Final exam in

# Web Security EITF05

Department of Electrical and Information Technology
Lund University

Oct 20, 2010, 14-19

- 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!

Paul & Martin

**Problem 1.** Explain the terms *safe method* and *idempotent method* in the context of HTTP. Are GET and POST safe methods? Are they idempotent methods?

**Answer**
A safe method does not have any side-effects on the server, it should just return a response. GET is safe, and POST in non-safe. An idempotent method should have the same side-effect if it is used multiple times as if it was used only once. GET is idempotent and POST is non-idempotent. (3 points)

---

**Problem 2.** Give the Base64 encoding of the word "ACES".

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**
The encoding is "QUNFUw==". (3 points)

---

**Problem 3.** User tracking is used by many websites for different reasons. Explain and compare user tracking with first and third party cookies.

**Answer**
Third-party cookies: Some content, e.g., a picture, is downloaded from a third party, together with a cookie. If content from the same third party is downloaded at a later time, the cookie will be sent in the request. This will inform the third party that the user has visited both pages, which time the pages were visited etc. It can be used e.g., for placing ads suitable for specific users.
First party cookies: By running a javascript on a webpage, a cookie can be used and sent, together with other information, in a GET request to a third party. This is made possible because the javascript can read the cookie and it is possible to send information as part of the URL in a GET request.
Comparison: Many users disable third party cookies since they are mostly used for user tracking. However, first party cookies are also used for session handling so it is not convenient for users to disallow these cookies. Thus, first party cookies are more robust in this sense. On the other hand, they require that javascript is turned on. Moreover, user tracking with first party cookies is primarily used for tracking on the same domain, even though it is possible to do it over several domains if they are in agreement. Third party cookies can easily be used over several domains if they only use the same third party. (3 points)

---

**Problem 4.** In this problem we will make a toy example of the disclosure attack on a Chaum Mix. Assume that we know (or can guess with high probability) in which output set a message from Alice is sent. In the communication system there are in total $N = 26$ users, labeled A, B, C, ..., Y, Z. We know that Alice has $m = 4$ communication partners among the users. The Mix outputs $n = 5$ messages at each time and we know that exactly one of these is sent from Alice. In the first part of the attack we collect $m$ mutually disjoint sets:

(A,M,P,G,J), (B,Q,R,F,I), (C,N,E,S,T), (D,H,K,L,O)

In the second part of the attack we use new sets in order to reveal the $m$ communication partners of Alice. Complete the second part of the attack if the collected output destinations are:

| | | | |
|---|---|---|---|
| (F,B,K,V,M) | (R,I,U,B,V) | (G,D,X,L,T) | (Y,N,Q,M,D) |
| (C,J,F,O,Z) | (Q,C,E,Z,U) | (R,S,H,A,L) | (F,T,P,A,W) |

**Answer**
For simplicity we denote the disjoint sets by $d_0, d_1, d_2$ and $d_3$ in the order presented in the problem. Similarly the additional observed sets are denoted $a_0, a_1, \ldots, a_7$. Since the sets $d_i$ are mutually disjoint we known that exactly one recipient in each set is a communication partner. If we look at the set $a_1$ we see that $a_1 \cap d_0 = \emptyset, a_1 \cap d_2 = \emptyset$ and $a_1 \cap d_3 = \emptyset$. Thus, a communication partner is in the intersection of $a_1 \cap d_1$ and we update the set $d_1$ as $d_1 \leftarrow a_1 \cap d_1$. The updates continue in the same way. In summary:
1. $a_1$ updates $d_1$ to (B,R,I)

2. $a_5$ updates $d_2$ to (C,E)
3. $a_7$ updates $d_0$ to (A,P)
4. $a_2$ updates $d_3$ to (D,L)
5. $a_0$ updates $d_1$ to (B)
6. $a_3$ updates $d_3$ to (D)
7. $a_4$ updates $d_2$ to (C)
8. $a_6$ updates $d_0$ to (A)

Note that it is possible to apply some steps in different order. The end result would still be the same, i.e., Alice's communication partners are A, B, C and D. (3 points)

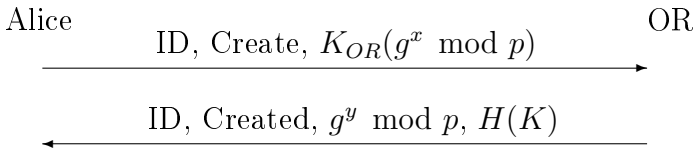
---

**Problem 5.** Write a regular expression that checks if a string is a base64 encoding of a SHA-256 hash.

**Answer**
256 bits is 32 bytes. When this is converted to base64 it has to be padded to a multiple of 3 bytes, so it is padded by one byte. 33 bytes will be turned into 44 base64 characters. Since one byte was used as padding, there will be 43 letters from the base64 alphabet followed by one "=". A regular expression for this would be `^[a-zA-Z0-9+/]{43}=$` (3 points)

