SymfonyLive Berlin 2023

Bekomm die Supply Chain deiner Projekte in den Griff

Nils Adermann
@naderman

Private Packagist
https://packagist.com

Supply Chain?
Lieferkette?


18
MSC HOME TERMINAL
17
MSC HOME TERMINAL
16
15
14
13
MSC TRIESTE


67
65
63
61

https://commons.wikimedia.org/wiki/File:Geely_assembly_line_in_Beilun,_Ningbo.JPG

Supply Chain

A supply chain is a complex logistics system that consists of facilities that convert raw materials into finished products which are later distributed to end consumers or end customers.

https://en.wikipedia.org/wiki/Supply_chain

Eine Lieferkette ist ein komplexes logistisches System, das aus Einrichtungen und Abläufen besteht, die Rohstoffe in fertige Produkte wandeln, die an Endkunden oder Konsumenten verteilt werden.

Supply Chain - Aber für Software?!

Rohstoffe	Source code
Raffinieren, verarbeiten, konstruieren	Build Prozess
Produktkomponenten	Abhängigkeiten, Hardware, Netzwerk
Montage, Logistik	Paketverwaltung
Qualitätssicherung	QA / Tests / CI Service
Lieferung/Fulfillment	Deployment Prozess

Eher ein hinkender Vergleich, nicht so wörtlich nehmen

Software Supply Chain

A software supply chain is composed of the components, libraries, tools, and processes used to develop, build, and publish a software artifact.

https://en.wikipedia.org/wiki/Software_supply_chain

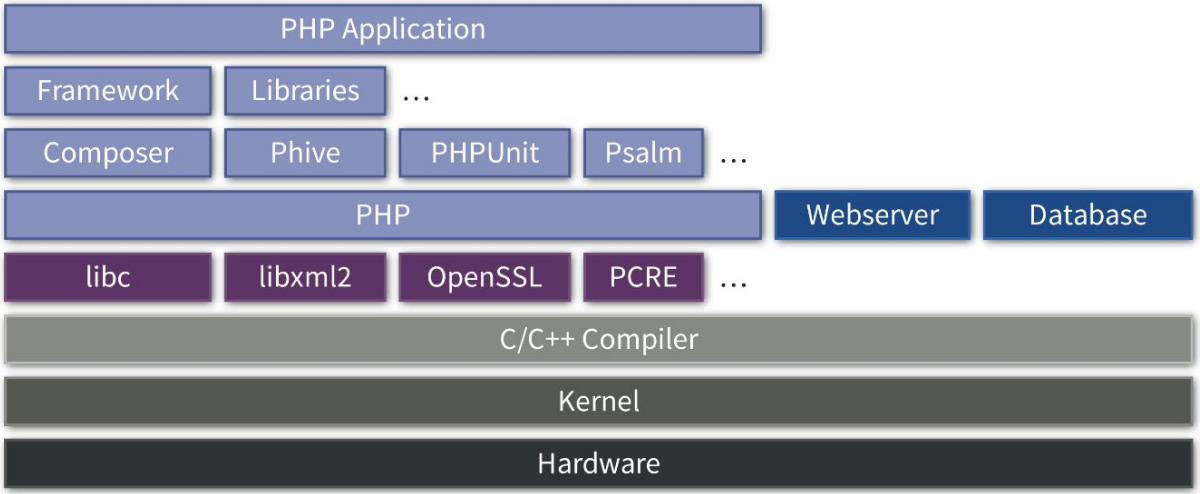
Eine Softwarelieferkette besteht aus den Komponenten, Bibliotheken, Werkzeugen und Prozessen mit denen ein Softwareartefakt entwickelt, gebaut und publiziert wird.

Software Supply Chain

In anderen Worten:

Der “Full-Stack” und alle Prozesse und Werkzeuge die darin involviert sind ihn zu schaffen und zusammenzusetzen.

Full-Stack


PHP Application
Framework
Libraries
…
Composer
Phive
PHPUnit
Psalm
…
PHP
Webserver
Database
libc
libxml2
OpenSSL
PCRE
…
C/C++ Compiler
Kernel
Hardware

Hardware
Webbrowser
Framework
Libraries
...
PHP Application
Framework
Libraries
...
Composer
Phive
PHPUnit
Psalm
...
PHP
Webserver
Database
libc
libxml2
OpenSSL
PCRE
...
C/C++ Compiler
Kernel
Hardware

Warum sollte euch das interessieren?

- Business Continuity
  - Was tust du wenn dein Rechenzentrum in Flammen steht?
  - Was passiert wenn deine CI Platform den Betrieb einstellt?
  - Was planst du wenn eine Abhängigkeit nicht mehr maintained wird?
  - Was machst du wenn eine Abhängigkeit gelöscht wird?
- Sicherheit
  - Supply Chain Attacks

Supply Chain Attacks

- Heartbleed - https://heartbleed.com/ - 2014
  - Der Heartbleed-Bug ist ein schwerwiegender Programmfehler in älteren Versionen der Open-Source-Bibliothek OpenSSL, durch den über verschlüsselte TLS-Verbindungen private Daten von Clients und Servern ausgelesen werden können. Ein großer Teil der Online-Dienste, darunter auch namhafte Websites wie auch VoIP-Telefone, Router und Netzwerkdrucker waren dadurch für Angriffe anfällig.

