Shielding the Foundation
Security Across Your Software Supply Chain

Stephen Chin, VP of Developer Relations
@ JFrog
THE WORLD RUNS ON OSS & 3RD-PARTY COMPONENTS ACROSS EVERY STAGE OF THE SDLC

- CURATE
- CREATE
- PACKAGE
- PROMOTE
- DISTRIBUTE
- DEPLOY
- RUN

python, rpm, docker, Maven, GO, Gradle, debian, COCOA PODS, Ruby, npm, NuGet, php
SOFTWARE SUPPLY CHAIN ATTACKS ARE UP 100x
DEVELOPERS ARE THE CLEAR TARGETS
BUT THEY ARE TIRED OF SECURITY
THE RATE OF **PUBLISHED CVEs** IS INCREASING

CREATING **CONSTANT PRESSURE** ON DEV & SECURITY TEAMS

---

**CVEs Disclosed Per Month**

- **All CVEs**
- **Critical CVEs**
YET
CRITICAL CVES
IN COMMON COMPONENTS
CAN HAVE
NO REAL SECURITY IMPACT
Yet **CRITICAL CVEs** in common components can have NO REAL SECURITY IMPACT

Curl is used **daily** by virtually **EVERY INTERNET-USING HUMAN** on the planet

```
curl://
```
YET CRITICAL CVES IN COMMON COMPONENTS CAN HAVE NO REAL SECURITY IMPACT

All made up.

4 long days later....

Current Description

**DISPUTED**

integer overflow vulnerability in tool_operate.c in curl 7.65.2 via a large value as the retry delay. NOTE: many parties report that this has no direct security impact on the curl user; however, it may (in theory) cause a denial of service to associated systems or networks if, for example, -retry 2 is set with a 'retry value' much smaller than what was intended. This is not especially plausible because the overflow only happens if the user wants to specify that curl should wait weeks (or longer) before trying to recover from a transient error.
MANY CRITICAL CVES IN COMMON COMPONENTS ARE NON-EXPLOITABLE IN 99% OF CASES

120,000 BINARIES
0 EXPLOITABLE CASES
AND WHILE DEVELOPERS ARE SWAMPED, ATTACKERS ARE COMING UP WITH NEW ATTACKS
ABUSING **SECRETS** IN BINARIES LEAKED TO PUBLIC REPOSITORIES

PRIVATE REPOSITORY

LEAK

OVER **250K TOKENS** DETECTED BY JFROG!

NON-COMPANY ACCOUNT
ON PUBLIC REPOSITORY

Read & Modify Source Code

Read All Messages

DevSecOps Day
EMEA Unlock 2024
THE RATE OF MALICIOUS PACKAGE ATTACKS IS INCREASING

NPM Malicious Packages Reports

2020/H1 2020/H2 2021/H1 2021/H2 2022/H1 2022/H2 2023/H1

0 1000 2000 3000 4000 5000 6000 7000

162 6,561
ML MODELS? YET ANOTHER MALICIOUS PACKAGE!

ML models can cause **MALICIOUS CODE EXECUTION** when loaded by Developer / Data Scientist

Public repositories for models **ARE NOW A TARGET**

These malicious models **WILL SEEM COMPLETELY SAFE** on the Hugging Face website
A SUPPOSEDLY LEGITIMATE MODEL - JUST DATA, RIGHT?
import tensorflow as tf
from keras.preprocessing import image
from keras.models import load_model
import numpy as np

# Load the model
model = load_model('vgg16_light/tf_model.h5')

img = image.load_img('./cat.jpeg', target_size=(224, 224))
img = np.asarray(img)
img = np.expand_dims(img, axis=0)
output = model.predict(img)
if output[0][0] > output[0][1]:
    print("cat")
else:
    print('dog')
HOW? MALICIOUS CODE IS HIDDEN IN THE BINARY DATA
ATTACKERS ARE EXPLOITING GENERATIVE AI FOR INJECTING MALICIOUS PACKAGES

1. Question

2. Answer with nonexistent package

3. Publish malicious package

4. Question

5. Answer with malicious package name

6. npm install

7. Malicious code executes

AI PACKAGE HALLUCINATION
NOW EVEN GENERATIVE AI INTRODUCES SECURITY VULNERABILITIES!

Requesting generation of an endpoint that returns file contents
# Write an endpoint that returns a file from user input
@app.route("/file/<filename>")
def file(filename):
    return open(filename, "r").read()

The generated code is **vulnerable** to path traversal -
?filename=../../../../etc/passwd

Requesting generation of a **secure** endpoint also doesn’t help
# Write a **secure** endpoint that returns a file from user input and prevents directory traversal
@app.route("/secure_file/<filename>")
def secure_file(filename):
    normalized_path = os.path.normpath(filename)
    return open(normalized_path, "r").read()

