

**VERACODE**  
You change the world, we'll secure it.

**.NET CORE SUMMER EVENT**

Niels Tanis

## The Rise of Software Supply-Chain Attacks

How Secure is your .NET Application?

## Who am I?

- Niels Tanis
  - Security Researcher @ Veracode
  - Background in .NET Development
  - Application Security Consultancy
  - Pen-testing & Ethical Hacking
  - ISC<sup>2</sup> CSSLP



01 @nielstanis



Picture is from Veracode report/site:

<https://www.veracode.com/sites/default/files/pdf/resources/whitepapers/everything-you-need-to-know-about-open-source-risk/index.html>

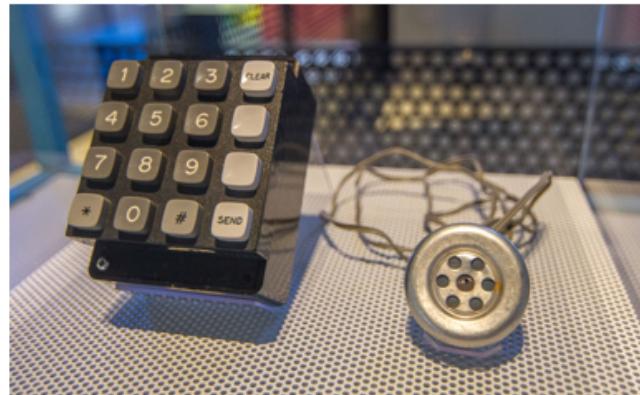
## Agenda

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

01 @nielstanis

## Hacking History

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



O1 @nielstanis

[https://www.wikiwand.com/en/Blue\\_box](https://www.wikiwand.com/en/Blue_box)

[https://en.wikipedia.org/wiki/Kevin\\_Mitnick](https://en.wikipedia.org/wiki/Kevin_Mitnick)

<https://www.youtube.com/watch?v=8s4b2ZKyPHc>

## 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
```

01 @nielstanis

[https://en.wikipedia.org/wiki/Security\\_Administrator\\_Tool\\_for\\_Analyzing\\_Networks](https://en.wikipedia.org/wiki/Security_Administrator_Tool_for_Analyzing_Networks)  
<https://nmap.org>

## 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

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX  
Smashing The Stack For Fun And Profit  
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

by Aleph One  
alephi@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 losage, overrun screw.



<http://phrack.org/issues/49/14.html#article>

## 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
```



<http://www.phrack.org/issues/54/8.html>

## Code Red & SQL Slammer

- Microsoft Internet Information Server, July 2001

```
GET /default.ida?NNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN  
%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
```



[https://en.wikipedia.org/wiki/Code\\_Red\\_\(computer\\_worm\)](https://en.wikipedia.org/wiki/Code_Red_(computer_worm))

[https://en.wikipedia.org/wiki/SQL\\_Slammer](https://en.wikipedia.org/wiki/SQL_Slammer)

## 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

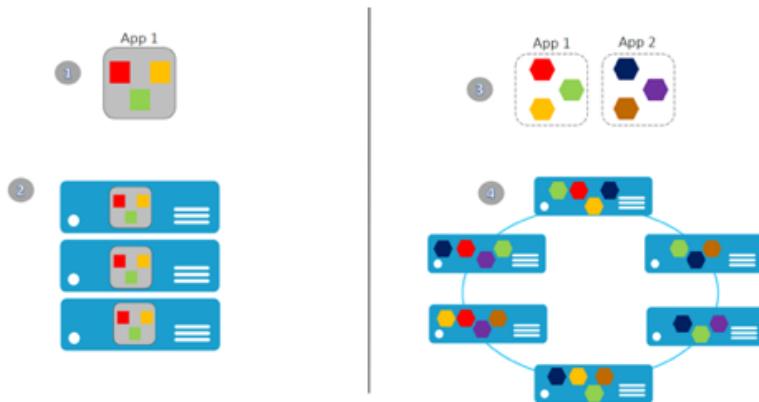
Every few years I have sent out a memo talking about the highest priority for Microsoft. Two years ago, it was the kickoff of our .NET strategy. Before that, it was several memos about the importance of the Internet to our future and the ways we could make the Internet truly useful for people. Over the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

