Composerを使った サプライチェーン攻撃の様子を 眺めてみる

第187回 PHP勉強会@東京

Hideki Kinjyo
GitHub: o0h / X: @o0h_

[公開用]

※発表後の追記

あなたがリポストしました

Jordi Boggiano @seldaek · 4時間

Composer 2.10 is out.

Native malware filtering via @AikidoSecurity, enabled by default on @Packagist. Plus a unified config.policy framework, deprecated source fallback, and wildcards in --with.

#php #phpc #composerphp

5 49 135 6,738

以下、発表内容

Composerを使った サプライチェーン攻撃の様子を 眺めてみる

第187回 PHP勉強会@東京

Hideki Kinjyo
GitHub: o0h / X: @o0h_

[公開用]

今日の登場人物

Composer

パッケージ入れる君

Packagist

パッケージ情報を
教えてくれるさん
（電話帳的な？）

自己紹介

- 金城秀樹 / きんじょうひでき
- GitHub: @oOh / 𝕏: @oOh_
- アイコンは美味しい鮭親子丼の写真です
- 来週の今頃は札幌にいます
- 最近はPodcastをやっています
  - ハッシュタグ: #readlinefm

Hideki Kinjyo
oOh

https://twitter.com/oOh_

今日の話

ここのところ、
毎週くらいのペースで
脆弱性やサプライチェーン攻撃の話を
聞くじゃんね〜〜〜〜〜〜〜

今日の話

Composer界隈の皆さんと、
「それってどうやって起こるの？？」を
見ていきたい！！

今日の話

という話です

（裏？ばなし）

きっと、今日より踏み込んだ(？)話は
PHP Conference Japanで
誰かから聴けるでしょう！！

気になるプロポーザルに
★を付けて盛り上げよう💪
https://fortee.jp/phpcon-2026/proposal/all?q=サプライチェーン&f=all

話すこと

- Composer+Packagistにおけるサプライチェーンアタックについて
- 汚染/侵害されたサードパーティパッケージが混入してくる、
  というケースのみに限定します
  - ソフトウェアサプライチェーンアタックはもっと色々ある

話さないこと

↓は話さない

- サプライチェーンアタック一般への(専門的)な対策方法
- 今回は「Composerの場合」を追うのが主なので
  一般的な話は、そういう文献等を当たってください

おしながき

1. その攻撃は、どういう風に実行されるの
2. その攻撃は、どういう風に入り込むの
3. 昨今のComposer側の対策

参考記事(日本語)

- 前に書いた:
  『昨今のComposerは(サプライチェーンアタックについて)どうなってるんすかね？？って軽く調べ - 大好き！にちようび』
  https://daisuki.nichiyoubi.land/entry/2026/04/03/005936
- あと、コドモンさんの記事:
  『できることから始めるPHPプロジェクトのOSSサプライチェーン攻撃対策 - コドモン Product Team Blog』
  https://tech.codmon.com/entry/2026/04/27/092802
  - 自分が勢いで書いたコンテンツより、整ってるんじゃないですかねぇ

その攻撃は、どういう風に実行されるの

そもそも超大前提的な

開発者（パッケージのユーザー）が
意図しないタイミングで
変な動きをする！！！

・・・が困る、って話

例えるなら

インストールしただけなのに

”何もしていないのに”感

手順を守った(or自動化された)
やり方なのに

コードいじってないのに

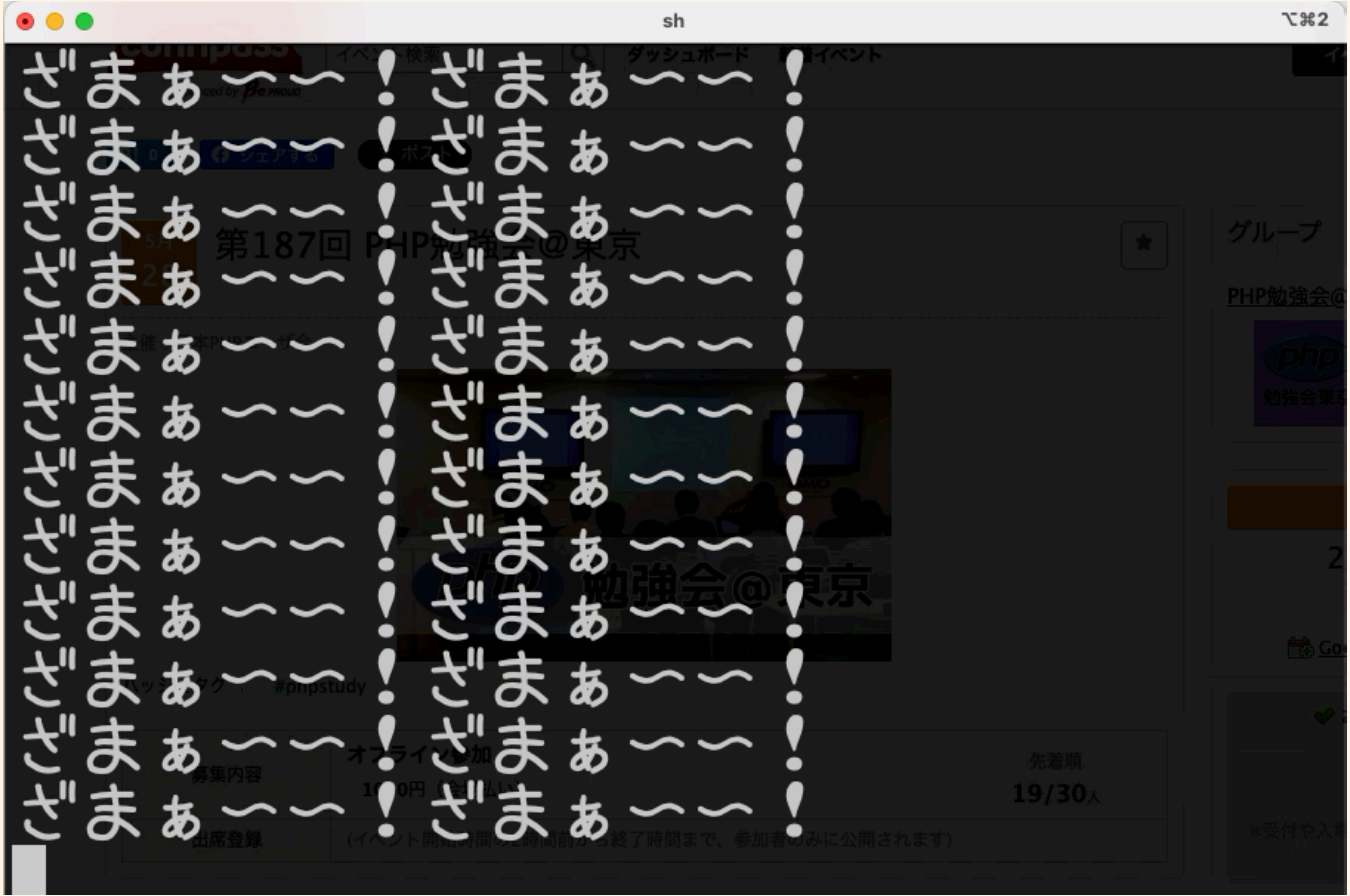
(怖い)デモ:
いつも通りの`composer install`を
ぶっ壊します

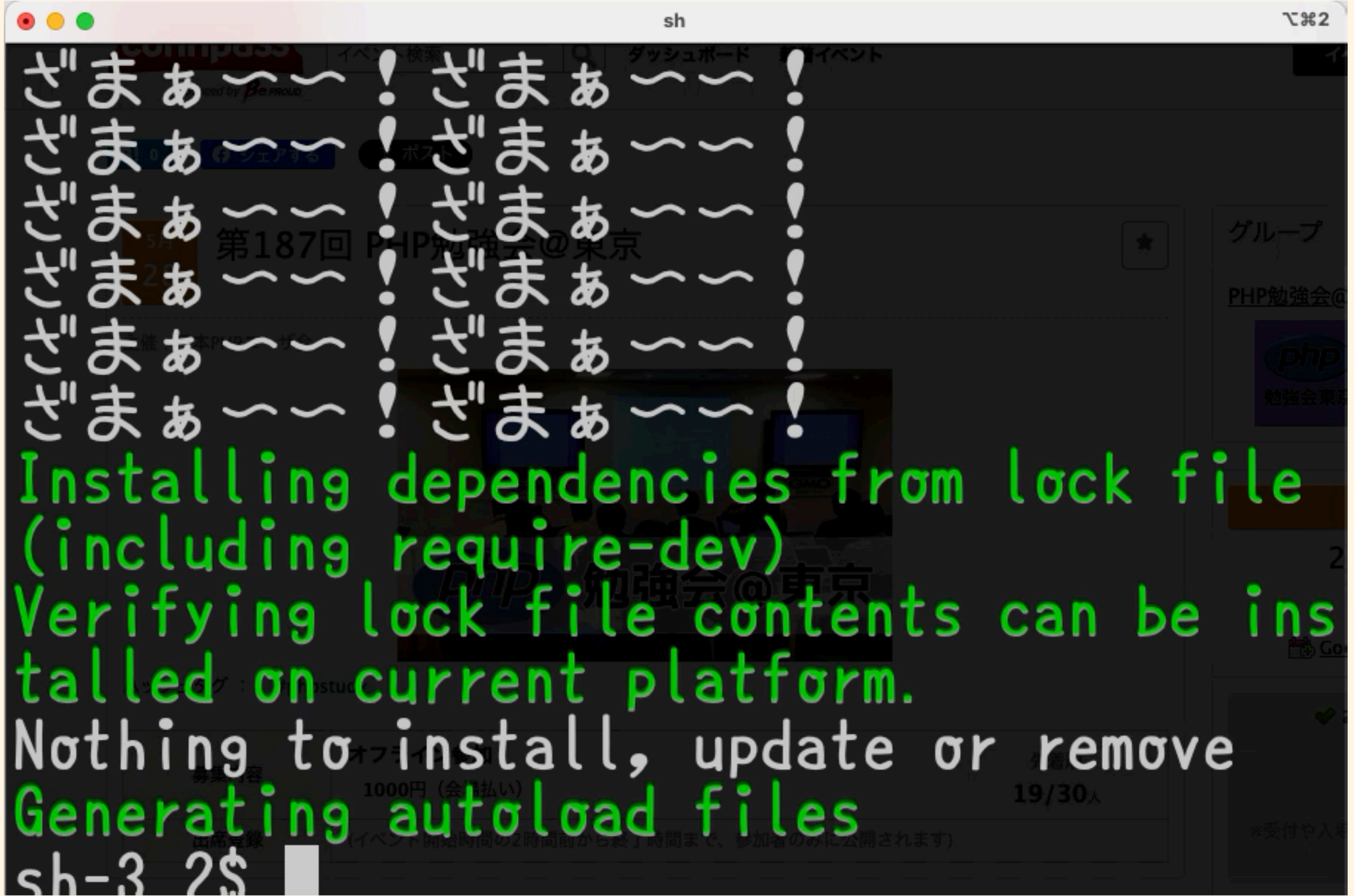
sh ⌥⌘2

```
sh-3.2$ ../composer-2.9.8.phar install
```

sh ⌥⌘2

```
sh-3.2$ ../composer-2.9.8.phar install
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
```


sh
⌥⌘2
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！


sh
⌥⌘2
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
ざまぁ〜〜！ざまぁ〜〜！
Installing dependencies from lock file
(including require-dev)
Verifying lock file contents can be ins
talled on current platform.
Nothing to install, update or remove
Generating autoload files
sh-3.2$

怖いですね

いつもの`composer install`が、
いつもと全然違う挙動

怖いですね

これは「文字列をechoする」だけだが、

「何かを出来ている」状態にありますね？

怖いですね

「ざまぁ〜」する代わりに、
「本番用のビルドにおいてAWSのクレデンシャルやSSHキーを環境変数etcから抜いてどっかにPOST」でも良いわけです

Composerにおいて
「自動的にコードを動かせる」のは、どこ？

自動実行を差し込める所

1. プラグインとして動作して、任意のイベントで発火
   ➡ installなど、Composerコマンドの実行時
2. オートロード(イーガーロード)ファイルとして動作させる
   ➡ Webリクエストやバッチ実行時など、`autoload.php`読込み時

Pluginでの実行

Composerのプラグインの仕組み

- Composerは、そのコマンド実行時に
  ライフサイクルに応じたイベントを発行している
- プラグインは、そのイベントを購読して、独自処理を発火する仕組み

