



VERACODE

You change the world, we'll secure it.

Niels Tanis

The Rise of Software Supply-Chain Attacks

How Secure is your .NET Application?

CODECAMP 

Who am I?

- Niels Tanis
 - Security Researcher @ Veracode
 - Background in .NET Development
 - Application Security Consultancy
 - Pen-testing & Ethical Hacking
 - ISC² CSSLP



The Rise of Software Supply-Chain Attacks

How Secure is your .NET Application?



Agenda

- Hacker History
- Definition Software Supply-Chain
 - Development of .NET application
 - Building / Releasing / Deploying
- Securing our Software Supply-Chain
- Conclusion and Q&A

Hacking History

- Started out with phreaking in late '50-'60.



Getting connected!

- SATAN (Security Administrator Tool for Analyzing Networks) by Wietse Venema and Dan Farmer released 1995.
- NMAP (Network Mapper) by Gordon Lyon released in 1997

```
Starting Nmap 7.60 ( https://nmap.org ) at 2018-01-14 11:05 CET
Nmap scan report for scanme.nmap.org (45.33.32.156)
Host is up (0.16s latency).
Other addresses for scanme.nmap.org (not scanned): 2600:3c01::f03c:91ff:fe18:bb2f
Not shown: 996 closed ports
PORT      STATE    SERVICE
22/tcp    open     ssh
25/tcp    filtered smtp
80/tcp    open     http
9929/tcp  open     nping-echo
```

Smashing the Stack...

- Phrack #49 in November 1996

Aleph One wrote about buffer overflows

.oo Phrack 49 Oo.

Volume Seven, Issue Forty-Nine

File 14 of 16

BugTraq, r00t, and Underground.Org
bring you

XXXXXXXXXXXXXXXXXXXXXXXXXXXX
Smashing The Stack For Fun And Profit
XXXXXXXXXXXXXXXXXXXXXXXXXXXX

by Aleph One
aleph1@underground.org

`smash the stack` [C programming] n. On many C implementations it is possible to corrupt the execution stack by writing past the end of an array declared auto in a routine. Code that does this is said to smash the stack, and can cause return from the routine to jump to a random address. This can produce some of the most insidious data-dependent bugs known to mankind. Variants include trash the stack, scribble the stack, mangle the stack; the term mung the stack is not used, as this is never done intentionally. See spam; see also alias bug, fandango on core, memory leak, precedence lossage, overrun screw.

SQL Injection

- Phrack #54 in December 1998

Rain Forest Puppy wrote about
SQL injection

----[ODBC and MS SQL server 6.5

Ok, topic change again. Since we've hit on web service and database stuff, let's roll with it. Onto ODBC and MS SQL server 6.5.

I worked with a fellow WT'er on this problem. He did the good thing and told Microsoft, and their answer was, well, hilarious. According to them, what you're about to read is not a problem, so don't worry about doing anything to stop it.

- WHAT'S THE PROBLEM? MS SQL server allows batch commands.

- WHAT'S THAT MEAN? I can do something like:

```
SELECT * FROM table WHERE x=1 SELECT * FROM table WHERE y=5
```

Exactly like that, and it'll work. It will return two record sets, with each set containing the results of the individual SELECT.

- WHAT'S THAT REALLY MEAN? People can possibly piggyback SQL commands into your statements. Let's say you have:

```
SELECT * FROM table WHERE x=%criteria from webpage user%
```

Now, what if %criteria from webpage user% was equal to:

```
SELECT * FROM sysobjects
```

It would translate to:

```
SELECT * FROM table WHERE x=1 SELECT * FROM sysobjects
```

Code Red & SQL Slammer

- Microsoft Internet Information Server, July 2001

```
GET /default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
%u9090%u6858%ucbd3%u7801%u9090%u6858%ucbd3%u7801  
%u9090%u6858%ucbd3%u7801%u9090%u9090%u8190%u00c3  
%u0003%u8b00%u531b%u53ff%u0078%u0000%u00=a HTTP/1.0
```

Bill Gates – Email to all MS FTE

BILL GATES BUSINESS 01.17.02 12:00 PM

Bill Gates: Trustworthy Computing

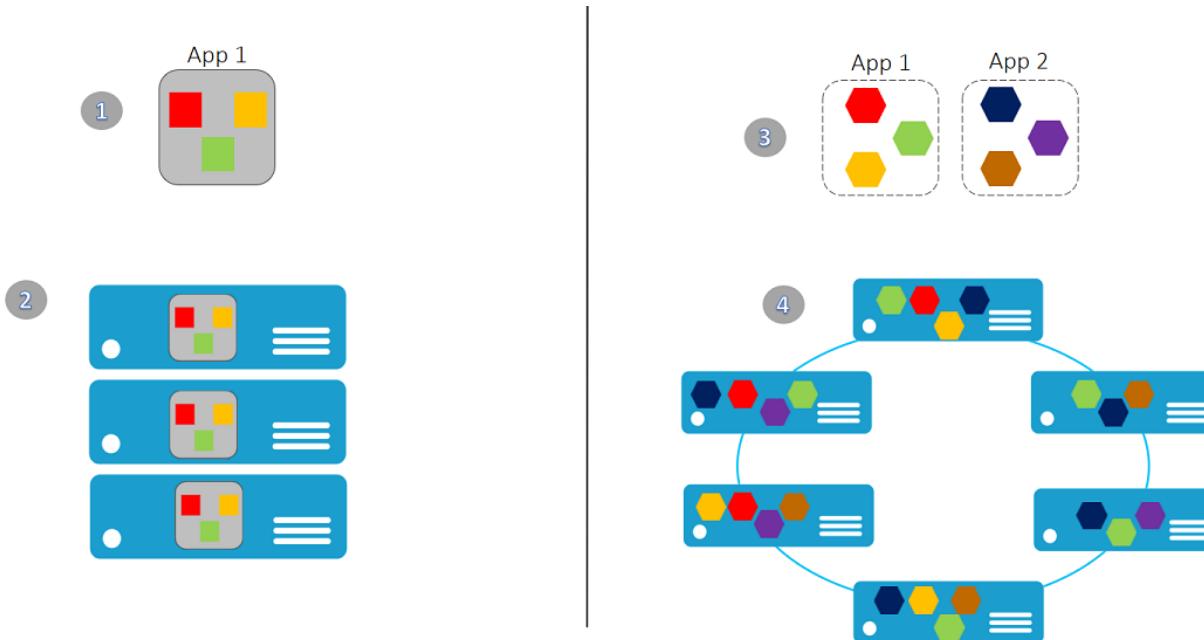
*This is the e-mail Bill Gates sent to every full-time employee at Microsoft, in which he describes the company's new strategy emphasizing security in its products.*From: Bill Gates
Sent: Tuesday, January 15, 2002 5:22 PM

To: Microsoft and Subsidiaries: All FTE
Subject: Trustworthy computing

Every few years I have sent out a memo talking about the highest priority for Microsoft. Two years ago, it was the kickoff of our .NET strategy. Before that, it was several memos about the importance of the Internet to our future and the ways we could make the Internet truly useful for people. Over the last year it has become clear that