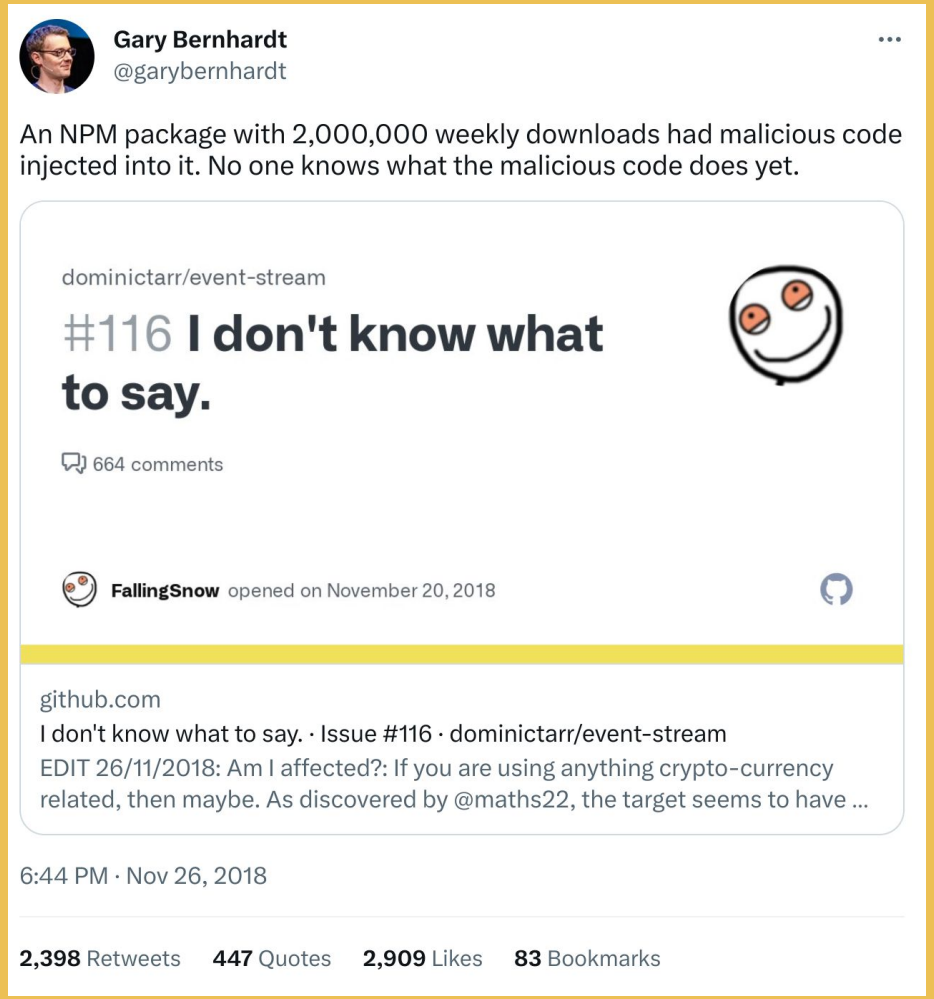
Supply Chain Attacks

- Stuxnet
  - entdeckt 2010, existiert vermutlich schon seit 2005
  - Kombination von 4 zero-days, Windows, Siemens Step7, Infektion durch USB Sticks
  - rootkit das auf PLCs (programmable logic controllers) zielt
  - vermutlich von USA und Israel gebaut um das iranische Atomprogramm zu stören
- SolarWinds Orion / 2020 United States federal government data breach
  - Angreifer hatten Zugang zu Buildsystem, vermutlich durch Office 365 Account
  - modifizierte Software Updates mit Remote Zugriff zu jeder Maschine die Orion installiert
  - 18,000 Kunden inklusive großer Teile der US Regierung und anderer Regierungen
  - vermutlich russische Hacker
  - Im Dezember 2020 entdeckt, genutzt seit September 2019

Supply Chain Attacks

- Log4Shell
  - https://en.wikipedia.org/wiki/Log4Shell
  - Log4j Vulnerability, Standard Java Logging Bibliothek
  - existierte von 2013 - 24. November, 2021
  - Arbitrary code execution, sehr weit verbreitet, CVSS Score 10/10

Eine Dependency wurde ohne böse Absicht an eine Person mit schlechten Absichten übertragen