主なイベントの例:

composer.lockの更新完了時・パッケージのDL時・autoloadファイルの生成時etc

例えばこんなプラグイン

- cweagans/composer-patches
  - パッケージインストール後、vendorファイルに任意のパッチを当てる
- php-http/discovery
  - 依存更新(composer.lock更新)時に、
    「対象HTTPクライアントが何かしら入っているか」をチェックする

さっきのデモの中身

```php
class EvilPlugin implements PluginInterface, EventSubscriberInterface
{
    public function activate(Composer $composer, IOInterface $io): void {}
    public function deactivate(Composer $composer, IOInterface $io): void {}
    public function uninstall(Composer $composer, IOInterface $io): void {}
    public static function getSubscribedEvents(): array
    {
        return [ScriptEvents::POST_INSTALL_CMD => 'onPostInstall'];
    }

    public function onPostInstall(): void
    {
        for ($i = 0; $i < 100; $i++) {
            echo "ざまぁ〜〜！ざまぁ〜〜！\n";
            usleep(50_000);
        }
    }
}
```

autoloadでの実行

sh ⌥⌘2

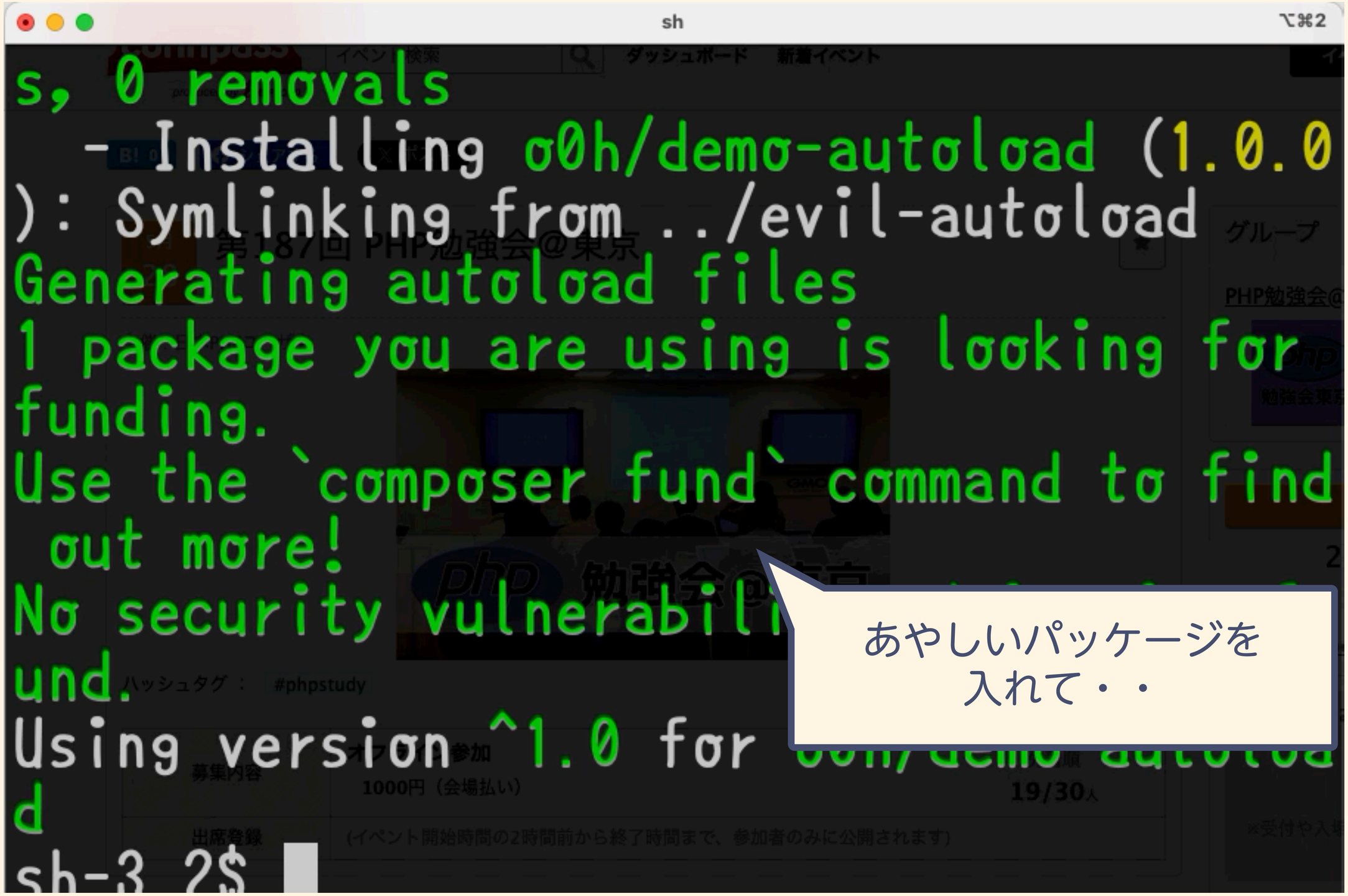
```
sh-3.2$ php app.php calc 10 2 +
= 12
sh-3.2$ php app.php calc 10.1 2.9 +
= 13
sh-3.2$
```

何の変哲もない
四則演算プログラム

sh ⌥⌘2

```
sh-3.2$ php app.php calc 10 2 +
= 12
sh-3.2$ php app.php calc 10.1 2.9 +
= 13
sh-3.2$ composer require o0h/demo-autol
oad
```

あやしいパッケージを
入れて・・


sh
⌥⌘2
s, 0 removals
- Installing o0h/demo-autoload (1.0.0
): Symlinking from ../evil-autoload
Generating autoload files
1 package you are using is looking for
funding.
Use the `composer fund` command to find
out more!
No security vulnerabili
und.
Using version ^1.0 for o0h/demo-autoloa
d
sh-3.2$
あやしいパッケージを
入れて・・

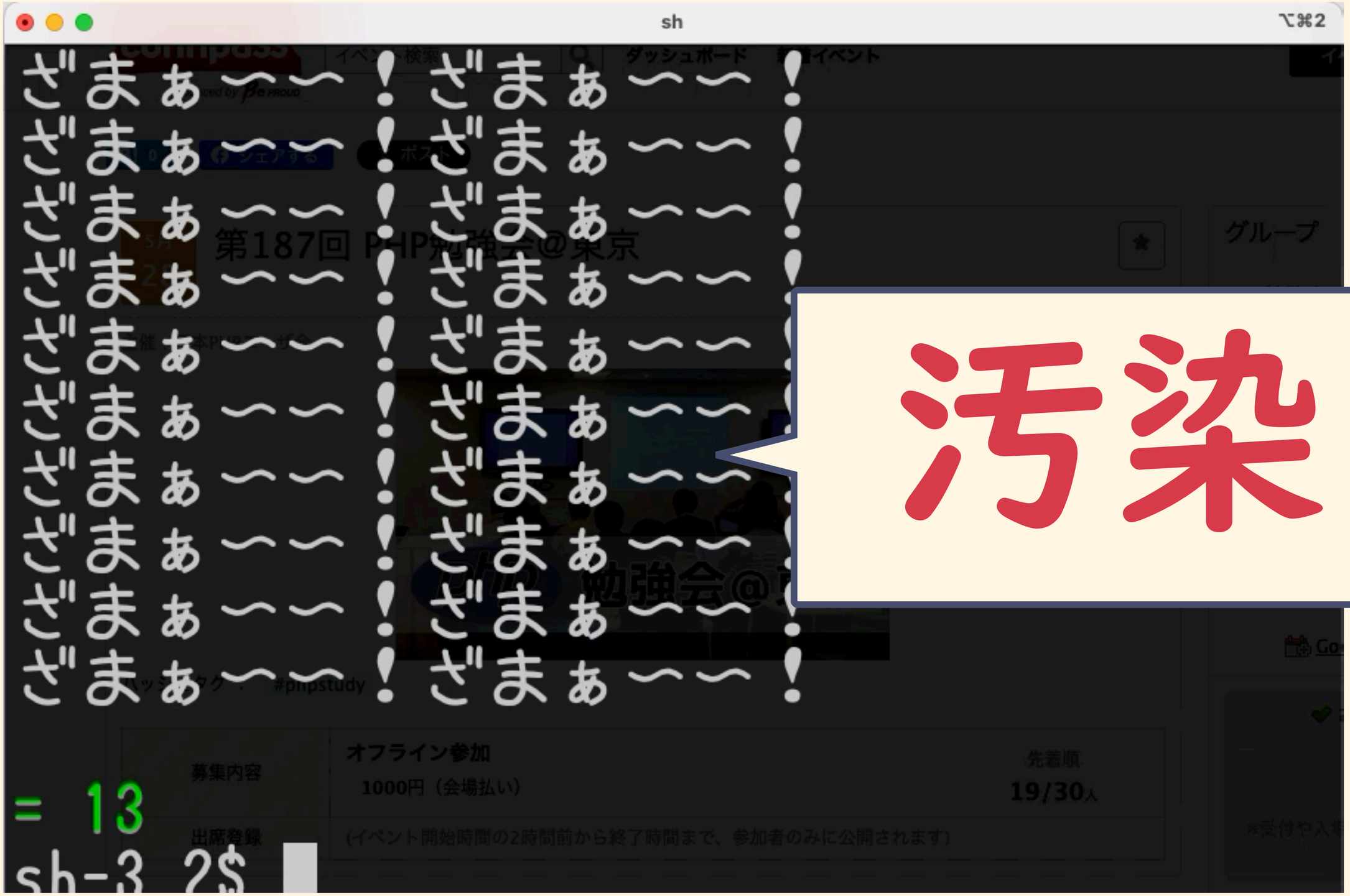
sh

⌥⌘2

```
sh-3.2$ php app.php calc 10.1 2.9 +
```

ただ普通に
プログラムを実行


sh
⌥⌘2
sh-3.2$ php app.php calc 10.1 2.9 +
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
ざまぁ～～！ざまぁ～～！
汚染


sh
ざまぁ〜〜！
= 13
sh-3.2$
汚染

何をされているのか

- Composerに「オートロード」ありますよね
  - 「PSR-4」とかのやつです
  - ↑の場合、composer.jsonにnamespaceと対応ディレクトリを指定する
- オートロードの種別に **`files`** というものがあります
  - これは「クラス(like)定義」以外に使います
  - 有名どころで言うと、symfony/polyfillとかがメッチャ使う
- **指定されたファイルが自動で読み込まれる**ようになる
  - PSR-4などは、「遅延読み込み」のための仕組み

さっきのデモの中身：パッケージ定義

```
{
    "name": "o0h/demo-autoload",
    "description": "An useful utility package :)",
    "version": "1.0.0",
    "autoload": {
        "files": [
            "src/bootstrap.php"
        ]
    }
}
```

さっきのデモの中身：核心のコード

src/bootstrap.php

```php
<?php

for ($i = 0; $i < 100; $i++) {
    echo "ざまぁ〜〜！ざまぁ〜〜！\n";
    usleep(50_000);
}
```

さっきのデモの中身: autoloadfiles

vendor/autoload.php から
読み込まれているファイル

```php
<?php

// autoload_files.php @generated by Composer

$vendorDir = dirname(__DIR__);
$baseDir = dirname($vendorDir);

return array(
    '(省略)' => $vendorDir . '/minicli/minicli/src/helpers.php',
    '(省略)' => $vendorDir . '/o0h/demo-autoload/src/bootstrap.php',
);
```

さっきのデモの中身: autoloadfiles

vendor/autoload.php から
読み込まれているファイル

```php
<?php

// autoload_files.php @generated by Composer

$vendorDir = dirname(__DIR__);
$baseDir = dirname($vendorDir);

return array(
    '(省略)' => $vendorDir . '/minicli/minicli/src/helpers.php',
    '(省略)' => $vendorDir . '/o0h/demo-autoload/src/bootstrap.php',
);
```

さっきのデモの中身: autoloadfiles

vendor/autoload.php から
読み込まれているファイル

```
class ComposerAutoloaderInitaXXXXX
{
// 省略
    public static function getLoader()
    {
// 省略
        $filesToLoad = \Composer\Autoload\ComposerStaticInitaXXXXX::$files;
        $requireFile = \Closure::bind(static function ($fileIdentifier, $file) {
            if (empty($GLOBALS['__composer_autoload_files'][$fileIdentifier])) {
                $GLOBALS['__composer_autoload_files'][$fileIdentifier] = true;

                require $file;
            }
        }, null, null);
        foreach ($filesToLoad as $fileIdentifier => $file) {
            $requireFile($fileIdentifier, $file);
        }

        return $loader;
    }
}
```

