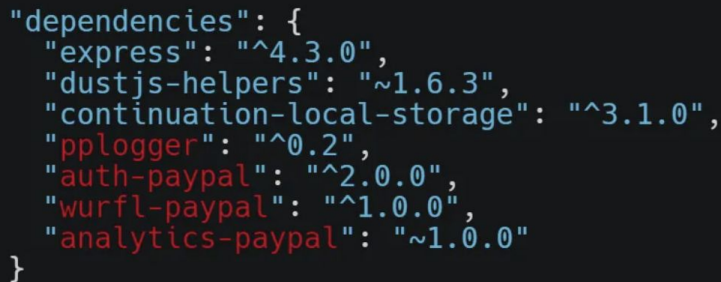


Dependency Confusion

The Idea

While attempting to hack PayPal with me during the summer of 2020, Justin Gardner (@Rhynorater) shared an interesting bit of Node.js source code found on GitHub.

The code was meant for internal PayPal use, and, in its `package.json` file, appeared to contain a mix of public and private dependencies — public packages from npm, as well as non-public package names, most likely hosted internally by PayPal. These names did not exist on the public npm registry at the time.



```
"dependencies": {
  "express": "^4.3.0",
  "dustjs-helpers": "~1.6.3",
  "continuation-local-storage": "^3.1.0",
  "pplogger": "^0.2",
  "auth-paypal": "^2.0.0",
  "wurfl-paypal": "^1.0.0",
  "analytics-paypal": "~1.0.0"
}
```

Dependency

Millions of GitHub repos likely vulnerable to RepoJacking, researchers say

By **Bill Toulas**

What is RepoJacking?

Repojacking

Build

First clone the project from github:

```
git clone https://github.com/socraticorg/mathsteps.git
cd mathsteps
```

Install the project dependencies:

```
npm install
```

Repojacking cont.

<> Code 🔗 Pull requests 🗃 Projects 🛡 Security 📈 Insights

🔑 master ▼

yesgraph-Dominus / install.sh

Code

Blame

Executable File · 15 lines (10 loc) · 442 Bytes

```
1  #!/bin/sh
2
3  echo "[DOMINUS]: Loading Dominus..."
4
5  DOWNLOAD_URI='https://github.com/YesGraph/Dominus/archive/master.zip'
6  TARGET_DIR="."
7
8  mkdir -p "${TARGET_DIR}/Dominus/"
9  curl -L $DOWNLOAD_URI | tar xvz -C "${TARGET_DIR}/Dominus/" --strip-components=1
10
11  echo "[DOMINUS]: Downloaded & Unpacked from GitHub."
12  echo "[DOMINUS]: Running integration system..."
13
```

Nearly 100,000 NPM Users' Credentials Stolen in GitHub OAuth Breach

📅 May 27, 2022 👤 Ravie Lakshmanan



Security

Security alert: Attack campaign involving stolen OAuth user tokens issued to two third-party integrators

On April 12, GitHub Security began an investigation that uncovered evidence that an attacker abused stolen OAuth user tokens issued to two third-party OAuth integrators, Heroku and Travis-CI, to download data from dozens of organizations, including npm. Read on to learn more about the impact to GitHub, npm, and our users.

Leaked Credentials

Find the Fraud

urllib3-1.21.1.tar.gz



urllib3-1.21.1.tar.gz

TypoSquatting

urllib3-1.21.1.tar.gz



urllib3-1.21.1.tar.gz



AUGUST 2023

Malicious PyPI package imitates VMware vSphere connector module

A malicious PyPI package called “VMConnect” imitated the legitimate VMware vSphere connector module but contained hidden sinister code. This package, downloaded hundreds of times, was part of an ongoing campaign dubbed “PaperPin.” When installed, it executed code from an attacker-controlled URL, along with similar packages “ethter” and “qu8uantiumbase.” These packages were reported and removed from PyPI, emphasizing the importance of obtaining legitimate software from official sources to avoid such threats.



JUNE 2023




JULY 2023





A French-meme-inspired PyPI package targets Windows with an info-stealer







A PyPI package named “feur” was found to contain a Windows Remote Access Trojan (RAT) alongside a meme-related name. Upon installation, it executed the RAT, which possessed surveillance capabilities, including clipboard access, network monitoring, webcam usage, and screenshot capture. Although initially appearing as a prank with low downloads, the RAT’s potential for misuse highlighted the importance of safeguards like Sonatype Repository Firewall.