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

PRIVATE PACKAGIST

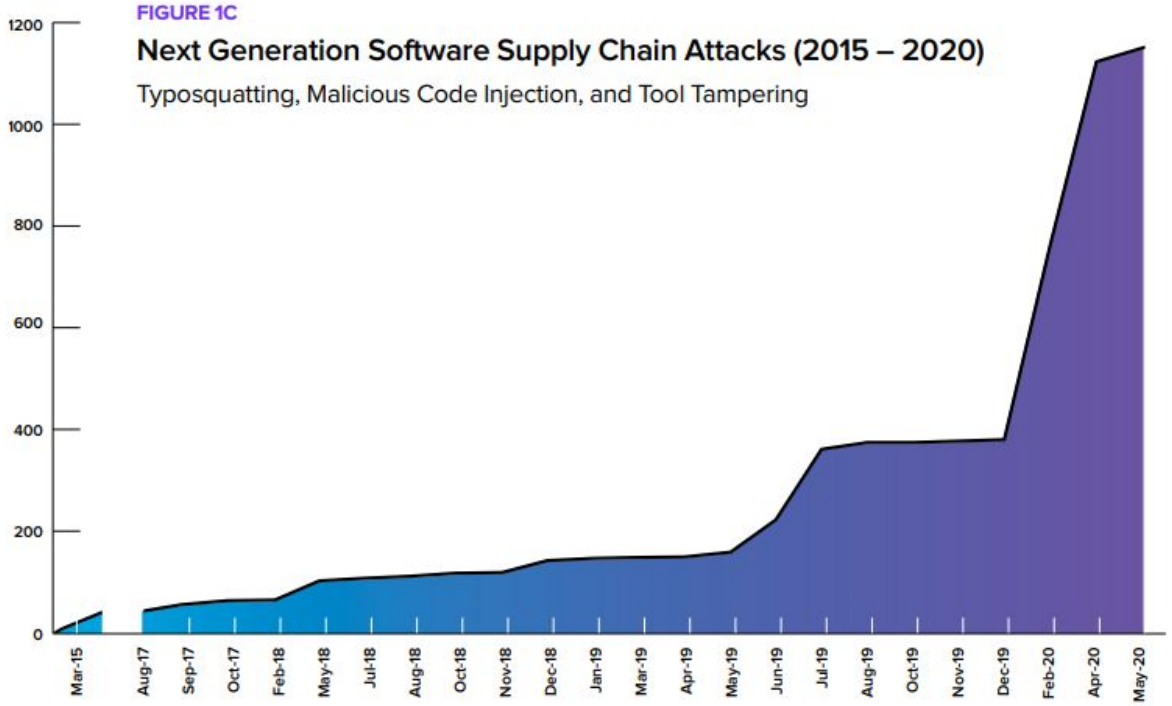
Supply Chain Attacks

- Depublikation von left-pad
  - https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code
- PyPi Typosquatting mit bösartigem Code
  - https://blog.phylum.io/phylum-discovers-revived-crypto-wallet-address-replacement-attack/
- Public Travis CI Logs (Still) Expose Users to Cyber Attacks
  - https://blog.aquasec.com/travis-ci-security
- Bösartige commits in php-src mit den Namen von Rasmus Lerdorf und Nikita Popov
  - https://news-web.php.net/php.internals/113838

Andere Supply Chain Probleme

- Jira: Atlassian customers frustrated by weeks-long outage, lack of communication from company
  - https://www.techrepublic.com/article/atlassian-customers-frustrated-by-weeks-long-outage-lack-of-communication-from-company/
- Following theft of GitHub OAuth tokens from Heroku, GitHub resets tokens but Salesforce takes weeks to reset passwords and restore functionality
  - https://www.zdnet.com/article/heroku-to-begin-user-password-reset-almost-a-month-after-github-oauth-token-theft/

Supply Chain Attacks

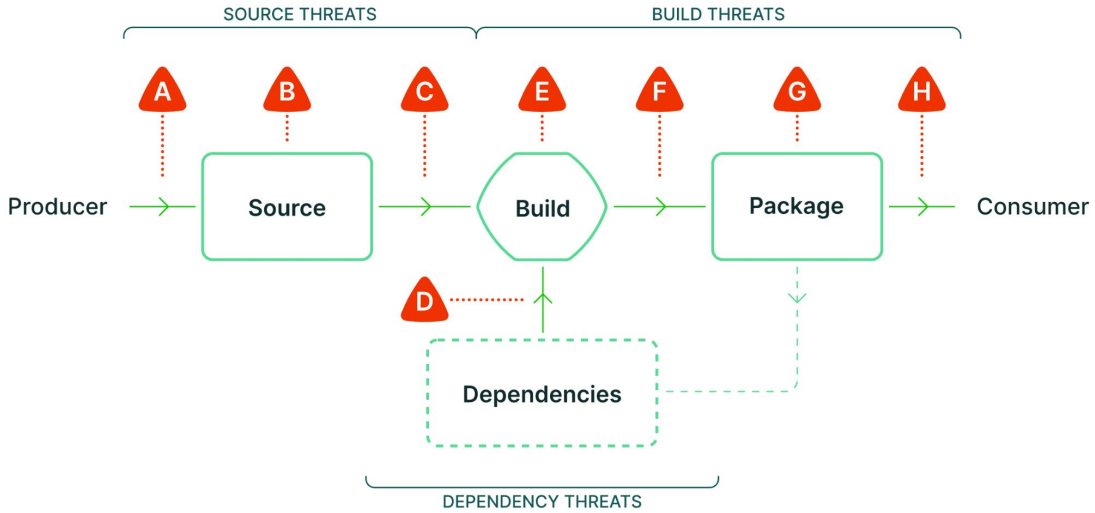


“2020 State of the Software Supply Chain” by sonatype
https://www.sonatype.com/hubfs/Corporate/Software%20Supply%20Chain/2020/SON_SSSC-Report-2020_final_aug11.pdf#page=7

PRIVATE PACKAGIST

Supply Chain Attacks

2021 Google stellt SLSA vor: “Supply-chain Levels for Software Artifacts” - https://slsa.dev/



SOURCE THREATS

- A Submit unauthorized change
- B Compromise source repo
- C Build from modified source

DEPENDENCY THREATS

- D Use compromised dependency

BUILD THREATS

- E Compromise build process
- F Upload modified package
- G Compromise package repo
- H Use compromised package