要するに

Composer利用時に気をつけたいこと

- Pluginは比較的自由が効くので、安易に使わない
  - Composerが提供している安全機構については後述
- autoload.filesは・・・厄介ですねぇ
  - 使うべきでない、というのもちょっと現実的ではない
  - 気をつけることは出来るかもだけど
- 両者で、発火タイミングが違う

その攻撃は、どういう風に入り込むの

まずは
Composer/Packagistの情報管理

レジストリ利用者としてのComposer

- 今回はデフォルトレポジトリ = Packagistの話に限定しますが
- Packagist自体は、パッケージの実コードやバイナリを持たない
  - パッケージの本体は、(主に)GitHubを案内している
- 代わりに、メタ情報だけを管理している
  - 提供しているパッケージ名と、バージョンの情報
  - 各バージョンに対応するハッシュ(Gitコミットハッシュ)

composer.json

```
{
    "require": {
        "php": ">=8.2",
        "cakephp/cakephp": "5.3.*",
        "cakephp/migrations": "^5.0",
        "cakephp/plugin-installer": "^2.0",
        "mobiledetect/mobiledetectlib": "^4.8.03"
    },
```

（基本的には）
利用したいバージョンの
「範囲」を指定する

composer.lock


```
{
    "content-hash": "1d69f3bd865733ecf989ff03a8dd25a9",
    "packages": [
        {
            "name": "cakephp/cakephp",
            "version": "5.3.6",
            "source": {
                "type": "git",
                "url": "https://github.com/cakephp/cakephp.git",
                "reference": "cdaca8c3b710789e8545bff5a83194a6b19cad46"
            },
            "dist": {
                "type": "zip",
                "url":
"https://api.github.com/repos/cakephp/cakephp/zipball/cdaca8c3b710789e8545bff5a83194a6b19cad46",
                "reference": "cdaca8c3b710789e8545bff5a83194a6b19cad46",
                "shasum": ""
            },
```

composer.lock

```
{
    "content-hash": "1d69f3bd865733ecf989ff03a8dd25a9",
    "packages": [
        {
            "name": "cakephp/cakephp",
            "version": "5.3.6",
                "url": "https://github.com/cakephp/cakephp.git",
                "reference": "cdaca8c3b710789e8545bff5a83194a6b19cad46"
            },
            "dist": {
                "type": "zip",
                "url":
"https://api.github.com/repos/cakephp/cakephp/zipball/cdaca8c3b710789e8545bff5a83194a6b19cad46",
                "reference": "cdaca8c3b710789e8545bff5a83194a6b19cad46",
                "shasum": ""
            },
```

バージョンが
明示的に指定され

composer.lock

```
{
    "content-hash": "1d69f3bd865733ecf989ff03a8dd25a9",
    "packages": [
        {
            "name": "cakephp/cakephp",
            "version": "5.3.6",
            "source": {
                "type": "git",
                "url": "https://github.com/cakephp/cakephp.git"
            },
            "dist": {
                "type": "zip",
                "url":
"https://api.github.com/repos/cakephp/cakephp/zipball/cdaca8
            },
```

実コードの
取得先が示される

パッケージ情報のソース

- この辺りの「バージョンとかハッシュとかdistのURL」を届けてくれているのが、PackagistのAPI
  - https://repo.packagist.org/p2/***/***.json
- Composerはコレを参照して、取り込んでいる

package.json

```
{
    "packages":{
        "cakephp/cakephp":[
            {
                "name":"cakephp/cakephp",
                "homepage":"https://cakephp.org",
                "version":"5.4.0-RC1",
                "version_normalized":"5.4.0.0-RC1",
                "license":[
                    "MIT"
                ],
                "dist":{

"url":"https://api.github.com/repos/cakephp/cakephp/zipball/ab7e118f3e72c90ec71f4eb4ac51cd5c22b7
394f",
                    "type":"zip",
                    "shasum":"",
                    "reference":"ab7e118f3e72c90ec71f4eb4ac51cd5c22b7394f"
                }
            }
```

めっちゃ割愛した
情報

package.json

```
{
    "packages":{
        "cakephp/cakephp":[
            {
                "version":"5.4.0-RC1",
                "version_normalized":"5.4.0.0-RC1",
                "license":[
                    "MIT"
                ],
                "dist":{
"url":"https://api.github.com/repos/cakephp/cakephp/zipball/ab71f4eb4ac51cd5c22b7394f",
                    "type":"zip",
                    "shasum":"",
                    "reference":"ab7e118f3e72c90ec71f4eb4ac51cd5c22b
                }
            }
```

バージョンごとに
distが入る

ヤバいコードはどうやってくるか

composer.lockがない場合

- composer.lockがない(もしくは更新される)場合、
  「(制約を満たす範囲で)**何が入ってくるかが保証されない**」。
- 更新される場合？
  - composer update
  - composer require

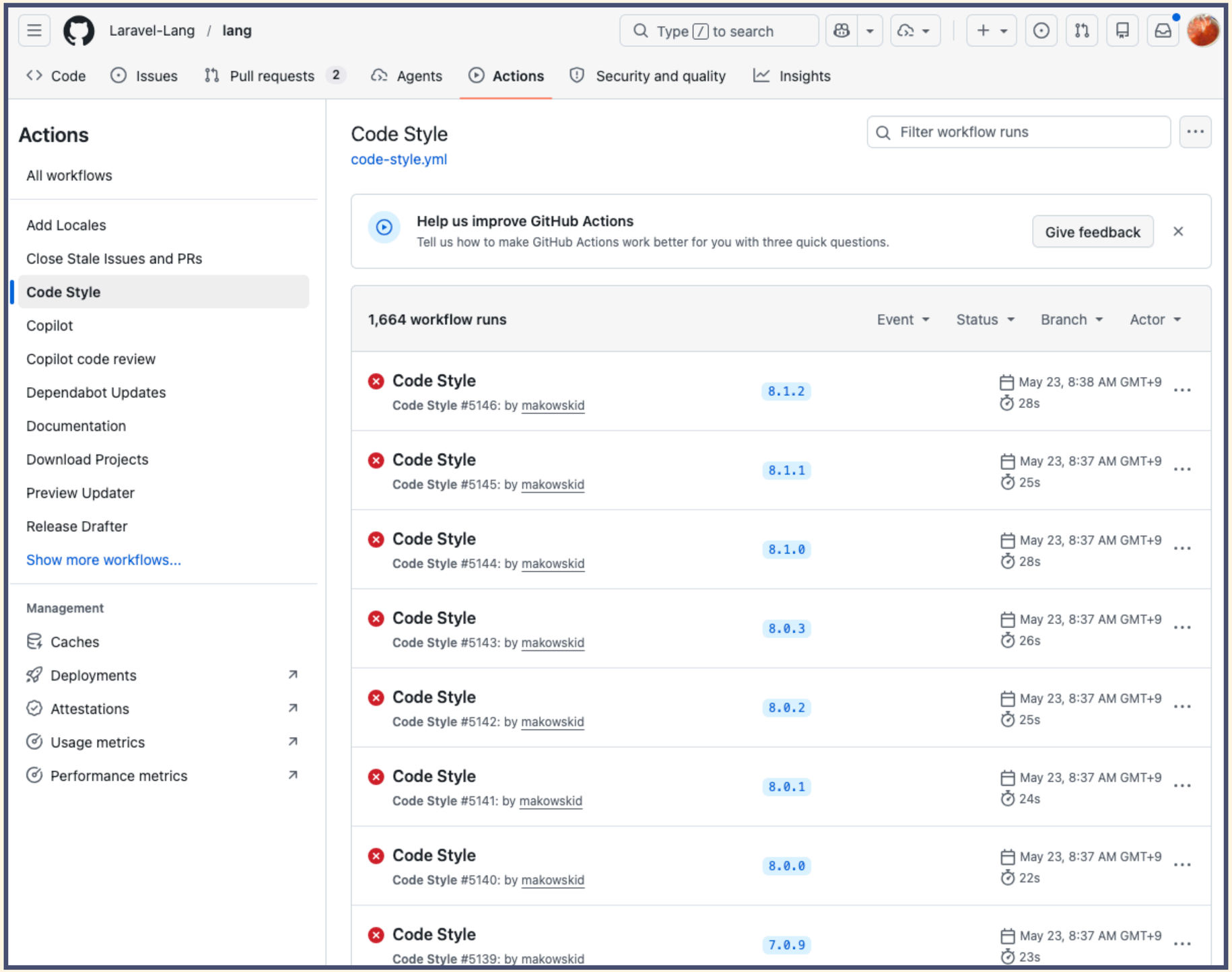
composer.jsonで「具体的な指定」をしていても？

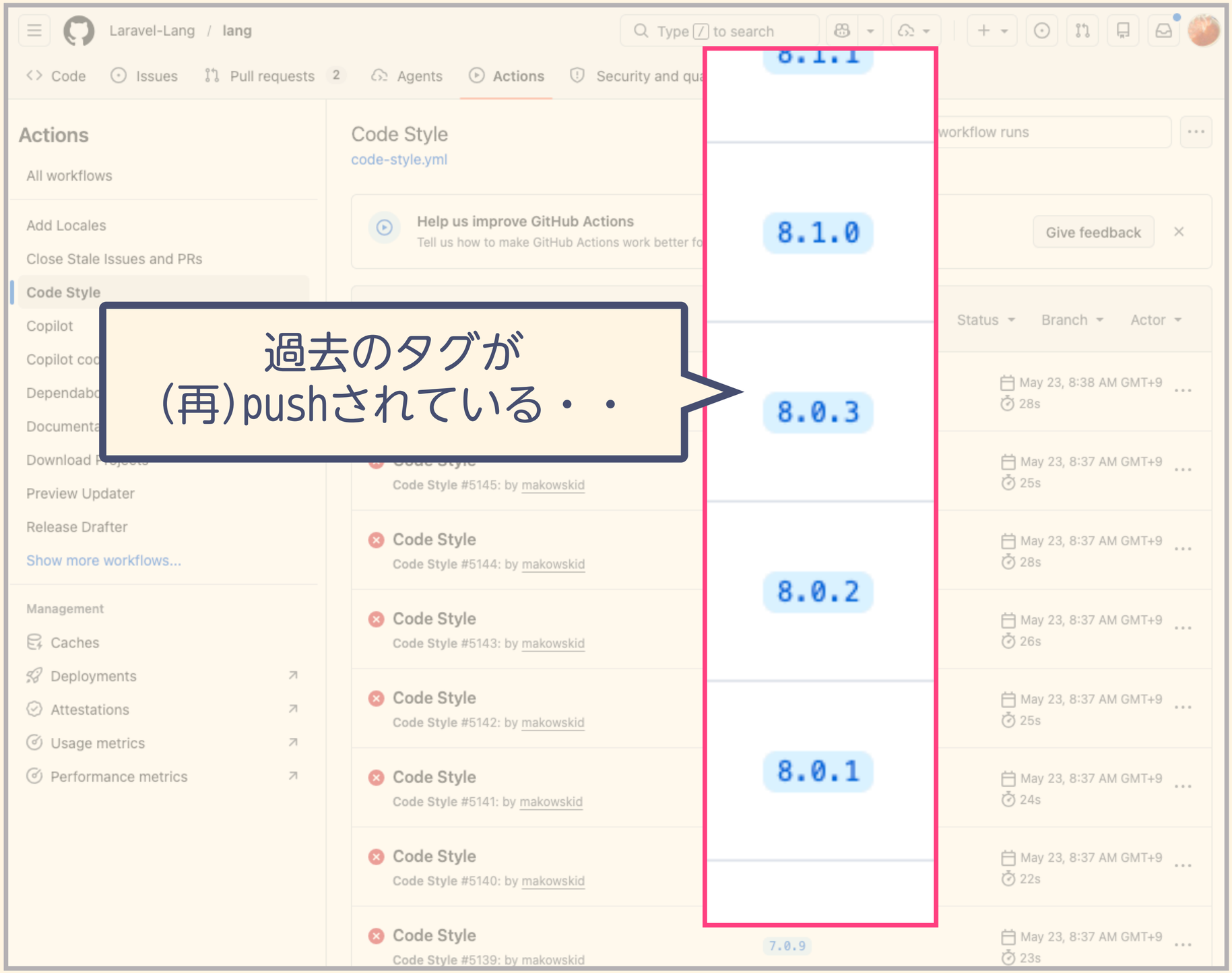
- 例えば、`composer require cakephp/cakephp:5.4.0` を指定
- ・・・しても、中身が同じ事は**保証されていない**
- Packagistの場合、同じリリース(タグ)での更新が可能
  - force push出来ちゃう