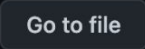

Malicious Packages



 ossf / malicious-packages Public


generated from [ossf/project-template](#)


 Notifications  Fork 11  Star 86 

 Code  Issues 2  Pull requests 7  Actions  Security  Insights

 main  8 branches  0 tags  

github-actions Assign IDs  b5c7ff5 14 hours ago  619 commits

 .github Bump github/codeql-action from 2.22.0 to 2.22.3 (#196) last week

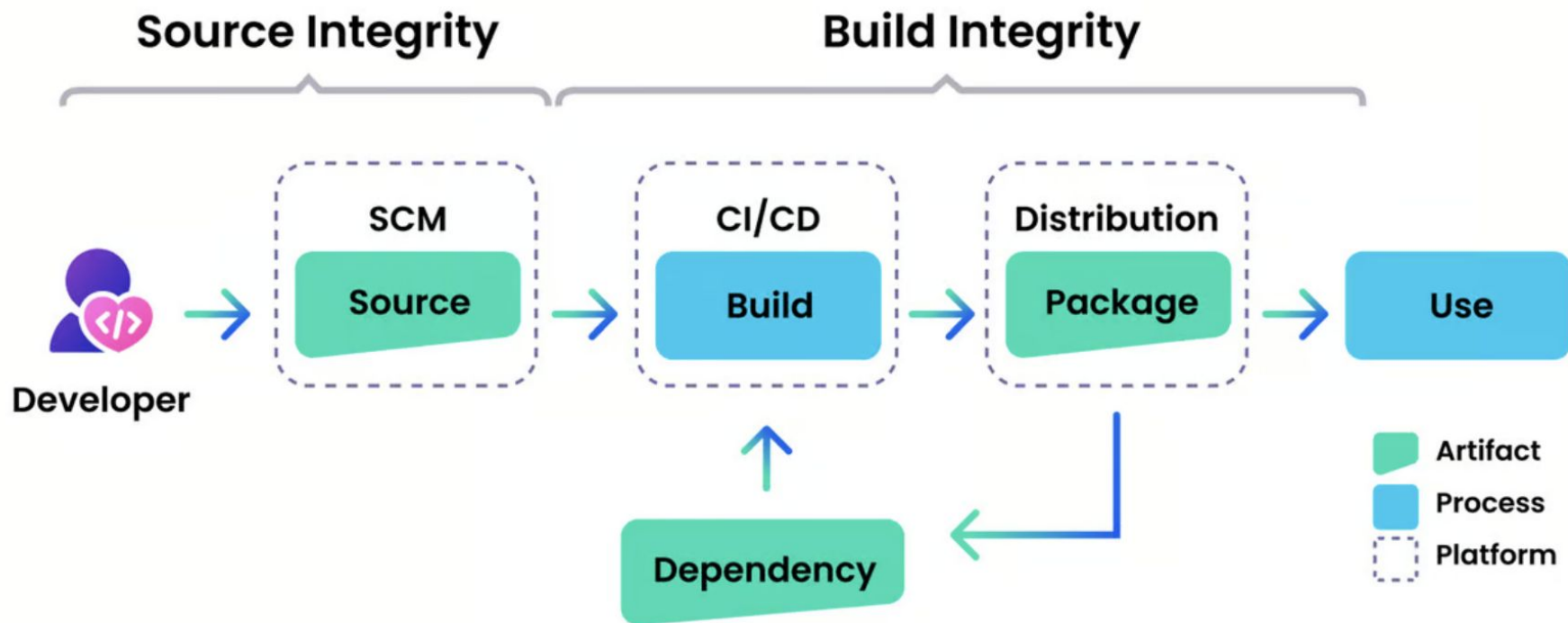
 cmd Add "id" and "modified_time" to origin for more ways to dedu... 3 months ago