Still vulnerable...
What about ?filename=/etc/passwd
THE JFROG SECURITY APPROACH

RELEASE FAST - WITH TRUST AND CONFIDENCE
THE SSC **BEGINS** WHEN ANYTHING ENTERS

**AND ENDS IN**

PRODUCTION
IF YOU CAN’T CONTROL IT COMPLETELY

YOU CAN’T SECURE IT COMPLETELY

CODE

BINARIES

CREATE

PACKAGE

PROMOTE

DISTRIBUTE

DEPLOY

RUN

CURATE

CREATE

PACKAGE

PROMOTE

DISTRIBUTE

DEPLOY

RUN
OSS PACKAGES
HIGH VOLUME & NO CONTROL

Go: 65K
PyPI: 20K
npm: 95K
NuGet: 15K
Maven: 5K
OSS PACKAGES
HIGH VOLUME & NO CONTROL

- Go
  - New Packages: 65K
  - New Versions: 500K
- PyPI
  - New Packages: 20K
  - New Versions: 100K
- npm
  - New Packages: 125K
  - New Versions: 95K
- NuGet
  - New Packages: 15K
  - New Versions: 175K
- Maven
  - New Packages: 5K
  - New Versions: 200K
IS THE PACKAGE **SAFE TO USE?**

- MALICIOUS?
- VULNERABLE?
- LICENSE?
- MAINTAINED?
SOFTWARE SUPPLY CHAIN
THE CURRENT APPROACH

CURATE  CREATE  PACKAGE  PROMOTE  DISTRIBUT
DEPLOY  RUN

DETECT & REMEDIATE
AN OUNCE of PREVENTION

is worth

A POUND of CURE

Benjamin Franklin
Centralize Visibility & Control of 3rd party (OSS) package downloads

Frictionless Package Consumption by Developers by proactively preventing & blocking malicious and unwanted packages

Automate Curation of 3rd Party Packages to provide your developers with a trusted source of software components

Improve DevSecOps Experience & Realize Cost Savings with seamless integration & reduced remediation later in your SDLC
JFROG CURATION
AVAILABLE NOW

jfrog.com/curation

Supported Packages

- npm
- docker
- npm
- v

Supported in
SAAS & SELF HOSTED

Supported Policy Rules for

- MALICIOUS
- LICENCE CLEARANCE
- OPERATIONAL RISK
- VULNERABILITIES

Upcoming

- GO
- R
- php
- R

Upcoming
HOW CAN DEVELOPERS TAKE CONTROL?
BACKBONE of JFrog SECURITY
POWERED BY RESEARCH & ENGINEERING

JFrog Augmented CVE Information
Dependencies
Licenses
OpenSSF Scorecards
Recommended Fix Version
....
BACKBONE of JFrog SECURITY
POWERED BY RESEARCH & ENGINEERING

- UP TO DATE
- COVERAGE
- ACCURACY
  - JFrog Augmented CVE Information
  - Dependencies
  - Licenses
  - OpenSSF Scorecards
  - Recommended Fix Version
  - ....
ANNOUNCING

JFROG CATALOG
JFROG CATALOG

The “Google Search” of OSS Packages available to developers, DevOps Engineers, AppSec, etc.

Application & Automations via GraphQL

It’s all about the Data and how it’s integrated across multiple use cases

Coming soon – Private Catalog enabling custom properties with external/private sources
JFROG CATALOG
AVAILABLE NOW

jfrog.com/catalog

Supported Packages
- npm
- docker
- Jenkins
- Vercel

Upcoming
- Go
- Rust

Supported in
WEB APPLICATION + GRAPHQL SUPPORT

DevSecOps Day EMEA
CURATION & CATALOG
DEVELOPERS NEED A TOOL TO DETECT, LEARN, FIX AND SHIP FAST!

- **Fast and Accurate** for Optimized Development
- **Efficiently Find and Fix** Source Code Vulnerabilities
- **Seamless Shift Left** Developer-Focused Experience
- **Centralized** Visibility and Governance
ANNOUNCING
JFROG SAST
JFROG SAST
AVAILABLE NOW

Jfrog.com/xray

Supported Technologies

Upcoming

Platform support
SAST
ESSENTIAL (XRAY)

- SCA
- MALICIOUS PACKAGE DETECTION
- LICENCE CLEARANCE
- OPERATIONAL RISK POLICIES

ADVANCED SECURITY

- CONTEXTUAL ANALYSIS
- SECRETS DETECTION
- MISCONFIGURATIONS
- IAC SECURITY ANALYSIS

Launched October 2022
ESSENTIAL (XRAY)

- SCA
- MALICIOUS PACKAGE DETECTION
- LICENCE CLEARANCE
- OPERATIONAL RISK POLICIES

ADVANCED SECURITY

- SAST
- CONTEXTUAL ANALYSIS
- SECRETS DETECTION
- MISCONFIGURATIONS
- IAC SECURITY ANALYSIS

Launched October 2022
ADVANCED SECURITY

END TO END APPLICATION SECURITY TESTING (AST)

SOURCE TO BINARY | 1ST PARTY TO 3RD PARTY
THE SSC BEGINS WHEN ANYTHING ENTERS

AND ENDS IN

PRODUCTION
INTRODUCING SECURITY INSIGHTS

Available
- Artifact
- Build
- Release Bundle

Coming soon
- Repository
- Project
SECURITY INSIGHTS
AVAILABLE NOW

JFROG CURATION

JFROG CATALOG

JFROG SAST

JFROG SECURITY INSIGHTS
Thank you