とても分かりやすい話

先日のlaravel-langのやつ

- Laravel Lang Compromised with RCE Backdoor Across 700+ Versions
  https://socket.dev/blog/laravel-lang-compromise
- 権限を取られた(断定してないかも)様子がある
- 攻撃者が、org内のコード等を操れる状態に
- CIの履歴を見ると、生々しく現場の様子が残ってる


Laravel-Lang / lang
Type / to search
Code
Issues
Pull requests 2
Agents
Actions
Security and quality
Insights
Actions
All workflows
Add Locales
Close Stale Issues and PRs
Code Style
Copilot
Copilot code review
Dependabot Updates
Documentation
Download Projects
Preview Updater
Release Drafter
Show more workflows...
Management
Caches
Deployments
Attestations
Usage metrics
Performance metrics
Code Style
code-style.yml
Filter workflow runs
Help us improve GitHub Actions
Tell us how to make GitHub Actions work better for you with three quick questions.
Give feedback
1,664 workflow runs
Event
Status
Branch
Actor
Code Style
Code Style #5146: by makowskid
8.1.2
May 23, 8:38 AM GMT+9
28s
Code Style
Code Style #5145: by makowskid
8.1.1
May 23, 8:37 AM GMT+9
25s
Code Style
Code Style #5144: by makowskid
8.1.0
May 23, 8:37 AM GMT+9
28s
Code Style
Code Style #5143: by makowskid
8.0.3
May 23, 8:37 AM GMT+9
26s
Code Style
Code Style #5142: by makowskid
8.0.2
May 23, 8:37 AM GMT+9
25s
Code Style
Code Style #5141: by makowskid
8.0.1
May 23, 8:37 AM GMT+9
24s
Code Style
Code Style #5140: by makowskid
8.0.0
May 23, 8:37 AM GMT+9
22s
Code Style
Code Style #5139: by makowskid
7.0.9
May 23, 8:37 AM GMT+9
23s


Laravel-Lang / lang
Type / to search
Code
Issues
Pull requests 2
Agents
Actions
Security and qua
Actions
All workflows
Add Locales
Close Stale Issues and PRs
Code Style
Copilot
Copilot coc
Dependabc
Documenta
Download P
Preview Updater
Release Drafter
Show more workflows...
Management
Caches
Deployments
Attestations
Usage metrics
Performance metrics
Code Style
code-style.yml
workflow runs
Help us improve GitHub Actions
Tell us how to make GitHub Actions work better fo
Give feedback
Status
Branch
Actor
過去のタグが
(再)pushされている・・
8.1.1
8.1.0
8.0.3
8.0.2
8.0.1
7.0.9
Code Style #5145: by makowskid
May 23, 8:38 AM GMT+9
28s
May 23, 8:37 AM GMT+9
25s
Code Style
Code Style #5144: by makowskid
May 23, 8:37 AM GMT+9
28s
Code Style
Code Style #5143: by makowskid
May 23, 8:37 AM GMT+9
26s
Code Style
Code Style #5142: by makowskid
May 23, 8:37 AM GMT+9
25s
Code Style
Code Style #5141: by makowskid
May 23, 8:37 AM GMT+9
24s
Code Style
Code Style #5140: by makowskid
May 23, 8:37 AM GMT+9
22s
Code Style
Code Style #5139: by makowskid
May 23, 8:37 AM GMT+9
23s

CIのジョブに紐づいているコミット

composer.json +5 -2

```
@@ -47,7 +47,10 @@
47      "autoload": {                                  47      "autoload": {
48          "psr-4": {                                 48          "psr-4": {
49              "LaravelLang\\Lang\\": "src/"          49              "LaravelLang\\Lang\\": "src/"
50  -       }                                          50  +       },
                                                       51  +       "files": [
                                                       52  +           "src/helpers.php"
                                                       53  +       ]
51      },                                             54      },
52      "autoload-dev": {                              55      "autoload-dev": {
53          "psr-4": {                                 56          "psr-4": {
@@ -86,4 +89,4 @@
86          "style": "vendor/bin/pint --parallel",     89          "style": "vendor/bin/pint --parallel",
87          "test": "vendor/bin/phpunit --             90          "test": "vendor/bin/phpunit --
    colors=always"                                         colors=always"
88      }                                              91      }
89  - }                                                92  + }
```

CIのジョブに紐づいているコミット

composer.json

```
@@ -47,7 +47,10 @@
47      "autoload": {
48          "psr-4": {
49              "LaravelLang\\Lang\\": "src/"
50  -       }
51      },
52      "autoload-d
53          "psr-4"
@@ -86,4 +89,4 @@
86          "style": "vendor/bin/pint --parallel",
87          "test": "vendor/bin/phpunit --
    colors=always"
88      }
89  - }
```

```
"autoload": {
    "psr-4": {
        "LaravelLang\\Lang\
    },
    "files": [
        "src/helpers.php"
    ]
```

```
90          "test": "vendor/bin/phpunit --
    colors=always"
91      }
92  + }
```

イーガーロードに
追加されている

こうなると、どうなる？

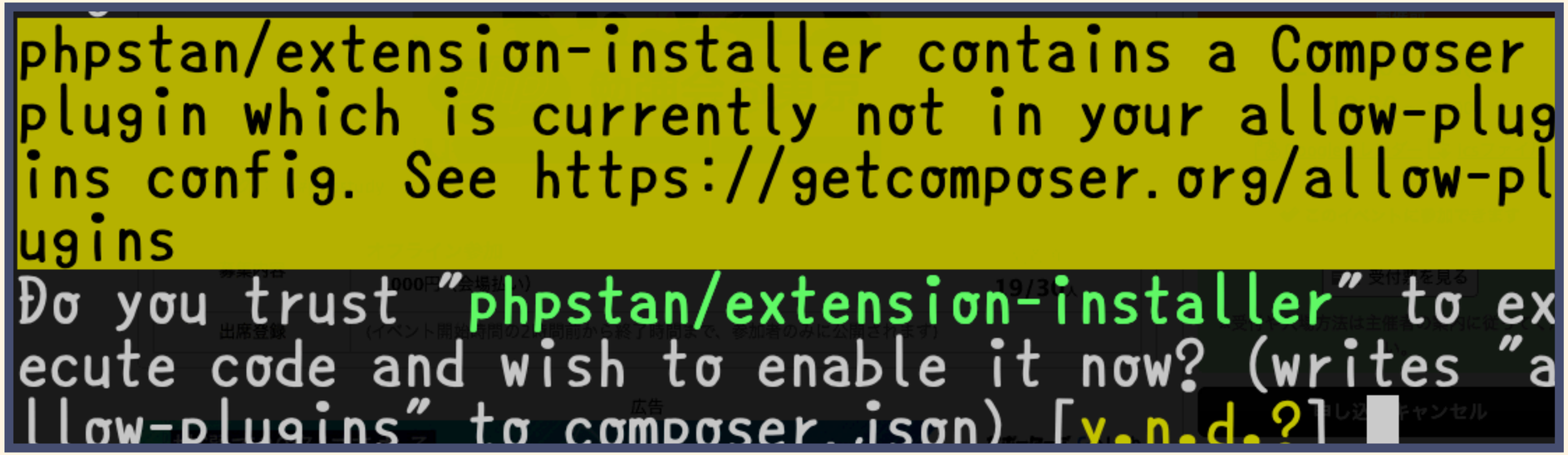
- 「正当だったバージョン」の内容は書き換えられて
  「違うコミットハッシュ」になっている
- composer.lockが変わっていなければ、
  「いま汚染されたファイル」を食わされないで済む
- 一方で、composer require/update系の
  composer.lockの更新は、汚染されたバージョンを食う

昨今のComposer側の対策

Plugin周り

プラグインは「明示的に許可されたもの」のみ動く

- Composerでは、プラグインは予め許可したものしか実行されない
  - 2.2.0〜（2021年リリース）
- 許可できるのは、PJのrootにある `composer.json`のみ
  - つまり、require/updateで入ったパッケージからは使えない

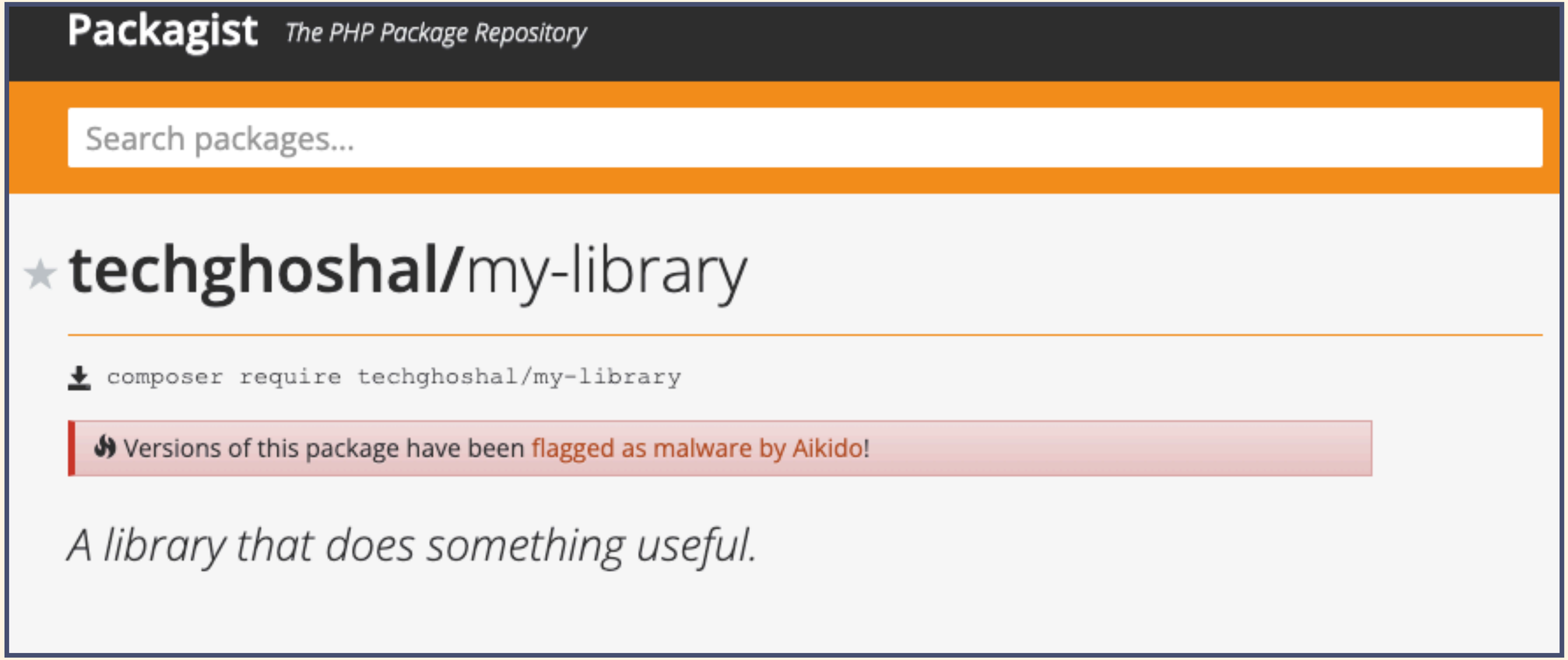


マルウェアフィルター

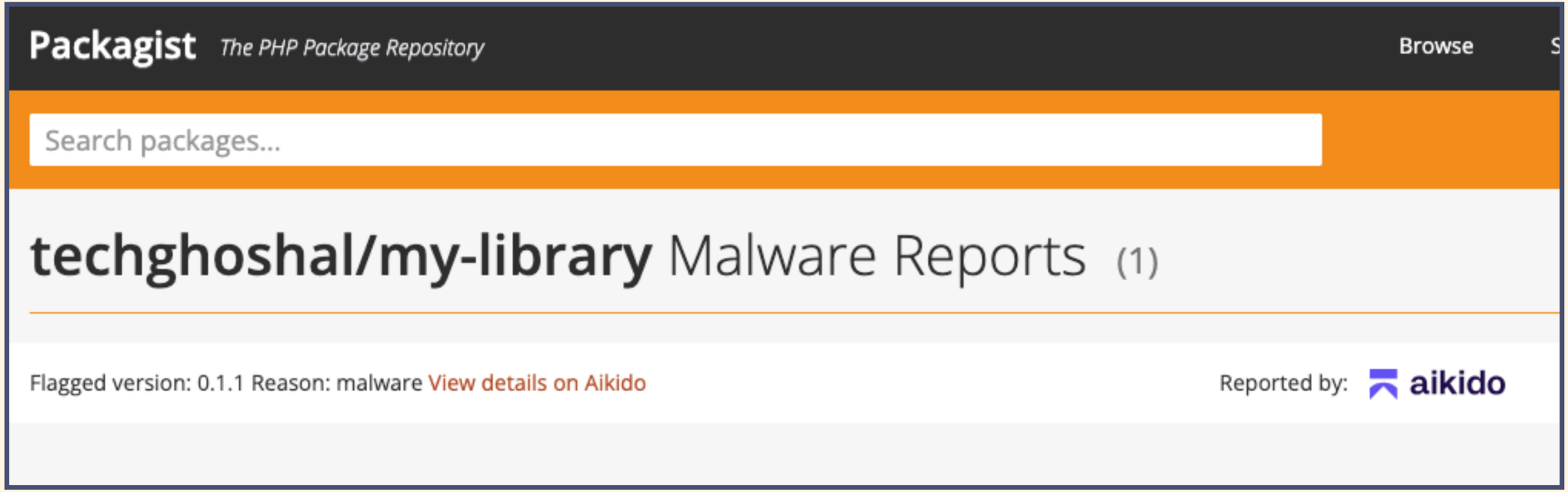
汚染されたバージョンのブロック

- https://github.com/composer/composer/pull/12766
- 次のバージョン(2.10)から
- Packagist側が、「汚染されたバージョン」フラグを提供する
  これをみて、危ないものが入りにくいようブロックする
- Aikido Securityによる提供