PRIVATE PACKAGIST

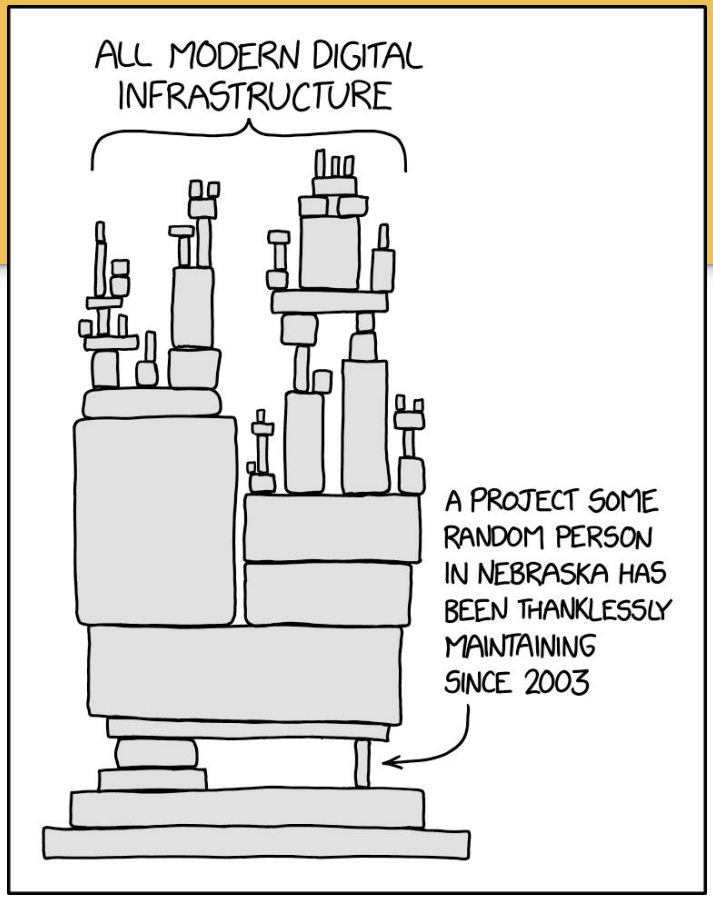
Supply Chain Attacks
Heartbleed

- $2,000 Spenden pro Jahr an OpenSSL
- $841 in 3 Tagen nach Heartbleed

- Gründung der Core Infrastructure Initiative bei der Linux Foundation, jetzt Open Software Security Foundation (OpenSSF)
  - https://openssf.org/
  - > $10 million raised by 2021

- Deutschland: Sovereign Tech Fund
  - https://sovereigntechfund.de

- PHP: PHP Foundation
  - https://thephp.foundation/



May 12, 2021

US Regierung: Executive Order 14028

- Führt Anforderung für SBOMs ein (Software Bill of Materials)
- Linux Foundation SPDX SBOMs
  - https://spdx.dev/
  - Kann aus einem GitHub Dependency Graph exportiert werden
- OWASP CycloneDX
  - https://cyclonedx.org/
  - Composer plugin: `cyclonedx/cyclonedx-php-composer`

Bekomm die Supply Chain deiner Projekte in den Griff

- Supply Chain identifiziert und dokumentieren
  - Alle Werkzeuge und Abhängigkeiten: SBOMs
  - Alle Services die verwendet werden: Wer sind die Anbieter? Checklisten um Informationen zu sammeln.
  - Alle Prozesse und Infrastruktur, die verwendet wird
  - Regelmäßig aktuell halten

- Risikoanalyse
  - Wahrscheinlichkeit von Problemen
  - Auswirkungen von Problemen

Alessandro Ranellucci @alranel · Jan 4, 2022

Dear $bigcorp, I'm an #opensource maintainer and not a provider. Please confirm which steps YOU are taking to ensure the software you're getting for free and using for your business is secure and maintained. #facepalm

> *Dear Provider,*
>
> *[redacted] is reaching out to you as a provider of the Slic3r software utilized by [redacted] for running its business.*
>
> *[redacted] are reaching out to you in response to the zero day log4j vulnerability the details are published by Apache: https://logging.apache.org/log4j/2.x/security.html*
>
> *Please confirm whether the system provided by you to [redacted] is susceptible to the log4j vulnerability.*
>
> *Please confirm which steps [redacted] is to take in order to protect its assets from possible attacks related to the software vulnerability.*
>
> Best regards / Cordialement.

56 689 2,669

David Longenecker
@dnlongen

I absolutely get your point, and it's 100% a valid point. At the same time, I have to tip my hat to $bigcorp whose software supply chain inventory is comprehensive enough to contact individual open source maintainers.

3:36 PM · Jan 5, 2022

https://twitter.com/dnlongen/status/1478737214179844100

Bekomm die Supply Chain deiner Projekte in den Griff

- Risikominimierung
  - Regelmäßig veraltete Software identifizieren und aktualisieren
    - so sehr automatisieren wir möglich
  - Lieferanten auditieren
    - Man kann nicht alles selbst machen, z.B. hat ein großer Cloud Hoster vermutlich einen besseren Überblick über Hardware und Firmware Sicherheitsupdates als ihr
  - Wählt Prozesse, die Risiken minimieren
    - getestete Artefakte deployen, statt Buildprozess beim Deployment, ggf. mit Unterschieden zu CI
    - Wählt deklarativen Zustand statt Zustand der über Zeit mutiert wird