About

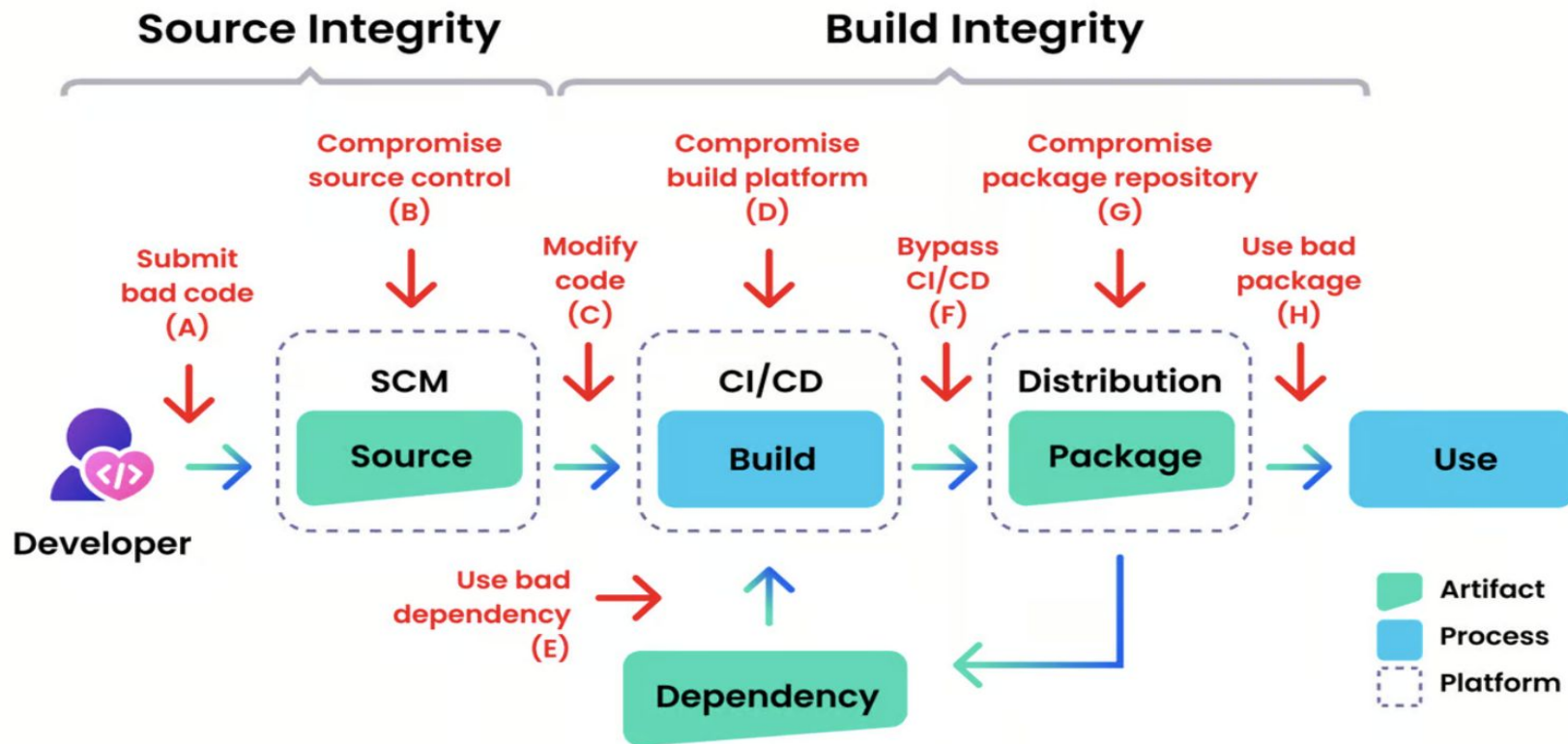
A repository of reports of malicious packages identified in Open Source package repositories, consumable via the Open Source Vulnerability (OSV) format.

How to protect against these attacks?

Software Supply Chain



Layers of Supply Chain Attacks





Suppliers to the new BMW 4 Series

CLOSE-COUPLED SYSTEMS FOR EXHAUST GAS CLEANING

EBERSPAECHER

SOUND SYSTEM

HARMAN

CRASH REINFORCEMENT ENGINE MOUNT

GESTAMP

LOW-VOLTAGE POWER CABLE SYSTEMS

GEBAUER & GRILLER

SHIELDING SYSTEMS (ENGINE)

ELRINGKLINGER

GASOLINE PARTICULATE FILTER

BOYSEN

ALLOY WHEELS

BORBET / RONAL

FASTENERS

A RAYMOND

VISCOUS DAMPER

AAM

DASH INSULATOR

AURIA

BRAKE CALIPER

BREMBO

SUN VISOR

GRUPO ANTOLIN

AC COMPRESSOR

DENSO

REAR WHEEL KNUCKLE

FONDERIE MARIO MAZZUCCONI

OTHER TRANSMISSION PARTS

HOERBIGER

CLIMATE CONTROL PANEL REAR

PHTC

Automotive News Europe

REAR KNUCKLE WITH BALL JOINT & RUBBER BUSHING

MARELLI

BEARINGS/ BUSHINGS (ENGINE MAIN PARTS)