flagged as malware by Aikido



flagged as malware by Aikido



laravel-langはこんな感じ

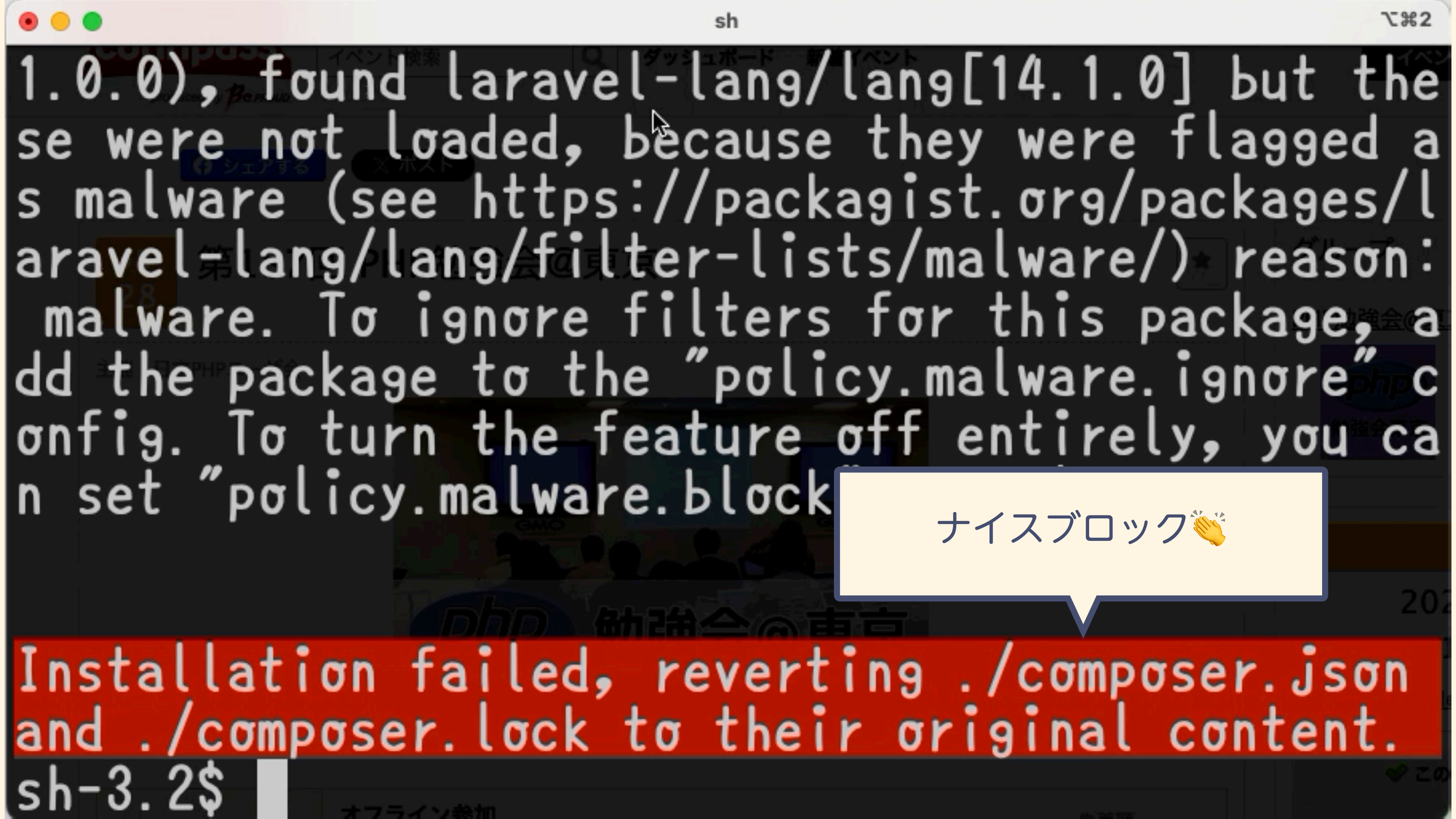
laravel-lang/lang Malware Reports (218)

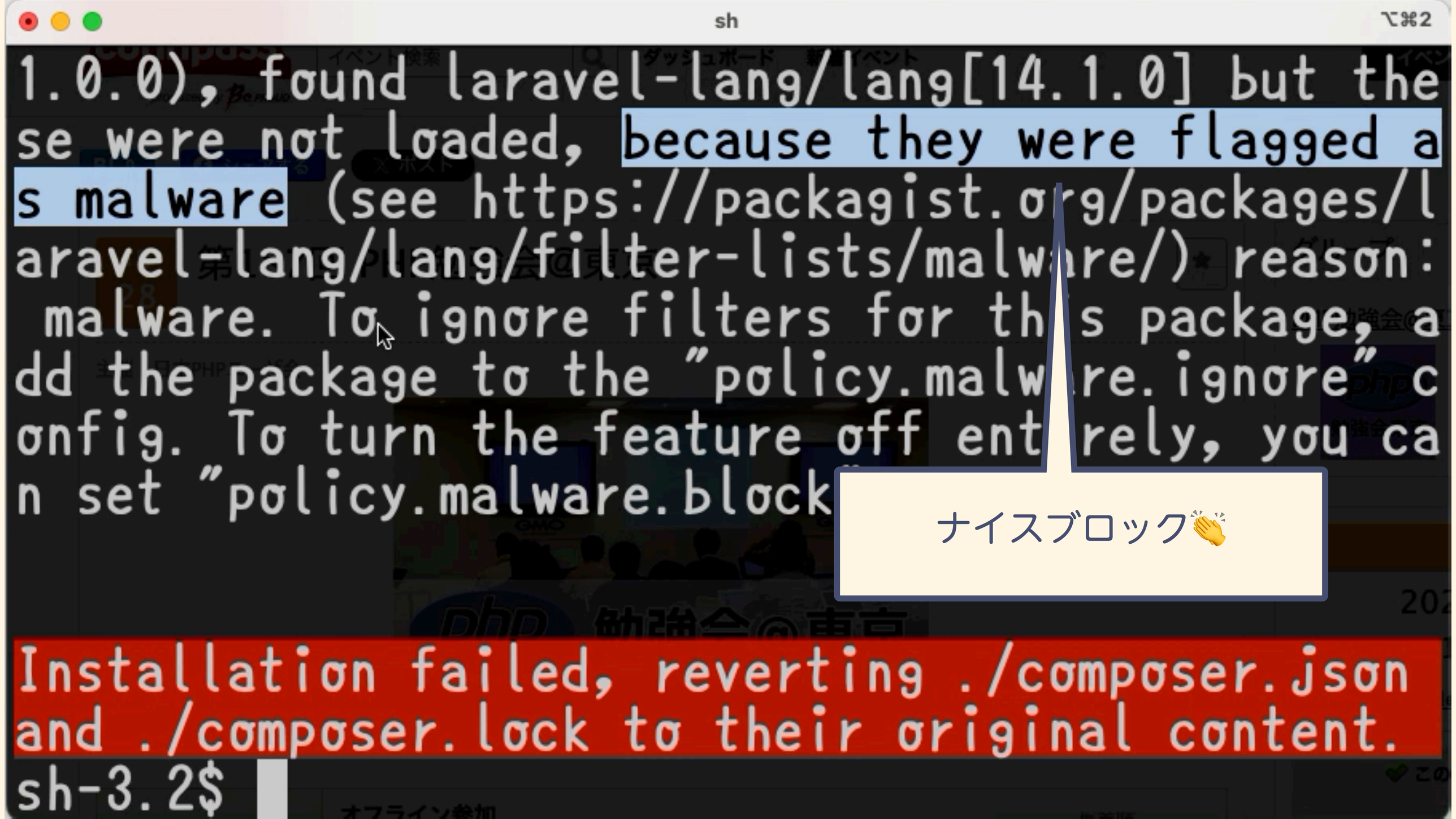
Flagged version: 1.0.2 Reason: malware View details on Aikido	Reported by: aikido
Flagged version: 10.0.0 Reason: malware View details on Aikido	Reported by: aikido
Flagged version: 10.0.1 Reason: malware View details on Aikido	Reported by: aikido
Flagged version: 10.0.2 Reason: malware View details on Aikido	Reported by: aikido
Flagged version: 10.1.0 Reason: malware View details on Aikido	Reported by: aikido
Flagged version: 10.1.1 Reason: malware View details on Aikido	Reported by: aikido

sh

⌥⌘2

```
sh-3.2$ ../composer-2.10.0-RC2.phar ¥
>     require laravel-lang/lang:14.1.0
```


sh
⌥⌘2
1.0.0), found laravel-lang/lang[14.1.0] but these were not loaded, because they were flagged as malware (see https://packagist.org/packages/laravel-lang/lang/filter-lists/malware/) reason: malware. To ignore filters for this package, add the package to the "policy.malware.ignore" config. To turn the feature off entirely, you can set "policy.malware.block
ナイスブロック👏
Installation failed, reverting ./composer.json and ./composer.lock to their original content.
sh-3.2$


sh
⌥⌘2
1.0.0), found laravel-lang/lang[14.1.0] but these were not loaded, because they were flagged as malware (see https://packagist.org/packages/laravel-lang/lang/filter-lists/malware/) reason: malware. To ignore filters for this package, add the package to the "policy.malware.ignore" config. To turn the feature off entirely, you can set "policy.malware.block
ナイスブロック👏
Installation failed, reverting ./composer.json and ./composer.lock to their original content.
sh-3.2$