01 @nielstanis

<https://www.wired.com/2002/01/bill-gates-trustworthy-computing/>

## Changes in Software Architecture

- Monolith → Microservices → Serverless



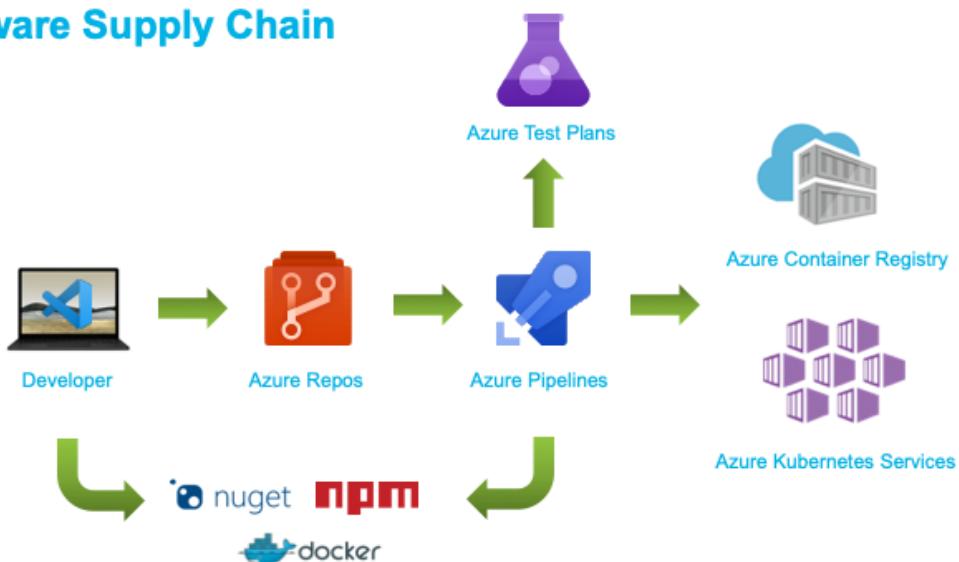
O1 @nielstanis

## What's a Supply-Chain?



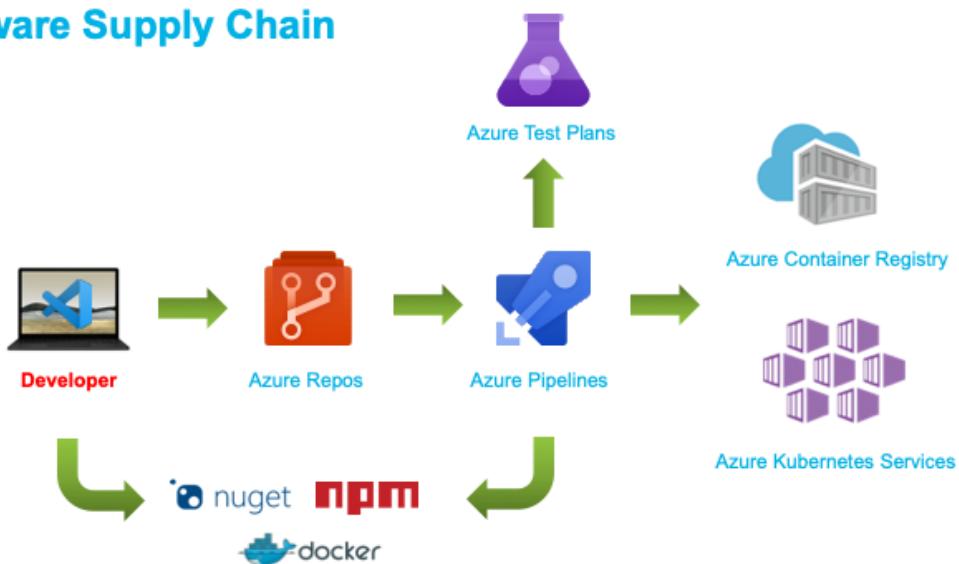
O1 @nielstanis

## Software Supply Chain



O1 @nielstanis

## Software Supply Chain



O1 @nielstanis

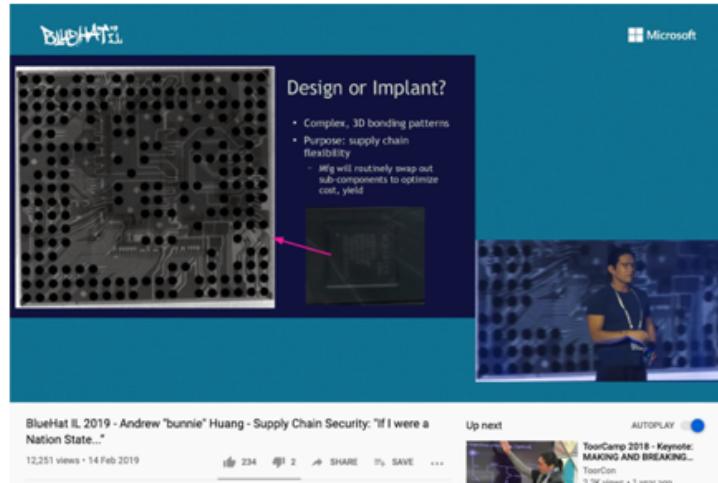
## Development Machine

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



OI @nielstanis

## Hacking Hardware



01 @nielstanis

<https://www.youtube.com/watch?v=RqQhWitJ1As>

## Development Machine



- Installs on machine – HomeBrew on Mac

```
• curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh
```

01 @nielstanis

## Development Machine



- Installs on machine – Chocolately 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://chocolately.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](#).

01 @nielstanis