Composer Tips
zu Supply Chain Security

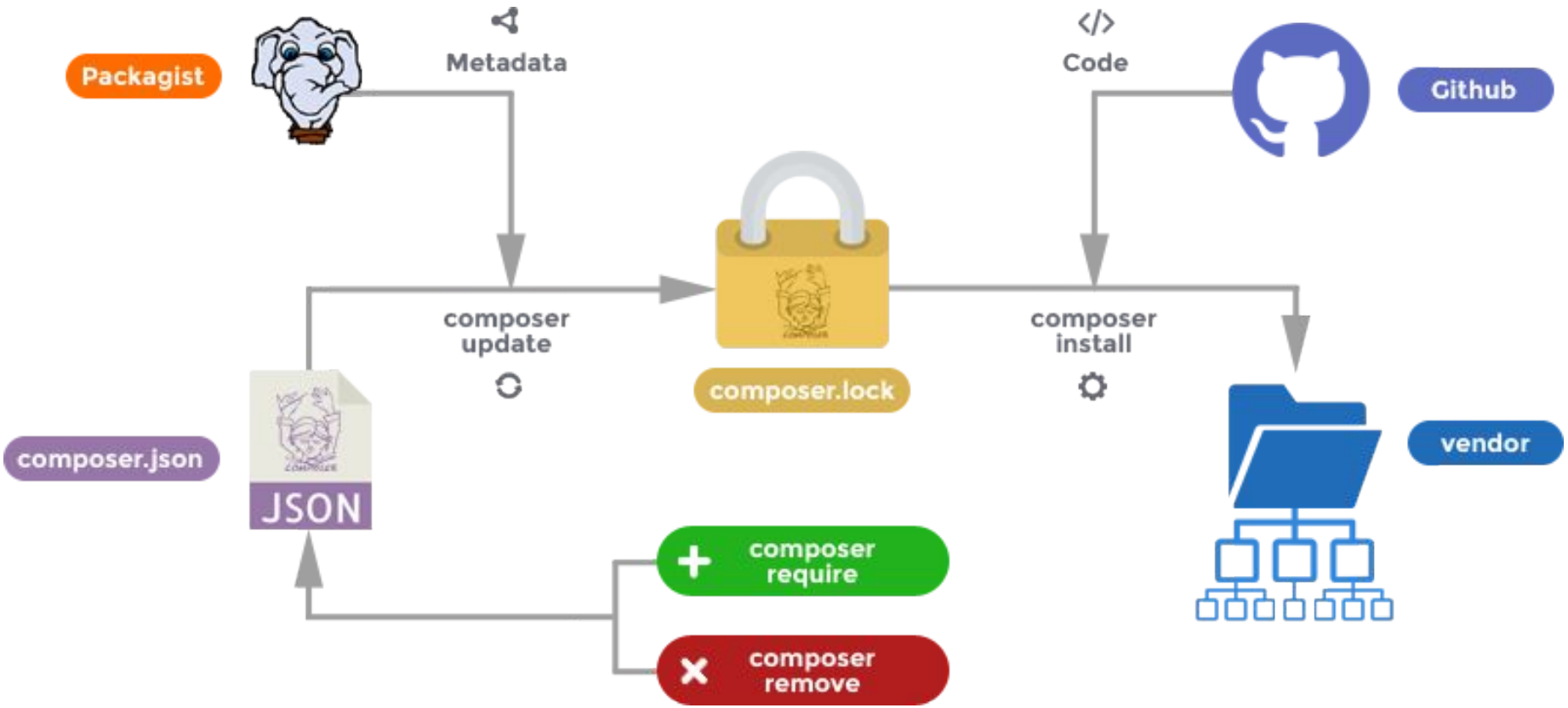
PRIVATE PACKAGIST

Composer 2.4: composer audit

- **`composer audit`** Command
  - Liste von Paketversionen mit bekannten Sicherheitslücken in composer.lock
  - Verwendet packagist.org Vulnerability DB API
    - GitHub advisory database
    - FriendsOfPHP/security-advisories
  - Non-zero status code falls Lücken gefunden -> zB in CI prüfen und fehlschlagen lassen

- `composer update implies audit --format=summary`
- `composer require --dev roave/security-advisories:dev-latest`

composer update vs. composer install



Packagist.org

- Nur Metadaten
- Keine Checksums für Pakete von GitHub
  - https://github.com/sansecio/composer-integrity-plugin
- Keine Signaturen
  - https://www.drupal.org/project/infrastructure/issues/3325040 - Automatic Updates / TUF
- Kein Mechanismus um Code hochzuladen
- Packagist.org maintainer account takeover
  https://blog.packagist.com/packagist-org-maintainer-account-takeover/
  - Bearbeiten von Source URLS nicht mehr erlaubt ab 50k Installs

Supply Chain Attacks

- Apr 13, 2022: Composer Command Injection Vulnerability
  - https://blog.packagist.com/cve-2022-24828-composer-command-injection-vulnerability/
  - Code Execution durch Git oder Mercurial Branch Namen
- Apr 27, 2021: Composer Command Injection Vulnerability
  - https://blog.packagist.com/composer-command-injection-vulnerability/
  - Code Execution durch Mercurial Repository URLs
- Mar 11, 2021: Git Clone Security Vulnerability
  - https://blog.packagist.com/git-clone-security-vulnerability/
  - Git Sicherheitslücke auf Case-insensitive Dateisystemen kann durch Composer exploited werden, wenn man Abhängigkeiten cloned

Also vendor Verzeichnis commiten?

- Wer kennt den git command um alle Änderungen in vendor zu commiten?

Also vendor Verzeichnis commiten?