---

**Problem 6.** Consider the key negotiation with the first Tor node as presented below.




Assume that we record *all* transactions between Alice and the OR during some time period. Much later, the private key of the OR is disclosed. Is it then possible to use this key to decrypt the recorded traffic? Motivate your answer clearly.

**Answer**
Using the private key it is possible to decrypt and get $g^x \bmod p$. However, having $g^x \bmod p$ and $g^y \bmod p$ is not enough to get the secret key $g^{xy} \bmod p$. Diffie-Hellman key exchange is vulnerable to man-in-the-middle attacks, but not to eavesdropping. Hence, we would not be able to decrypt the recorded traffic. This is the perfect forward secrecy property. (3 points)

---

**Problem 7.** In Hashcash, the string

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

is hashed using SHA-1. However, not all such strings are valid Hashcash strings. Explain how a valid Hashcash string is computed. Why is an email containing a Hashcash header most likely legitimate?

**Answer**
A valid hashcash string will have the property that the hash of the string starts with $x$ number of zeros. The value of $x$ depends on how difficult it should be to find the string. The string is hashed for different values of "counter" until the hash has the desired property. If the value of $x$ is large enough, ($\approx 20 - 25$), it means that the sender must

have spent a significant amount of time preparing the email. Typically, spammers do not have time to spend several seconds for each email. (3 points)

---

**Problem 8.** Consider the following piece of (edited) PHP code.

```
$uname = $_POST['username'];
$pass = $_POST['passwd'];

$db = mysqli_connect();

...[some hidden code]...

/* bind parameters and result, execute and fetch parameters */
mysqli_stmt_bind_param($stmt, "ss", $uname, $pass);
mysqli_stmt_execute($stmt);
mysqli_stmt_bind_result($stmt, $u_name, $u_pass, $u_email);
mysqli_stmt_fetch($stmt);

if ($u_name) {
  /* user is authenticated */
  session_regenerate_id();
  ...
```

a) Which technique is used to prevent SQL-injection? Give one other way to prevent this type of attack.

b) Explain, without writing code, how the database connection call can be made without parameters.

c) What is the purpose of `session_regenerate_id()`?

**Answer**
a) A prepared statement is used. Another protection is to filter the input using e.g., `mysql_real_escape_string()`.
b) The host, user and password can be put in a configuration file instead of in the source code.
c) The purpose is to prevent session fixation attacks. If an attacker can force a user to use a particular session id, this id will be useless for the attacker anyway after the user has logged in. (3 points)

---

**Problem 9.** Although it would have been technically possible, DNSSEC was not designed to use digital certificates. How are keys verified in DNSSEC? In what way is the trust model in DNSSEC similar to that of digital certificates?

**Answer**
The key is verified by letting the parent domain sign a hash of the public key. This hash is stored in a DS record. The signature is verified by asking for the public key of the parent domain. If the signature is valid we know that the key belongs to the domain

provided that we trust the public key of the parent domain. This key can in turn be signed by the next domain and so on until we get a public key that we explicitly trust. In digital certificates the public key is signed by a CA. The CA certificate, which includes the public key of the CA, can in turn be signed by another CA. This is done until we find a certificate that is trusted. Hence, the model is essentially the same, but in the case of certificates any CA can sign any certificate while in DNSSEC a public key is always signed by the parent. (3 points)

---

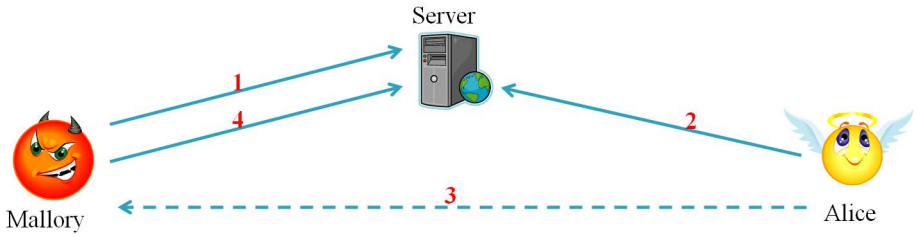
**Problem 10.** Early versions of the BIND DNS server used sequential transaction IDs when making queries, i.e., if the transaction ID in one query was $x$, the transaction ID used for the next query was $x + 1$ etc. Describe how you would mount a DNS cache poisoning attack in this situation.

