Packaging in Microsoft Intune

Fully Third-Party · Managed · DevOps — Security · Control · Speed · Effort.
Pick three.

onedeploy

Windows Deployment Reimagined

Insight

control up

robopack

numecent

Microsoft

eido

Rimo3

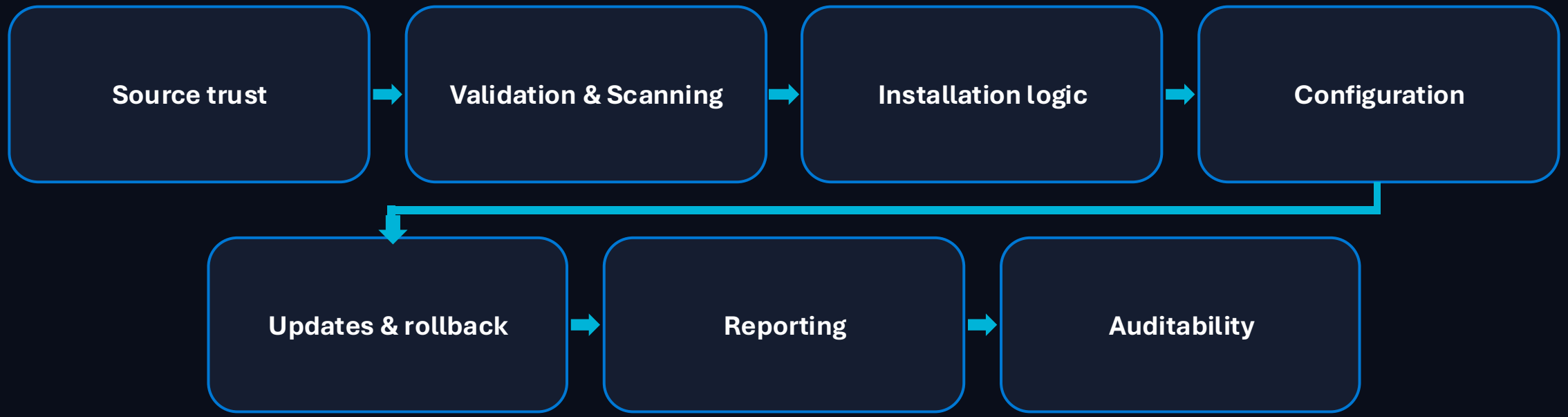
Raise your hands

WHO PACKAGES APPLICATIONS?

WHO MAINTAINS THEM?

WHO GETS CALLED WHEN SOMETHING BREAKS?

Packaging is a Process, not a file



Why this topic, why now

Increased automation

Larger attack surface

Higher compliance pressure

Niels Kok

- Sysadmin/Consultant -> DevOps Engineer
- Microsoft MVP – Intune + Azure Virtual Desktop
- Freelance

BRAINPULSE IT
CONSULTANCY & TRAINING

Stefan Dingemanse

Microsoft Cloud Consultant & Architect

Microsoft MVP · AVD / Windows 365 · MCT

Dordrecht, Netherlands · Brainpulse IT

Co-Founder AVD Community & Dutch VDI User Group

Hybrid Athlete - Running & Weightlifting

What we’ll cover

01 The real problems behind "packaging"

02 How we got here

03 Common solutions

04 Trade-offs, not opinions

05 The North Star — and the four pillars

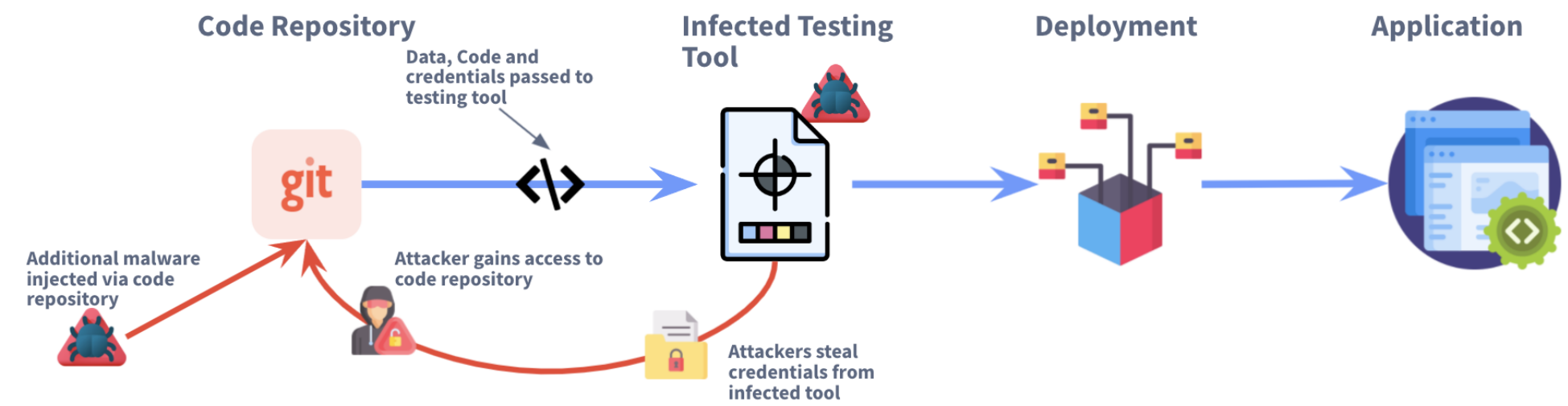
06 Your trade-off: which pillar can you defend losing?

