

Learning Objectives:

- HTTP Protocol
- Burp Suite Basics

Web Application Security:

- There are four fundamental aspects of web application security:
 - HTTP Protocols
 - Cookies
 - Sessions
 - Same Origin Policy

HTTP Protocol Basics:

- How does this support my pentesting career?
 - The ability to exploit web applications and find vulnerabilities in web servers and services.
 - Web applications technology is used market-wide also by desktop or mobile applications.
- **Hypertext Transfer protocol (HTTP)** – the most used application protocol on the internet.
 - It is the client-server protocol used to transfer web pages and web application data.
 - In HTTP, the client, usually a web browser, connects to a web server such as MS IIS or Apache HTTP Server. HTTP is also used under the hood by many mobile and modern applications.
- During an HTTP communication, the client and server exchange messages.
 - The client sends **requests** to the server and the server sends back **responses**.
 - HTTP works on top of TCP protocol.
 - First a TCP connection is established and then the client sends its requests and waits for a response. The server processes the request and sends back its answer, providing a status code and appropriate data.
- To end lines in HTTP, you have to use \r (carriage return) and the \n (newline) characters.
- Every header contains a request followed by some header fields. The format is as follows: Header-name: header value.

HTTP Requests:

This request has an empty body, as there is nothing after the two empty lines following the headers.

```
</>  
GET / HTTP/1.1  
Host: www.elearnsecurity.com  
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:31.0) Gecko/20100101 Firefox/31.0  
Iceweasel/31.2.0  
Accept: text/html  
Accept-Language: en-US,en;q=0.5  
Accept-Encoding: gzip, deflate  
Connection: keep-alive  
  

```

PTSv4: Section 1, Module 3 - Caendra Inc. © 2019 | p.17

HTTPS

- HTTP Secure (HTTPS), or HTTP over SSL/TLS, is a method to run HTTP which is a clear-text protocol over SSL/TLS, a cryptographic protocol.
- Why is HTTPS important?
 - o An attacker on the path cannot sniff the application layer communication.
 - o An attacker on the path cannot alter the application layer data.
 - o The client can tell the real identity of the server and, sometimes, vice-versa.
- A network user will be able to sniff the HTTP traffic but will not be able to read the HTTP Request headers, body, target domain, HTTP Response headers, or body.
- Network user may recognize:
 - o Target IP address.
 - o Target port
 - o DNS or similar protocols may disclose which domain user tries resolve.
- **IMPORTANT- HTTPS does not protect against web app flaws. The extra encryption layer only protects data exchanged between the client and the server. It does not protect from an attack against the application itself.**
 - o **Attacks such as XSS and SQL injections will still work.**

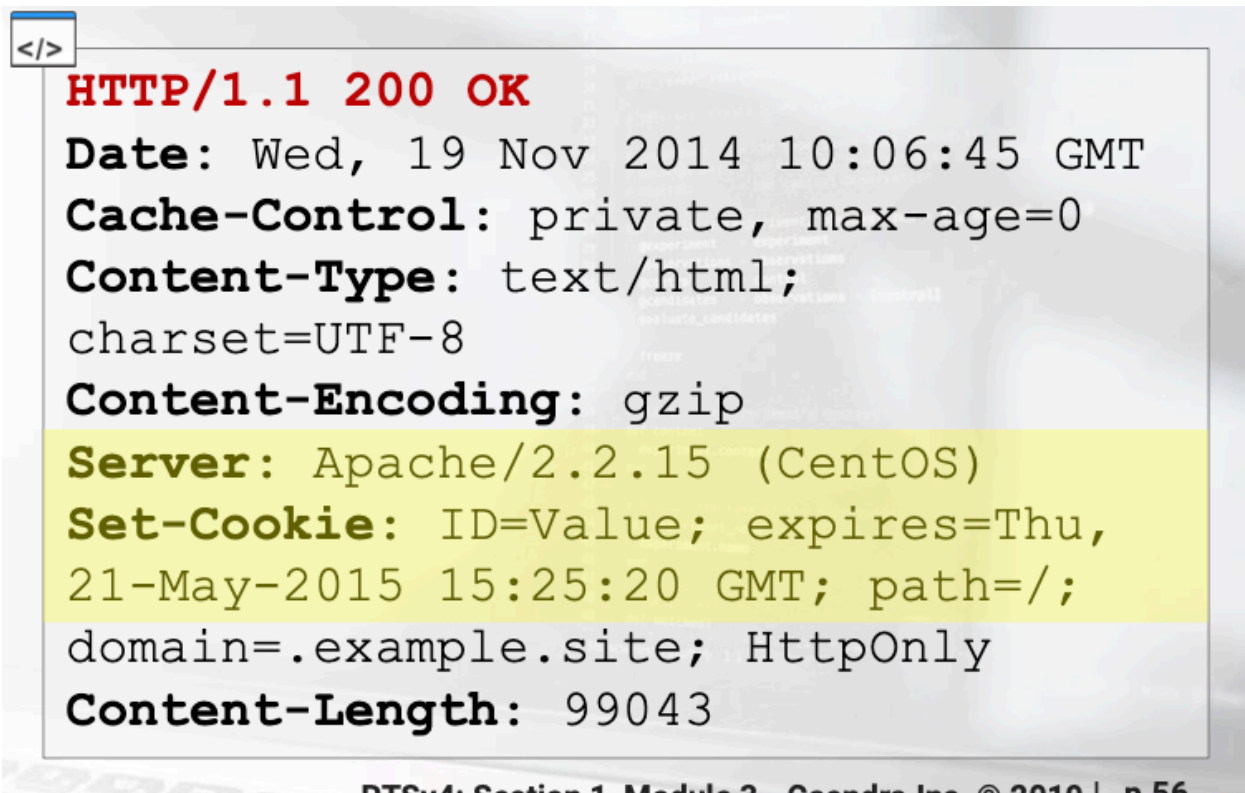
Cookies

- HTTP is a stateless protocol; this means that websites cannot keep the state of a visit across different HTTP requests.
 - o Every HTTP request is completely unrelated to the ones preceding and following it.
- Sessions and cookies were invented in 94'
- Netscape, invented cookies to make HTTP stateful.
- Cookies are the foundation of authorization of many applications