Changes in Software Architecture

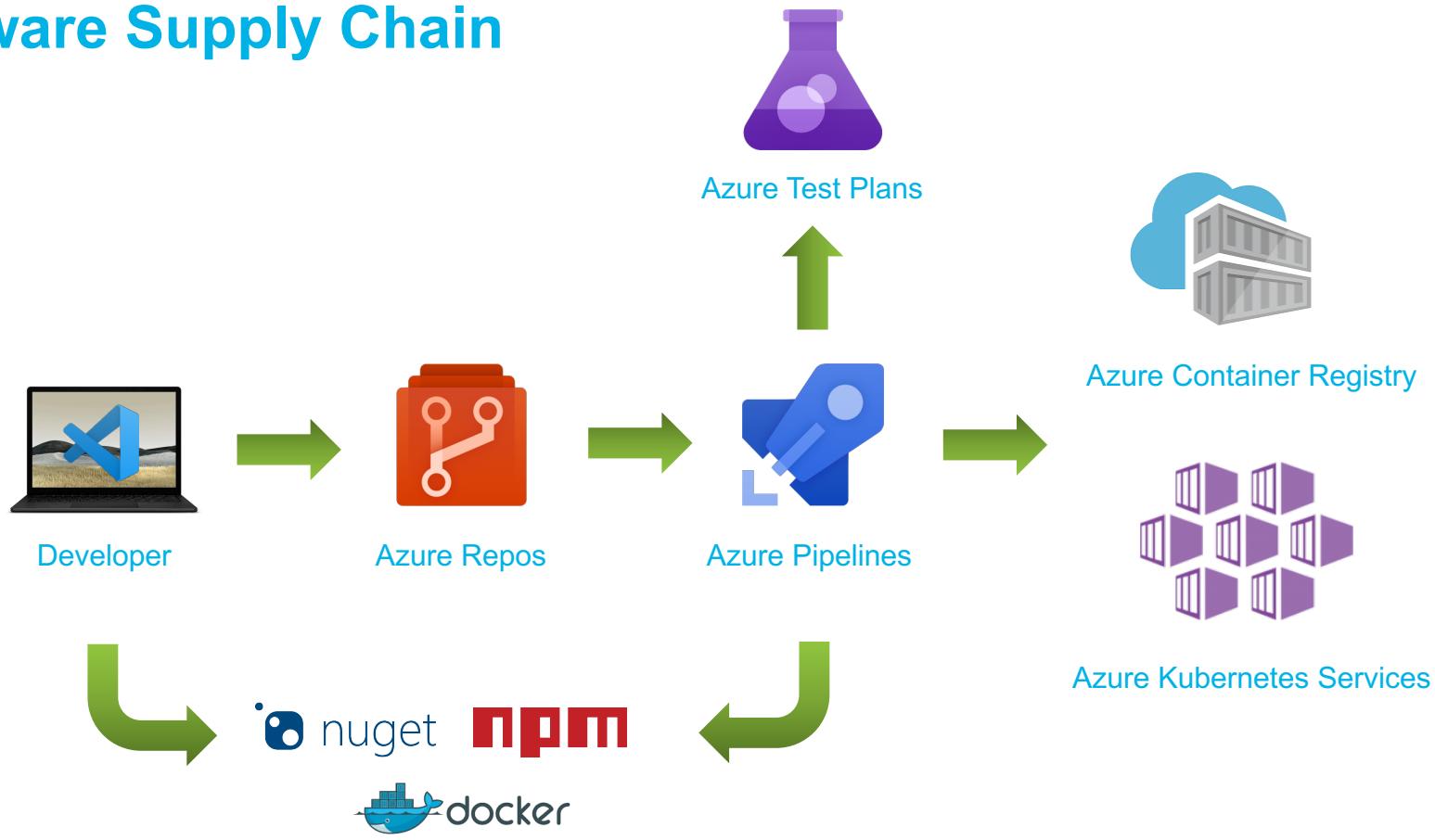
- Monolith → Microservices → Serverless



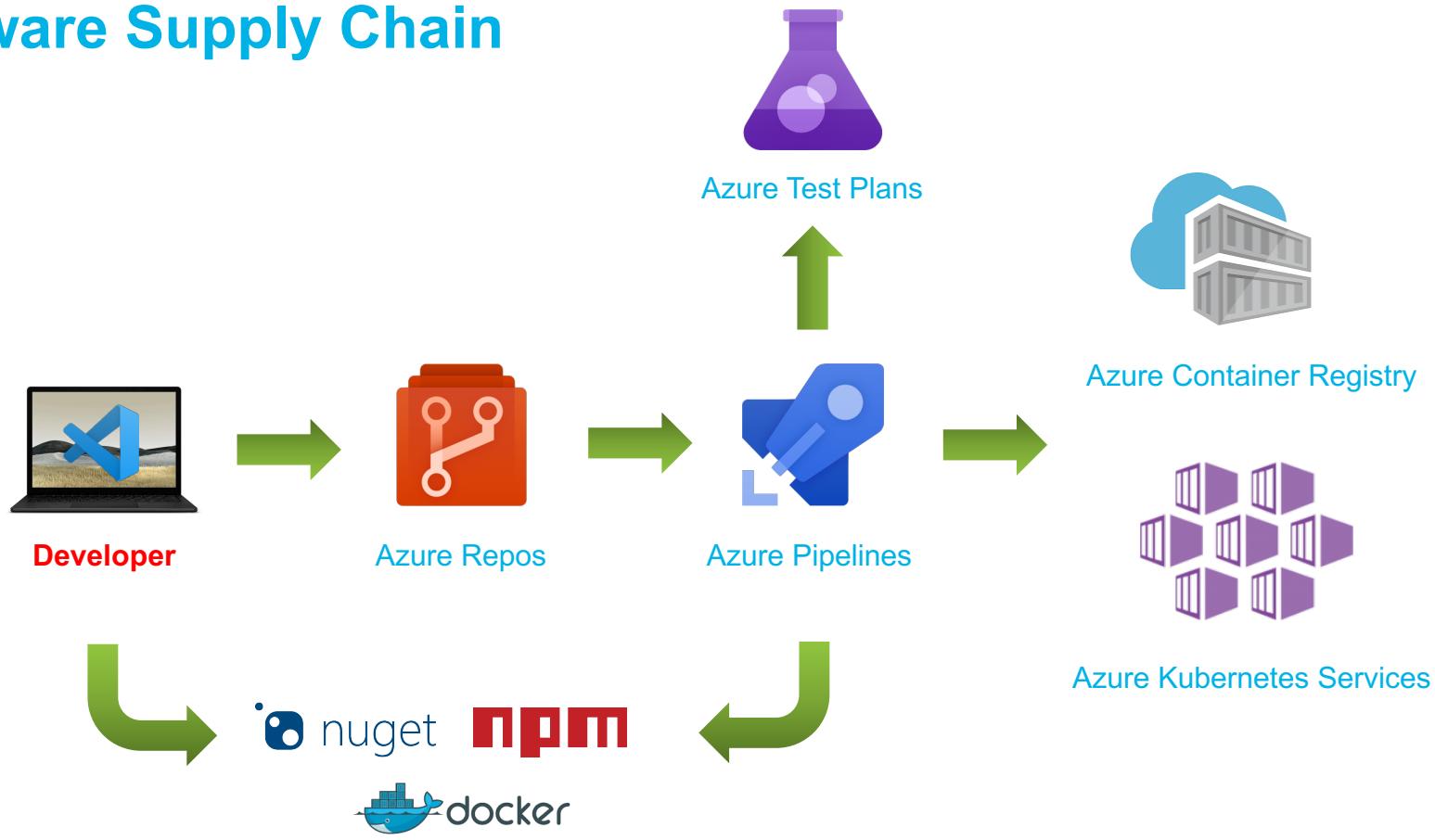
What's a Supply-Chain?



Software Supply Chain



Software Supply Chain



Development Machine

- Secure Boot & Trusted Platform Module (TPM)
- Encrypt disk, harden operating system install updates
- But can you trust the hardware?



Hacking Hardware

Design or Implant?

- Complex, 3D bonding patterns
- Purpose: supply chain flexibility
 - Mfg will routinely swap out sub-components to optimize cost, yield

BlueHat IL 2019 - Andrew "bunnie" Huang - Supply Chain Security: "If I were a Nation State..."

12,251 views • 14 Feb 2019

234 2 SHARE SAVE ...

Up next



AUTOPLAY

ToorCamp 2018 - Keynote:
MAKING AND BREAKING...
ToorCon
2.9K views • 1 year ago

Development Machine



- Installs on machine – HomeBrew on Mac
- ```
/bin/bash -c "$(curl -fSSL
https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"
```

# Development Machine



- Installs on machine – Chocolatey on Windows

## Chocolatey Install:

[Individual](#)   [Organization](#)

1. First, ensure that you are using an [administrative shell](#) - you can also install as a non-admin, check out [Non-Administrative Installation](#).
2. Install with powershell.exe

**NOTE:** Please inspect <https://chocolatey.org/install.ps1> prior to running any of these scripts to ensure safety. We already know it's safe, but you should verify the security and contents of *any* script from the internet you are not familiar with. All of these scripts download a remote PowerShell script and execute it on your machine. We take security very seriously. [Learn more about our security protocols](#).