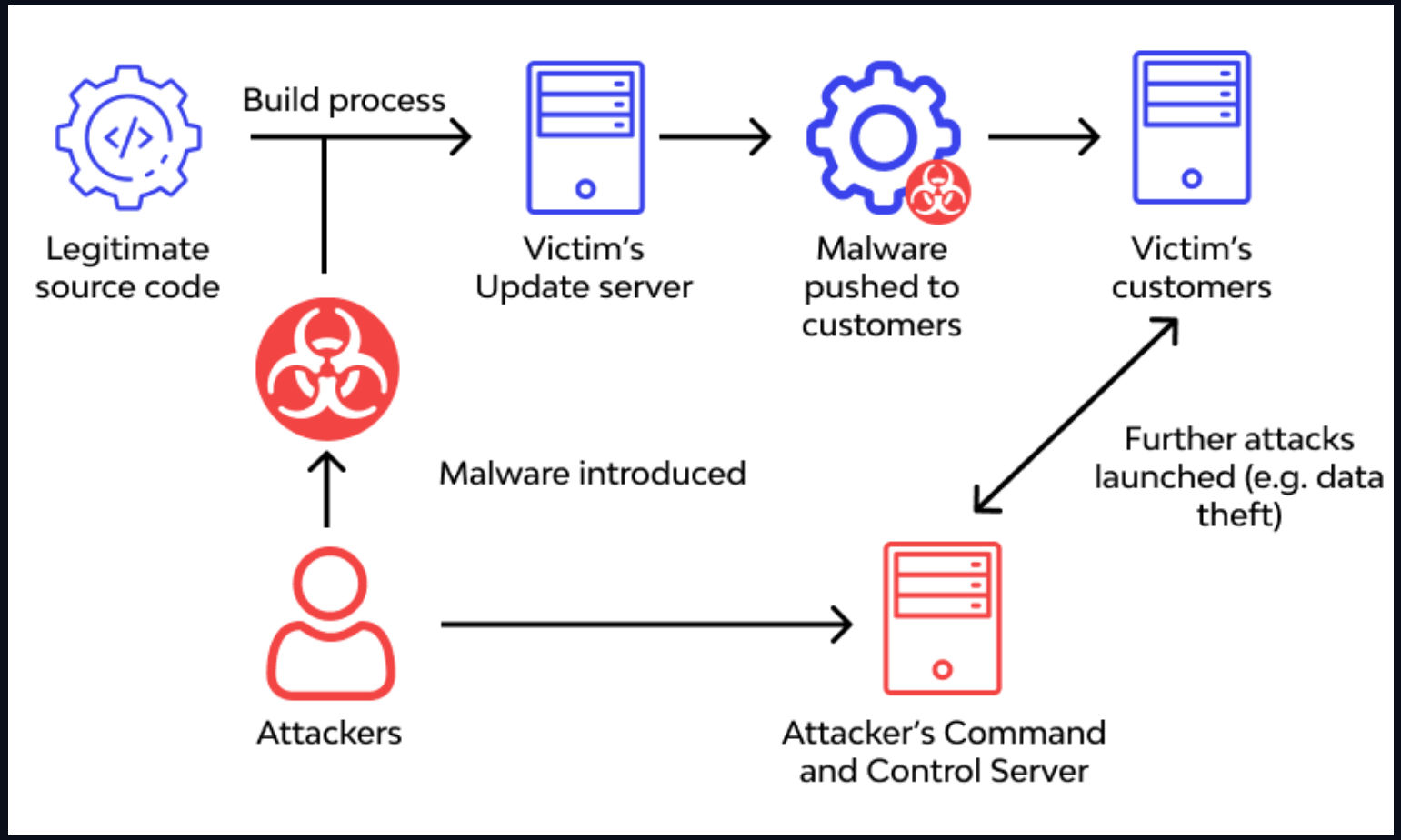

DEPENDENCY ISSUES
BUILD FAILURES
PROBLEMS
WITH APPLICATION PACKAGING
BIG FILE SIZE
ENVIRONMENT ERRORS
SLOW PERFORMANCE
BROKEN THERE

Problem / issues

- Supply chain attacks (Shai-Hulud 2.0) - **Security**
- Public/Private repositories - **Security** - **Control**
- Chocolatey / Winget Deployment flow - **Control** · **Effort**
- Winget Publishing Process - **Security** · **Control**
- Custom – Windows – Legacy - **Control** · **Effort**
- Time consuming / Costs - **Effort**
- Consistency Packages (PSADT / MSIs/ etc) - **Control** · **Effort**
- Poor Reporting - **Security** · **Control**

Attack through deployment and testing tools

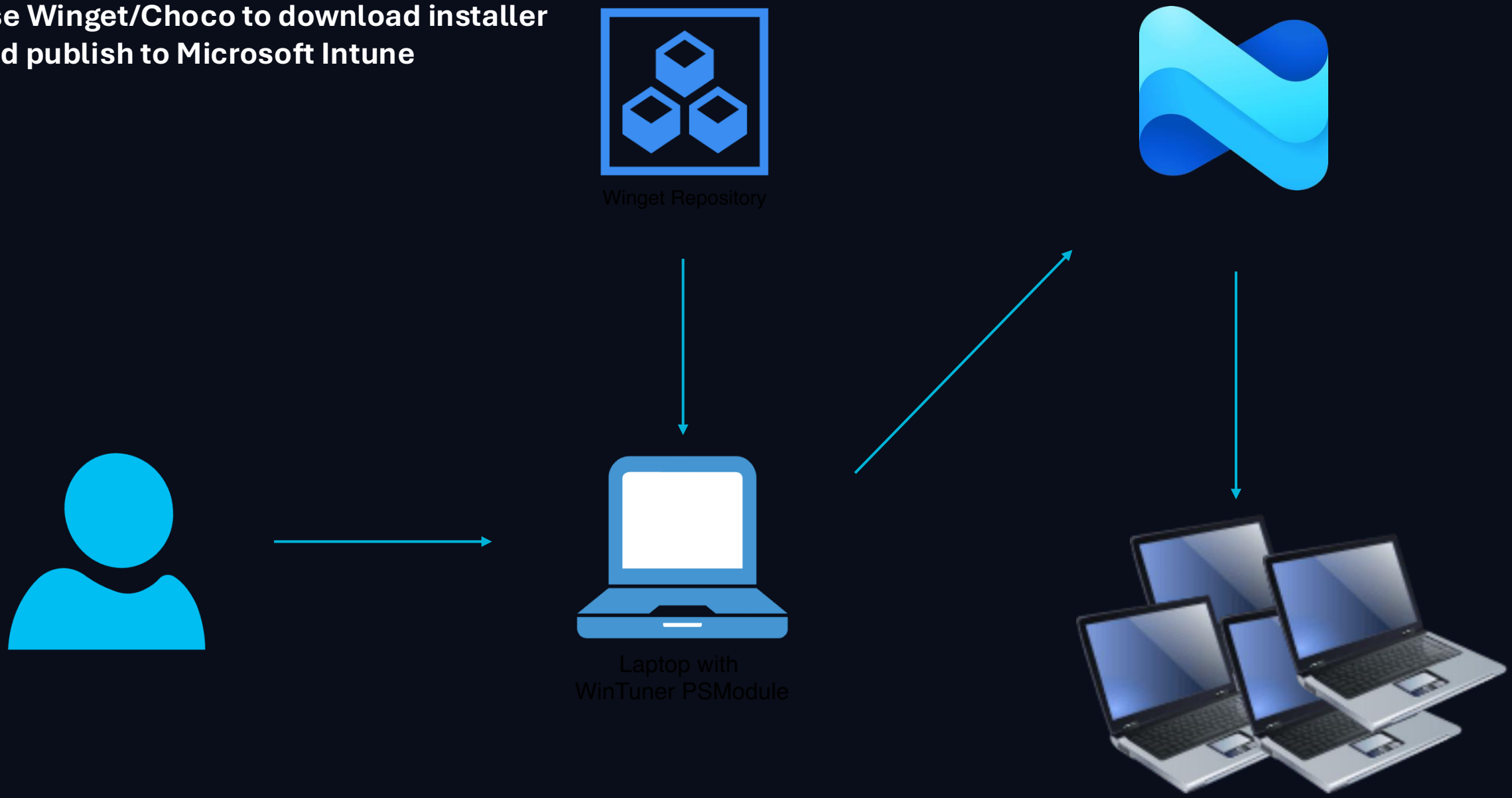



Legitimate source code
Build process
Victim's Update server
Malware pushed to customers
Victim's customers
Malware introduced
Attackers
Attacker's Command and Control Server
Further attacks launched (e.g. data theft)

