Content

Introduction ......................................................................................................................... 3
Description of the vulnerability ......................................................................................... 4
Addressing vulnerability ................................................................................................. 5
Sources ............................................................................................................................. 6
Introduction

OneDev is an all-in-one devops platform. It features git repository management with code intelligence, customizable issue management with boards, etc.

At the beginning of 2021, was discovered a vulnerability with which users can run arbitrary code. The vulnerability was discovered by GitHub user @pwntester.

The US National Vulnerability Database (NVD) rated the severity of this vulnerability with a rating of 8.8 on a 10-point scale. Also GitHub rated the severity with 9.6 points (critical).

Hereby give the University of Tartu permission to publicly display this essay until 2026. Käesolevaga luban Tartu Ülikoolil seda referaati avalikult eksponeerida kuni aastani 2026.

Aleksei Parm
Description of the vulnerability

The TextInput class is used for text parameters. There is method getPropertyDef():

```java
public static String getPropertyDef(InputSpec inputSpec, Map<String, Integer> indexes,
          @Nullable String pattern, DefaultValueProvider defaultValueProvider) {
```

A **pattern** can be controlled by the user.

The following pattern can be defined:

```java
foo") public String foo() {return "";} static {Runtime.getRuntime().exec("touch /tmp/pwned1");} //
```

The pattern is used in the following lines:

```java
if (pattern != null)
    buffer.append(" @Pattern(regexp="\" + pattern + "\", message="Should match regular expression: " + pattern + "\")\n\n```

And, when the pattern is used, it will result in:

```java
... @Pattern(regexp="foo") public String foo() {return "";} static {Runtime.getRuntime().exec("touch /tmp/pwned1");} //
public String input1() {
    ...
```

So, it is possible to run arbitrary code.

The vulnerability falls into category CWE-74 (Improper neutralization of special elements in Output Used by a Downstream Component ('Injection')).
Addressing vulnerability

Vulnerability was addressed in version 4.0.3 by escaping special characters from user input.

Fix - use existing in this project method `InputSpec.escape()` for pattern field.

TextInput.java: `getPropertyDef()`

```java
public static String getPropertyDef(InputSpec inputSpec, Map<String, Integer> indexes, String pattern, DefaultValueProvider defaultValueProvider) {
    pattern = InputSpec.escape(pattern);
    int index = indexes.get(inputSpec.getName());
    StringBuffer buffer = new StringBuffer();
    inputSpec.appendField(buffer, index, "String");
}
```

InputSpec.java: `escape()`

```java
public static String escape(String string) {
    String escaped = JavaEscape.escapeJava(string);
    // escape $ character since it has special meaning in groovy string
    escaped = escaped.replace("\$", "\\\$");

    return escaped;
}
```

Method `JavaEscape.escapeJava()` escapes java special characters (such as \b, \t, \" etc) and all non ASCII characters.
In addition to this, method `escape()` escapes $ character, because it has special meaning in Groovy.

This fix is enough, because there escaped the quote symbol.
Sources

1. https://github.com/theonedev/onedev/commit/39d95ab8122c5d9ed18e69dc024870cae08d2d60