**HTTP Request Methods**

Each time you visit a website, submit a form, or interact with a web application, your browser sends an HTTP request to the server. This request includes a method that specifies the intended action.

Below are the most commonly used HTTP request methods, along with explanations tailored for those just starting out:

**1. GET**  
The GET method is used to **retrieve data** from the server. It is the most common method used when accessing a web page. Any parameters (data) are passed in the URL using query strings (e.g., ?param=value).  
Example: Viewing a product page or searching using a keyword in the URL.

**2. POST**  
The POST method is used to **send data to the server**. This method supports a wide variety of data types, such as text, PDFs, images, or other binary data. Unlike GET, the data is sent in the request **body**, not the URL.  
Example: Submitting login credentials, uploading a document, or submitting a contact form.

**3. HEAD**  
HEAD is similar to GET, but it only returns the **headers** from the server, not the actual content (such as HTML or images). It is typically used to check if a resource exists or to determine its size before downloading.  
Example: Used by tools to verify whether a link is live without downloading the full file.

**4. PUT**  
The PUT method is used to **create or replace** a resource on the server. It sends data to the server, similar to POST, but is generally used in REST APIs to update existing data.  
Security Note: If a server accepts PUT requests without proper validation, an attacker could upload malicious files or overwrite critical data.

**5. DELETE**  
As the name suggests, DELETE is used to **remove a resource** from the server. It is typically used in REST APIs to delete database records.  
Security Note: Misconfigured DELETE endpoints can be exploited to remove important files or cause denial of service (DoS).

**6. OPTIONS**  
OPTIONS is used to retrieve information about the **communication options** available on the target server, such as which HTTP methods are supported. It is often used in CORS (Cross-Origin Resource Sharing) preflight requests.

**7. PATCH**  
PATCH is used to **partially update** an existing resource. Unlike PUT, which replaces the entire content, PATCH modifies only the specific fields provided in the request.

These methods define how clients interact with web servers and APIs. Insecure implementation of any of these methods can introduce vulnerabilities in web applications, making them potential targets in bug bounty hunting and penetration testing.

**HTTP Response Codes**

Once the server receives and processes a request, it returns a **response code**. This code informs the client about the status of the request—whether it succeeded, failed, or requires some form of correction.

HTTP response codes are divided into five categories:

**1xx – Informational Responses**  
These codes indicate that the request has been received and is being processed, but no final response is available yet. These are rarely seen in day-to-day browsing.

**2xx – Success**  
These codes indicate that the request was successfully received, understood, and processed.

* **200 OK**: The request succeeded, and the server returned the requested content.
* **204 No Content**: The request succeeded, but there is no content to return (often used after DELETE or PUT operations).

**3xx – Redirection**  
These codes indicate that further action needs to be taken by the client to complete the request.

* **301 Moved Permanently**: The requested resource has been moved to a new URL.
* **302 Found**: The requested resource is temporarily located at a different URL. Often used after login or redirect operations.

**4xx – Client Errors**  
These codes mean the client sent an invalid or incorrect request.

* **400 Bad Request**: The request was malformed or improperly formatted.
* **403 Forbidden**: The server understood the request but refuses to authorize it. This may occur when the user lacks permissions or when malicious activity is detected.
* **404 Not Found**: The requested resource does not exist on the server.

**5xx – Server Errors**  
These codes indicate that the server failed to fulfill a valid request.

* **500 Internal Server Error**: A generic error indicating that the server encountered an unexpected condition.
* **502 Bad Gateway / 503 Service Unavailable**: The server is either down or temporarily unable to handle the request.

**Why This Knowledge is Crucial in Cybersecurity**

Understanding HTTP methods and response codes is essential for anyone entering the field of cybersecurity, ethical hacking, or bug bounty hunting. Here’s why:

* It helps identify how data flows between the client and server.
* It allows you to test for vulnerabilities in input handling and resource management.
* It makes it easier to interpret the behavior of web applications during security testing.
* It helps you craft more effective payloads when testing for issues like IDOR (Insecure Direct Object Reference), broken access control, file upload flaws, and more.

If you're using browser developer tools or tools like Burp Suite or cURL, observe which HTTP methods are used and how the server responds. These insights are the starting point for mapping the attack surface of a web application.