jirutka / asciidoctor-include-ext Public

Code   Issues 1   Pull requests   Actions   Security 1   Insights

Fix command injection vulnerability

Browse files

```
Reported by Joern Schneeweisz from GitLab GmbH:

> During an internal assessment on some GitLab code I came across a way
> to execute arbitrary commands in your asciidoctor-include-ext Gem.
>
> The following adoc snippet demonstrates the issue:
>
> ```
> :app-name: |id # + \
> http://test.com
>
> include::{app-name}[]
> ```
>
> It uses a linebreak to bypass the `target_uri` check here
> https://github.com/jirutka/asciidoctor-include-ext/blob/master/lib/asciidoctor/include_ext/include_processor.rb#L97
> and feed a command with the `|` prefix to open/IO.foreach.
>
> You can verify this with the above snippet by rendering it like this
>
> ```
> asciidoctor -r asciidoctor-include-ext  -a allow-uri-read home.asciidoc
> ```

See-Also: https://sakurity.com/blog/2015/02/28/openuri.html
```

master
v0.4.0

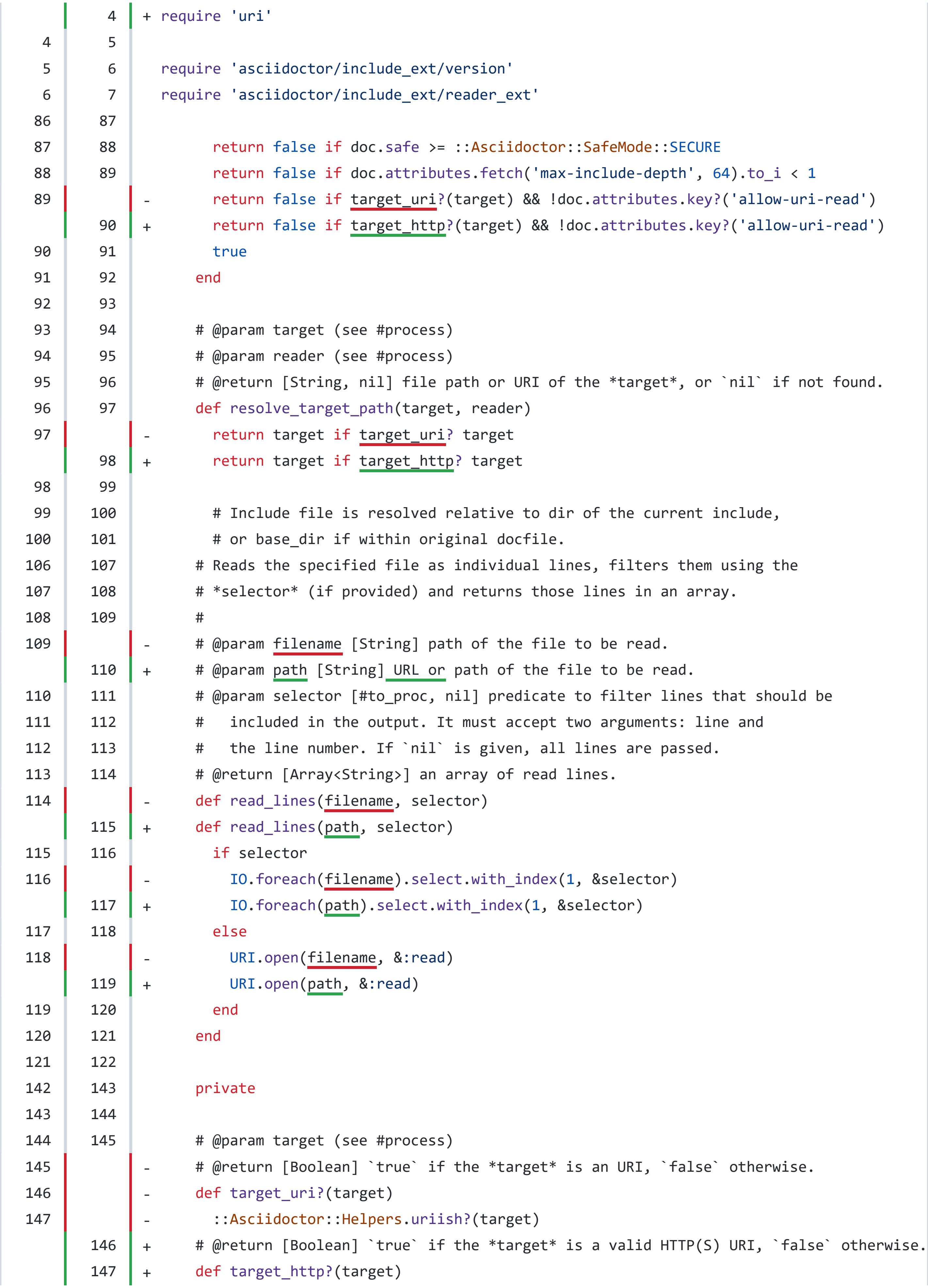
jirutka committed on Mar 291 parent 7227a60   commit c7ea001a597c7033575342c51483dab7b87ae155