The screenshot shows a GitHub research page titled "Octopus Scanner - NetBeans". The page has a dark header with "Back to GitHub.com" and a "Security Lab" logo. Below the header, there are navigation links for "Bounties", "CodeQL", "Research" (which is underlined), "Advisories", "Get Involved", and "Events". The main content area features a large dotted graphic on the left and a title card on the right. The title card includes the date "May 28, 2020", the title "The Octopus Scanner Malware: Attacking the open source supply chain", and the author's name "Alvaro Muñoz" with a small profile picture. A short summary follows the title.

**Octopus Scanner - NetBeans**

Back to GitHub.com

Security Lab

Bounties CodeQL **Research** Advisories Get Involved Events

May 28, 2020

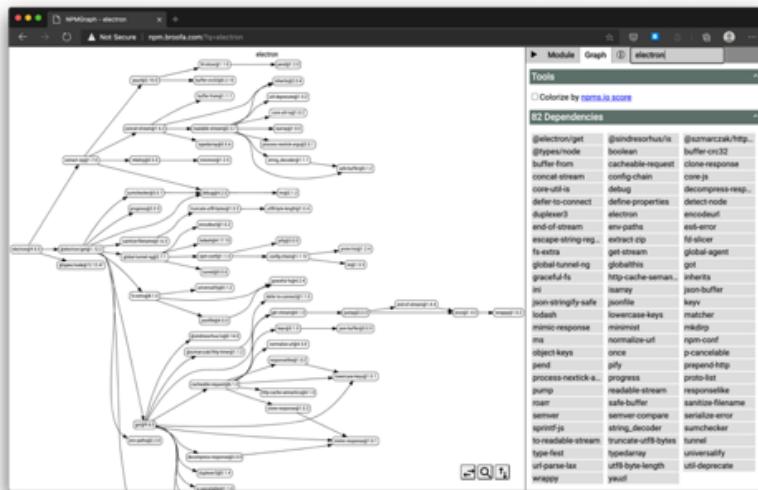
**The Octopus Scanner Malware:  
Attacking the open source  
supply chain**

Alvaro Muñoz

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.