- Exploits often rely on stealing cookies.
- The cookie jar is the storage space where a web browser stores the cookies.

Cookies Format

- A server can set a cookie via the Set-Cookie HTTP header field in a response message. Cookies contain the following attributes:
 - The actual content
 - An expiration date
 - A path
 - The domain
 - Optional flags:
 - Http only flag
 - Secure flag.



```

HTTP/1.1 200 OK
Date: Wed, 19 Nov 2014 10:06:45 GMT
Cache-Control: private, max-age=0
Content-Type: text/html;
charset=UTF-8
Content-Encoding: gzip
Server: Apache/2.2.15 (CentOS)
Set-Cookie: ID=Value; expires=Thu,
21-May-2015 15:25:20 GMT; path=/;
domain=.example.site; HttpOnly
Content-Length: 99043
  
```

- **Cookie Handling**
 - Cookies are sent only to the valid domain/path when they are not expired and according to their flags.
- **Cookie Domain**
 - Domain and path field set the scope of the cookie.
 - When a cookie has the domain attribute set to:
 - Domain = elearnsecurity.com or domain = .elearnsecurity.com
 - The browser will send the cookie to:
 - www.elearnsecurity .com

- all subdomains of elearnsecurity.com
 - elearnsecurity.com
- If the server does not specify the domain attribute, the browser will automatically set the domain as the server domain and set the cookie host-only flag; this means that the cookie will be sent only to that precise hostname.
- **Cookie Path**
 - The path and the domain attributes set the scope of a cookie.
 - The browser will send a cookie to the right domain and to any sub-path of the path field value.

When a cookie has the path attribute set to:

- **path**=/the/path

The browser will send the cookie to the right domain and to the resources in:

- /the/path
- /the/path/sub
- /the/path/sub/sub/sub/path

But, it will not send it to /otherpath.

- **Cookie Expires Attribute**
 - The expires attribute sets the validity time window of a cookie.
 - A browser will not send an expired cookie to the server. Session cookies expire with the HTTP session.
- **Cookie HTTP-Only Attribute**
 - When the server installs a cookie into a client with the http-only attribute, the client will set the http-only flag for that cookie. This mechanism prevents JavaScript, Flash, Java, and any other non-HTML technology from reading the cookie, thus preventing cookie steal via XSS.
- **Cookie Secure Attribute**
 - Secure flag creates secure cookies that will only be sent over an HTTPS connection (they will NOT be sent over HTTP).
- **Cookie Content**
 - Can carry a number of values.
 - A server can set multiple values with a single Set-Cookie header by specifying multiple KEY=Value pairs.

Sessions

- Sessions are a mechanism that lets the website store variables specific for a given visit on the server side.

- Every user session is identified by a session id, or token, which the server assigns to the client.
- The client presents this ID for each subsequent request, thus being recognized by the server.
- The server retrieves the state of the client and all of its associated variables. The server stores Session IDS inside text files in its storage.

Session Cookies

- Websites running PHP install session cookies by using the “PHPSESSID” parameter, while JSP websites use “JSESSIONID”.
 - Every development language has its own default session parameter name.
 - Developers can also customize their own parameter names
- Servers install session cookies after a browser performs some kind of activity, like:
 - Opening a specific page
 - Changing settings in the web app
 - Logging in
- The browser can then use the cookie in subsequent requests.
 - Sessions can contain many variables, so sending small cookies can keep the bandwidth usage low.

Same Origin Policy (SOP)

- Critical web app security.
 - This policy prevents JavaScript code from getting or setting properties on a resource coming from a different origin.
 - In order to determine if JavaScript can access a resource: Hostname, port, and protocol must match.
- Example of what resources JavaScript has access to:

When a cookie has the path attribute set to:

- **path**=/the/path

The browser will send the cookie to the right domain and to the resources in:

- /the/path
- /the/path/sub
- /the/path/sub/sub/sub/path

But, it will not send it to /otherpath.

- Example of what JavaScript does not have access to:

- `https://www.elearnsecurity.com/path`
(same protocol and domain but different port)
- `http://www.elearnsecurity.com:345/path`
(same port and domain but different protocol)
- `https://www.heralab.net:345/path`
(same port and protocol but different domain)

- SOP applies only to the actual code of a script.
 - External resources can still be included such as HTML tags like `img`, `script`, `iframe`, `object`, etc.
- The entire Web application security is based on SOP.

/=====STOP=====

Burp Suite Notes are in "Burp Suite Notes" Document.

Notes:

- Web applications are applications that run on web servers and accessible via web browser (chrome, edge, etc.)
- Almost every website on the internet includes some kind of intelligence in its web page
 - Intelligence could either be on a client side or server side.
- The web app world is extremely heterogeneous. Which means there are many ways to accomplish the same task, but that also means there are many ways to mess things up.