- Wer kennt den git command um alle Änderungen in vendor zu commiten?
  - `git add vendor/` löscht Dateien nicht, evtl. Bugs/Sicherheitsprobleme
  - Verwendet `git add -A vendor/`
- vendor Verzeichnisinhalte können vom erwarteten Inhalt abweichen
  - Wir prüft ihr, dass die Inhalte der lock Datei entsprechen?
    - z.B. wurden Dateien aus gelöschten Paketen wirklich gelöscht?
- Konflikte lösen in größeren Teams wird noch schwieriger als die Inhalte der lock Datei zu verwalten

Also vendor Verzeichnis commiten?

- Angriffsszenarien, z.B. verärgerter Mitarbeiter
  - Szenarien
    - Code in Composer unbekanntem Verzeichnis - sieht aus wie legitime Abhängigkeit
    - Kann Code existierender Pakete in vendor/ ändern
  - Würde euer Review Prozess soetwas als Teil eines großen Updates bemerken?
  - Falls nicht, habt ihr Werkzeuge, die den Unterschied bemerken würden?
    - Ist es einfacher/günstiger das zu bauen als privates Composer Repository zu benutzen?

Allgemein: **Nein, das vendor Verzeichnis nicht commiten**

Benutzt ein eigenes Composer Repository

- Satis
- JFrog Artifactory
- Sonatype Nexus Repository
- Cloudsmith
- GitLab Package Registry
- …

