Legal Informatics, Privacy and Cyber Crime

Part One: Web Application Security

Sandro Etalle

02/05/2018

2017/2018
Study material: Book Targeted Cyber Attacks

- Available (also in electronic format) at Elsevier.com and other bookstores
- Ch 1, 2, 3
- Ch 4: only 4.1, 4.2, 4.5
- Skip Chapters 5 & 6
- Ch 7 is facultative
- Ch 8
Other Material

  – (look for my home page – forgive me the old picture – and find the link “Material for the MDTM Master (Bologna Business School)”)

• Also, and interesting read
• Internet Security Threat Report
• Available at Symantec.com
  – (requires signing-up)
About this part of the course

- Topic: Web Security
- Underlying material: Slides only
Disclaimer

• Any actions and or activities related to the material contained in these slides is solely your responsibility. The misuse of the information presented here can result in criminal charges brought against the persons in question.

• The author(s) and the TU/e do not promote, encourage and or endorse ANY illegal activities. The author(s) and the TU/e will not be held responsible in the event any criminal charges be brought against any individuals misusing such information to break the law.
You need to get your hands a bit dirty

• **Please install**
  • Something to tamper with the http request, such as liveHTTPheaders or TamperData.
  • liveHTTPheaders and TamperData do not work with the last version of Firefox. As alternative you can use Chrome with TamperChrome (a bit more complicated to use, but the result is the same, there is also a video tutorial).
  • 2018: What now works with my firefox is the add-on HTTP header live.
  • Install Firefox, look for a plug-in for “headers” and install it.
The trouble with web applications

• Until a few years ago attacks Used to be on the Operating Systems

• Now it is easier to attack the (web) applications
  • Either for a direct attack (steal information)
  • Or an indirect attack (attack a user using the web application)

• Let’s take a look at OWASP
Owasp.org

• If you have to develop any webapplication, you should look here for information how to avoid security problems
  • Look at the cheat sheets.
• Interesting for us is the OWASP top 10
  • There is a tentative 2017 version
OWASP Top Ten (2013 Edition)

A1: Injection
A2: Broken Authentication and Session Management
A3: Cross-Site Scripting (XSS)
A4: Insecure Direct Object References
A5: Security Misconfiguration
A6: Sensitive Data Exposure
A7: Missing Function Level Access Control
A8: Cross Site Request Forgery (CSRF)
A9: Using Known Vulnerable Components
A10: Unvalidated Redirects and Forwards

credit to owasp.org
Live - Asscher: 'Dit kabinet plaatst multinationals boven mensen'

Binnenland Het regeerakkoord van VVD, CDA, D66 en ChristenUnie is openbaar. De partijleiders gaven zojuist elk een korte toelichting op het akkoord. Daarbij hebb…

Willem Vissers: Kijk vanavond nog één keer naar Arjen Robben, met een traan in de ooghoek

Opinie

MEEST GELEZEN
1. 30 jaar na Tsjernobyl ramp duiken opeens radioactieve
2. VERoordeelde zedendelinquent
3. Verdachte vermissings Anne
4. Onderzoekers verzwegen dood ‘wondercompatiënt’
5. Medicijn taalijzmuziek niet vergoed wegens

Direct aan de slag en lekker bijverdienen?

krantenbezorgmen.nl

BEKIJK HELE LIJST

— ADVERTENTIE
What happens when you go to nu.nl?

• **Your browser (Internet Explorer, Firefox, Safari, Opera) will**
  • Display content coming from nu.nl
  • Display content coming from external sites (*tens of sometimes hundreds of them*)
    - So the browser will also transmit some information to these external sites
  • Display aggregate content from a number of advertisers

• Carry out some operations
  - Some of them determined by NU.nl
  - Some of them determined by some external sites
  - Input to these operations is partly determined by them
Simple Exercise

• **Easy**
  - Start using TamperChrome (or if you have an old version of Firefox TamperData) and take a look at all the [http request](http://www.nu.nl) that are originated by your browser when you go to [www.nu.nl](http://www.nu.nl)
  
  - login Facebook

• **For the advanced user: install a proxy**
  - Start burpproxy
    - under Kali: applications > web application analysis > burpsuite
    - make sure your browser is connected to the right proxy
    - typically: localhost:8081 (see the options in burpsuite)
Result

- This is a small fraction of the requests that are generate when you go to nu.nl
- Notice that they are mostly external sites
- ....
- (this is using tamperdata and clicking on submit every time, tamperchrome and livehttpheaders have a different interface)
This is what it is

- Each box is a software component and it has vulnerabilities

- And third party content is particularly untrusted
Let’s play a bit
TamperData allows you to (and alter) an HTTP request ...