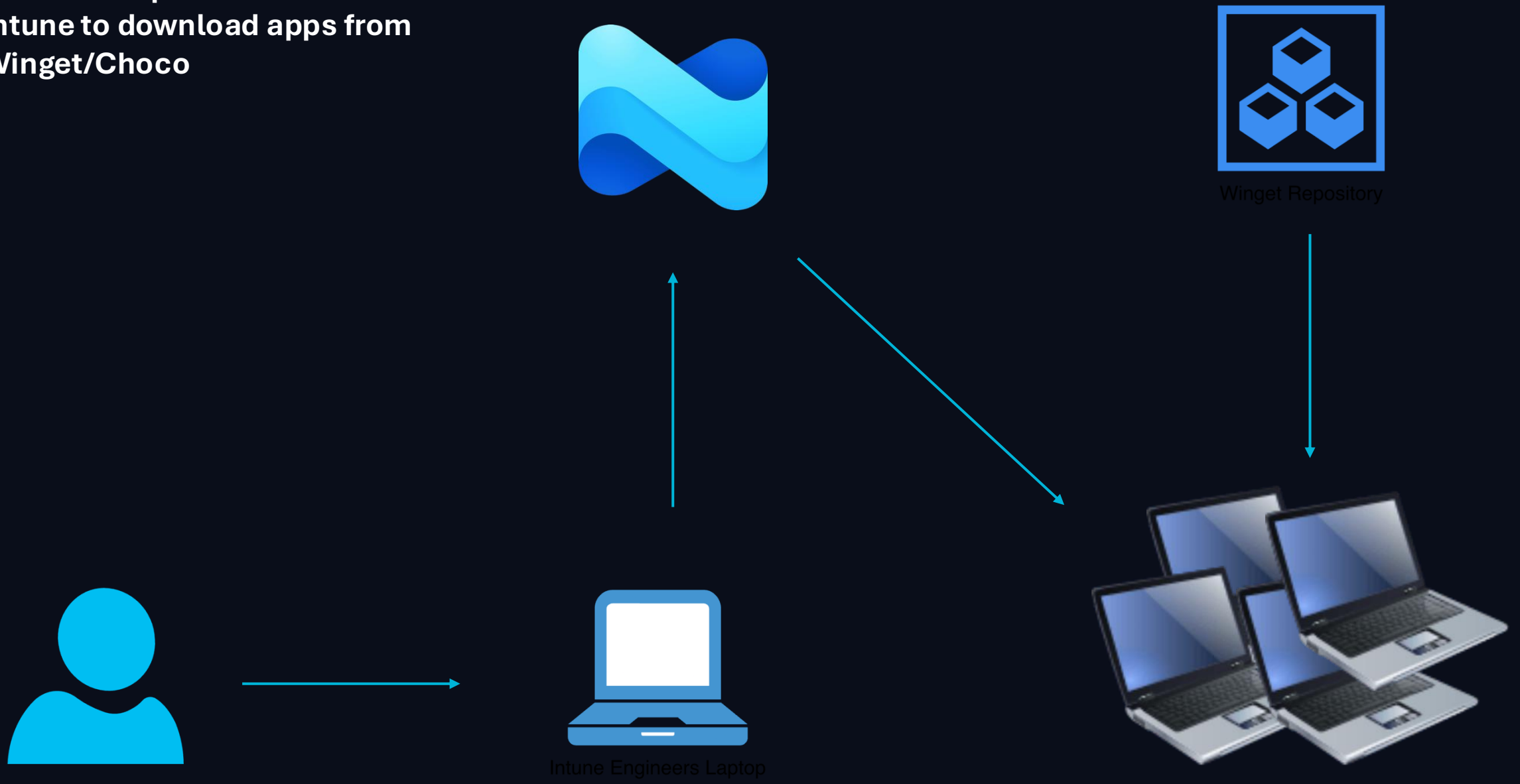
Public vs Private Repositories: Who Do You Trust With Your Binaries?

Public Repository	Private Repository
Community or vendor controlled	Organisation controlled
Fast, wide coverage	Slower onboarding
Low effort	Higher effort
Shared responsibility	Full accountability
Limited auditability	Audit-ready

Use Winget/Choco to download installer and publish to Microsoft Intune



Publish script as installer in Microsoft Intune to download apps from Winget/Choco



Update all Winget/Choco Applications

Winget Repository

Run Remediation/Win32App to run:
“Winget/Choco Upgrade all”

Files

master

Go to file t

.config
.github
.vscode
DevOpsPipelineDefinitions
Tools
doc
fonts
manifests
schemas
.editorconfig
.gitattributes
.gitignore
CODE_OF_CONDUCT.md
CONTRIBUTING.md
LICENSE
README.md
SECURITY.md
SUPPORT.md

Trenly Fix Link for Authoring.md (#145849) 88c58f9 · 2 years ago History

Preview Code Blame 188 lines (116 loc) · 11.5 KB · Raw

Windows Package Manager Community Repository Contributor's Guide

Below is our guidance for how to report issues, propose new features, and submit contributions via Pull Requests (PRs).

Open Development Workflow

The Windows Package Manager team is VERY active in this GitHub Repository. In fact, we live in it all day long and carry out all our development in the open!

When the team finds issues we file them in the repository. When we propose new ideas or think-up new features, we file new feature requests. When we work on fixes or features, we create branches and work on those improvements. And when PRs are reviewed, we review in public - including all the good, the bad, and the ugly parts.

The point of doing all this work in public is to ensure that we are holding ourselves to a high degree of transparency, and so that the community sees that we apply the same processes and hold ourselves to the same quality-bar as we do to community-submitted issues and PRs. We also want to make sure that we expose our team culture and "tribal knowledge" that is inherent in any closely-knit team, which often contains considerable value to those new to the project who are trying to figure out "why the heck does this thing look/work like this???"

Repository Bot

Windows 10's package manager flooded with duplicate, malformed apps

By Ax Sharma

June 1, 2021 11:15 AM 4

Last week, Microsoft released the first stable version of its Windows 10 package manager, Winget, which enables users to manage apps via command-line.

Much like package managers available on other platforms, Winget lets Windows users automate app management when it comes to installing, configuring, upgrading, and uninstalling applications.

But, over the weekend, multiple users flooded Winget's software registry with pull requests for apps that are either duplicate or malformed, thereby raising concerns about the integrity of the Winget ecosystem.

Exploits & Vulnerabilities

ZDI-23-1527 and ZDI-23-1528: The Potential Impact of Overly Permissive SAS Tokens on PC Manager Supply Chains

In ZDI-23-1527 and ZDI-23-1528 we uncover two possible scenarios where attackers could have compromised the Microsoft PC Manager supply chain.

