**Testing for HTTP Parameter Pollution (HPP)** 🛡️

HTTP Parameter Pollution (HPP) happens when **multiple parameters with the same name** are passed in an HTTP request, and the server or application **fails to handle them properly**. Attackers exploit this to **manipulate logic, bypass filters, or inject payloads**.

**🔎 How to Test (Step by Step in Burp Suite / Browser)**

1. **Identify input parameters**
   * Query strings, form inputs, headers, cookies.  
     Example:
2. https://example.com/page?id=123
3. **Duplicate the parameter with different values**
4. https://example.com/page?id=123&id=456
   * Observe how the server handles it.
     + Some servers take the **first value**.
     + Some take the **last value**.
     + Some **merge both** (e.g., id=123,456).
5. **Inject a payload into one parameter**
6. https://example.com/page?id=123&id=<script>alert(1)</script>
   * If executed → possible **XSS via HPP**.
7. **Test hidden parameters (POST requests)**
   * Intercept with Burp Suite and **add duplicate fields** in body.  
     Example:
8. username=admin&username=test&password=123
9. **Check for bypasses**
   * Some WAFs or filters validate only the **first parameter**, but execution uses the **last one** → bypassing security.

**✅ Key Things to Verify**

* How does the application resolve multiple same-name parameters?
* Does HPP allow **authentication bypass**?
* Can it be used to **inject payloads** into DB/JS/HTML?

**⚡ Recommended Tools**

* **Burp Suite Intruder** → send multiple parameters with fuzzed payloads.
* **OWASP ZAP** → HPP scanning.

👉 Do you want me to now create **MCQs (with answers) for HTTP Parameter Pollution testing** in our fixed format (Q + 🟦 Answer)?