Obligatory exercise: activate “tamper data” and log in a site you know.
- In most cases the password is sent in clear (can be intercepted – unless https is uses)
- notice that you are able to modify the parameters
An HTTP request contains **parameters**

- **The usual ones**
  - Host
  - Referer. “The **HTTP referer** (originally a misspelling of **referrer**\(^1\)) is an **HTTP header field** that identifies the address of the webpage (i.e. the **URI** or **IRI**) that linked to the resource being requested. By checking the referer, the new webpage can see where the request originated
  - Cookies

- **The unusual ones**
  - username,
  - password

- **Important to understand:** there exist two kind of parameters: **GET** and **POST** parameters.
Get parameter by example

- This is a page taken from the level 1 challenge on SQL injections.
- Type “ouch” in the “search” (Zoeken) field, press enter and you see....
Get parameter by example (2)

- That the parameter of the search is passed in the URL
- This is thus a GET method parameter
- Notice that you see something “different” in the URL also when you click on one of the buttons.
POST method by example (1)

- The old Facebook example,
- When you try to login you don’t see your login name and password in the url of facebook,
The parameters are now “hidden” in the request: they are thus POST parameters.
You need to use temperdata to see them.
What are the consequences of this difference security-wise?

![Tamper Popup](image)
### Some differences (from w3schools.com)

<table>
<thead>
<tr>
<th>GET</th>
<th>POST</th>
</tr>
</thead>
<tbody>
<tr>
<td>BACK button/Reload</td>
<td>Data will be re-submitted (the browser should alert the user that the data are about to be re-submitted)</td>
</tr>
<tr>
<td>Bookmarked</td>
<td>Cannot be bookmarked</td>
</tr>
<tr>
<td>Cached</td>
<td>Not cached</td>
</tr>
<tr>
<td>Encoding type</td>
<td>application/x-www-form-urlencoded or multipart/form-data. Use multipart encoding for binary data</td>
</tr>
<tr>
<td>History</td>
<td>Parameters are not saved in browser history</td>
</tr>
<tr>
<td>Restrictions on data length</td>
<td>No restrictions</td>
</tr>
<tr>
<td>Restrictions on data type</td>
<td>No restrictions. Binary data is also allowed</td>
</tr>
<tr>
<td>Security</td>
<td>POST is a little safer than GET because the parameters are not stored in browser history or in web server logs</td>
</tr>
<tr>
<td>Visibility</td>
<td>Data is not displayed in the URL</td>
</tr>
</tbody>
</table>

- **GET parameter:** visible in the URL
  - You can bookmark it
  - Less secure: (e.g. when you only need to craft a link your victim has to click on)

- **POST parameter:** not visible in the URL
  - But sometimes the web application accepts transforming a post into a get parameter

---

Never use GET when sending passwords or other sensitive information!
The response contains interesting information:
- the status of the request.
- parameters that may be reused.
Nb: some parameter are encoded

• This is to make the request more digestible by the webserver, webapplication etc.
• Very often requests are encoded,
• Very often the webapplication accepts several different encodings
  • Hexadecimal encoding: 0x41 = A, etc
  • URL encodings
    – %3d =
    – %25 %
    – %20 space
    – %0a new line
    – %00 null byte
HTTP is connectionless

In particular it is sessionless, and does not have a “state”

- Small exception: HTTP keep-alive, or HTTP connection reuse in which you reuse the same TCP connection (Connection: Keep-Alive), this is however often unlinked to the application logic, so as far as we are concerned there is no connection state.

So, think about a web shop, in which you change “state” by putting things in the car etc.

The webapplication has no memory of what you have done so far.

