HTTP

Http is the protocol that supports communication between web browsers and web servers.

Client Request form: Request Method, URL, Protocol version. Followed by MIMI-like message containing request modifiers, client information, and body content.

Server response: status line including the protocol version and a success error or error code. Followed by mine like message containing serve information and body content.

User Agent

Web client is user agent but not always. Spiders, crawlers, mobile phone.

Identifies application type, OS, language, software vendor or revision.

Can be spoofed.

Content type tells client what data to expect

Same Origin Policy

Allows cookies and JS to silently exist in the client without interfering with each other.

Origin Server: the server on which a given resource resides or is to be created. Server on which the HTTP response is generated. Disregards proxies, gateways. Permits running scripts on pages with same origin but prevents cross side scripting. Enforced by the browser.

Origin: domain name(host name). Application layer protocol and TCP port.

URI

Uniform Resource Identifier: Identify a name or resource on the internet.

Protocol://[user:password@]host.domain-name[:port]/resource[?param=value]

HTTP Verbs

- GET: Requests information, allows pages to be bookmarked. Not secured as the information is in address bar. Can only store name value pairs
- POST: requests a page but passes params in payload. Name value pairs submitted in body of the request. It is secured as name value pairs not displayed din location bar.
- HEAD: asks for response identical to the one that would correspond to a get request but without the response body. Useful for retrieving meta information.
- OPTIONS: Asks server for allowed methods.
- TRACE: echoes back the received request. Tells you what server got.
- CONNECT: converts request connection to TCPIP tunnel. Used for connection through proxy, used for SSL connections
- PUT: Uploads data in the body to the locstion. WebDAV for managing files on servers.
- DELTE: deletes resource specified

HTTP Status codes

3xx: redirection, 4xx: client error, 5xx: server error

Basic Authentication

Username and Password encoded in base64

Get request, auth required 401, get request with auth basic and encoded username and pass.

Digest Authentication

Hashes password using MD5

Get, 401, get with auth digest and salt and hashed user and pass.

Client server Auth

Certs on client and server. Encrypted.

Integrated Windows Auth

IIS and windows. Intranets.

Form based Auth

Client requests page, server response with form. Submits form, server processes. Auth happens.

Vulnerable to SQLi, XSS redirects

Session Management

HTTP sessions are stateless. Each event is unique. Maintaining state requires overhead.

Cookies

Simple session storage management. Designed to track status information about user visits. Set in http headers or Javascript. Consist of 2 headers, set and request. Key value pairs

Used for session management.

Structure

Name, Value, expiry date, domain, path.

Name is followed by attributes as pairs. Max age = lifetime, discard.

Http only prevents cookies form being read by client side scripts.

Secure. Only sent to SSL enabled site.

Recommended to use non-persistent cookies for session management. Stored in memory and not on disc.

Url based

Session token stored in URL and sent with get requests. Or on hidden form fields

Hidden form fields

Hidden form fields with information

Session Fixation

Authentication without invalidating any existing sessions. Allows hacker to steal auth session.

Fixed by invalidating sessions on login or changing session id

Hijacking sessions Credentials

Changing non-persistent cookies. Custom browser/app that lets you view a page and associated cookies. Use a proxy to change the cookie in the browser.

Defend by digitally signing, encrypting, large IDs so thy can't be brute forced or guessed. Set timeouts

Account Management Issues

Get valid usernames by trying to register them.

Use captchas, approval, and IP blocking. Enforce email addresses as usernames, send email to registered account to activate. Multi factor auth.

Deleting an account

Remove all info or replace with junk before deleting. Change to complex password.

Storing Passwords

Store encrypted. Hash and salt. Bcrypt, PBKDF2, HMAC. Don't limit length. Don't respond with bad password notification.

Command injection

Separate code and data. Sanitize inputs. Separate commands, interpolate strings in commands.

An attack in which the goal is execution of arbitrary commands on the host system.

Server Testing

Comments in server side code (debug). VPS shared hosting, port based virtual hosting.

Load Balancing

One site on many servers. Increases availability of the site and provides redundancy.

All should be same server and config but sometimes one gets patched when others don't.

Sticky sessions.

Uri analysis

- www.example.com, www2.exmaple.com or using a tool to identify all hostnames within a domain.
- Timestamp analysis (server farms in different places). Use the HTTP HEAD method to look at the date in the response header. See if it goes backwards or moves a large amount. Can also use the last modified field in the http response.
- SSL differences. Sometimes SSL config doesn't match across servers. Version may be different. Browse to site to test,
- Cookie analysis (cookie says what its used for) used to implement stickiness. Strings such as LB, Balance, load..
- HTML source differences. Sometimes apps have hidden fields to identify the server.
- TCP/IP. Flood them with ICMP. See if return IP differs or any differences in response. TTL and window size.

Pen Testing

Identify HTTP methods (should not support PUT DELETE, OPTIONS AND TRACE) – can use NetCat

Is PHP supported

Default pages. Sometimes default pages are left on server and can sometimes identify the server software

XSS

Causing a JavaScript popup usually suffices to demonstrate that a site is vulnerable because if alter works then window.open() will work. Open a window to attackers site and run all the scripts they want.

Scripts execute on the client

Reflective, stored and Dom based.

Reflected: get user to click on the link and it brings them to a website and what it sends to the web server (vulnerable) is reflected to the browser (the JavaScript) which is then executed. (sometimes called non-persistent).

Stored: Script in a comment on a forum. Executes on any client that visits it. Script stored on the servers db. (called persistent)

Dom Based: document.url, document.referrer. Payload is executed because of Dom manipulation. Only noticeable at runtime (in the client with page rendered). Example. Virus replacing ads in web pages.

Countermeasures

Sanitize variables, httponly cookie, new echo variables directly. Treat everything in the https request from the browser as untrusted

HTML entity encoding < &

OWASP

1. HTML escape the data before inserting data into the HML context
2. Attribute escape
3. JavaScript escape
4. Css escape
5. URL escape
6. Use httponly cookie flag