# Octopus Scanner - NetBeans

[Back to GitHub.com](#)

 Security Lab

Bounties CodeQL **Research** [Advisories](#) [Get Involved](#) [Events](#)



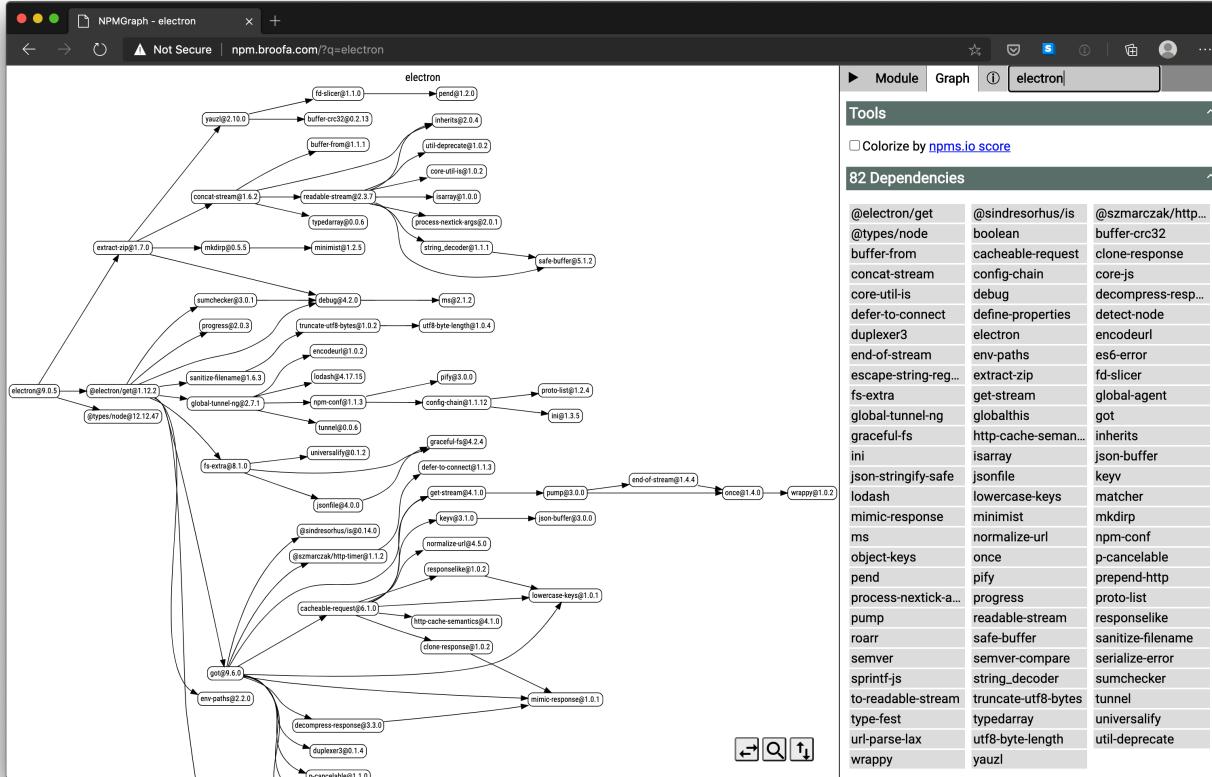
May 28, 2020

## The Octopus Scanner Malware: Attacking the open source supply chain

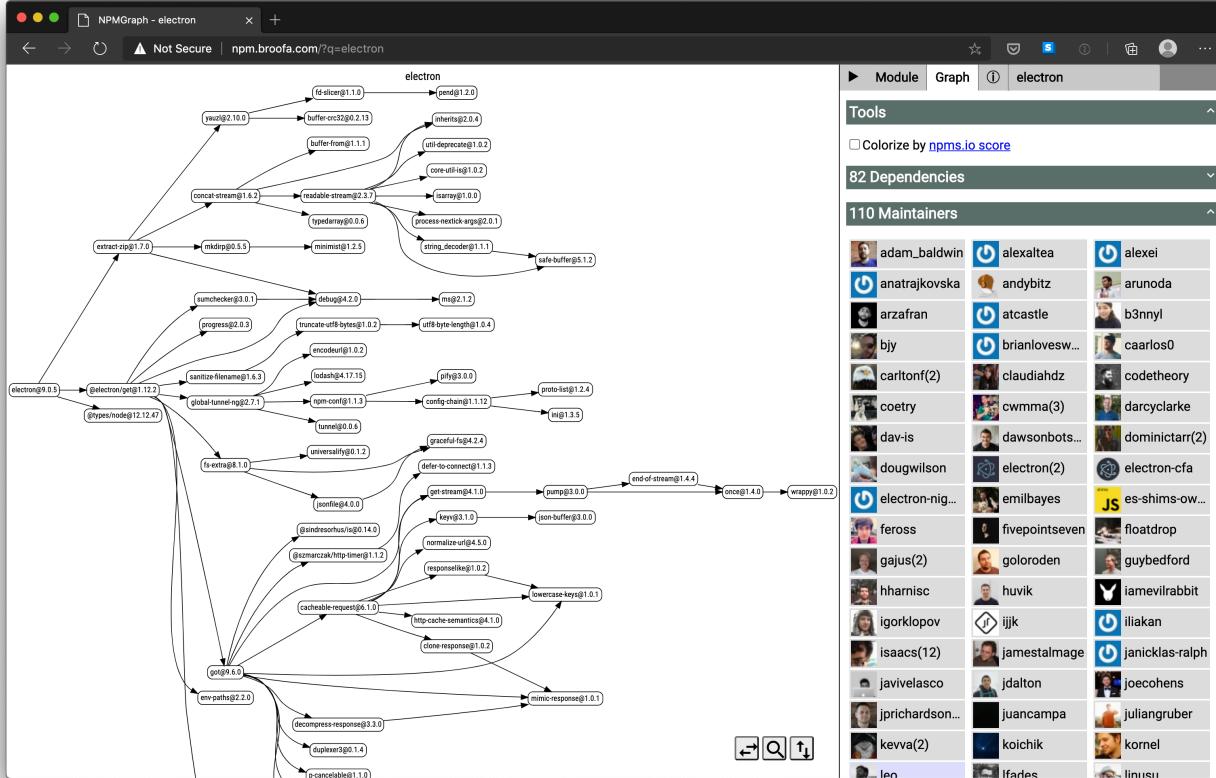


Securing the open source supply chain is an enormous task. It goes far beyond a security assessment or just patching for the latest CVEs. Supply chain security is about the integrity of the entire software development and delivery ecosystem. From the code commits themselves, to how they flow through the CI/CD pipeline, to the actual delivery of releases, there's the potential for loss of integrity and security concerns, throughout the entire lifecycle.