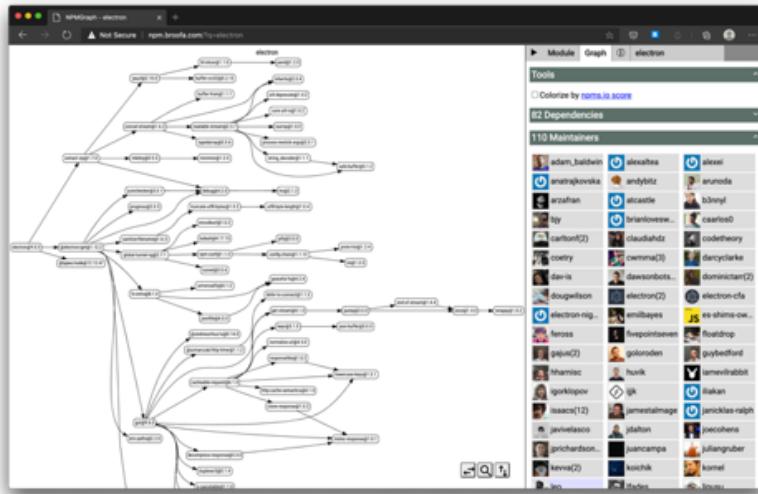
<https://securitylab.github.com/research/octopus-scanner-malware-open-source-supply-chain>

## Visual Studio Code



O1 @nielstanis

## Visual Studio Code



O1 @nielstanis

## Visual Studio Code

The screenshot shows a web browser window displaying a blog post. The title of the post is "CVE-2018-1000136 - Electron nodeIntegration Bypass". The author is listed as "Brendan Scarvell" and the date is "May 10, 2018". Below the title, there is a paragraph of text explaining the vulnerability. At the bottom of the post, there is a note about the WebView tag feature.

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

OI @nielstanis

<https://www.trustwave.com/en-us/resources/blogs/spiderlabs-blog/cve-2018-1000136-electron-nodeintegration-bypass/>

## Visual Studio Code

The screenshot shows a Microsoft web page for a security advisory. The main content area displays the following information:

**CVE-2020-1192 | Visual Studio Code Python Extension Remote Code Execution Vulnerability**

Published: 05/12/2020  
MITRE CVE-2020-1192

A remote code execution vulnerability exists in Visual Studio Code when the Python extension loads workspace settings from a notebooks 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 open a specially crafted file in Visual Studio Code with the Python extension installed.

The update address the vulnerability by modifying the way Visual Studio Code Python extension enforces user settings.

**Exploitability Assessment**

The following table provides an [exploitability assessment](#) for this vulnerability at the time of original publication.

Publicly Disclosed	Exploited	Latest Software Release	Older Software Release	Denial of Service

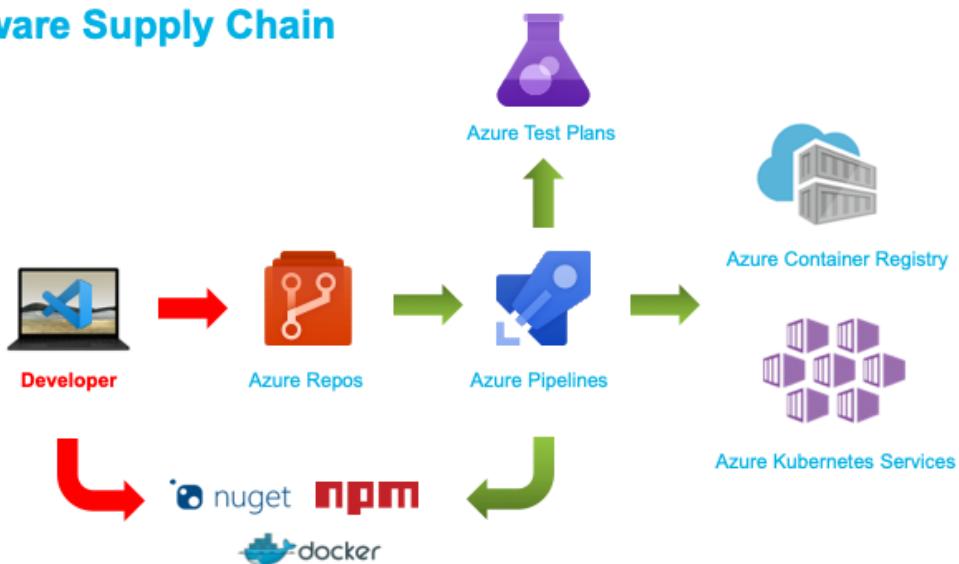
**On this page**

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

**At the bottom left:** OI @nielstanis

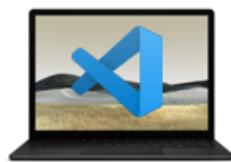
<https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-1192>