By: Nitesh Surana
Apr 15, 2025
Read time: 12 min (3247 words)

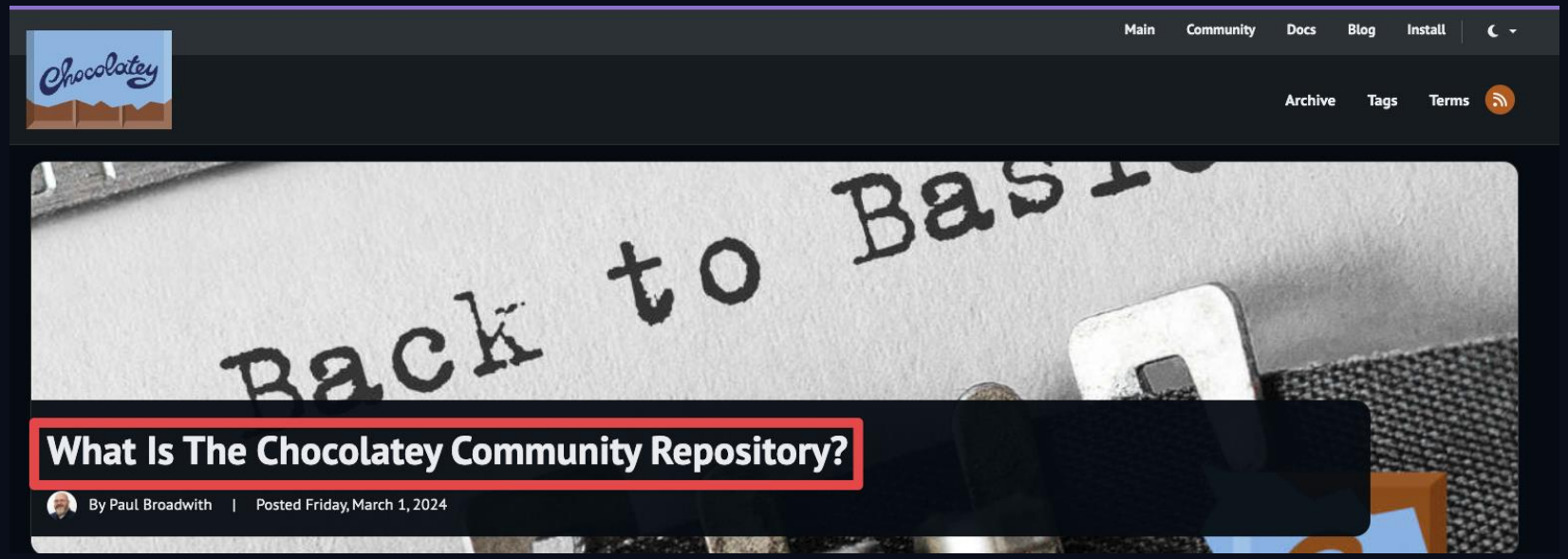
Authors

Nitesh Surana
Senior Threat Researcher

CONTACT US

Summary:

- This blog explores two possible scenarios where PC Manager releases could have been hijacked by attackers via WinGet repository, 'aka.ms' URLs, and an official subdomain of Microsoft, due to overly permissive SAS tokens.
- If an attack had been carried out, cybercriminals could have compromised software supply chains for distribution of malware, allowed them to replace software releases, and alter distributed PC Manager executables.
- These issues detailed in this blog have been reported to Microsoft and have since been resolved. This blog details recommendations on how to avoid similar potential compromises.
- This blog entry documents the outcome of security research conducted in 2023. Microsoft has since provided additional guidance and updates to the rules of engagement of the Azure Bounty Program. This blog is for educational purposes only.



Who Creates and Maintains Packages?

You. Me. People like us. Members of the Chocolatey Communi. Anybody can create and maintain a package.

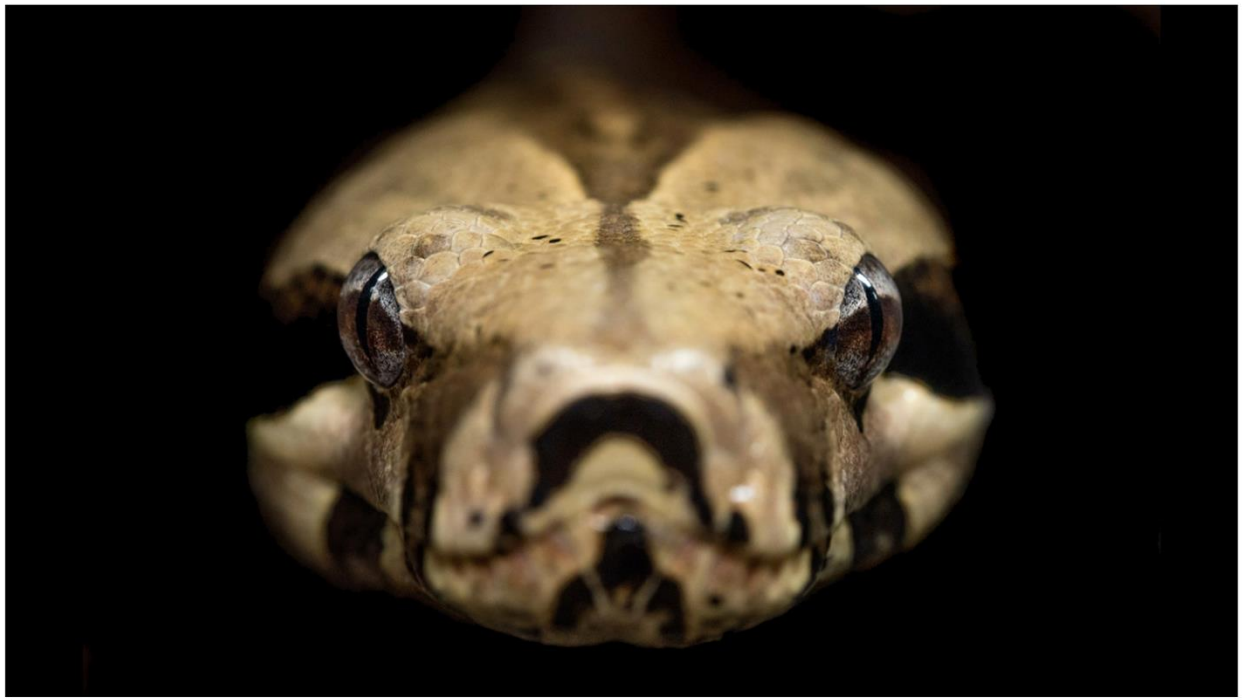
A package maintainer ensures that a package is up-to-date, works as it should and commits to continue that maintenance. There is process for packages that are out of date or no longer being maintained.