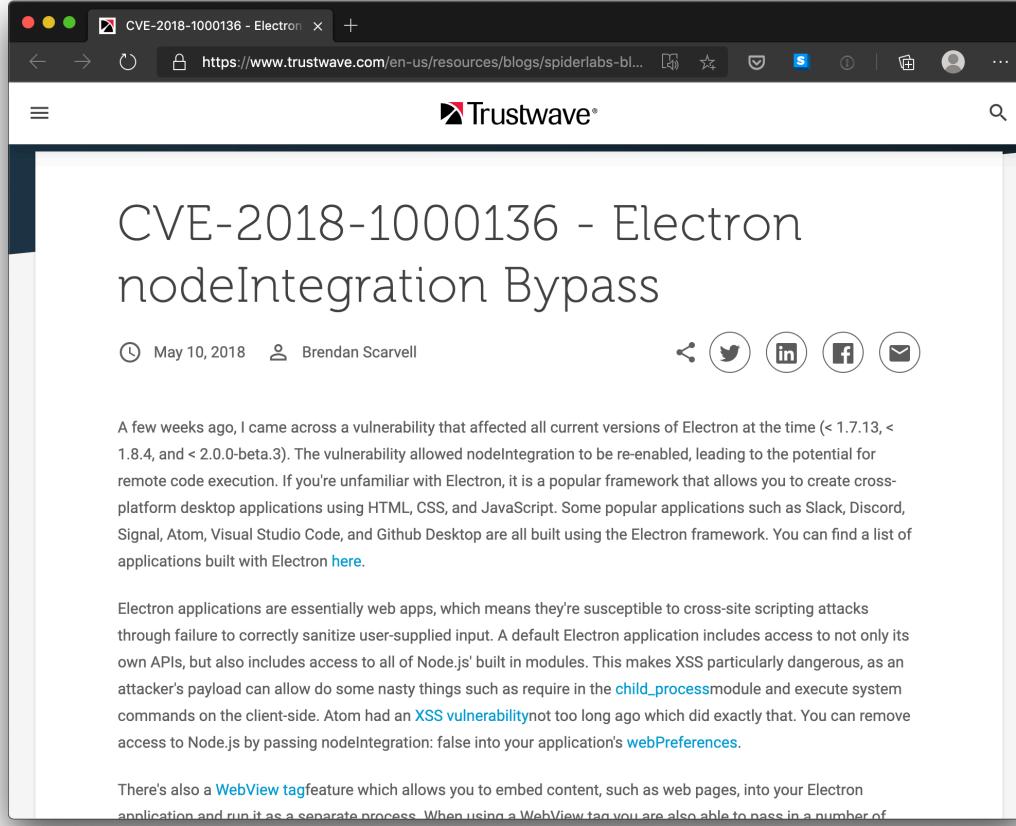
# Visual Studio Code



# Visual Studio Code



# Visual Studio Code



The screenshot shows a web browser window with the title "CVE-2018-1000136 - Electron" and the URL "https://www.trustwave.com/en-us/resources/blogs/spiderlabs-bl...". The page content is from Trustwave's blog, featuring a large heading "CVE-2018-1000136 - Electron nodeIntegration Bypass", a timestamp "May 10, 2018", the author "Brendan Scarvell", and social sharing icons for Twitter, LinkedIn, Facebook, and Email. The main text discusses a vulnerability in Electron that allows remote code execution through nodeIntegration bypass.

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 vulnerability allowed nodeIntegration to be re-enabled, leading to the potential for remote code execution. If you're unfamiliar with Electron, it is a popular framework that allows you to create cross-platform desktop applications using HTML, CSS, and JavaScript. Some popular applications such as Slack, Discord, Signal, Atom, Visual Studio Code, and Github Desktop are all built using the Electron framework. You can find a list of applications built with Electron [here](#).

Electron applications are essentially web apps, which means they're susceptible to cross-site scripting attacks through failure to correctly sanitize user-supplied input. A default Electron application includes access to not only its own APIs, but also includes access to all of Node.js' built in modules. This makes XSS particularly dangerous, as an attacker's payload can allow do some nasty things such as require in the `child_process` module and execute system commands on the client-side. Atom had an [XSS vulnerability](#) not too long ago which did exactly that. You can remove access to Node.js by passing `nodeIntegration: false` into your application's `webPreferences`.

There's also a [WebView tag](#) feature which allows you to embed content, such as web pages, into your Electron application and run it as a separate process. When using a `WebView tag` you are also able to pass in a number of

# Visual Studio Code

The screenshot shows a web browser window displaying a Microsoft Security Response Center (MSRC) page. The title bar reads "CVE-2020-16881 | Visual Studio". The URL in the address bar is <https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-16881>. The page header includes the Microsoft logo, "MSRC", "Report an issue", "Customer guidance", "Engage", "More", "All Microsoft", "Sign in", and a language selector for "United States (English)". The main content area has a breadcrumb navigation "Security Update Guide > Details". The main title is "CVE-2020-16881 | Visual Studio JSON Remote Code Execution Vulnerability". Below it is the section title "Security Vulnerability". A timestamp "Published: 09/08/2020" and the "MITRE CVE-2020-16881" identifier are listed. The main text describes a remote code execution vulnerability in Visual Studio Code. A sidebar titled "On this page" lists links to "Executive Summary", "Exploitability Assessment", "Security Updates", "Mitigations", "Workarounds", "FAQ", "Acknowledgements", "Disclaimer", and "Revisions".

CVE-2020-16881 | Visual Studio JSON Remote Code Execution Vulnerability

Security Vulnerability

Published: 09/08/2020  
MITRE CVE-2020-16881

A remote code execution vulnerability exists in Visual Studio Code when a user is tricked into opening a malicious 'package.json' file. An attacker who successfully exploited the vulnerability could run arbitrary code in the context of the current user. If the current user is logged on with administrative user rights, an attacker could take control of the affected system. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights. To exploit this vulnerability, an attacker would need to convince a target to clone a repository and open it in Visual Studio Code. Attacker-specified code would execute when the target opens the malicious 'package.json' file. The update address the vulnerability by modifying the way Visual Studio Code handles JSON files.

On this page

- Executive Summary
- Exploitability Assessment
- Security Updates
- Mitigations
- Workarounds
- FAQ
- Acknowledgements
- Disclaimer
- Revisions

# Visual Studio Code

The screenshot shows a GitHub issue page for the Microsoft/VSCode repository. The issue is titled "(Assumed) Fix for CVE-2020-16881 can be bypassed" and has the ID #107951. It is marked as "Closed" by justinsteven 12 days ago with 0 comments.

**Issue Details:**

- Assignees:** aeschli
- Labels:** insiders-released, verified
- Projects:** None yet

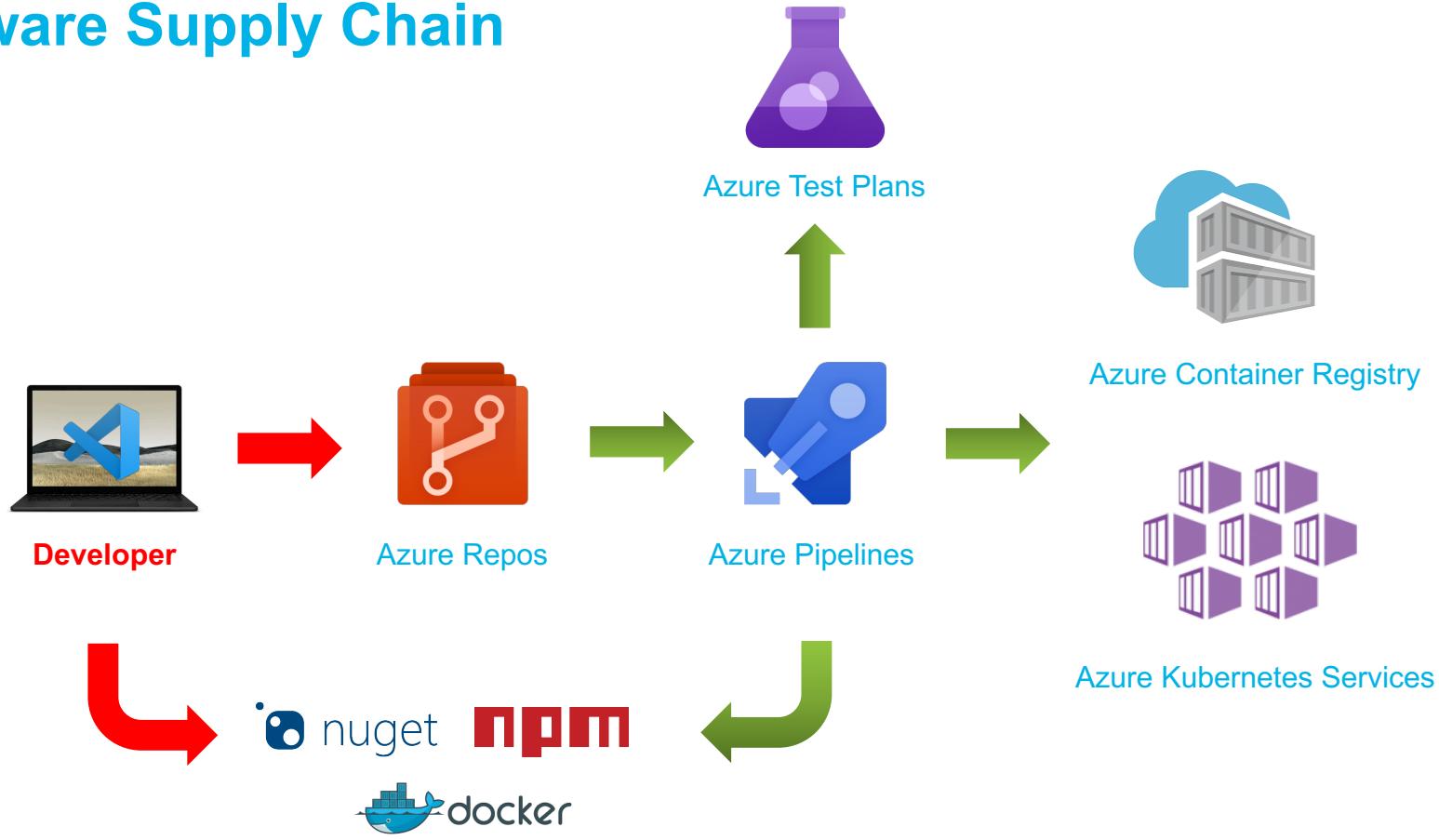
**Comment by justinsteven:**

- VSCode Version: 1.49.2
- OS Version: Can be reproduced on Debian Linux and Windows

<https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-16881>. says that "A remote code execution vulnerability exists in Visual Studio Code when a user is tricked into opening a malicious 'package.json' file."

