🔍 Technology Fingerprinting

✂️ Check HTTP Response Headers (FIRST)

Goal: Identify server-side technologies and security posture.

How:

```
curl -I https://example.com
```

Look for:

- `Server: nginx / apache / cloudflare`
- `X-Powered-By: PHP / ASP.NET`
- Presence or absence of security headers

Why important:

- Outdated servers → known CVEs
- Missing headers → misconfiguration issues

✒️ Identify Web Technology / Framework

Goal: Detect CMS, backend framework, and frontend stack.

Find:

- CMS: WordPress, Joomla, Drupal
- Framework: Laravel, Django, Spring
- Frontend: React, Angular

How:

- Page source (`view-source`)
- Common paths: `/wp-admin` , `/wp-content`
- Cookies: `laravel_session` , `JSESSIONID`

Why:

- CMS + version → targeted vulnerability testing

🛏️ Detect Cloud, CDN, or WAF

Goal: Understand infrastructure protections and routing.

Look for:

- Cloudflare, AWS, Azure, GCP
- WAF behavior or blocking

Indicators:

- Headers like `cf-ray`
- IP belongs to CDN ranges
- Different behavior via IP vs domain

Why:

- CDN/WAF affects attack strategy
- Some bugs appear only behind WAF

🌂 Match Technology → Likely Bugs (MOST IMPORTANT)

Technology Found	What to Test
WordPress	Plugin vulnerabilities, file uploads
Laravel	Debug mode exposure, mass assignment
API backend	IDOR, authentication bypass
Old Apache	Path traversal
No WAF	Broader manual testing possible

⚠️ Fingerprinting is useless unless mapped to vulnerabilities.

🌡️ JavaScript & Client-Side Fingerprinting

Goal: Discover hidden endpoints, APIs, and logic exposed in client code.

Look for: - `.js` files (main.js, app.js, vendor.js) - Hardcoded API endpoints - Secrets, tokens, environment names

Why: - JS often leaks undocumented endpoints - Many IDOR and auth bugs start here

---

🎒 Error Handling & Debug Fingerprinting

Goal: Identify verbose errors and debug configurations.

Look for: - Stack traces - Framework error pages - JSON error responses

Why: - Errors reveal backend logic - Debug modes expose sensitive data

---

🎓 Authentication & Session Fingerprinting

Goal: Understand how auth and sessions are implemented.

Look for: - Cookie flags (`HttpOnly`, `Secure`, `SameSite`) - Token-based auth (JWT, OAuth) - Session rotation behavior

Why: - Weak auth leads to account takeover - Session issues lead to privilege escalation

---

🎩 File & Resource Exposure Fingerprinting

Goal: Identify exposed sensitive files.

Look for: - `.env`, `.git`, backups, logs - Open config files

Why: - Direct data exposure = high-impact bugs

🎽 Match Technology → Likely Bugs (MOST IMPORTANT)

Technology Found	What to Test
WordPress	Plugin vulnerabilities, file uploads
Laravel	Debug mode exposure, mass assignment
API backend	IDOR, authentication bypass
Old Apache	Path traversal
No WAF	Broader manual testing possible

⚠️ Fingerprinting is useless unless mapped to vulnerabilities.

📀 Why This Step Matters

- Without fingerprinting → random testing
- With fingerprinting → intelligent, targeted testing