Chocolatey Package Submission

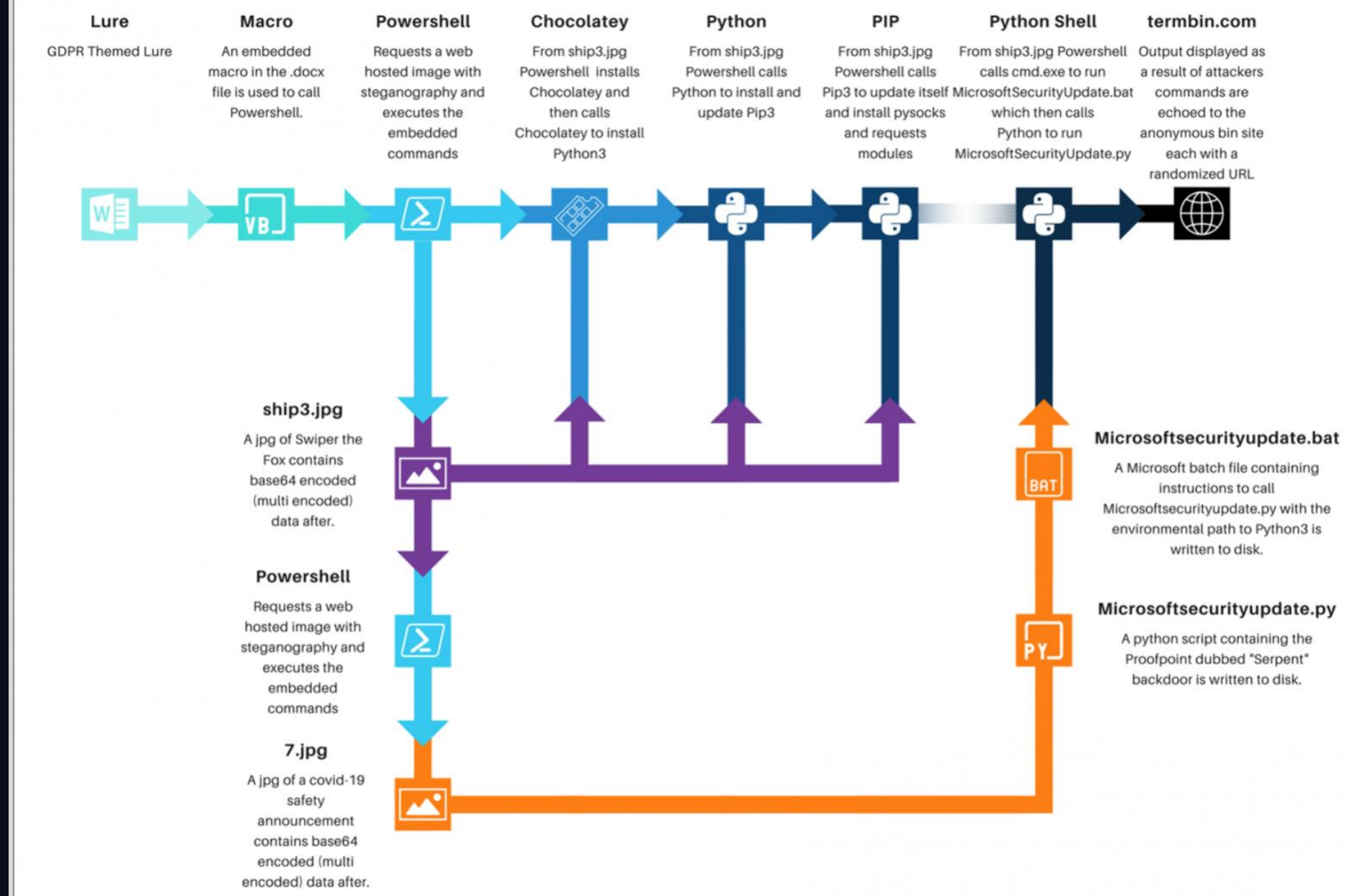
Serpent malware campaign abuses Chocolatey Windows package manager

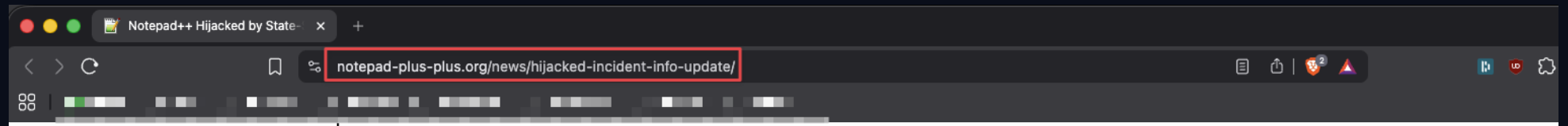
By Bill Toulas

March 21, 2022 01:10 PM 3



Threat actors are abusing the popular Chocolatey Windows package manager in a new phishing campaign to install new 'Serpent' backdoor malware on systems of French government agencies and large construction firms.


Lure
GDPR Themed Lure
Macro
An embedded macro in the .docx file is used to call Powershell.
Powershell
Requests a web hosted image with steganography and executes the embedded commands
Chocolatey
From ship3.jpg Powershell installs Chocolatey and then calls Chocolatey to install Python3
Python
From ship3.jpg Powershell calls Python to install and update Pip3
PIP
From ship3.jpg Powershell calls Pip3 to update itself and install pysocks and requests modules
Python Shell
From ship3.jpg Powershell calls cmd.exe to run MicrosoftSecurityUpdate.bat which then calls Python to run MicrosoftSecurityUpdate.py
termbin.com
Output displayed as a result of attackers commands are echoed to the anonymous bin site each with a randomized URL
ship3.jpg
A jpg of Swiper the Fox contains base64 encoded (multi encoded) data after.
Powershell
Requests a web hosted image with steganography and executes the embedded commands
7.jpg
A jpg of a covid-19 safety announcement contains base64 encoded (multi encoded) data after.
Microsoftsecurityupdate.bat
A Microsoft batch file containing instructions to call Microsoftsecurityupdate.py with the environmental path to Python3 is written to disk.
Microsoftsecurityupdate.py
A python script containing the Proofpoint dubbed "Serpent" backdoor is written to disk.



Notepad++ Hijacked by State-Sponsored Hackers

2026-02-02

Following the security disclosure published in the v8.8.9 announcement
https://notepad-plus-plus.org/news/v889-released/
the investigation has continued in collaboration with external experts and with the full involvement of my (now former) shared hosting provider.

According to the analysis provided by the security experts, the attack involved infrastructure-level compromise that allowed malicious actors to intercept and redirect update traffic destined for notepad-plus-plus.org. The exact technical mechanism remains under investigation, though the compromise occurred at the hosting provider level rather than through vulnerabilities in Notepad++ code itself. Traffic from certain targeted users was selectively redirected to attacker-controlled malicious update manifests.

