HTTP

🔷 What is an HTTP Request?

HTTP (HyperText Transfer Protocol) is how your browser communicates with websites. When you visit a site, your browser sends an **HTTP request** to the server, and the server responds with an **HTTP response** (like a webpage).

What is HTTP?

The Hypertext Transfer Protocol (HTTP) is designed to enable communications between clients and servers.

HTTP works as a request-response protocol between a client and server.

Example: A client (browser) sends an HTTP request to the server; then the server returns a response to the client. The response contains status information about the request and may also contain the requested content.

HTTP Methods

- **GET**
- **POST**
- **PUT**
- **HEAD**
- **DELETE**
- **PATCH**
- **OPTIONS**
- **CONNECT**
- **TRACE**

The two most common HTTP methods are: GET and POST.

The GET Method

GET is used to request data from a specified resource.

Note that the query string (name/value pairs) is sent in the URL of a GET request:

```
/test/demo_form.php?name1=value1&name2=value2
```

Some notes on GET requests:

- GET requests can be cached
- GET requests remain in the browser history
- GET requests can be bookmarked
- GET requests should never be used when dealing with sensitive data
- GET requests have length restrictions
- GET requests are only used to request data (not modify)

The POST Method

POST is used to send data to a server to create/update a resource.

The data sent to the server with POST is stored in the request body of the HTTP request:

```
POST /test/demo_form.php HTTP/1.1
```

```
Host: w3schools.com
```

```
name1=value1&name2=value2
```

Some notes on POST requests:

- POST requests are never cached
- POST requests do not remain in the browser history
- POST requests cannot be bookmarked
- POST requests have no restrictions on data length

Compare GET vs. POST

The following table compares the two HTTP methods: GET and POST.

	GET	POST
BACK button/Reload	Harmless	Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)

Bookmarked	Can be bookmarked	Cannot be bookmarked
Cached	Can be cached	Not cached
Encoding type	application/x-www-form-urlencoded	application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data
History	Parameters remain in browser history	Parameters are not saved in browser history
Restrictions on data length	Yes, when sending data, the GET method adds the data to the URL; and the length of a URL is limited (maximum URL length is 2048 characters)	No restrictions
Restrictions on data type	Only ASCII characters allowed	No restrictions. Binary data is also allowed
Security	GET is less secure compared to POST because data sent is part of the URL Never use GET when sending passwords or other sensitive information!	POST is a little safer than GET because the parameters are not stored in browser history or in web server logs
Visibility	Data is visible to everyone in the URL	Data is not displayed in the URL

The PUT Method

PUT is used to send data to a server to create/update a resource.

The difference between POST and PUT is that PUT requests are idempotent. That is, calling the same PUT request multiple times will always produce the same result. In contrast, calling a POST request repeatedly have side effects of creating the same resource multiple times.

The HEAD Method

HEAD is almost identical to GET, but without the response body.

In other words, if GET /users returns a list of users, then HEAD /users will make the same request but will not return the list of users.

A HEAD request is useful for checking what a GET request will return before actually making a GET request - a HEAD request can read the Content-Length header to check the size of the file, without actually downloading the file.

The DELETE Method

The DELETE method deletes the specified resource.

The PATCH Method

The PATCH method is used to apply partial modifications to a resource.

The OPTIONS Method

The OPTIONS method describes the communication options for the target resource.

The CONNECT Method

The CONNECT method is used to start a two-way communications (a tunnel) with the requested resource.

The TRACE Method

The TRACE method is used to perform a message loop-back test that tests the path for the target resource (useful for debugging purposes).

🔶 Basic Structure of an HTTP Request

Here's a simple example of a **GET request**:

makefile

```
GET /login HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0
Cookie: sessionid=abc123
```

Breakdown:

- GET /login → You're asking for the /login page.
- Host: → The domain you're sending the request to.

- User-Agent: → Info about the browser/client.
- Cookie: → Authentication/session data.

🔶 Common HTTP Methods

Method	What It Does	Example Use
GET	Retrieve data	View a webpage
POST	Send data	Submit a form
PUT	Update data	Edit user info
DELETE	Remove data	Delete account
OPTIONS	Ask what methods are allowed	Pre-flight request in CORS

🔶 Important Parts of HTTP for Bug Hunters

1. **Headers**
    - **Authorization:** May contain tokens (JWTs, API keys).
    - **Cookie:** Session identifiers—can be tested for session fixation.
    - **Content-Type:** Tells the server what format you're sending (e.g., application/json, application/x-www-form-urlencoded).
    - **User-Agent / Referer / Origin:** Sometimes useful in bypassing filters or crafting CSRF attacks.
2. **Body (only in POST/PUT/DELETE)**
   Example POST request:

   ```pgsql
   POST /login HTTP/1.1
   Host: example.com
   Content-Type: application/x-www-form-urlencoded

   username=admin&password=1234
   ```

🔶 Real-World Bug Hunting Use Cases

- **SQL Injection:** Tampering with GET/POST parameters.

```sql
GET /search?query=1' OR '1'='1
```

- **XSS (Cross-site Scripting):** Injecting scripts via GET/POST.

```javascript
POST /comment
comment=<script>alert(1)</script>
```

- **CSRF (Cross-Site Request Forgery):** Abusing authenticated requests via the victim's browser.
- **Broken Authentication:** Stealing cookies from Set-Cookie headers or JWT tokens from Authorization.
- **Directory Traversal / File Inclusion:**

```bash
GET /download?file=../../etc/passwd
```

🔶 Tools for Inspecting HTTP Requests

- **Burp Suite** (must-have for bug bounty)
- **Browser DevTools → Network tab**
- **Postman / cURL** for testing requests manually
- **OWASP ZAP**

✅ Summary for Bug Hunting

- Understand the structure of requests.
- Learn how parameters are passed (query, body, headers).
- Practice tampering with values.
- Use Burp Suite to intercept and modify HTTP requests.
- Watch for authentication, input validation, and misconfigurations.