RHEINMETALL AUTOMOTIVE

POWER DISTRIBUTION UNIT

MTA

SUSPENSION COMPONENTS

ROLLAX

WINDSHIELD BONDING

SIKA

TORQUE CONVERTER

ZF

OIL PAN

IBS FILTRAN / KIRPART

FRONT END

KIRCHHOFF AUTOMOTIVE

BRAKE LINE

TI FLUID SYSTEMS

SENSOR LINKAGES

MACLEAN-FOGG

FRONT BUMPER

MOTHERSON

CENTER CONSOLE

PREH

HOOD LATCH

KIEKERT

HEATING LINE

TEKLAS

DOOR PANEL

MEGATECH INDUSTRIES

EXHAUST COLD END

TENNECO



Bill of Materials

SBoms (Software Bill of Materials)

- Lists all component parts and software dependencies used in application development and delivery.
 - What is being deployed where?
 - Where it came from?
 - What's in it?
- Formats
 - Software Package Data Exchange (SPDX)
 - OWASP CycloneDX
 - Standard for software identification (SWID)

Source Code Layer

- Code Reviews
- Software Application Security Testing (SAST)
- Secure Defaults
- Securing Source Code Management(SCM) systems
- Code Environments and IDE extensions

Build and Pipeline Layer

- Software Composition Analysis(SCA) - Dependencies
 - Identifying open source software in the code base
- Continuous Integration/Continuous Delivery (CI/CD) Pipeline
 - Jenkins
 - Github Actions
- Containers and Registries
 - Docker image or Kubernetes
 - Docker Hub and Amazon' Elastic Container Registry (ECR)

Packaging and Deployment Layer

- Software Bill of Materials (SBOM)
 - A machine-readable inventory of software components, build tools and dependencies.
 - Details of the components within a software product:
 - Name
 - Version
 - License
 - Vulnerabilities
- Code Provenance and Signature
 - Helps understand the origin of a piece of code or software
- Artifact Repository
 - Compiled code, libraries or executable files

Supply Chain Security Frameworks

- Supply Chain Levels for Software Artifacts (SLSA)
 - Pronounced SALSA
- Secure Software Development Framework (SSDF)
- OpenSSF Scorecard

SLSA

- SLSA 1
 - Documented Build Process
- SLSA 2
 - Running builds on a hosted platform that generates provenance
 - Publish provenance
- SLSA 3
 - Hardened Build Platforms
 - Signed Provenance
- SLSA 4
 - Two Party Reviews
 - Hermetic Builds

SLSA

Vulnerability	Category	Real-world example
(A) Inject bad code	Source threat	Linux hypocrite commits: Researcher intentionally introduces vulnerabilities into Linux kernel via patches
(B) Compromise source control	Source threat	PHP: Attacker compromised PHP's self-hosted git server and submits two malicious commits
(C) Build from modified source	Build threat	Webmin: Attacker modifies build infra to use source files not matching source control
(D) Compromise build process	Build threat	SolarWinds: Attacker compromised build platform
(E) Vulnerable dependency	Source threat Build threat	Event-stream: Attacker added innocuous dependency then turned dependency malicious
(F) Bypass CI/CD + inject bad artifact	Build threat	CodeCov: Leaked credentials used to upload malicious artifact to GCS bucket
(G) Compromise package repo/signing	Build threat	Attacks on Package Mirrors: Mirrors run on popular package repos
(H) Compromise deploy process	Deployment and runtime threat	Browserify typosquatting: Similar name to original used

SCA Tools

- Dependency Track
- Dependabot
- Trivy
- Dependency Check
- CycloneDX tools
- Snyk (Paid)
- BlackDuck (Paid)
- Endor Labs (Paid)
- Sonatype (Paid)

Automation

- Existing Code
 - Parse repos with Graphql
 - Generate sboms
 - Send them to a SCA platform
- New Code
 - Build Phase
 - Github Actions / Jenkins Pipeline
 - Generate sboms
 - Send them to a SCA platform
- Alerts
 - Slack - Based on Criticality/EPSS
- Analyze

Demo

If you wish to make an apple pie from scratch, you must first invent the universe.

Carl Sagan