The advisory links to [https://code.visualstudio.com/updates/v1\\_48](https://code.visualstudio.com/updates/v1_48). There is no mention of

# Software Supply Chain



# Development Machine

- Package manager e.g. NuGet / NPM
- Transport-Layer Security (TLS)
  - Root Authority Trust
  - Downgrade, TLS 1.0 – 1.1 deprecated on NuGet
- Domain Name Service (DNS)
  - DNSSEC → NuGet.org and GitHub.com don't support it



# Canonical GitHub Account

The screenshot shows a news article from ZDNet. At the top, there's a navigation bar with the ZDNet logo, a search icon, a 'MENU' button, a user profile icon, and a 'EU' link. A red banner at the top of the main content area reads: 'MUST READ: Windows 10 critical process failure: Microsoft admits June updates are triggering reboots'. Below this, the main headline is 'Canonical GitHub account hacked, Ubuntu source code safe'. A subtext below it says, 'Ubuntu source code appears to be safe; however Canonical is investigating.' There are social sharing icons for LinkedIn, Facebook, Twitter, and Email. Below the article, there's a photo of a man and the author's bio: 'By Catalin Cimpanu for Zero Day | July 7, 2019 -- 10:38 GMT (11:38 BST) | Topic: Security'. The main content area shows a screenshot of the Canonical Ltd GitHub repository page. The repository has a purple circular icon, is located in London, UK, and has a homepage at <http://www.canonical.com>. It lists 'Repositories', 'People', and 'Projects'. A search bar at the top of the GitHub page includes fields for 'Find a repository?', 'Type: All', and 'Language: All'. Below the search bar, a repository named 'CAN\_GOT\_HAXXD\_10' is shown, updated 20 hours ago. To the right, there are sections for 'Top languages' (Python, Go, Shell, C++) and 'Most used topics' (ubuntu, canonical, ceph, kubernetes).

# Microsoft GitHub Account

**threatpost**

Podcast: Shifting Cloud Security Left With Infrastructure-as-Code → Hackers Breach 3.5 Million MobiFriends Dating App Credentials →

## Report: Microsoft's GitHub Account Gets Hacked



The Shiny Hunters hacking group said it stole 500 GB of data from the tech giant's repositories on the developer platform, which it owns.

Author: Elizabeth Montalbano  
May 8, 2020 / 11:36 am

INFOSEC INSIDER

**Helping Remote Workers Overcome Remote Attacks**  June 10, 2020

**Understanding the Payload-Less Email Attacks Evading Your Security Team**  June 4, 2020

**Long Tail Analysis: A New Hope in the Cybercrime Battle**  May 21, 2020

**The Windows 7 Postmortem: What's at Stake**  May 19, 2020

**VPN Concerns with Unplanned Remote Employees**  May 5, 2020

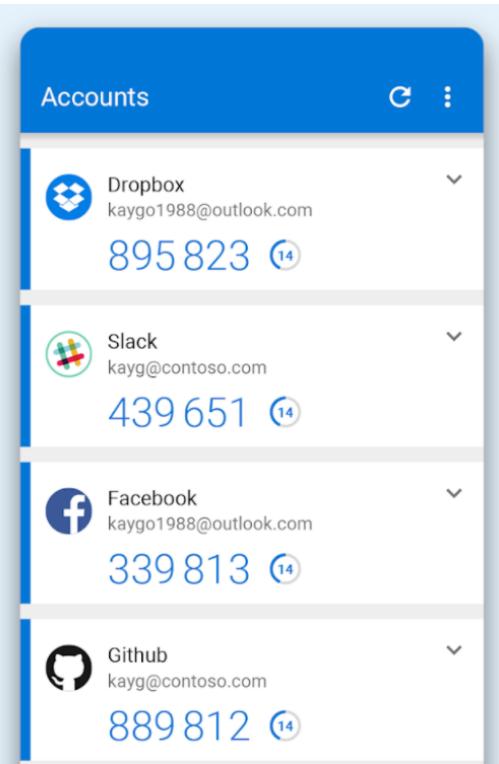
Newsletter 

**Subscribe to Threatpost Today** 

Join thousands of people who receive the latest breaking cybersecurity news every day.

**Subscribe now**

# Use MFA on source-repository



# GIT Commit Signing

The screenshot shows a GitHub repository page for `dotnet / roslyn`. The main navigation bar includes links for Why GitHub?, Team, Enterprise, Explore, Marketplace, Pricing, and a search bar. Below the header, the repository name `dotnet / roslyn` is displayed, along with metrics: Watch (1.1k), Star (13.1k), Fork (3k). The `Code` tab is selected, showing issues (5,000+), pull requests (271), discussions, projects (46), wiki, security, and more.

The commit history is filtered to show only signed commits. A dropdown menu indicates the branch is `master`.

- Commits on May 20, 2020:**
  - Merge pull request #44257 from 333fred/test ...  
jaredpar committed on 21 May ✗  
Verified 64d1878 ca0a72f
- Commits on May 5, 2020:**
  - Merge pull request #43954 from jaredpar/maintain-perf ...  
jaredpar committed on 5 May ✗  
Verified ca0a72f 8c74
- Commits on May 4, 2020:**
  - Enforce analyzer consistency in our builds ...  
jaredpar committed on 4 May ✓  
This commit was created on GitHub.com and signed with a **verified signature** using GitHub's key.  
GPG key ID: 4AEE18F83AFDEB23  
Learn about signing commits
- Commits on Apr 30, 2020:**
  - Move MS.CA.VisualBasic to be multi-targeted (#43805) ...  
jaredpar committed on 30 Apr ✗  
Verified f77d925



# EvenStream NPM

- November 2018
- Is transitive dependency of 2000 other libraries



Gary Bernhardt  
@garybernhardt

Follow

An NPM package with 2,000,000 weekly downloads had malicious code injected into it. No one knows what the malicious code does yet.



I don't know what to say. · Issue #116 · dominictarr...

EDIT 26/11/2018: Am I affected?: If you are using anything crypto-currency related, then maybe. As discovered by @maths22, the target seems to have b...

[github.com](https://github.com)