Showing **2 changed files** with **28 additions** and **9 deletions**.

Split   Unified

23 🟩🟩🟩🟥⬜ lib/asciidoctor/include_ext/include_processor.rb

```
@@ -1,6 +1,7 @@
# frozen_string_literal: true
require 'logger'
require 'open-uri'
```



```diff
+ require 'uri'

  require 'asciidoctor/include_ext/version'
  require 'asciidoctor/include_ext/reader_ext'

        return false if doc.safe >= ::Asciidoctor::SafeMode::SECURE
        return false if doc.attributes.fetch('max-include-depth', 64).to_i < 1
-       return false if target_uri?(target) && !doc.attributes.key?('allow-uri-read')
+       return false if target_http?(target) && !doc.attributes.key?('allow-uri-read')
        true
      end

      # @param target (see #process)
      # @param reader (see #process)
      # @return [String, nil] file path or URI of the *target*, or `nil` if not found.
      def resolve_target_path(target, reader)
-       return target if target_uri? target
+       return target if target_http? target

        # Include file is resolved relative to dir of the current include,
        # or base_dir if within original docfile.
      # Reads the specified file as individual lines, filters them using the
      # *selector* (if provided) and returns those lines in an array.
      #
-     # @param filename [String] path of the file to be read.
+     # @param path [String] URL or path of the file to be read.
      # @param selector [#to_proc, nil] predicate to filter lines that should be
      #   included in the output. It must accept two arguments: line and
      #   the line number. If `nil` is given, all lines are passed.
      # @return [Array<String>] an array of read lines.
-     def read_lines(filename, selector)
+     def read_lines(path, selector)
        if selector
-         IO.foreach(filename).select.with_index(1, &selector)
+         IO.foreach(path).select.with_index(1, &selector)
        else
-         URI.open(filename, &:read)
+         URI.open(path, &:read)
        end
      end

      private

      # @param target (see #process)
-     # @return [Boolean] `true` if the *target* is an URI, `false` otherwise.
-     def target_uri?(target)
-       ::Asciidoctor::Helpers.uriish?(target)
+     # @return [Boolean] `true` if the *target* is a valid HTTP(S) URI, `false` otherwise.
+     def target_http?(target)
+    # First do a fast test, then try to parse it.
+    target.downcase.start_with?('http://', 'https://') \
+     && URI.parse(target).is_a?(URI::HTTP)
+   rescue URI::InvalidURIError
+    false
    end
   end
  end
```

14 🟩🟩🟩🟩🟩 spec/integration_spec.rb

```
     should match /let s = SS.empty;;/
     should_not match /(?:tag|end)::snippet\[\]/
    end
+
+   it 'does not allow execution of system command when allow-uri-read is set' do
+    options.merge!(attributes: { 'allow-uri-read' => '' })
+    given <<~ADOC
+     :app-name: |cat LICENSE # + \\
+     http://test.com
+
+     include::{app-name}[]
+    ADOC
+
+    should match /unresolved/i
+    should_not match /The MIT License/
+   end
+
   end
  
  
```

0 comments on commit c7ea001

Please sign in to comment.