Question: How can a server (dealing with 100s of connection at the same time)

- “know” which requests comes from you and which ones come from someone else?
- Know the status of request, and what you have done earlier? Think about cheking out at a webshop.
HTTP “weak” Sessions (2)

- All the information needed to process your request
  - Proof of who you are,
  - Information about what you have done so far (elements you have in the basket)
- IS STORED IN YOUR BROWSER

nu.nl

webapplication

webserver

internet

Browser

nu.nl

webapplication

webserver

internet

Browser

HTTP "weak" Sessions (2)

nu.nl

webapplication

webserver

internet

Browser

nu.nl

webapplication

webserver

internet

Browser

HTTP "weak" Sessions (2)

nu.nl

webapplication

webserver

internet

Browser

nu.nl

webapplication

webserver

internet

Browser

HTTP "weak" Sessions (2)

nu.nl

webapplication

webserver

internet

Browser

nu.nl

webapplication

webserver

internet

Browser

HTTP "weak" Sessions (2)

nu.nl

webapplication

webserver

internet

Browser

nu.nl

webapplication

webserver

internet

Browser

HTTP "weak" Sessions (2)

nu.nl

webapplication

webserver

internet

Browser

nu.nl

webapplication

webserver

internet

Browser

HTTP "weak" Sessions (2)

nu.nl

webapplication

webserver

internet

Browser

nu.nl

webapplication

webserver

internet

Browser
So, why don’t we use SSL and thereby “solve” all our problems?

- SSL DOES NOT HELP against the attacks we just mentioned!
- Actually, we should talk about TLS “Transport Layer Security”
- Where does SSL help?
Where SSL acts

- SSL takes care of securing the communication (TCP/IP).
- E.g. eavesdropping if you are at an internet café
  - Or at home using Wifi & WEP
  - Or WPA
- The security problems we mentioned before have nothing to do with the application vulnerabilities.
- Also: there are other ways of intercepting traffic.
Cookies
Cookies in a nutshell (1)

- The HTTP answer (from the webapplication to the browser), may contain cookies that set a value in the browser
  - Eg. The response coming from amazon.com may contain
    - Set-Cookie: SessId=191041-1042
    - Set-Cookie: UID=1042
    - Set-Cookie: DiscountAgreed=25
- The next time we visit amazon.com the browser will include in the header the following information
  - Cookie: SessId=191041-1042
  - Cookie: UID=1042
  - Cookie: DiscountAgreed=25
- Nb: you can change this with temperdata
How cookies work

• Each cookie has a domain and a path
• It will be submitted to each request in which
  • The domain is a subdomain of the cookie domain
  • The path is a subdirectory of the cookie path
• Domain and path can be determined by the server in the so-called cookie policy
  • Set-Cookie: LSID=DQAAA…; Domain=docs.foo.com; Path=/accounts
  • Provided that foo.com is the actual domain (or a subdomain) of the http request, otherwise the cookie will not be accepted for security reasons.
• Some cookies have an expiration moment
  • Some last only for the duration of the session
  • Some have an expiration date (e.g. June 15, 2025)
Third-party cookie and privacy

• What happens when you change these settings?
• Think about it a second.
Cookies and tracking

• We have seen that an HTTP request to nu.nl actually generates requests to other sites as well: google.com, amazon.com, advertiser.nl.
  • These sites (let’s call them collateral sites) will not be able to see the cookies set by nu.nl, but they will receive the cookies they have set in our browsers, even in previous sessions

• So what happens when we block “third party” cookies?
  • That these collateral sites will not be able to set new cookies in the browser, but they are still able to read their own (older) cookies. So tracking can take place anyhow.
Supercookies

- Like cookies, but associated to a first-level domain
  - E.g. .nl rather than nu.nl
- They are used also when the browser is in "private mode"
- Are not eliminated when you clean the cookies from your browser
- Allow sites to track you even better.
A very effective tracking method

Search

Default Search Engine
Choose the default search engine to use in the address bar and search bar.

- Google

- Provide search suggestions
- Show search suggestions in address bar results
A very effective tracking method

- **Search suggestions**
  - Don’t lie: it is probably “on” in your browser
- **Exercise: turn on tamper data, and then type**
  [www.yourfavoritesite.com](http://www.yourfavoritesite.com)
- **What do you see?**
- **Each time you type a letter a new query goes to google, including all google’s cookies.**
  - Google knows exactly what you type in the browser, all the time
  - And if you ever logged in google on that browser, google knows it’s you
Back to attacks (whether targeted or not)
Browser Exploitation Framework

- Take a look at beefproject.com
- Very nice video at
  - https://www.youtube.com/watch?time_continue=132&v=xdbvU_U42kY
- Features
  - Fingerprinting: Browser, OS, location fingerprinting
  - They adapt the response to the request (e.g. when they see a Firefox browser they put an exploit for Firefox in the response)
  - They can be used for targeted attacks (e.g. introduce the exploit only if browser originates from a given IP range)
Some BEPs

- Easy to find....