- **Private Packagist**

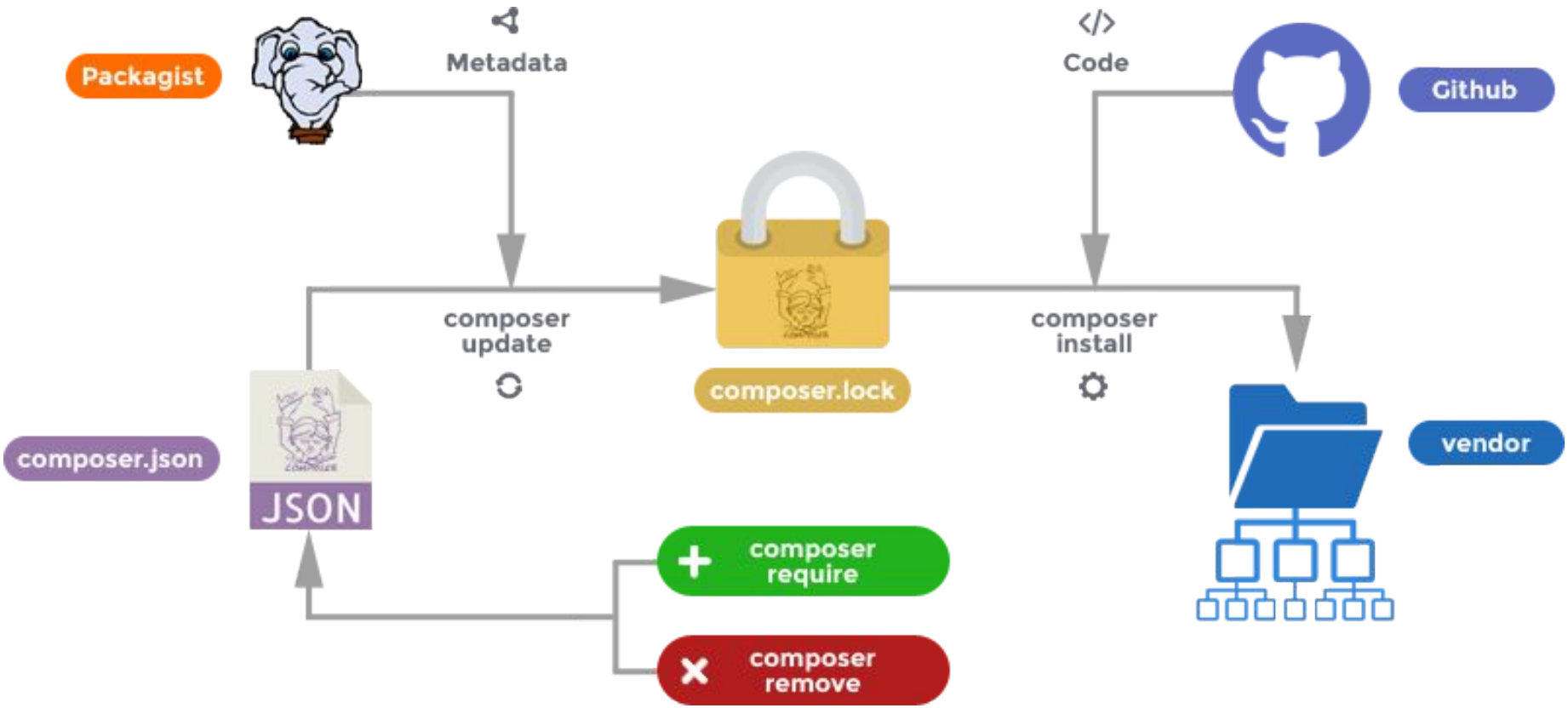
PRIVATE PACKAGIST

Private Packagist

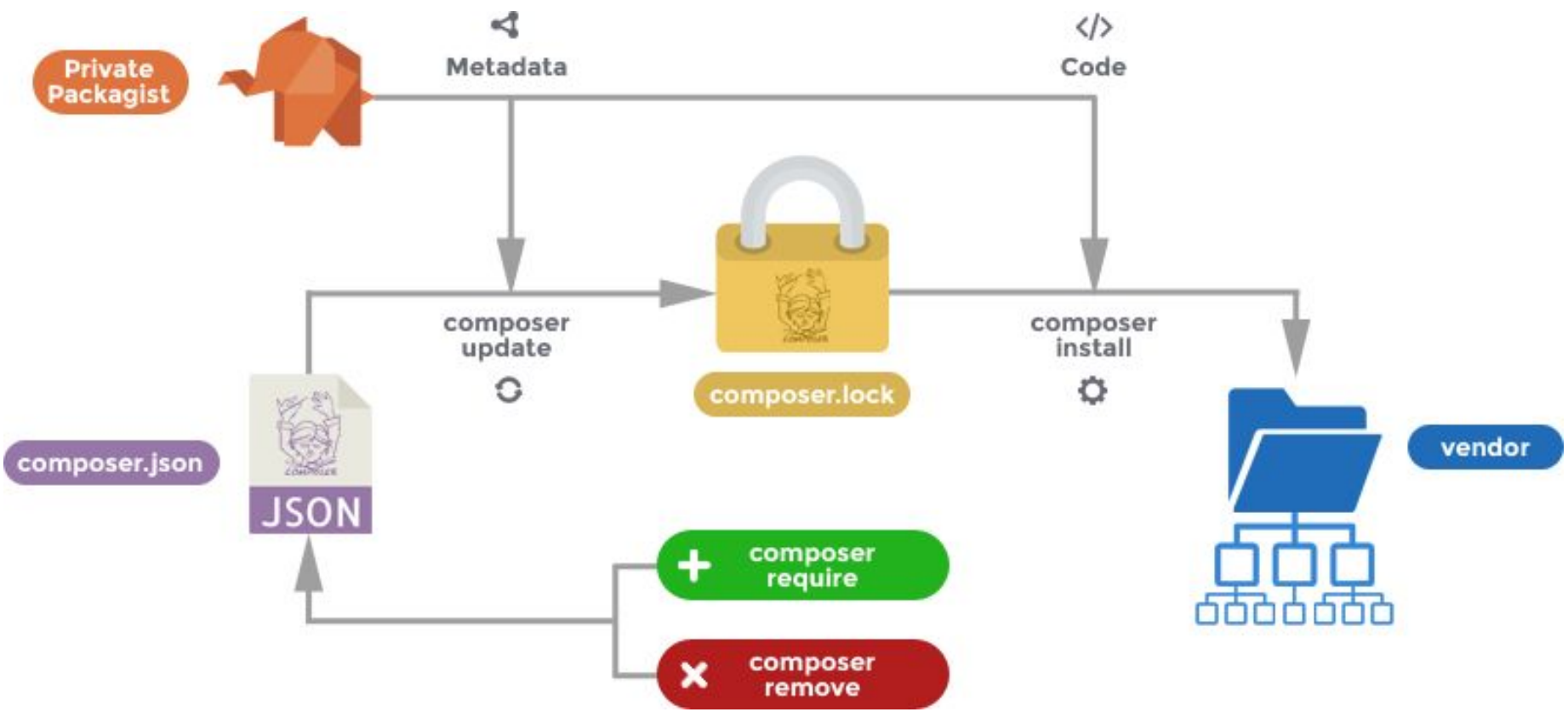
- Speichert eine Kopie aller benutzter Versionen eurer Abhängigkeiten
  - Sicher vor Löschen
  - Sicher vor Veränderung
- Liefert Metadata **und** Code
- Alternative Methoden das gleiche zu erreichen meist aufwendiger
  - z.B. alle Abhängigkeiten in Git Repositories kopieren, wie aktualisiert man diese dann?

PRIVATE PACKAGIST

Public packagist.org / GitHub



Private Packagist



Abhängigkeiten oft aktualisieren

- Legt einen Zeitplan und/oder regelmäßige Erinnerungen an Updates fest
- Nutzt Alerting für Sicherheitslücken in euren Abhängigkeiten
  - GitHub Dependabot
    https://docs.github.com/en/code-security/dependabot/dependabot-alerts/about-dependabot-alerts
  - Snyk
    https://snyk.io/product/open-source-security-management/
  - **Private Packagist** Security Monitoring
    https://packagist.com/features/security-monitoring

