Perfect, let’s tackle **Guessable User Accounts** — this is a classic weakness in web apps and is included in WSTG testing.

**🧾 Guessable User Account**

**1. What It Means**

A **guessable account** is when an application uses **predictable usernames or credentials** for important accounts, often left over from:

* Default installs (e.g., admin/admin, guest/guest)
* Weak conventions (firstname.lastname, user1, user2)
* Vendor or test accounts (support, demo)

👉 Attackers can easily guess them without brute-forcing.

**2. Common Examples**

* **Default usernames**:
  + admin
  + root
  + superuser
* **Default passwords**:
  + password, 123456, welcome
  + admin123, changeme
* **Application-created accounts**:
  + WordPress: admin is created by default unless changed.
  + Database consoles: root often has no password in test setups.

**3. Security Risks**

* **Immediate unauthorized access** if default password is unchanged.
* **Privilege escalation** because these accounts often have elevated rights.
* **Account lockout evasion** since attackers already know valid usernames.

**4. How to Test (WSTG Approach)**

🔹 **Step 1 – Identify possible usernames**

* From error messages (user enumeration).
* From email patterns (jdoe@company.com → jdoe).
* From open services (/etc/passwd leaks, default creds lists).

🔹 **Step 2 – Attempt login with defaults**

* Tools:
  + Hydra
  + Burp Suite Intruder
  + Medusa
* Payloads: admin:admin, root:toor, guest:guest.

🔹 **Step 3 – Observe Responses**

* Error messages (“invalid password” vs. “user does not exist”) leak info.
* Time delays or HTTP status codes (200 vs. 403).

**5. Defenses Against Guessable Accounts**

✅ Disable or rename default accounts (never keep admin).  
✅ Enforce **account lockout** on failed attempts.  
✅ Use strong password policies.  
✅ Monitor login attempts (SIEM integration).  
✅ Remove test/demo accounts before deployment.

**6. Real Example**

* Many IoT devices ship with admin:admin.
* Mirai Botnet exploited this by scanning the internet and logging in with those defaults → creating one of the largest DDoS attacks ever.

✅ **Summary**:  
Guessable accounts = *low-hanging fruit for attackers*. Testing involves checking for **default usernames, predictable patterns, and weak passwords**. Defense means **removing defaults, enforcing strong authentication, and monitoring logins**.

👉 Do you want me to build a **Challenge Lab: Exploiting Guessable Accounts with Hydra/Burp Suite** so you can practice this in a controlled way?