### Table 4.2 Most Widely Used BEPs List from Last 5 Years

<table>
<thead>
<tr>
<th>Cool Exploit Kit</th>
<th>BlackHole Exploit kit</th>
<th>Crime Boss Exploit Pack</th>
<th>Crime Pack</th>
<th>Bleeding Life</th>
</tr>
</thead>
<tbody>
<tr>
<td>CritXPack</td>
<td>EL Fiesta</td>
<td>Dragon</td>
<td>Styx Exploit Pack</td>
<td>Zombie Infection kit</td>
</tr>
<tr>
<td>JustExploit</td>
<td>iPack</td>
<td>Incognito</td>
<td>Impassioned Framework</td>
<td>Icepack</td>
</tr>
<tr>
<td>Hierarchy Exploit Pack</td>
<td>Grandsoft</td>
<td>Gong Da</td>
<td>Fragus Black</td>
<td>Eleonore Exploit Kit</td>
</tr>
<tr>
<td>Lupit Exploit Pack</td>
<td>LinuQ</td>
<td>Neosploit</td>
<td>Liberty</td>
<td>Katrin Exploit Kit</td>
</tr>
<tr>
<td>Nucsoft Exploit Pack</td>
<td>Nuclear</td>
<td>Mpack</td>
<td>Mushroom/ Unknown</td>
<td>Merry Christmas</td>
</tr>
<tr>
<td>Sakura Exploit Pack</td>
<td>Phoenix</td>
<td>Papka</td>
<td>Open Source/ MetaPack</td>
<td>Neutrino</td>
</tr>
<tr>
<td>Salo Exploit Kit</td>
<td>Safe Pack</td>
<td>Robopak Exploit Kit</td>
<td>Red Dot</td>
<td>Redkit</td>
</tr>
<tr>
<td>T-Iframer</td>
<td>Sweet Orange</td>
<td>Siberia Private</td>
<td>SofosFO aka Stamp EK</td>
<td>Sava/Pay0C</td>
</tr>
<tr>
<td>Zopack</td>
<td>Tornado</td>
<td>Techno</td>
<td>Siberia</td>
<td>SEO Sploit pack</td>
</tr>
<tr>
<td>Yang Pack</td>
<td>XPack</td>
<td>Whitehole</td>
<td>Web-attack</td>
<td>Unique Pack Sploit 2.1</td>
</tr>
<tr>
<td>Yes Exploit</td>
<td>Zero Exploit Kit</td>
<td>Zhi Zhu</td>
<td>Sibhost Exploit Pack</td>
<td>KaiXin</td>
</tr>
</tbody>
</table>
Watering Hole Attacks

• Gather Intelligence on the target
  • Discover suitable sites

• Infect steppingstone site
  • Possibly via 3rd party sites

• When target visits the site it is infected
  • Usually with a dropper

• Dropper on target contacts CC (command and control site) to download full malware

• Malware carries out the exploitation
  • Data exfiltration
  • Key logging
  • Etc etc etc
Phishing

- Email with malicious link/instructions
- Can be very targeted spear-phishing
- Like it appears coming from your boss
  - Sharing a document you should review
- In simple cases it redirects to a genuine-looking site where you need to fill some credentials
From: "MSU! Helpdesk" <macicchino@email.wm.edu>
Date: January 30, 2014 at 10:41:52 AM EST
To: undisclosed-recipients::<>
Subject: {Account Login Alert!}

This is an automated message to notify you that a valid password was used to login your MSU! account from an unrecognized device, Today Thursday, January 30th, 2014 at 09:00(UTC+02), in Mauritius, Port Louis (IP=41.136.181.172) as a result of that your account was temporarily suspended.

If you did this, you can safely disregard this email. If you didn’t do this, kindly follow our review link below to retrieve your account
http://cse-msuaccountreviewauthenticationforum.yolasite.com/
Sincerely,
The MSU! Helpdesk
[---001:000564:57449---]
Please do not reply to this message. Mail sent to this address cannot be answered.
There is a whole market for DDoS attacks, you can just buy one. We’ll see some examples.
QUESTIONS