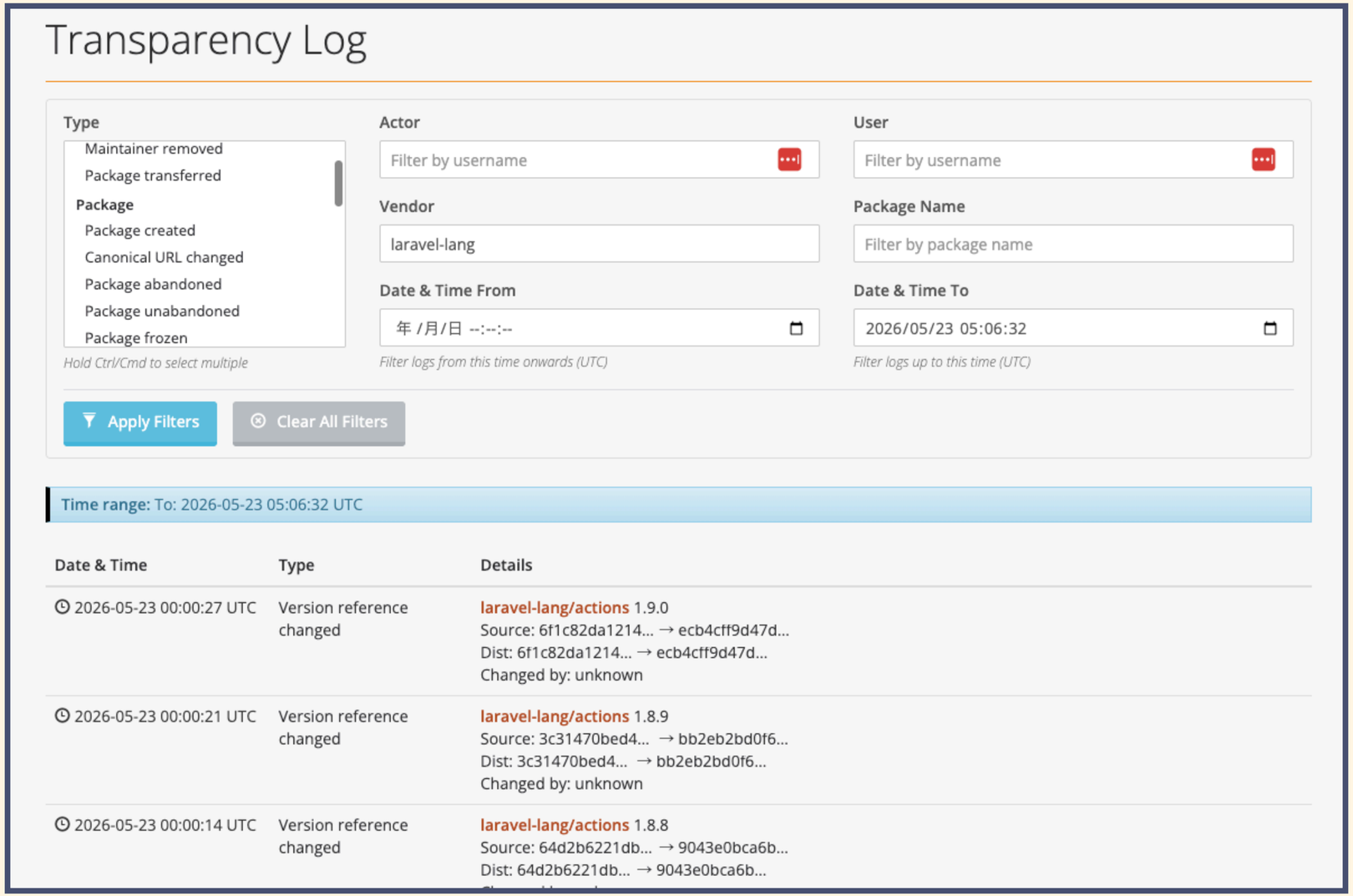
(安定版)バージョンの上書き不可に

リリースされたタグは上書きできなくなる

- Packagist側の修正
- 1度リリースされたバージョンは、
  内容が同一であるものと保証しやすくなる
- 5/28(JST)時点で、まだマージされていない
  - https://github.com/composer/packagist/pull/1742
  - ブログでは「in this week」のリリースと書かれている

その他の動き

Transparency Log



https://packagist.org/transparency-log?actor=&user=&vendor=laravel-lang&package=&datetime_from=&datetime_to=2026-05-23T05%3A06%3A32

Transparency Log

Date & Time	Type	Details
2026-05-23 00:00:27 UTC	Version reference changed	laravel-lang/actions 1.9.0 Source: 6f1c82da1214... → ecb4cff9d47d... Dist: 6f1c82da1214... → ecb4cff9d47d... Changed by: unknown
2026-05-23 00:00:21 UTC	Version reference changed	laravel-lang/actions 1.8.9 Source: 3c31470bed4... → bb2eb2bd0f6... Dist: 3c31470bed4... → bb2eb2bd0f6... Changed by: unknown
2026-05-23 00:00:14 UTC	Version reference changed	laravel-lang/actions 1.8.8 Source: 64d2b6221db... → 9043e0bca6b... Dist: 64d2b6221db... → 9043e0bca6b... Changed by: unknown
2026-05-23 00:00:07 UTC	Version reference changed	laravel-lang/actions 1.8.7 Source: f982195c759d... → 7ff13500af1b... Dist: f982195c759d... → 7ff13500af1b... Changed by: unknown
2026-05-22 23:59:59 UTC	Version reference changed	laravel-lang/actions 1.8.6 Source: 39d0617db1d... → 9108a615c6e...

https://packagist.org/transparency-log?actor=&user=&vendor=laravel-lang&package=&datetime_from=&datetime_to=2026-05-23T05%3A06%3A32

予定されているもの・関心が寄せられているもの

- (5/27) An Update on Composer & Packagist Supply Chain Security https://blog.packagist.com/an-update-on-composer-packagist-supply-chain-security/
- Coming in the next weeks and months:
  - Minimum-release-age / cooldown
- Longer-term direction:
  - Mandatory MFA across Packagist.org
  - Packagist.org hosting immutable build artifacts directly w/SLSA build provenance, Sigstore attestation

予定されているもの・関心が寄せられているもの

- (5/27) An Update on Composer & Packagist Supply Chain Security
  https://bl
  supply-cha
- Coming in
  - **Minimum-release-age / cooldown**
- Longer-term direction:
  - Mandatory MFA across Packagist.org
  - Packagist.org hosting immutable build artifacts directly w/SLSA build provenance, Sigstore attestation

補足:
現状、Packagistで配布するメタ情報にある`time`フィールドは、
パッケージの作者が任意に書き換え可能なので信頼しちゃ駄目

まとめ/今できること

「何が入っているか分からない」を防ぐ

- ちゃんと.lockファイルを使いましょう
- なるべく使うプラグインは減らしておいた方が良いかも
  - require-dev系でプラグインを使いたい時は、本番環境から隔離する
  - 本番には--no-dev インストールを使う
- vendor-binプラグインの活用 & 利用環境を分ける
- CIで`composer audit` を定期実行するのも

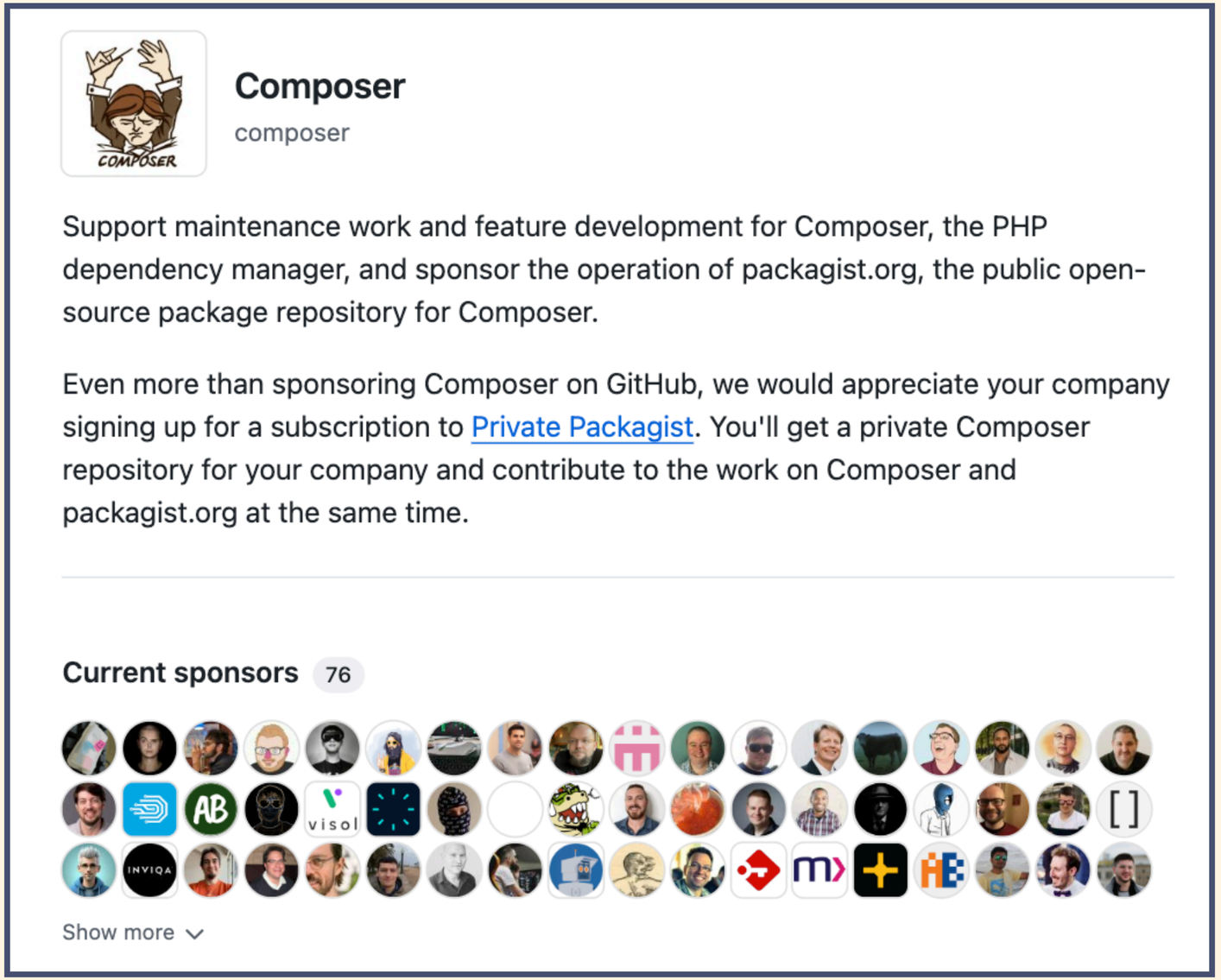
「何が入っているか分からない」を防ぐ

- Lockファイルの差分もPR単位で見るAction
  - https://github.com/marketplace/actions/composer-lock-diff
- ただ、今の時代なら、
  このくらいの軽量なものであれば自作しても良いかも
  - プラグインやGitHub Actionsを増やしまくるのに不安がある場合

Composer 2.10を待ちましょう、飛びつきましょう

- マルウェアブロック🙌
- 今週リリースらしい

ComposerとPackagistを支えよう



https://github.com/sponsors/composer

おしまい！

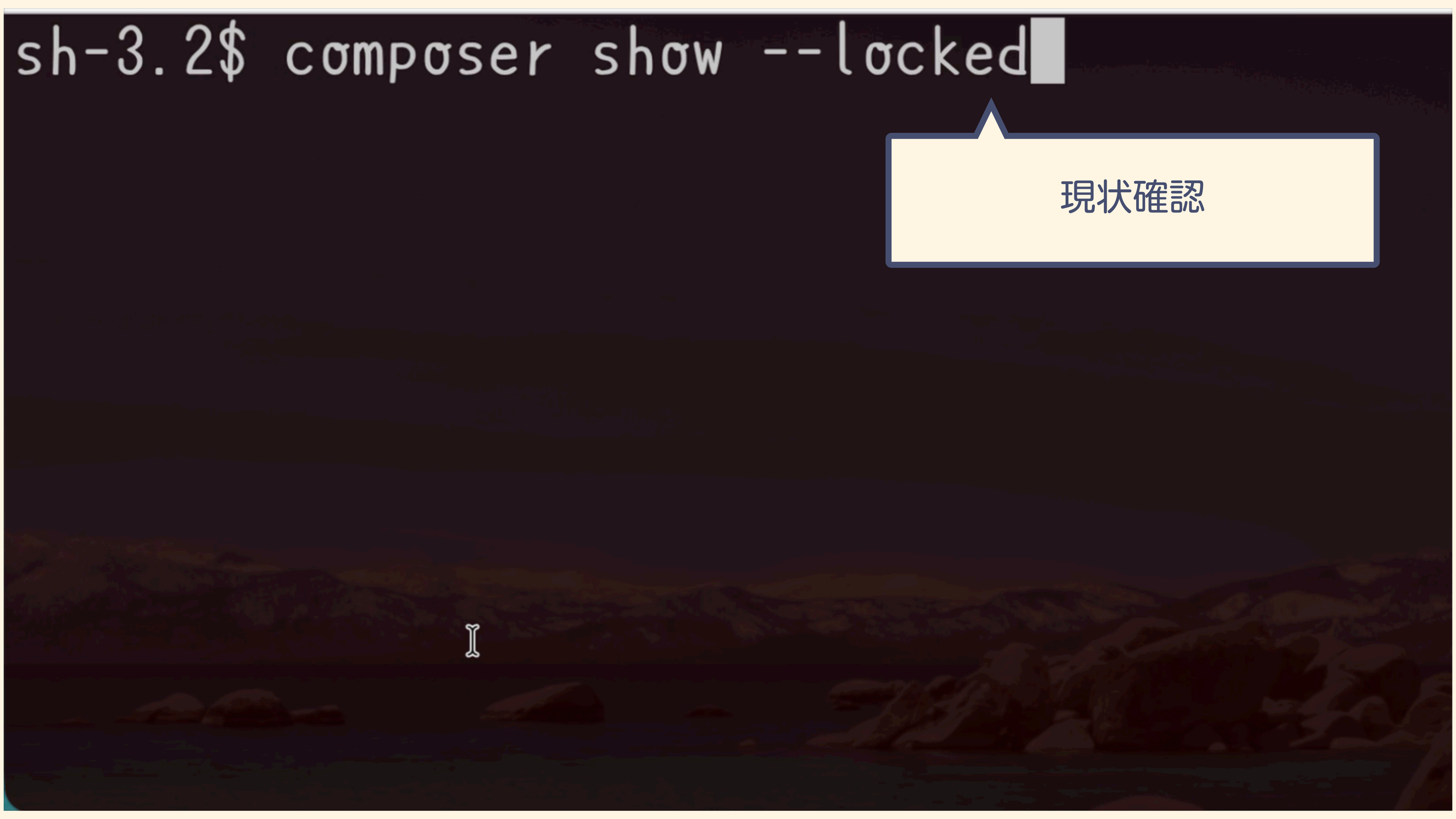
お付き合いいただき
ありがとうございました!!