## Software Supply Chain



O1 @nielstanis

## 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

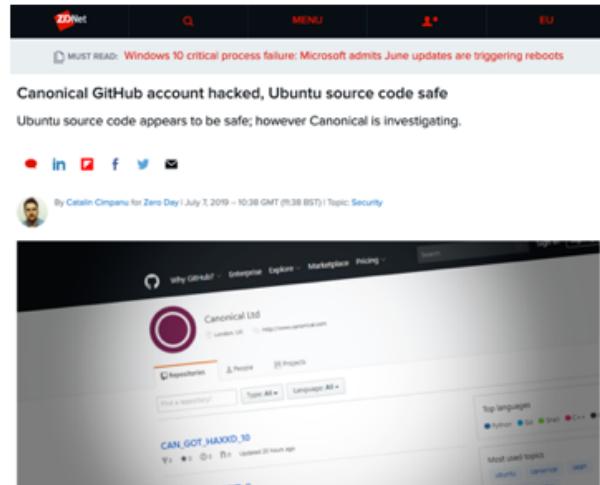
01 @nielstanis

[https://www.ssllabs.com/downloads/SSL\\_Threat\\_Model.png](https://www.ssllabs.com/downloads/SSL_Threat_Model.png)

<https://devblogs.microsoft.com/nuget/deprecating-tls-1-0-and-1-1-on-nuget-org/>

<https://dnssec-analyzer.verisignlabs.com/nuget.org>

## Canonical GitHub Account



01 @nielstanis

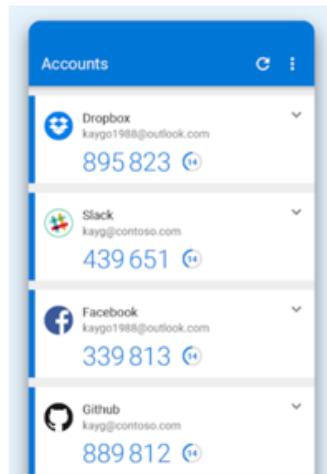
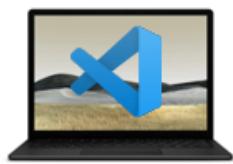
<https://www.zdnet.com/article/canonical-github-account-hacked-ubuntu-source-code-safe/>

## Microsoft GitHub Account

The screenshot shows a news article from threatpost.com. The title is "Report: Microsoft's GitHub Account Gets Hacked". Below the title is a large image of the GitHub logo. The article discusses a hacking group called "The Shiny Hunters" who stole 500 GB of data from Microsoft's GitHub repositories. The author is Elizabeth Montalbano, and the date is May 8, 2020. To the right of the article, there is a sidebar with several other news articles and a "Subscribe" button.

<https://threatpost.com/report-microsofts-github-account-gets-hacked/155587/>

## Use MFA on source-repository



OI @nielstanis

<https://help.github.com/en/github/authenticating-to-github/configuring-two-factor-authentication>

The screenshot shows a GitHub repository page for `dotnet/roslyn`. The main content area displays a list of commits, each with a green checkmark icon indicating it is verified. A tooltip for one commit states: "This commit was created on GitHub.com and signed with a verified signature using GitHub's key." A GPG key ID is also listed. At the top right of the page, there is a graphic of a laptop displaying the Visual Studio logo.

**GIT Commit Signing**

dotnet / roslyn

Code Issues 6,000+ Pull requests 271 Discussions Projects 46 Wiki Security

Branch: master

- Commits on May 20, 2020
  - Merge pull request #44297 from 333fred/test Verified 6401878
- Commits on May 5, 2020
  - Merge pull request #43954 from jaredpar/maintain-perf Verified caba72f
  - Enforce analyzer consistency in our builds Verified 8c74
- Commits on May 4, 2020
  - Move MS.CA.VisualBasic to be multi-targeted (#43805) Verified f77d925
- Commits on Apr 30, 2020
  - jaredpar committed on 30 Apr

01 @nielstanis

[https://www.hanselman.com/blog/HowToSetupSignedGitCommitsWithAYubiKeyNEO AndGPGAndKeybaseOnWindows.aspx](https://www.hanselman.com/blog/HowToSetupSignedGitCommitsWithAYubiKeyNEOAndGPGAndKeybaseOnWindows.aspx)