9:44 am - 26 Nov 2018

2,736 Retweets 3,193 Likes



69 2.7K 3.2K

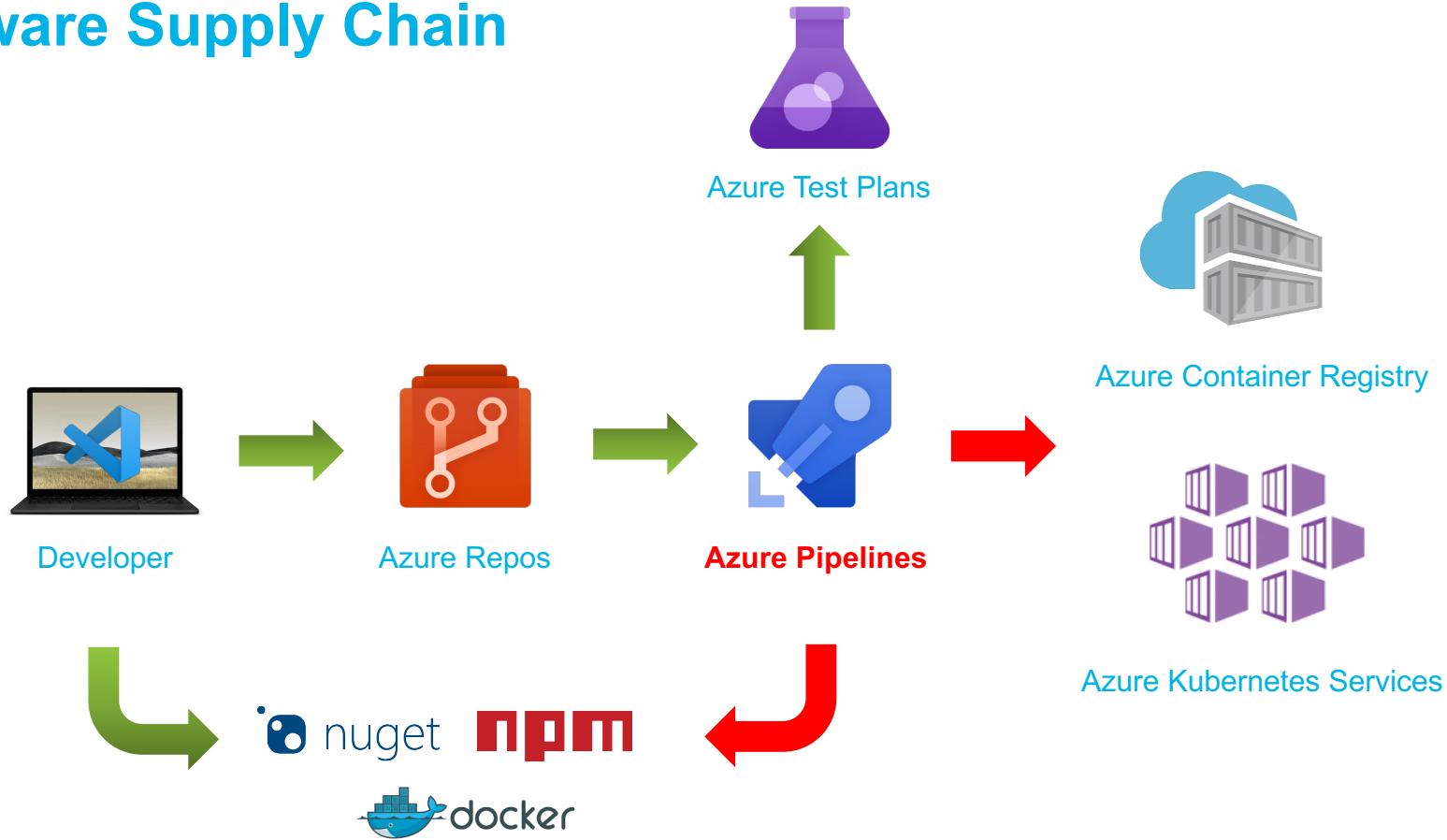


Gary Bernhardt @garybernhardt · 26 Nov 2018

There are basically two camps in that thread.

1) This is the original maintainer's fault for transferring ownership to someone they didn't know / trust.

# Software Supply Chain

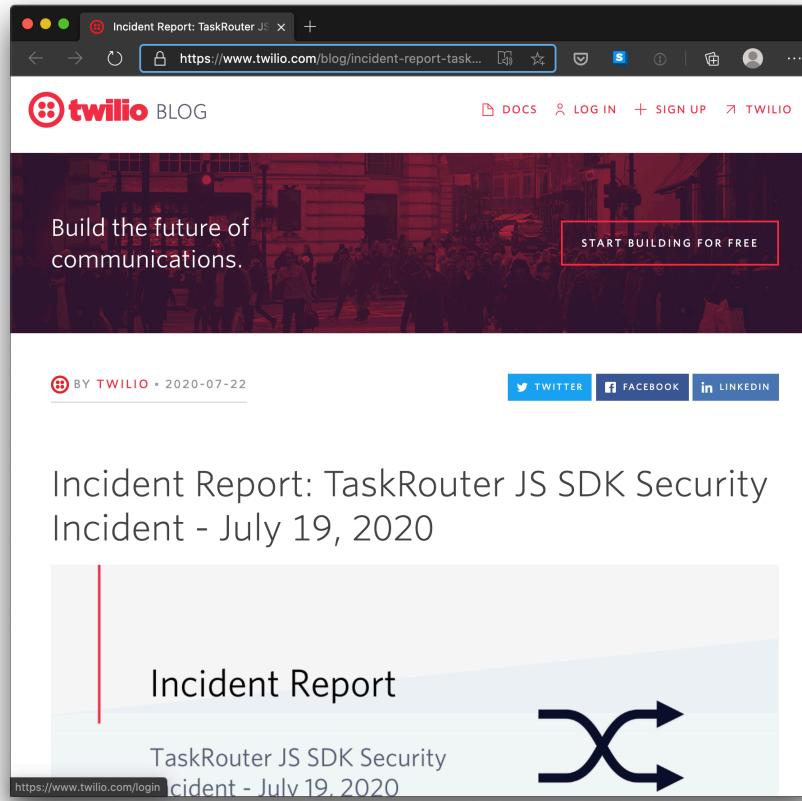


# Build / Deployment

- What about hardware? Vendor trust?
- TLS issues?
- Compromised Docker Images
  - Two-Factor authentication in beta
- Build Server can be compromised



# Twilio SDK



# Webmin Backdoor

The screenshot shows the official Webmin website. At the top, there's a navigation bar with links for Home, Downloads, Documentation, Usermin, Virtualmin, Cloudmin, and Community. Below the navigation is a sidebar with sections for "Download Webmin 1.941" (links for RPM, Debian Package, TAR file, Solaris Package, Development Versions, and Third-Party Modules) and "Webmin Links" (links for Introduction To Webmin, Supported Systems, Module Documentation, Screenshots, Standard Modules, Supported Languages, and Updated Modules). The main content area features a heading "Webmin 1.890 Exploit - What Happened?" followed by a detailed explanation of the vulnerability. It states that Webmin version 1.890 was released with a backdoor, and versions 1.900 to 1.920 also contained similar code. The exploit was only exploitable if the "Authentication" feature was enabled. The text also notes that the exploit was added after the file was reverted from GitHub and modified again by the attacker.

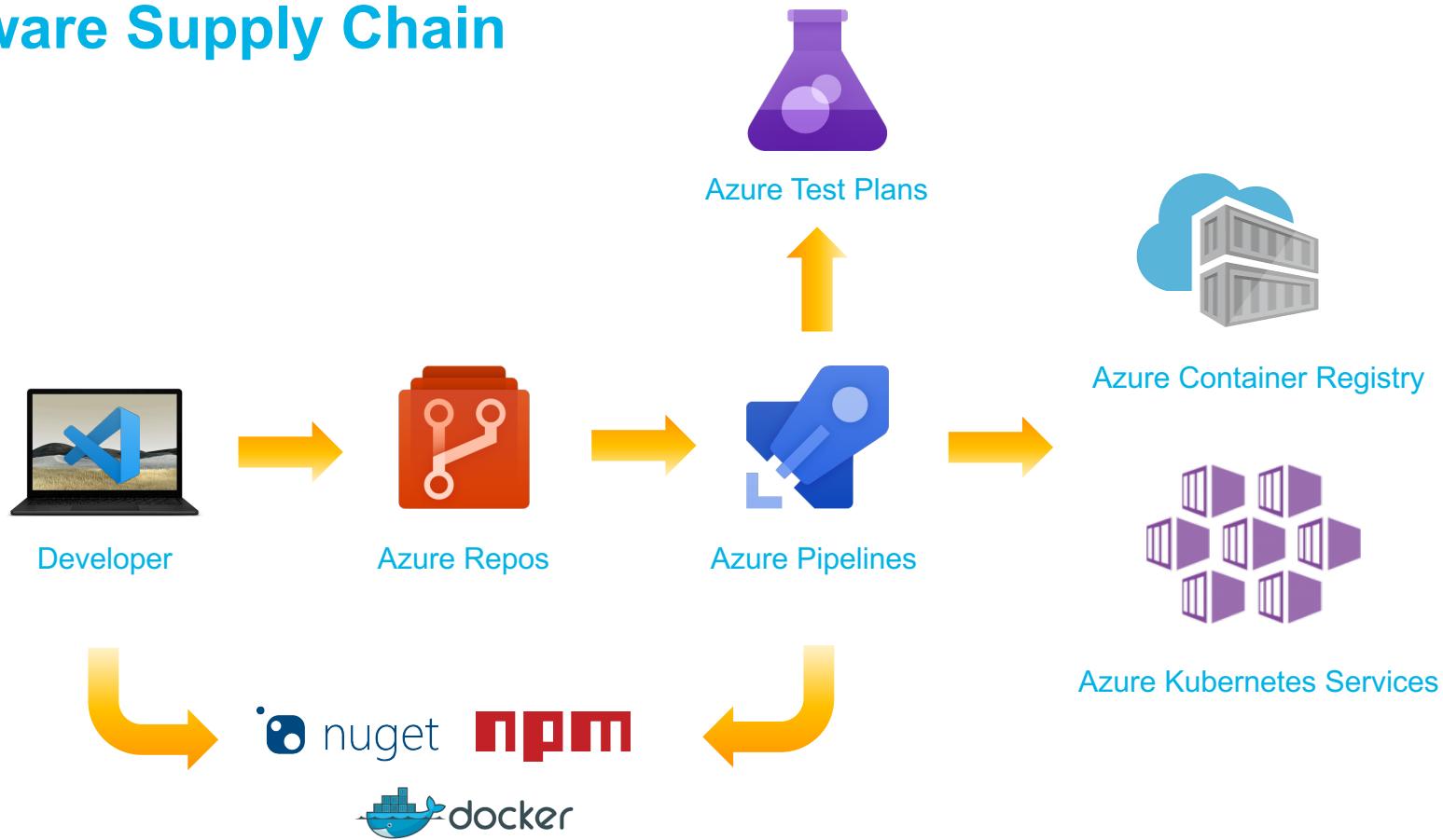
**Webmin 1.890 Exploit - What Happened?**