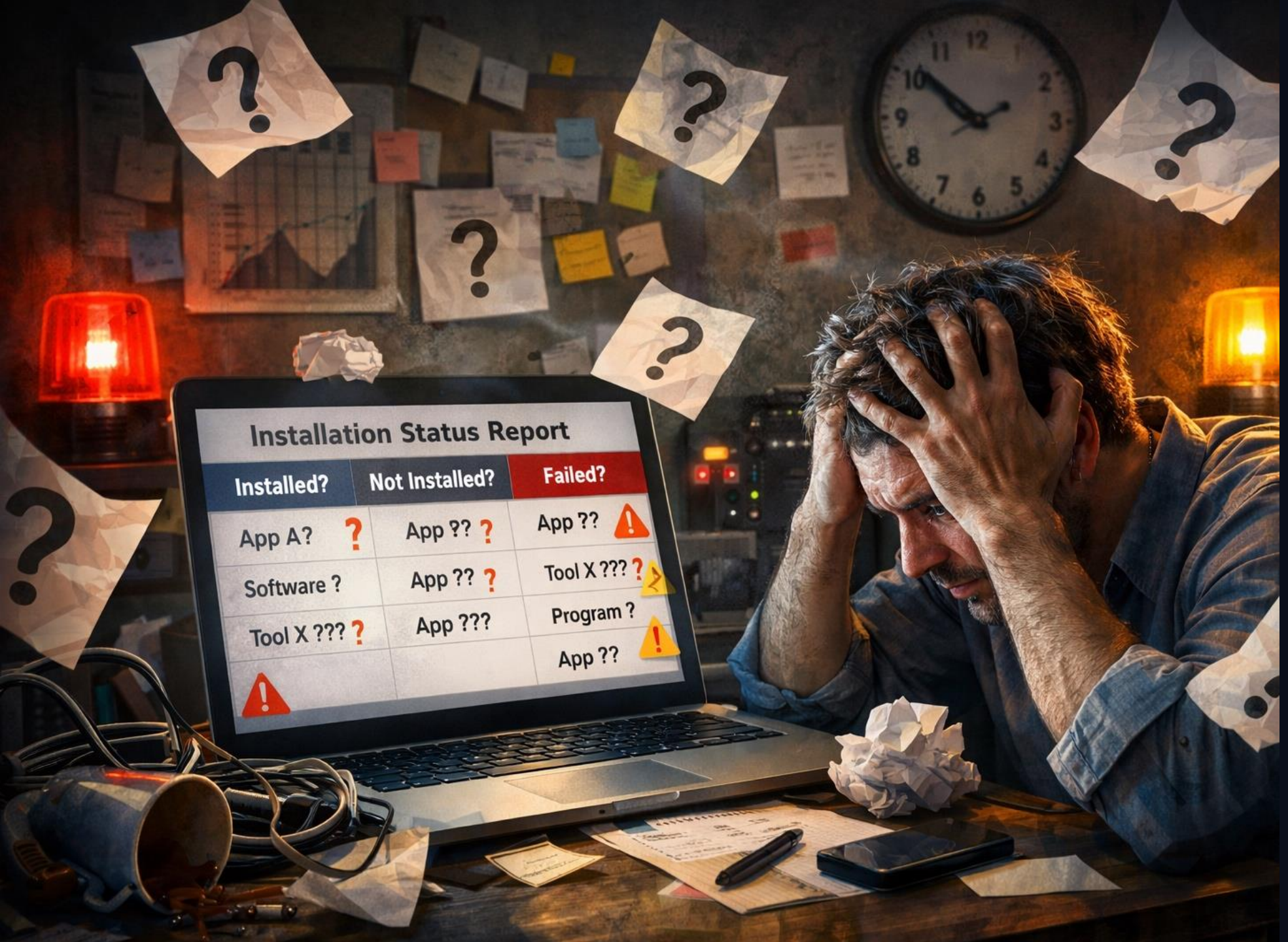
The incident began in June 2025. Multiple independent security researchers have assessed that the threat actor is likely a Chinese state-sponsored group, which would explain the highly selective targeting observed during the campaign.

An incident-response (IR) plan was proposed by the security expert, and I facilitated direct communication between the hosting provider and the IR team. After the IR team engaged with the provider and reviewed the situation, I received the following detailed statement from the provider:

Custom & legacy apps

Where catalogs stop helping

- MST transforms
- Custom configs
- Line-of-business apps



Reporting

- If answering this requires multiple tools or manual checks, you don’t have control, you have hope.


Intune
Apps
Win32 apps
App packages
SOLUTIONS
Win32
Win32
MSI
MSIX
MSIX
Appx

The North Star of Application Packaging for Intune

Solution Categories

CLASSIC APPROACH

Manual

Download, package, upload, assign manually. Works at small scale but hits a wall fast.

MANAGED TOOLING

Third-Party

Vendor platforms automate packaging, scanning, and delivery. Fastest path to scale.

DIY

Self-Hosted DevOps

Own the full pipeline in code. Maximum control, highest upfront effort.

Pillars

Security

- Where binaries come from
- How they're scanned and validated
- Who owns the risk when something goes wrong

Control

- Who decides what gets installed and when
- How updates are approved, delayed, or rolled back
- How custom logic and LOB apps are handled

Speed

- How fast apps reach production
- How quickly updates and patches flow
- How much manual work is removed

Effort

- Engineering time required
- Operational overhead
- Complexity of maintenance and governance

Manual deployment

Classic approach

- Download
- Package
- Upload
- Assign
- Fix issues manually

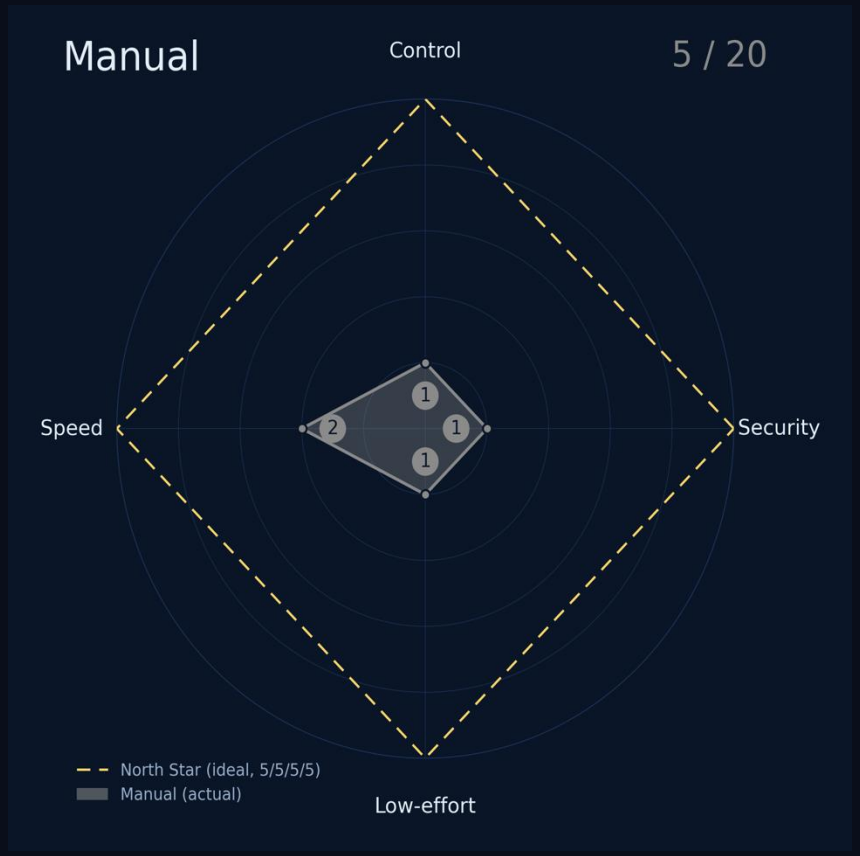
Score card – Manual Deployment

Problem	How Manual helps	What remains	★
Supply chain attacks	You manually choose sources	No systematic scanning	★
Public/private repositories	Direct downloads	No central control	★
Deployment flow	Flexible & simple	Not repeatable	★★
Custom & legacy apps	Works for all apps	Very time consuming	★★★★
Time & cost	No tooling cost	High labour cost	★
Consistency	Engineer dependent	Not scalable	★
Reporting & auditability	Best effort	Incomplete	★

Total score: 11/35

Manual Packaging: Solves Speed, Not Scale

Manual — Speed only. Everything else loses.



The shape fits one use case: the building is on fire, and you need something installed tonight.

Do it yourself

Fully DevOps

Defender running on this machine

Checks for:
Hashes
Malicious code
Etc.