## EvenStream NPM

- November 2018
- Is transitive dependency of 2000 other libraries

Gary Bernhardt  
@garybernhardt

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](#)

9:44 am - 26 Nov 2018

2,736 Retweets 3,193 Likes

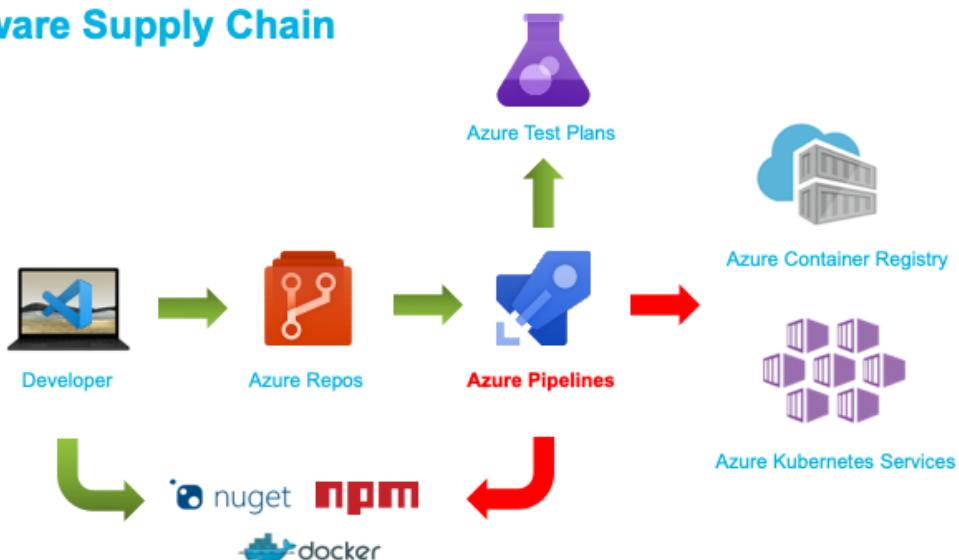
69 2.7K 3.2K

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

<https://twitter.com/garybernhardt/status/1067111872225136640>

## Software Supply Chain



O1 @nielstanis

## Build / Deployment



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

01 @nielstanis

<https://docs.docker.com/docker-hub/2fa/>

## XCodeGhost

The screenshot shows a web browser window displaying a blog post. The title of the post is "XcodeGhost malware infiltrates App Store". The post is by Thomas Reed, posted on September 21, 2015, and last updated on March 11, 2016. The content discusses the discovery of XcodeGhost malware, which was found in the App Store. The author's bio indicates he had a Mac before it was cool to have Macs, is a self-trained Apple security expert, and an amateur photographer. A sidebar on the right features a "FREE DOWNLOAD" button and an "ABOUT THE AUTHOR" section with a photo of Thomas Reed.

<https://blog.malwarebytes.com/cybercrime/2015/09/xcodeghost-malware-infiltrates-app-store/>

