THE BATTLE CARD

1. DETECTION (The "Smoke Test")

You suspect SSTI when user input is reflected on the page. You confirm it by injecting math.

- **The Polyglot Fuzzer:** `${{<%[%'"}}%\`
  - *Why:* Triggers syntax errors in almost every engine, often revealing the engine name in the stack trace.
- **The Math Test:**
  - Input: `{{7*7}}` or `${7*7}`
  - Reflected Output: `49` -> **VULNERABLE**
  - Reflected Output: `{{7*7}}` -> **Safe** (Treated as text)

---

2. CONTEXT & ENGINES (The "Adversaries")

Engine	Language	Signature / Indicator	Key Constraint	The Exploit / Bypass
Tornado	Python	`{{7*7}}` works. Tracebacks show `tornado`.	No `import` statements allowed.	**Dynamic Import:** `__import__('os').system('cmd')`
FreeMarker	Java	`${7*7}` works. Errors mention `freemarker`.	Sandbox blocks `?new()` on dangerous classes.	**Gadget Chain:** Access `product.class.protectionDomain.classLoader` to load `Execute` manually.
Handlebars	Node.js	`{{7*7}}` works. Errors mention `handlebars.js`.	"Logic-less" templates (no direct code).	**Helper Abuse:** Use `{{#with}}` blocks to walk the prototype chain and access `require('child_process')`.
Django	Python	`{{7*7}}` usually fails. `{% debug %}` works.	Tight sandbox. RCE is rare.	**Info Disclosure:** Use `{% debug %}` to find exposed objects (like `settings`) and read `SECRET_KEY`.

Engine	Language	Signature / Indicator	Key Constraint	The Exploit / Bypass
Twig / PHP	PHP	`{{7*7}}` works.	Hardened PHP environments block `system()`.	**Custom Gadgets:** Chain application logic (e.g., `user.setAvatar` + `user.delete`) to achieve file deletion.

---

3. THE KILL CHAIN (The Workflow)

1. **DETECT:** Fuzz inputs with math (`{{7*7}}`) to distinguish Text Injection from Template Injection.
2. **IDENTIFY:** Analyze error messages (stack traces) to pinpoint the Engine (Java/Python/Node) and Version.
3. **RESEARCH:**
   - **RTFM:** Read the docs for "Utility" classes or "Debug" tags.
   - **Context:** Are you in a text block (`<p>Here</p>`) or a code block (`render("User: " + input)`)?
4. **ESCAPE:**
   - **Syntax Escape:** `}}` to close existing tags.
   - **Sandbox Escape:** Use Reflection (Java), `__import__` (Python), or Prototype Pollution (Node) to break limits.
5. **EXECUTE:**
   - **Blind:** Use `sleep` or `ping` to verify execution.
   - **Visible:** Use `ls` or `cat` to retrieve flags.
   - **Destructive:** `rm` (Only as a last resort).