Webmin version 1.890 was released with a backdoor that could allow anyone with knowledge of it to execute commands as `root`. Versions 1.900 to 1.920 also contained a backdoor using similar code, but it was not exploitable in a default Webmin install. Only if the admin had enabled the feature at Webmin -> Webmin Configuration -> Authentication to allow changing of expired passwords could it be used by an attacker.

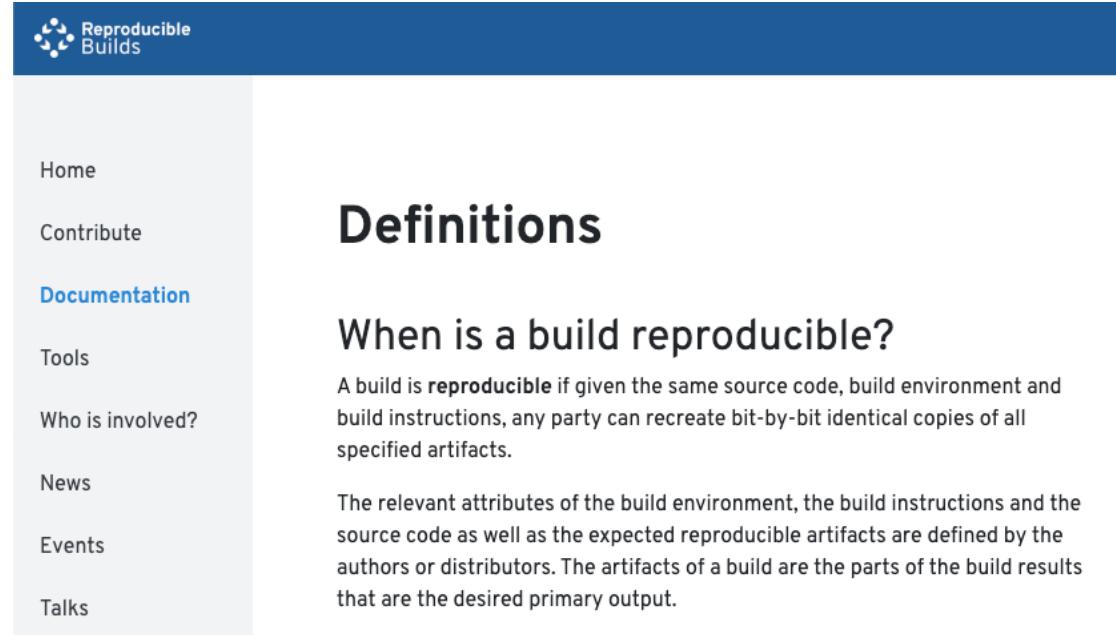
Neither of these were accidental bugs - rather, the Webmin source code had been maliciously modified to add a non-obvious vulnerability. It appears that this happened as follows :

- At some time in April 2018, the Webmin development build server was exploited and a vulnerability added to the `password_change.cgi` script. Because the timestamp on the file was set back, it did not show up in any Git diffs. This was included in the Webmin 1.890 release.
- The vulnerable file was reverted to the checked-in version from Github, but sometime in July 2018 the file was modified again by the attacker. However, this time the exploit was added to code that is only executed if changing of expired passwords is enabled. This was included in the Webmin 1.900 release.

# Software Supply Chain



# Reproducible/Deterministic Builds



The screenshot shows a website for "Reproducible Builds". The header features a blue bar with the "Reproducible Builds" logo, which consists of four white circles connected by lines, followed by the text "Reproducible Builds". Below this is a light gray sidebar containing links: Home, Contribute, Documentation (which is highlighted in blue), Tools, Who is involved?, News, Events, and Talks. The main content area has a dark blue header with the title "Definitions". Below the header, the text "When is a build reproducible?" is displayed. A detailed explanation follows: "A build is **reproducible** if given the same source code, build environment and build instructions, any party can recreate bit-by-bit identical copies of all specified artifacts. The relevant attributes of the build environment, the build instructions and the source code as well as the expected reproducible artifacts are defined by the authors or distributors. The artifacts of a build are the parts of the build results that are the desired primary output."

## Definitions

### When is a build reproducible?

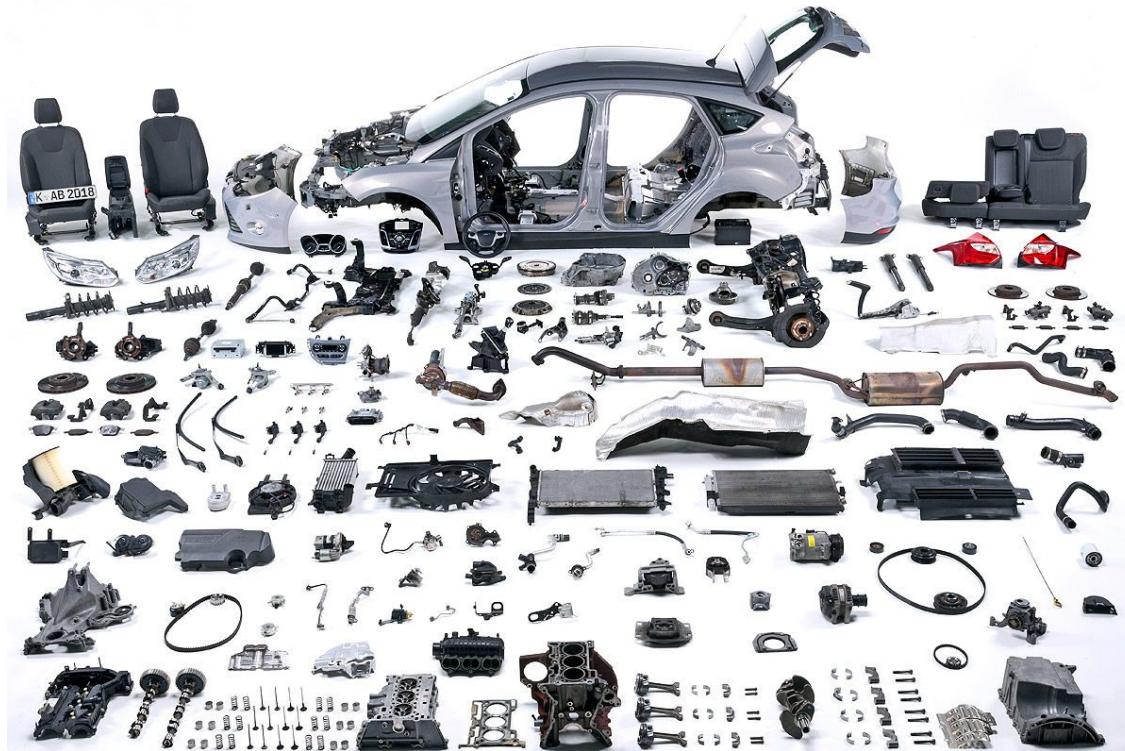
A build is **reproducible** if given the same source code, build environment and build instructions, any party can recreate bit-by-bit identical copies of all specified artifacts.