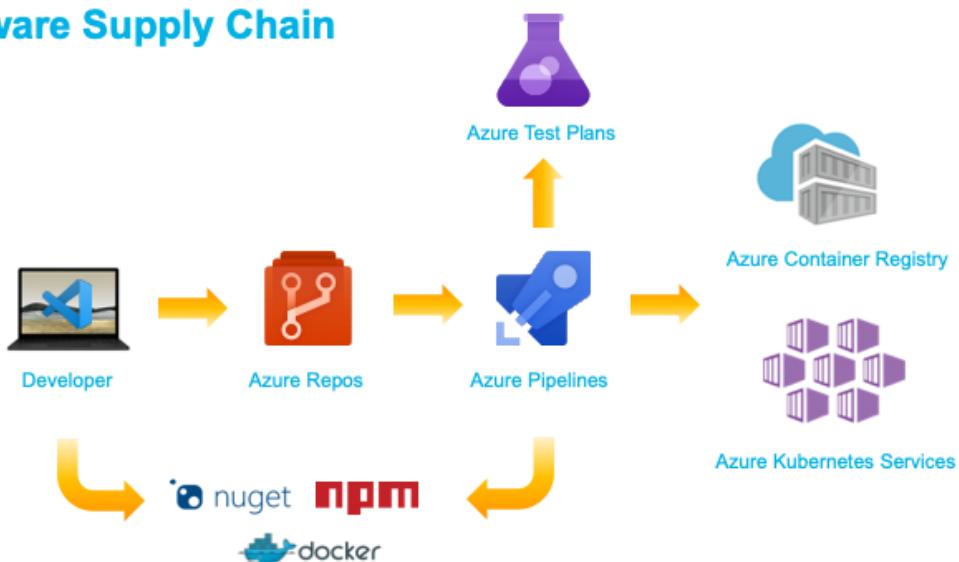
## Webmin Backdoor

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 titled "Download Webmin 1.941" containing links for RPM, Debian Package, TAR file, Solaris Package, Development Versions, and Third-Party Modules. Another sidebar titled "Webmin Links" includes 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 exploit. It mentions that 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. The text also notes that 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.

01 @nielstanis

## Software Supply Chain



O1 @nielstanis

## Reproducible/Deterministic Builds

The screenshot shows a website for "Reproducible Builds". The header features a blue bar with the title "Reproducible/Deterministic Builds". Below the header is a navigation menu with the following items: Home, Contribute, Documentation (which is currently selected), Tools, Who is involved?, News, Events, and Talks. The main content area is titled "Definitions" and contains a section titled "When is a build reproducible?". The text in this section states: "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." It also notes that the relevant attributes of the build environment, build instructions, and source code are defined by authors or distributors, and that artifacts are the parts of the build results that are the desired primary output.

OI @nielstanis

<https://reproducible-builds.org/docs/definition/>

## 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

01 @nielstanis

<https://blog.paranoidcoding.com/2016/04/05/deterministic-builds-in-roslyn.html>  
<https://github.com/dotnet/roslyn/blob/master/docs/compilers/Deterministic%20Inputs.md>  
<https://github.com/clairernovotny/DeterministicBuilds>

## Automotive Industry



O1 @nielstanis

## 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

01 @nielstanis

## 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

01 @nielstanis

<https://www.it-cisq.org/software-bill-of-materials/>

<https://cyclonedx.org/>

<https://owasp.org/www-project-dependency-check/>

## In-Toto



O1 @nielstanis

<https://in-toto.io/>

## 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

OI @nielstanis

<https://in-toto.io/>

## DataDog & In-Toto



01 @nielstanis

<https://www.datadoghq.com/blog/engineering/secure-publication-of-datadog-agent-integrations-with-tuf-and-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



OI @nielstanis

<https://grafeas.io/>

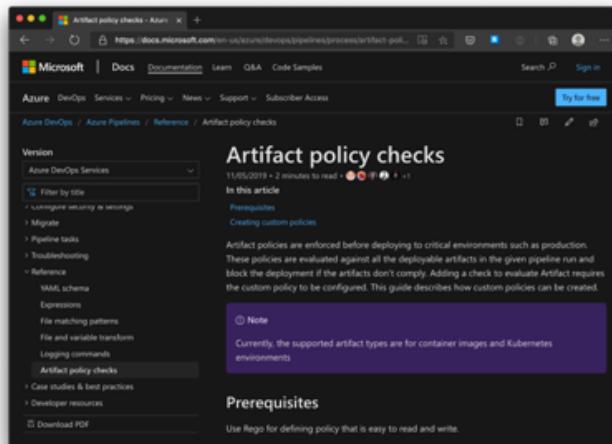
<https://github.com/grafeas/kritis/blob/master/docs/binary-authorization.md>

<https://www.infoq.com/presentations/supply-grafeas-kritis/>

<https://www.youtube.com/watch?v=hOzH3mOApjs>

<https://www.youtube.com/watch?v=05zN-YQxEAM>

## Azure Pipelines Artifact Policy



OI @nielstanis

<https://devblogs.microsoft.com/devops/secure-software-supply-chain-with-azure-pipelines-artifact-policies/>

<https://docs.microsoft.com/en-us/azure/devops/pipelines/process/artifact-policy?view=azure-devops>

## 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).

OI @nielstanis



Thanks! Questions?

**VERACODE**

You change the world, we'll secure it.

<https://github.com/nielstanis/dncse2020>

ntanis at veracode.com

@nielstanis on Twitter