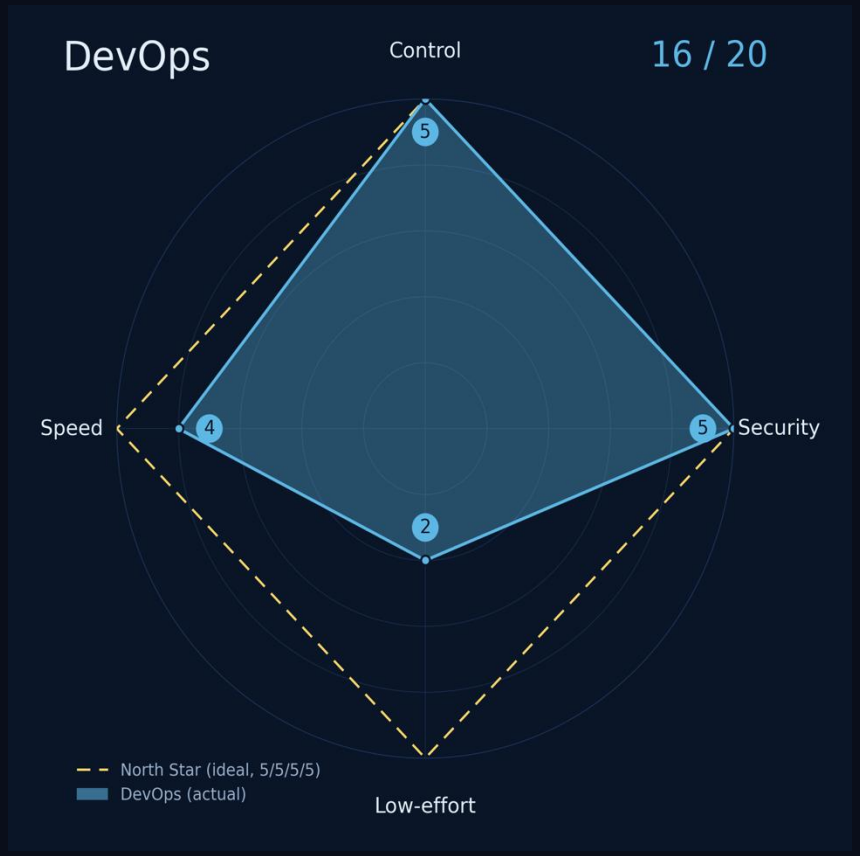
Score card - DevOps

Problem	How DevOps helps	What remains	★
Supply chain attacks	Mirror, scan, validate	You own the work	★★★★★
Public/private repositories	Internal artifact feed	Needs maintenance	★★★
Deployment flow	Fully controlled pipeline	Engineering effort	★★★★
Custom & legacy apps	First-class citizens	Requires standards	★★★★★
Time & cost	Lower long-term cost	Higher upfront cost	★
Consistency	Pipeline enforced	Discipline required	★★★★★
Reporting & auditability	End-to-end audit trail	Build effort	★★★

Total score: 26/35

DevOps: Solves Control, Auditability, and Ownership, but costs Effort

DevOps - Security, Control, Speed at scale. Pays in Effort.



Maximum Security and Control at scale. The cost lives in Effort.

Third-Party

Third-party tooling

Third Party

What these tools typically solve

- Ready-made packages at scale
- Automated updates and patching
- Standardised install logic
- Less scripting knowledge required
- Faster time to production
- Lower operational effort

robopack

LIQUIT | RECAST SOFTWARE

COMPARISON

Are third-party tools the same?

No, key differences across vendors

Security scanning

Depth of vulnerability scanning and source verification varies between vendors.

Custom support

Some vendors handle custom and legacy apps; others stick to catalogs only.

Update strategy

Patching cadence, ring rollouts, and auto-update behaviour are not standardised.

Licensing

Per-device, per-user, or flat-fee, feature tiers and contract terms vary widely.

Who curates source

Specialised vendor (PMPC, Action1, ManageEngine), Microsoft EAM (in Intune), or OEM/in-house catalogs (Lenovo, HP).

Different vendors. Same trade-off: someone else controls what gets in.

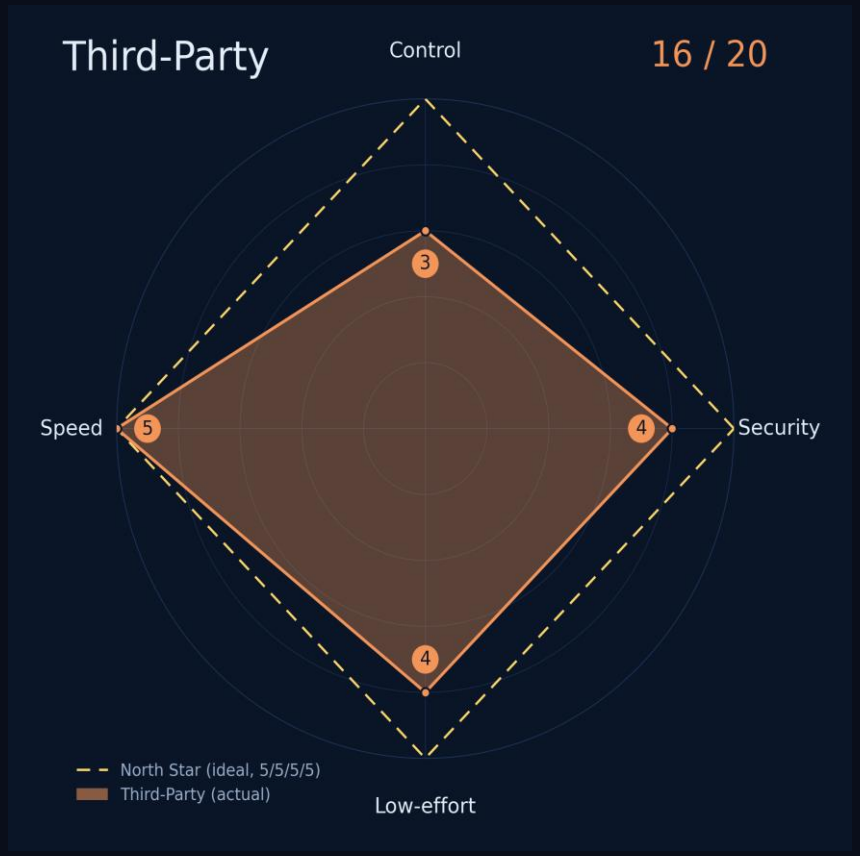
Score card – Third-Party

Problem	How Third-Party helps	What remains	★
Supply chain attacks	Vendor scanning & curation	Vendor's upstream still hits you	★★★
Public/private repositories	Managed catalogs	Vendor dependency	★★★
Deployment flow	Standardised & automated	Third-Party model limits	★★★★
Custom & legacy apps	Supported	Still special cases	★★★★
Time & cost	Massive time savings	License cost	★★★★
Consistency	Enforced packages	—	★★★★★
Reporting & auditability	Vendor-provided reports	Limited end-to-end	★★★★

Total score: 23/35

Third-Party Tools: Solves Speed and Consistency. Security depends on the vendor you trust.

