Introduction to InfoSec – Recitation 8

Nir Krakowski (nirkrako at post.tau.ac.il)
Itamar Gilad (itamargi at post.tau.ac.il)

Today

- Web 101
  - HTTP
  - Cookies
  - HTML
  - PHP
  - SQL
- Web Vulnerabilities –
  - SQL Injection
- If we have time –
  - HeartBleed
-

HTTP

- Hyper Text Transfer Protocol
- Simple textual protocol over TCP port 80, **stateless** request-response model
- Requests –
  - [METHOD] [URI]\r\n
  - Headers\r\n\r\n

  e.g.: "GET /\r\n\r\n"
  - Headers –
    - Client type – User agent
    - Will the client support compression – Accept Encoding
    - Client language
    - Last valid cache the client has
    - ….

HTTP Response

- Responses –
  - [Numeric code] [String]\r\n
  - Headers\r\n\r\n
  - Data

  - Codes –
    - 200 OK
    - 302 Redirect
    - 404 Not found
    - 500 Server Error
    - 502 Gateway Error
    - ….

Misc HTTP

- Extra things to know about –
  - HTTP Keep alive
  - HTTP Authentication
  - X-forwarded-for (and X-we-are-hiring…)
- HTTPS is SSL / TLS transporting regular HTTP

Cookies

- A way for the server to store something in the client's browser for later use
- Cookies default to being domain specific
- Cookies have an expiry date
- Most authentication schemes use something like –
  - Client logs in via form
  - Server authenticates user, sends back an encrypted and hashed cookie, valid for x days
  - Client browses through the site with no need to re-login for a few days
- Cookies come in a few flavors – 'regular', 'HTTP only' and 'secure'

Cookies – cont.

- Cookies are a valuable commodity –
  - If I have your cookie – I AM YOU
  - Cookie stealing used to be very easy (simple Javascript), but now is pretty impossible (thank the SOP – Same Origin Policy)

HTML

- Hyper Text Markup Language
- XML format representation of the DOM (Document Object Model)
- The DOM is the tree-like structure of the document
- You may interact with and modify the DOM via Javascript
- The browser renders the objects within the document and allows the user to interact with them
- HTML5 is geared towards the dynamic web, and provides many services (local storage, 3d API, Async calls)
- CSS is used for design, HTML is used for structure

Basic HTML Example

```
<html>
    <head>
        <title>My title</title>
    </head>
    <body>
        <h1>Big letters!</h1> <br />
        <h6>Small letters!</h6>
    </body>
</html>
```

Forms and AJAX

- The two major ways to send user data as part of a web application are HTML forms and AJAX (Async Javascript And XML)
- HTML Forms –
  - <form action="/target.php" method="post" />
    - <input name="username" type="text" />
    - <input name="password" type="password" />
    - <input value="Submit" type="submit" />
  - </form>
  - Will send data as POST parameters to target.php upon clicking the submit button
- AJAX –
  - Read about JQuery and AsyncHttpRequest()

PHP

- Server-side processing language, commonly used in web applications
- Hybrid perl & C syntax
- Once the web server support processing PHP files, all that's needed is –

```
<?php
    echo "Hello, world!";
?>
```

PHP – cont.

- Headers are sent using header() (all calls to header() must be before sending data)
- Data is sent via echo / print() calls (or anything that writes to stdout)
- Input is done via HTTP parameters - $_POST["var_name"]
- The body of the request can be had via reading from stdin directly / file_get_contents(php://input) or stream_get_contents(STDIN)

A little more

```
<?php
 mysql_connect("your.hostaddress.com",
   "username", "password") or
    die(mysql_error());
 mysql_select_db("Database_Name") or
    die(mysql_error());
?>
```

PHP – Tips and Tricks

- Don't forget the ";" at the end of each line
- Use the "or die();" syntax to quickly find when your code breaks
- Errors are hard to spot. Your machines should have error reporting enabled, but don't expect much
- May need to revert to "printf debugging"
- You can run php on a file in a terminal, but understand you won't have the environment available (There could be better tools out there)

SQL

- Structured Query Language
- Very powerful interface to relational databases
- Tables have fields (columns) and rows
- Actions –
  - Select – query, return valid row(s)
  - Insert – Add new row(s)
  - Update – Change existing row(s)
  - Maintenance – Create table, Drop table, Add column….
- + Many more operations

SQL

- For each action, you can select which fields to choose by, and which fields to return
- Examples -
  - FROM users SELECT * WHERE username = “mitsi”
  - FROM users SELECT password WHERE username = “mitsi”
  - UPDATE users SET password=“123456” WHERE username=“mitsi”
  - INSERT INTO users VALUES (“myuser”, “mypass”)

PHP & SQL

- PHP has support for sql (MySQL in our case)
- You'll need to connect to the DB, and then you may query to your heart's content
- See the example within the exercise

SQL Vulenarbilities

- There are quite a few, but SQL Injection is #1
- Example –

*statement = "SELECT * FROM users WHERE name = '" + userName + "';“*

- The attacker controls userName, and assuming there is no input sanitation, the attacker can set userName to be

  ' or '1'='1

- Will lead to the query always returning valid data

Adaptations

*statement = "SELECT * FROM users WHERE name = '" + userName + "';“*

- Blocking the rest of the query

' or '1'='1' -- ‘

' or '1'='1' ({ '

' or '1'='1' / ‘*

- Not really limited to the Web – can be done with RFID food tags, dog tags – anywhere someone is querying a DB without proper input sanitation

Further reading & Tools

- W3Cschools.com, codecademy.com & php.net have everything you need to know
- Also, Google ☺

- **Firefox Developer tools**
- Fiddler really helps when you want to research an existing site
- XML verifiers / code beautifiers

Questions?

HeartBleed 101

- In SSL/TLS, a "Heart-beat" packet is used to keep the connection alive / know when the connection has dropped
- Works like 'ping' – will echo sent data (built-in length field)
- OpenSSL is a very common SSL/TLS implementation (~66% of HTTPS servers on the Internet)
- OpenSSL allowed a peer to send a heart-beat packet while controlling the length field
- Attacker can send a small packet with a large length value → Attacker gets back a bigger answer, consisting of server memory

Implications

- Reading (dynamic) server memory
- Which may contain sensitive information
- And key material!

- If exploited, all your security are belong to us! (can decrypt / MITM that site's traffic)

Mitigation

- Proper Solution –
  - Update to a fixed version of OpenSSL
  - Generate new certificates
  - Change all passwords, re-check anything that happened in the mean time
  - Never sleep well again
- Reality –
  - Most of the Internet has been patched very quickly (<24hours)
  - Not all certificates have been / will be replaced
  - Most sites have not urged users to change passwords
  - Most users won't do it anyway…

Questions?

- More info at hearbleed.com