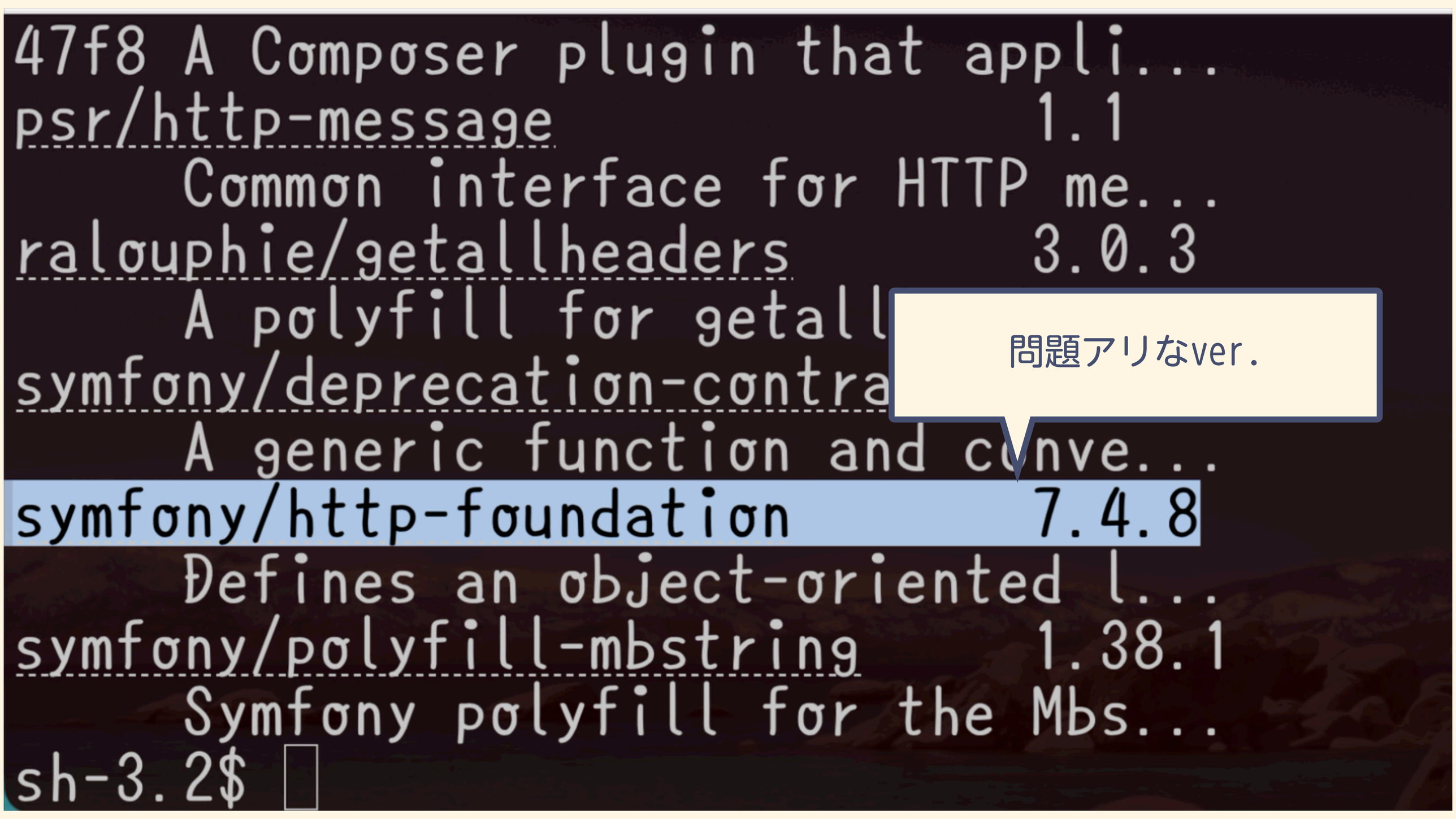
オマケ

auditfix的なコマンド

なに？

- npmとかには `audit fix` 的なコマンドがあるらしいと聞いた
- Composerでも、
  「パッチを当てる最小限の変更」ができたら良いのにねぇ


sh-3.2$ composer show --locked
現状確認


47f8 A Composer plugin that appli...
psr/http-message 1.1
Common interface for HTTP me...
ralouphie/getallheaders 3.0.3
A polyfill for getall
symfony/deprecation-contra
A generic function and conve...
symfony/http-foundation 7.4.8
Defines an object-oriented l...
symfony/polyfill-mbstring 1.38.1
Symfony polyfill for the Mbs...
sh-3.2$
問題アリなver.

```
sh-3.2$ composer audit
```

auditも見てみると

```
     Defines an object-oriented l...
symfony/polyfill-mbstring      1.38.1
     Symfony polyfill for the Mbs...
sh-3.2$
sh-3.2$ composer audit
Found 1 security vulnerability advisory af
fecting 1 package:
+-------------------+-----------------------
-----------------------------------------

------------------+
| Package          | symfony/http-foundat
ion
```

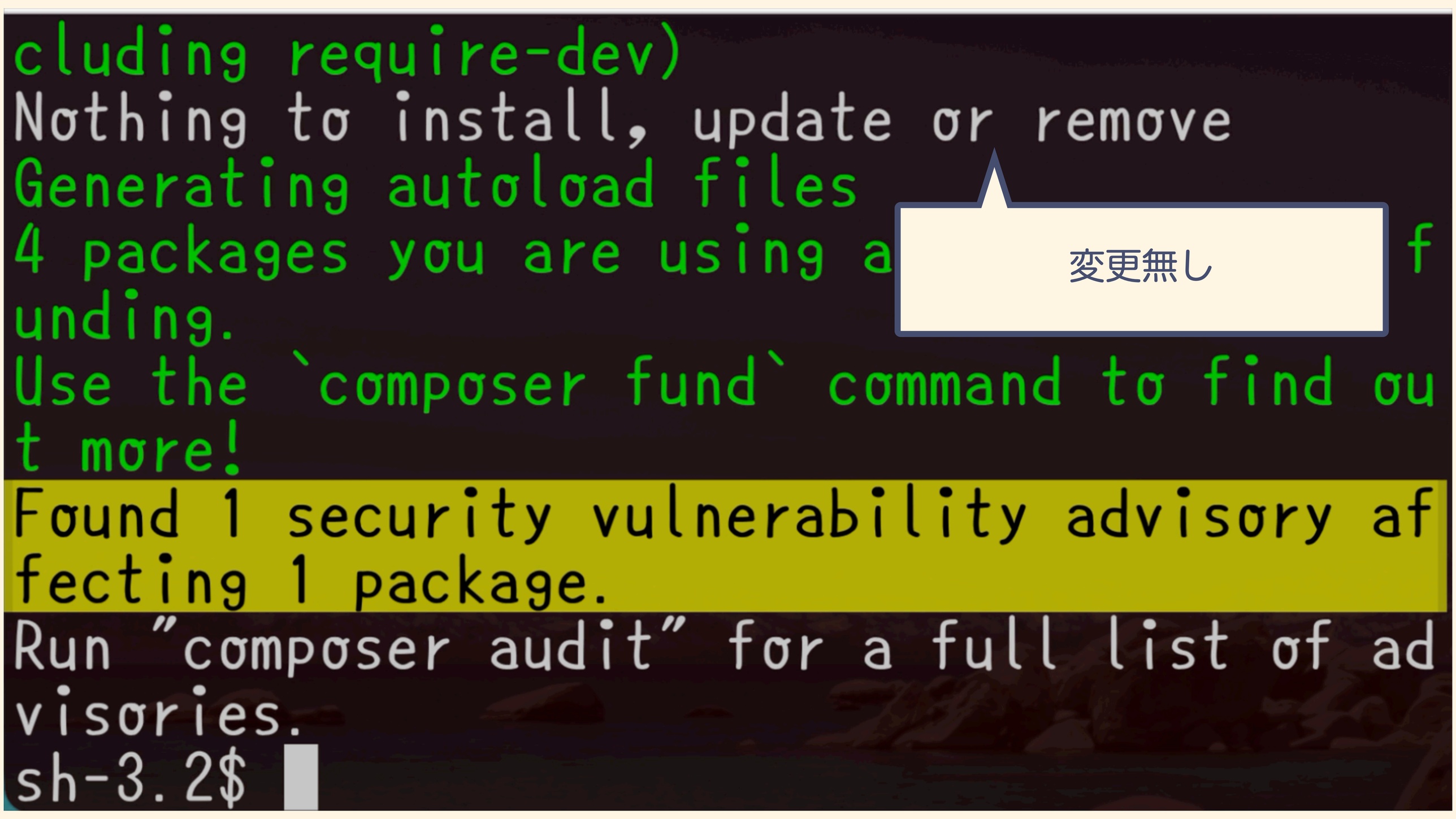
検出されている

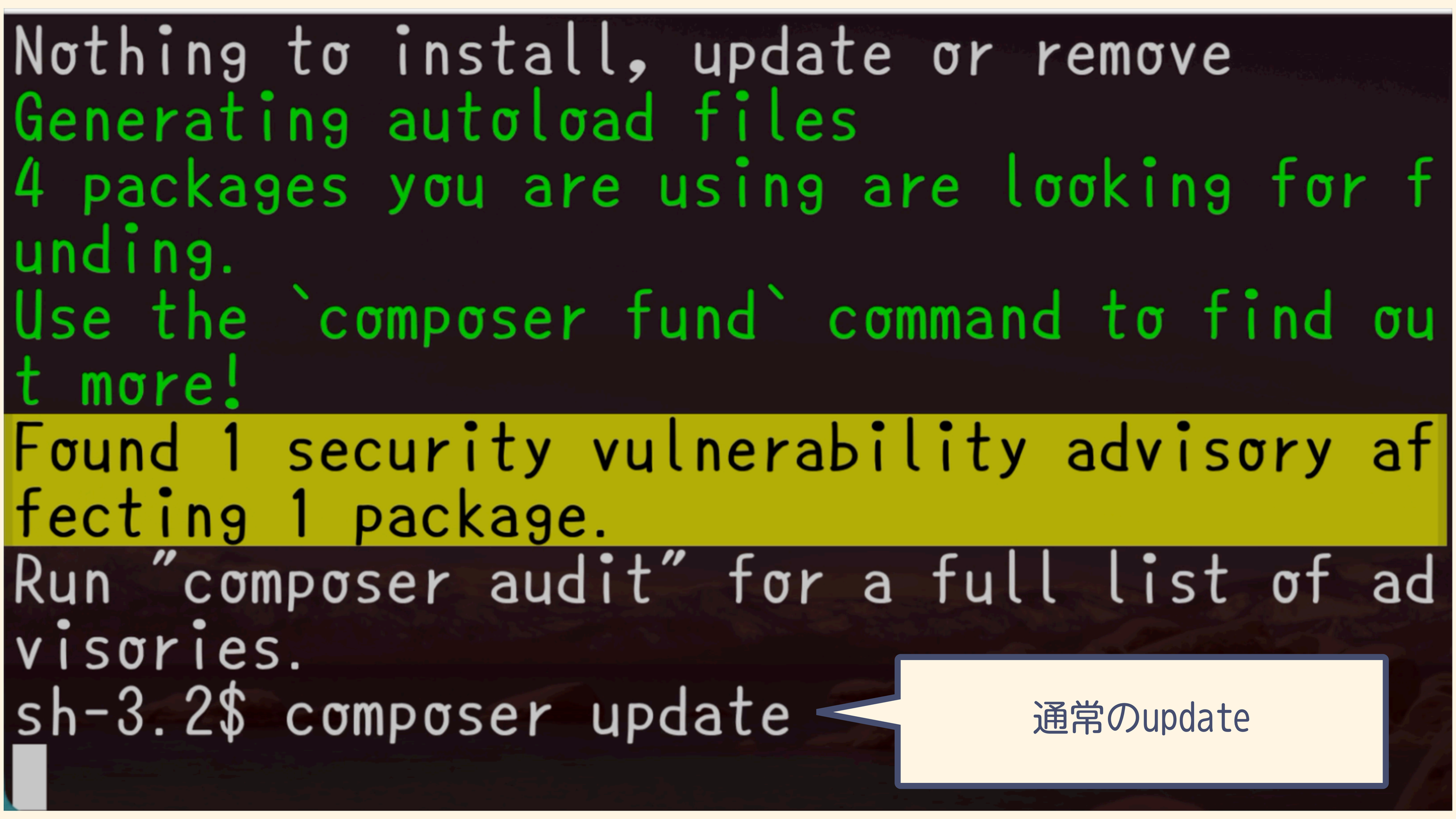

fecting 1 package:
+----------+----------

----------+
| Package | symfony/http-foundat
ion
| Severity |
| Advisory ID | PKSA-y6py-qpv1-h52p
検出されている

```
sh-3.2$ composer update ¥
>     --minimal-changes
```

minimal-changesだと


cluding require-dev)
Nothing to install, update or remove
Generating autoload files
4 packages you are using a
f
unding.
Use the `composer fund` command to find ou
t more!
Found 1 security vulnerability advisory af
fecting 1 package.
Run "composer audit" for a full list of ad
visories.
sh-3.2$
変更無し