HTML Error Messages

When a browser requests a service from a web server, an error might occur, and the server might return an error code like "404 Not Found".

It is common to name these errors HTML error messages.

But these messages are something called HTTP status messages. In fact, the server always returns a message for every request. The most common message is 200 OK.

Below is a list of HTTP status messages that might be returned:

1xx: Information

Message:	Description:
100 Continue	The server has received the request headers, and the client should proceed to send the request body
101 Switching Protocols	The requester has asked the server to switch protocols
103 Early Hints	Used with the Link header to allow the browser to start preloading resources while the server prepares a response

2xx: Successful

Message:	Description:
200 OK	The request is OK (this is the standard response for successful HTTP requests)
201 Created	The request has been fulfilled, and a new resource is created
202 Accepted	The request has been accepted for processing, but the processing has not been completed
203 Non-Authoritative Information	The request has been successfully processed, but is returning information that may be from another source
204 No Content	The request has been successfully processed, but is not returning any content
205 Reset Content	The request has been successfully processed, but is not returning any content, and requires that the requester reset the document view
206 Partial Content	The server is delivering only part of the resource due to a range header sent by the client

3xx: Redirection

Message:	Description:
300 Multiple Choices	A link list. The user can select a link and go to that location. Maximum five addresses
301 Moved Permanently	The requested page has moved to a new URL
302 Found	The requested page has moved temporarily to a new URL
303 See Other	The requested page can be found under a different URL

304 Not Modified	Indicates the requested page has not been modified since last requested
307 Temporary Redirect	The requested page has moved temporarily to a new URL
308 Permanent Redirect	The requested page has moved permanently to a new URL

4xx: Client Error

Message:	**Description:**
400 Bad Request	The request cannot be fulfilled due to bad syntax
401 Unauthorized	The request was a legal request, but the server is refusing to respond to it. For use when authentication is possible but has failed or not yet been provided
402 Payment Required	*Reserved for future use*
403 Forbidden	The request was a legal request, but the server is refusing to respond to it
404 Not Found	The requested page could not be found but may be available again in the future
405 Method Not Allowed	A request was made of a page using a request method not supported by that page
406 Not Acceptable	The server can only generate a response that is not accepted by the client
407 Proxy Authentication Required	The client must first authenticate itself with the proxy
408 Request Timeout	The server timed out waiting for the request
409 Conflict	The request could not be completed because of a conflict in the request
410 Gone	The requested page is no longer available
411 Length Required	The "Content-Length" is not defined. The server will not accept the request without it
412 Precondition Failed	The precondition given in the request evaluated to false by the server
413 Request Too Large	The server will not accept the request, because the request entity is too large
414 Request-URI Too Long	The server will not accept the request, because the URI is too long. Occurs when you convert a POST request to a GET request with a long query information
415 Unsupported Media Type	The server will not accept the request, because the media type is not supported

416 Range Not Satisfiable	The client has asked for a portion of the file, but the server cannot supply that portion
417 Expectation Failed	The server cannot meet the requirements of the Expect request-header field

5xx: Server Error

Message:	**Description:**
500 Internal Server Error	A generic error message, given when no more specific message is suitable
501 Not Implemented	The server either does not recognize the request method, or it lacks the ability to fulfill the request
502 Bad Gateway	The server was acting as a gateway or proxy and received an invalid response from the upstream server
503 Service Unavailable	The server is currently unavailable (overloaded or down)
504 Gateway Timeout	The server was acting as a gateway or proxy and did not receive a timely response from the upstream server
505 HTTP Version Not Supported	The server does not support the HTTP protocol version used in the request

511 Network Authentication Required	The client needs to authenticate to gain network access

HTTP Status Codes

When a browser request a service from a web service, a response code will be given.
These are the list of HTTP Status code that might be returned

1XX Information	
100	Continue
101	Switching Protocols
102	Processing
103	Early Hints
2XX Success	
200	OK
201	Created
202	Accepted
203	Non-Authoritative Information
205	Reset Content
206	Partial Content
207	Multi-Status (WebDAV)
208	Already Reported (WebDAV)
226	IM Used (HTTP Delta Encoding)
3XX Redirection	
300	Multiple Choices
301	Moved Permanently
302	Found
303	See Other
304	Not Modified
305	Use Proxy
306	Unused
307	Temporary Redirect
308	Permanent Redirect
4XX Client Error	
400	Bad Request
401	Unauthorized
402	Payment Required
403	Forbidden
404	Not Found
405	Method Not Allowed
406	Not Acceptable
Compiled by Ivan Tay.	

4XX Client (Continue)	
407	Proxy Authentication Required
408	Request Timeout
409	Conflict
410	Gone
411	Length Required
412	Precondition Failed
413	Payload Too Large
414	URI Too Large
415	Unsupported Media Type
416	Range Not Satisfiable
417	Exception Failed
418	I'm a teapot
421	Misdirected Request
422	Unprocessable Entity (WebDAV)
423	Locked (WebDAV)
424	Failed Dependency (WebDAV)
425	Too Early
426	Upgrade Required
428	Precondition Required
429	Too Many Requests
431	Request Header Fields Too Large
451	Unavailable for Legal Reasons
499	Client Closed Request
5XX Server Error Responses	
500	Internal Server Error
501	Not Implemented
502	Bad Gateway
503	Service Unavailable
504	Gateway Timeout
505	HTTP Version Not Supported
507	Insufficient Storage (WebDAV)
508	Loop Detected (WebDAV)
510	Not Extended
511	Network Authentication Required
599	Network Connect Timeout Error