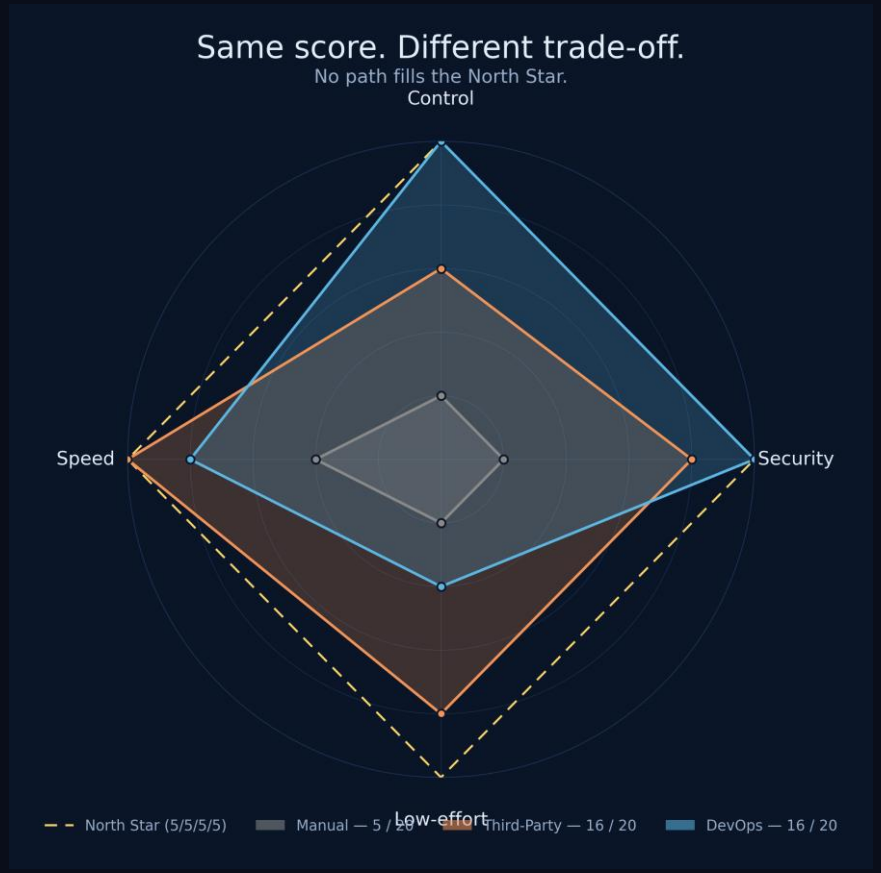
Third-Party — Speed + Effort solved. Security moves to the vendor.



Source ownership transfers — to the vendor, or to Microsoft (EAM). Same trade-off, different vendor.

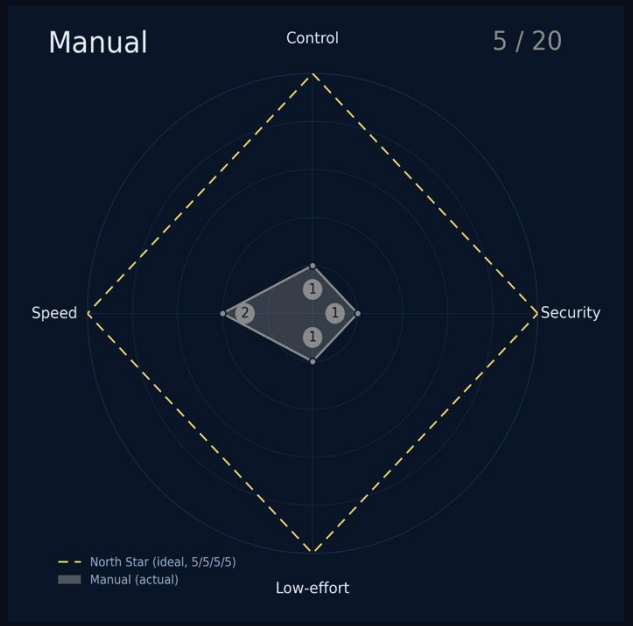
Same score. Different trade-off.

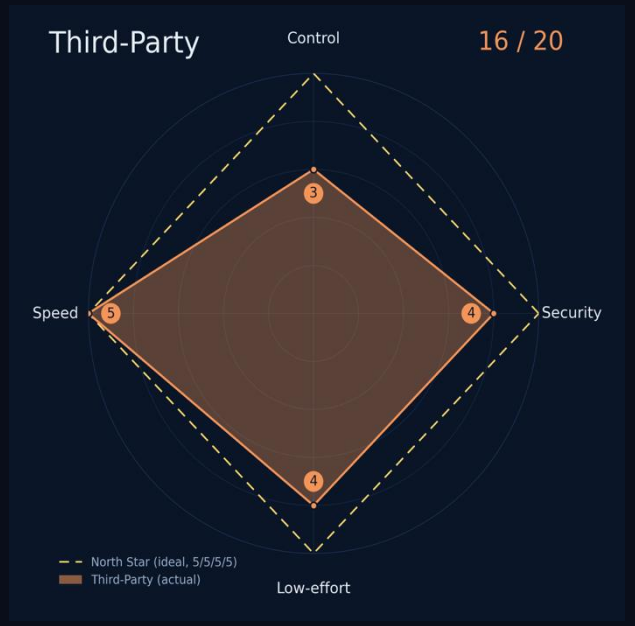
No path fills the North Star. Choose the shortfall you can defend.

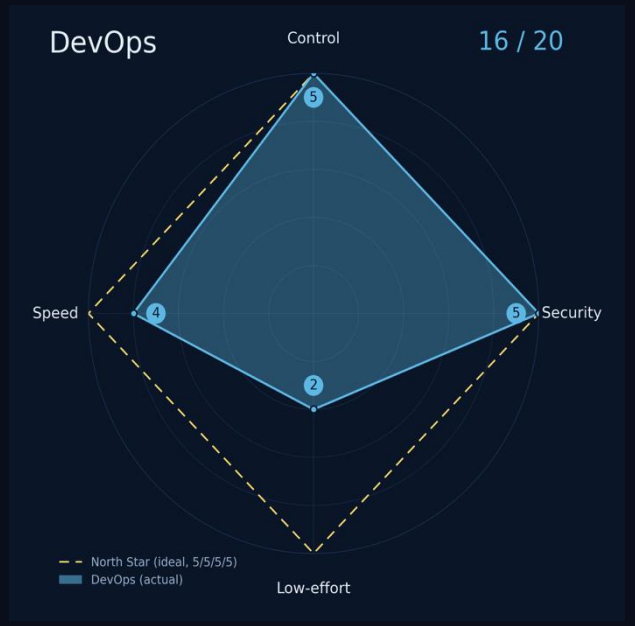


Manual is out of the race. Third-Party and DevOps tie at 16/20 — on different pillars.

Three shapes. Your choice.







Manual

Emergency only

Third-Party

Scale by default

DevOps

Control by design

Takeaways

Key Takeaways

01 Packaging is a supply-chain decision

If you don't control the source, you don't control the risk.

02 There is no "best" model

Security, Control, Speed, and Effort can't be maxed at the same time.

03 Third-party tools buy speed, not ownership

Reduced effort, more consistency, but responsibility stays with you.

04 DevOps is about control, not complexity

But you'll have to build it yourself.

05 Manual packaging solves urgency, not scale

It works, until it becomes your bottleneck and your risk.

06 Auditability is the real end goal

If you can't answer *what, where, and why*, you have hope, not control.

What matters most

01 No universal best solution

Every organisation must evaluate based on scale, risk, and capabilities

02 Security & traceability are mandatory

Know what is installed, where it came from, and why

03 Own your trade-offs

Security, Control, Speed, and Effort can't all be optimised at once.

Closing

Questions. Discussion. War stories.