HTTP Fundamentals

→ HTTP (Hypertext Transfer Protocol) is an application layer protocol for transmitting hypermedia documents, such as HTML.
→ Designed for communication between web browsers and web servers,
→ A client server model and stateless protocol (server doesn't keep any session data between two requests)
→ It is an application layer protocol that is sent over TCP or TLS encrypted TCP connection.
→ Proxies are numerous computers and machines between the web server and the web browser that relay the HTTP messages in the application layer.
Functions of proxies includes:-
> caching > filtering > load balancing > authentication > logging

Why HTTP being stateless causes auth vulnerabilities
> HTTP is stateless, so the server do not retain authentication state between requests & a result authentication relies on client-side data such as cookies

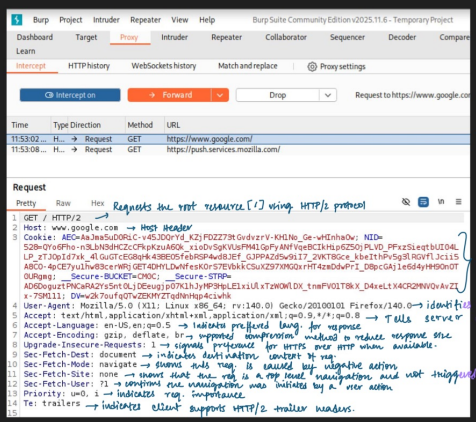
, session IDs or tokens being sent with every request. If those are stolen, reused or manipulated, attackers can impersonate users.

This statelessness enables vulnerabilities like session hijacking, replay attacks, CSRF & session fixation, making secure session management critical.

→ HTTP/1.1 introduced
> pipelining > persistent connections

→ HTTP header attack surface.

Header	Attacks
Host	Virtual Host routing attacks
Cookie	Session Hijacking
Authorization	Token abuse
Origin	CORS exploitation
Referer	CSRF bypass
Content-Type	Validation bypass
User-Agent	Logic flaws / bot detection bypass.
X-Forwarded-For	IP Spoofing



Cookie Header, sends stored browser cookies to maintain session state.

→ HTTP Methods:
> GET > POST > PUT > PATCH > DELETE > OPTIONS > HEAD

→ Below methods should exist on
> login endpoint — GET, POST
> user profile endpoint — GET, PUT, PATCH, POST

→ Some frameworks allow overriding HTTP methods using _method=DELETE and X-HTTP-Method-Override header.
This can be used by attackers to send a POST request but force backend to treat is as DELETE/PUT

→ Safe vs Unsafe HTTP methods

Safe	Unsafe
• Do not change server state	• Modify server state
• Used only for data retrieval	• Must have CSRF protection & auth checks
• GET, HEAD	• POST, PUT, PATCH, DELETE

→ Idempotent Vs Non-Idempotent methods

Idempotent	Non-Idempotent
• Multiple identical requests gives same result.	• Each request may cause new state.
• Safe to retry	• Replays are dangerous
• GET, PUT, DELETE	• POST

→ HTTP Status Codes

→ categories:
• 1xx - informational • 2xx — Success • 3xx — redirects • 4xx - client errors • 5xx — Server errors

→ Imp Codes meaning in security.
• 200 - often wide failure • 302 - Logic flaws • 401 - Missing auth • 403 - Authorization failure • 404 - fake protection • 500 - SQLi, RCE hints

→ 403 is better for security reasons as it:
• enforces explicit authorization checks.
• prevents logic flaws where missing auth.
• enables proper alerting for unauthorized access attempts
• Aligns with least privilege & OWASP guidance.