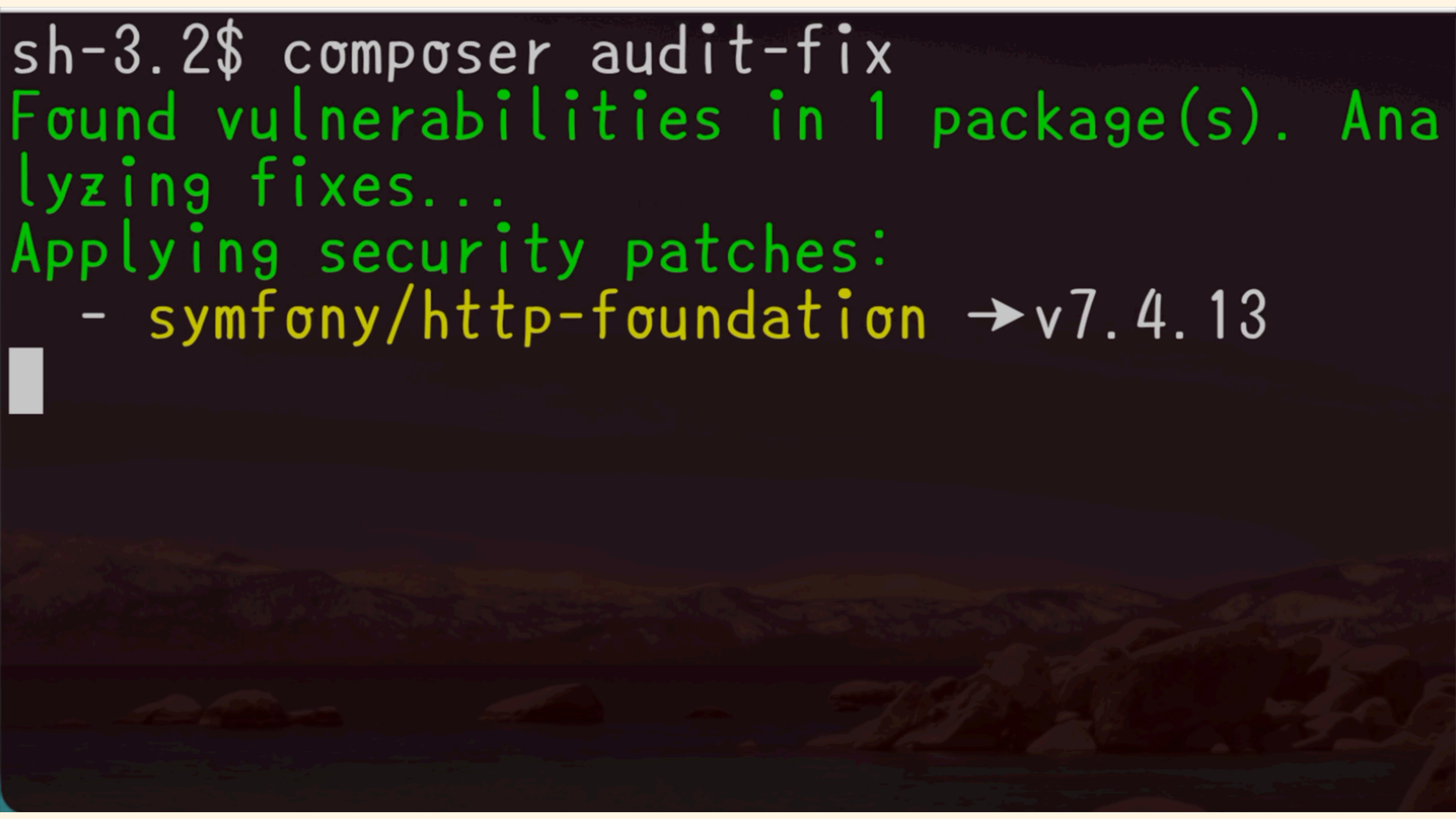

Nothing to install, update or remove
Generating autoload files
4 packages you are using are looking for f
unding.
Use the `composer fund` command to find ou
t more!
Found 1 security vulnerability advisory af
fecting 1 package.
Run "composer audit" for a full list of ad
visories.
sh-3.2$ composer update
通常のupdate


- Removing symfony/deprecation-contracts (v3.7.0)
- Upgrading symfony/http-foundation (v7.4.8 => v8.0.13): Extracting archive
Generating autoload files
3 packages you are using are looking for funding.
Use the `composer fund
t more!
No security vulnerability advisories found.
sh-3.2$
結構versionが変わった

コレをどうにかしたい

PoC的なものを作ってみた


sh-3.2$ composer audit-fix
audit-fix

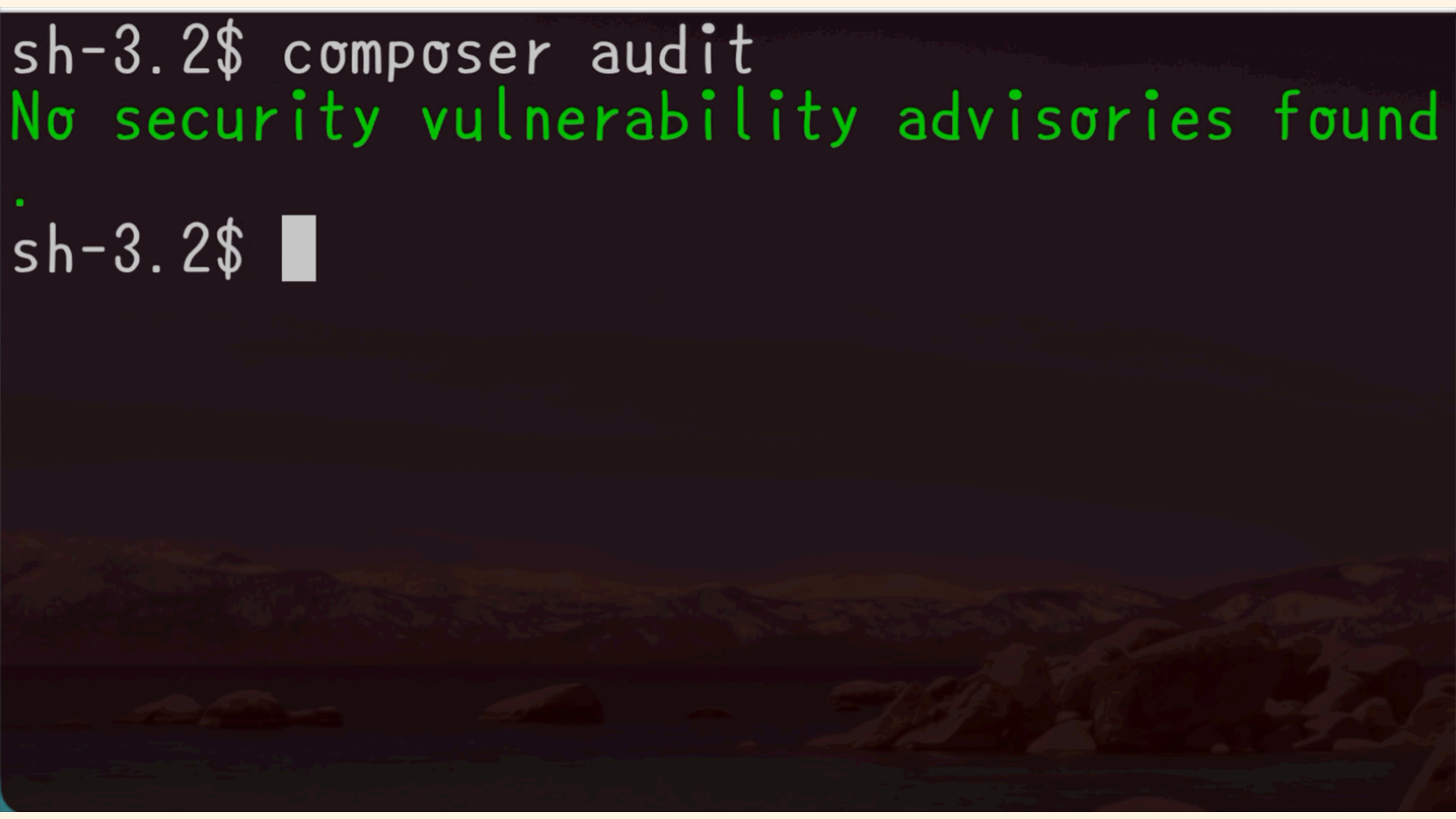

sh-3.2$ composer audit-fix
Found vulnerabilities in 1 package(s). Ana
lyzing fixes...
Applying security patches:
- symfony/http-foundation → v7.4.13

```
Package operations: 0 installs, 1 update,
0 removals
  - Upgrading symfony/http-foundation (v7.
4.8 => v7.4.13): Extracting archive
Generating autoload files
4 packages you are using are looking for f
unding.
Use the `composer fund` command to find ou
t more!
No security vulnerability advisories found
.
```

```
0 removals
  - Upgrading symfony/http-foundation (v7.
4.8 => v7.4.13): Extracting archive
Generating autoload files
4 packages you are using are looking for f
unding.
Use the `composer fund` command to find ou
t more!
No security vulnerabilit                    ound
.
sh-3.2$ composer show --locked
```

結果を見てみる

```
47f8 A Composer plugin that appli...
psr/http-message                    1.1
     Common interface for HTTP me...
ralouphie/getallheaders             3.0.3
     A polyfill for getallheaders.
symfony/deprecation-contracts 3.7.0
     A generic function and conve...
symfony/http-foundation             7.4.13
     Defines an object-oriented l...
symfony/polyfill-mbstring           1.38.1
     Symfony polyfill for the Mbs...
sh-3.2$
```


sh-3.2$ composer audit
No security vulnerability advisories found
.
sh-3.2$

参考資料とか

過去に出した資料

- Composerのメタ情報収集の流れについて
  - Composer 2.0って何？ どう変わるの？ 読んでみました！ (2020)
    https://speakerdeck.com/o0h/lets-read-composer2
  - 作って理解するComposer <クイックコース> (2024)
    https://zenn.dev/o0h/books/phpcon-2024-composer-ws
- プラグインの仕組みについて
  - 作って遊ぼう！Composer Plugin (2022)
    https://speakerdeck.com/o0h/phperkaigi-2022-composer-plugin-b

他のめっちゃ良い情報

- Jxckさんの記事。必読
  - サプライチェーン攻撃への防御策 | blog.jxck.io https://blog.jxck.io/entries/2025-09-20/mitigate-risk-of-oss-dependencies.html
- GMO Flatt Security CTO米内さんの記事。パッケージマネージャーの比較も
  - axios, LiteLLM...不使用だったのでOK、ではない。「次に備える」ソフトウェアサプライチェーン侵害への対策 - Speaker Deck https://speakerdeck.com/flatt_security/axios-litellm-dot-dot-dot-bu-shi-yong-datutanodeok-dehanai-ci-nibei-eru-sohutoueasapuraitienqin-hai-henodui-ce