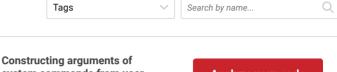


Empty statements should be removed



system commands from user input is security-sensitive

Analyze your code





Constructing arguments of system commands from user input is security-sensitive. It has led in the past to the following vulnerabilities:

- CVE-2016-9920
- CVE-2021-29472

Arguments of system commands are processed by the executed program. The arguments are usually used to configure and influence the behavior of the programs. Control over a single argument might be enough for an attacker to trigger dangerous features like executing arbitrary commands or writing files into specific directories

Ask Yourself Whether

- Malicious arguments can result in undesired behavior in the executed command.
- · Passing user input to a system command is not necessary.

There is a risk if you answered yes to any of those questions.

Recommended Secure Coding Practices

- · Avoid constructing system commands from user input when possible.
- Ensure that no risky arguments can be injected for the given program, e.g., typecast the argument to an integer.
- Use a more secure interface to communicate with other programs, e.g., the standard input stream (stdin).

Sensitive Code Example

Arguments like -delete or -exec for the find command can alter the expected behavior and result in vulnerabilities:

```
using System.Diagnostics:
Process p = new Process();
p.StartInfo.FileName = "/usr/bin/find";
p.StartInfo.ArgumentList.Add(input); // Sensitive
```

Compliant Solution

Use an allow-list to restrict the arguments to trusted values:

```
using System.Diagnostics;
Process p = new Process();
p.StartInfo.FileName = "/usr/bin/find";
if (allowed.Contains(input)) {
  p.StartInfo.ArgumentList.Add(input);
```

Code Smell Fields should not have public accessibility Code Smell URIs should not be hardcoded Code Smell

Types should be named in PascalCase

Code Smell

Track uses of "TODO" tags

Code Smell

See

- OWASP Top 10 2021 Category A3 Injection
- OWASP Top 10 2017 Category A1 Injection
- MITRE, CWE-88 Argument Injection or Modification
- SANS Top 25 Insecure Interaction Between Components
- CVE-2021-29472 PHP Supply Chain Attack on Composer

Available In:

sonarcloud sonarqube Developer Edition

© 2008-2022 SonarSource S.A., Switzerland. All content is copyright protected. SONAR, SONARSOURCE, SONARLINT, SONARQUBE and SONARCLOUD are trademarks of SonarSource S.A. All other trademarks and copyrights are the property of their respective owners. All rights are expressly reserved.

Privacy Policy