The relevant attributes of the build environment, the build instructions and the source code as well as the expected reproducible artifacts are defined by the authors or distributors. The artifacts of a build are the parts of the build results that are the desired primary output.

# Reproducible/Deterministic Builds

- Roslyn v1.1 started supporting some kind of determinism on how items are emitted
- Given same inputs, the compiled output will always be deterministic
- Inputs can be found in Roslyn compiler docs ‘Deterministic Inputs’
  - Current full directory path influences PDB location embedded in PE Header
- PE Header has values that need MSBuild **/deterministic** flag set (default .NET Core)
  - MVID: GUID identifying the PE which will be generated by compiler
  - PDB ID: GUID identifying the generated PDB matching PDB generated on each build
  - Date / Time stamp: Seconds since epoch, calculated on every build

# Automotive Industry



# Car Supply Chain



## Tata Steel Factory

- Iron Ore from Sweden
- ISO 6892-1 Tested/Certified
  - Batch #1234



## Bosch Factory

- Steel Batch #1234 Tata
- ECE-R90 Tested/Certified
  - Serie #45678
- Used by Ford, Volkswagen and KIA



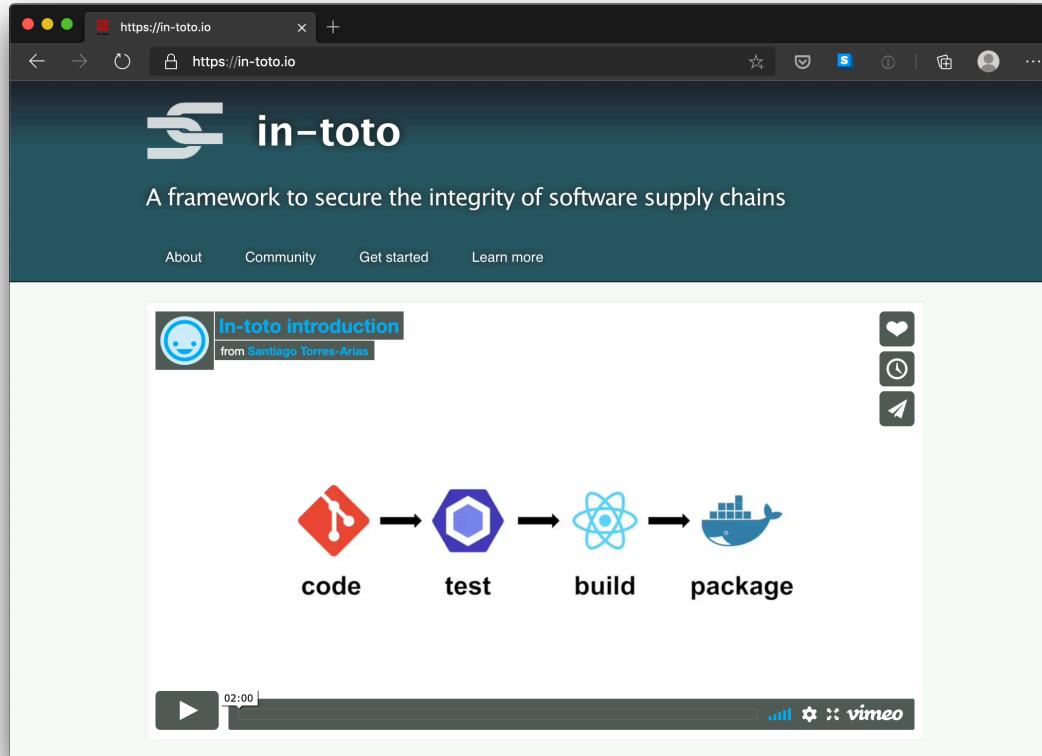
## Ford Manufacturing

- Bosch Disk #45678
- Bosal Exhaust #RE9876
- Goodyear Tires #GY8877
- Focus VIN 1234567890

# Software Bill of Materials (SBOM) & Policy

- Industry standard of describing the software
  - Producer Identity – Who Created it?
  - Product Identity – What's the product?
  - Integrity – Is the project unaltered?
  - Licensing – How can the project be used?
  - Creation – How was the product created? Process meets requirements?
  - Materials - How was the product created? Materials/Source used?
- CycloneDX – Lightweight SBOM with dependency graph
- National Telecommunications and Information Administration <https://www.ntia.gov/SBOM>

# In-Toto



# In-Toto – Demo - Terminology

- **Functionaries** that are identified by public key our supply chain.  
Niels (Project-Owner), Aimee (Developer) and Noud (Packager)
- **Project-Owner** defines a **(Supply Chain) Layout** that describes **what** happens and by **who** and what the produced **Materials** and **Byproducts** are.
- Link metadata is output of executed step in the **Layout**  
**Materials** are input, **Products** are output and can be used as **Materials** in later steps
- With executing the **Verification** all end-products and associated **Link** metadata will be verified

# DataDog & In-Toto

The screenshot shows a web browser displaying a blog post from DataDog. The title of the post is "End-to-end verification with in-toto". The post discusses how to prevent tampering in the software supply chain by using In-Toto to specify a fixed series of steps, each producing signed metadata, which a client like the Datadog Agent can inspect.

**DEVELOPERS**

- TAG → YUBIKEYS
- YUBIKEYS → TAG-LINK (dd-check/ setup.py: 0xA)

**CI/CD**

- WHEELS-BUILDER → WHEELS-BUILDER KEY
- WHEELS-BUILDER KEY → WHEELS-BUILDER LINK (dd-check/ setup.py: 0xA, dd-check/ whl: 0x8)
- WHEELS-SIGNER → WHEELS-SIGNER KEY
- WHEELS-SIGNER KEY → WHEELS-SIGNER.LINK (dd-check/ whl: 0x8)

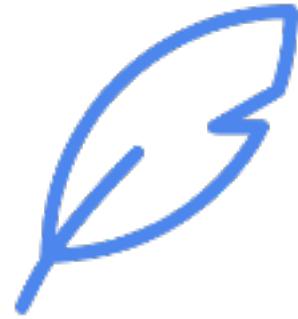
**AGENT**

- UNZIP → UNZIP.LINK (dd-check/ whl: 0x8, dd-check/ setup.py: 0xA)

The diagram illustrates the flow of data and signatures between Developers, CI/CD, and Agent, ensuring end-to-end verification through In-Toto.

# Grafeas and Kritis by Google

- Grafeas – Component Metadata API
  - Container Analysis API on Google Cloud Platform
- Kritis – Deployment Authorization for Kubernetes Applications
  - Binary Authorization on Google Cloud Platform



# Azure Pipelines Artifact Policy

The screenshot shows a web browser displaying the Microsoft Docs website for the "Artifact policy checks" article. The URL is <https://docs.microsoft.com/en-us/azure/devops/pipelines/process/artifact-policy-checks>. The page has a dark theme. On the left, there's a sidebar with navigation links for "Version" (Azure DevOps Services), "Filter by title", and several sections like "Configure security & settings", "Migrate", "Pipeline tasks", "Troubleshooting", "Reference", "YAML schema", "Expressions", "File matching patterns", "File and variable transform", "Logging commands", "Artifact policy checks" (which is highlighted in blue), "Case studies & best practices", "Developer resources", and "Download PDF". The main content area has a heading "Artifact policy checks", a note about supported artifact types (container images and Kubernetes environments), a "Prerequisites" section, and a note about using Rego for defining policies. There are also social sharing icons at the top right.

# Conclusion

- Be aware of your own (and other used) software supply chain(s).
- Know what you're consuming and pulling into software projects.
- Use MFA on all accounts!
- Integrate security into your software lifecycle.
- Learn more on Software Bill of Materials (SBOM).



Thanks! Questions?

**VERACODE**

You change the world, we'll secure it.

<https://github.com/nielstanis/codecamp2020>

ntanis at veracode.com

@nielstanis on Twitter