PRIVATE PACKAGIST

Abhängigkeiten oft aktualisieren

Noch besser: Automatische Updates

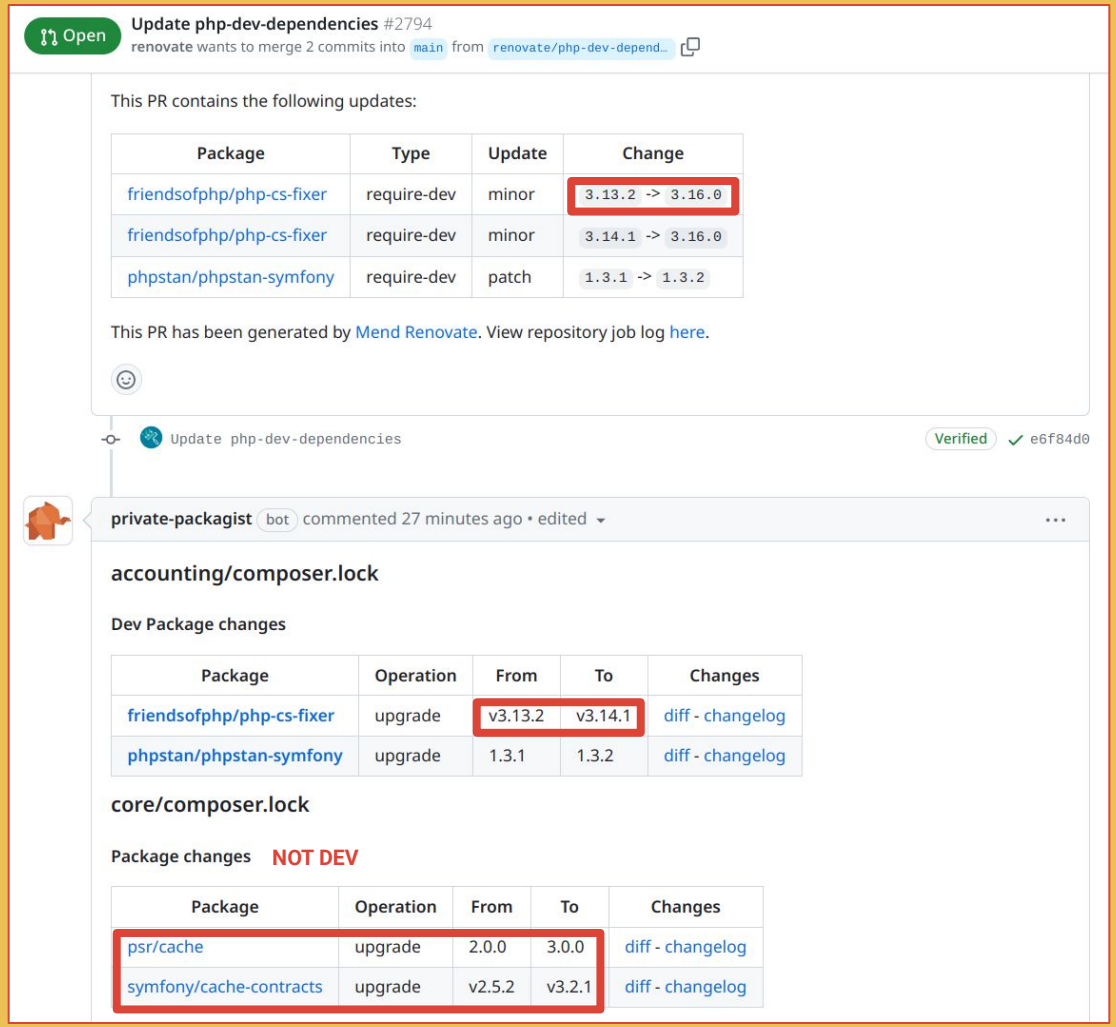
- Mend Renovate https://www.mend.io/renovate/
- GitHub Dependabot https://github.com/dependabot
- *(WIP: Private Packagist Automated Updates)*

Direkt einen Pull Request bekommen wenn ein Update nötig ist

Achtung!

Private Packagist Update Review

GitHub
BitBucket
GitLab



Open **Update php-dev-dependencies** #2794
renovate wants to merge 2 commits into `main` from `renovate/php-dev-depend…`

This PR contains the following updates:

Package	Type	Update	Change
friendsofphp/php-cs-fixer	require-dev	minor	`3.13.2` -> `3.16.0`
friendsofphp/php-cs-fixer	require-dev	minor	`3.14.1` -> `3.16.0`
phpstan/phpstan-symfony	require-dev	patch	`1.3.1` -> `1.3.2`

This PR has been generated by Mend Renovate. View repository job log here.

`Update php-dev-dependencies` Verified `e6f84d0`

private-packagist bot commented 27 minutes ago • edited

accounting/composer.lock

Dev Package changes

Package	Operation	From	To	Changes
friendsofphp/php-cs-fixer	upgrade	v3.13.2	v3.14.1	diff - changelog
phpstan/phpstan-symfony	upgrade	1.3.1	1.3.2	diff - changelog

core/composer.lock

Package changes **NOT DEV**

Package	Operation	From	To	Changes
psr/cache	upgrade	2.0.0	3.0.0	diff - changelog
symfony/cache-contracts	upgrade	v2.5.2	v3.2.1	diff - changelog

PRIVATE PACKAGIST

Composer Plugins & Scripts

- Seit Composer 2.2 müssen Plugins explizit aktiviert werden
  - `config.allow-plugins`
  - Beschützt euch vor unabsichtlichem Ausführen bösartigen Codes bevor die composer.lock Änderungen gereviewt werden können.

- Scripts & plugin Auswahl ist auf root composer.json begrenzt
  - Schützt vor Angriffen durch bösartige Maintainer, Dependency Confusion oder versehentlich hinzugefügten Abhängigkeiten
  - Ihr müsst immer noch Änderungen an der Lock Datei reviewen!

Empfohlene Verwendung von Composer im Deployment Prozess

- composer.lock commiten
- CI/CD
  - composer install (nicht update!)
  - Codegenerierung sofern nötig
  - Alles in ein Archiv packen
- deployment
  - Upload auf prod Server, dann verschieben
  - composer check-platform-reqs
  - Optimierten autoloader generieren
  - Webserver auf neuen Code umstellen

Ergebnis

- keine Überraschungen auf Prod
  - Gleiche Abhängigkeiten wie beim Testen
  - Keine Versionskonflikte beim Deploy
  - Code ändert sich zur Laufzeit nicht
- Deploy auf mehrere Server
  - Überall genau gleicher Zustand
  - Keine unnötig wiederholten Schritte

Fragen / Feedback?

Private Packagist
https://packagist.com

E-Mail: contact@packagist.com
X: @naderman Mastodon: @naderman@phpc.social