**Answer**
The current transaction ID is obtained by first making a request to a domain name for which I control the authoritative DNS server. I send this request to the intended victim. When the victim queries my DNS server for the name I get the victim's current transaction ID, denoted $x$. Then I make a query for e.g., bank.com to the victim and also immediately answer this query forging the IP of the DNS server authoritative for the domain bank.com. In the answer I use the transaction ID $x+1$. This attack will have a very high probability of success since there is no randomness at all. Note that we assume that the port is not random even though this is not explicitly stated. Randomizing ports was not added until later in BIND. (However, answers assuming that we have to guess the port would also give full points.) (3 points)

---

**Problem 11.** Explain how an XSS attack works. You may use the picture below for references. Also, state where the following script fits in and what it can be used for.

```
<script>
document.body.innerHTML=
  '<iframe src="http://www.server.com"
  width="100%"
  height="100%"
  frameborder="0" />';
</script>
```




**Answer**
Using the picture as reference:
1. The attacker (Mallory) injects a script on a vulnerable server. One example is the script provided in the problem.
2. Alice visits the server and runs the script.
3. The script is used to send information to Mallory. Using the script in the problem, Mallory will see a different webpage than the one that is written in the address bar of

the browser. The could be a webpage provided by Mallory that is very similar to the real webpage, but instead sends e.g., passwords to mallory instead of to the real webpage. Another example is cookie stealing. The script has access to Alice's cookie shared with the server. This cookie can be sent to Mallory as data in a GET or POST.
4. Using the stolen password, or alternatively the stolen cookie, Mallory can authenticate as Alice to the server. (5 points)

---

**Problem 12.** Consider the Digest Authentication (1999) protocol with *qop=auth* given below.

$$md5(md5(A1) : nonce : nc : cnonce : qop : md5(A2))$$
$$A1 : \quad username : realm : password$$
$$A2 : \quad method : URI$$

a) What purposes do the *realm* and *cnonce* parameters serve?

b) If you are a man-in-the-middle that can alter messages, how would you rank the following options from best to worst. Motivate your answer.

1) Replace the entire Digest Authentication header with a Basic Authentication header.

2) Replace all server-provided items and perform a TMTO attack on the remaining information.

3) Perform a dictionary attack on md5(A1).

**Answer**
a) The realm is used to notify the user which password to use in case the server has different access control for different parts of the server. It is also used a salt to the password when it is hashed. The cnonce is used to provide additional randomness to the outer hashfunction. This will completely defeat any sensible TMTO attack. It is also used as challenge if the client also wants to authenticate the server.
b) The best alternative is (1) since it will immediately reveal the password to the attacker (but requires that the user does not notice the change). The second best is a dictionary attack (3) since it gives the attacker the password in case it can be found in the dictionary. The TMTO attack (2) is completetely useless in this case since it will be infeasible to build the tables for the attack in advance due to the cnonce. Buildning the tables after the response is received is the only option, but this will be as costly (or actually slightly more costly) as a brute force attack. Any clever attacker would try a dictionary attack before trying brute force. (5 points)

---

**Problem 13.** A DKIM signature header of an email is given below.

```
DKIM-Signature:
v=1;
a=rsa-sha256;
c=simple/relaxed;
d=gmail.com;
s=gamma;
```

```
h=received:message-id:date:from:to:subject:mime-version:content-type;
bh=9gicsZnlcLK7yYh6VIrgyAMMRZiWsSbWqSPIhc78RRk=;
b=k4ofvpHPkaQmvuSoGVhRrnCsPK+JEuv9
  KUrZO7aiypvf/6Y1N2iIatvLvdzwOnZX
  /W6Kxyx6Z4Ybuk8Dqk/vNTIE7Jpy+GQU
  UHFvM0NFtmZo1CbGRvo8DdHnXRBB/qWw
  lV+Z6wxw/mq7lNuJknVprOAaTLws5mwc
  Z+AWL8KwHg0=
```

a) What is a "Selector"?

b) How many bits are in the RSA signature?

c) Which parts of the email are integrity protected?

d) How does the client obtain the public key?

e) How does one know that the public key belongs to the signer?

**Answer**
a) The selector is the filed that determines which public key should be used to verify the signature. It is used to allow a domain to have several public keys.
b) The number of base64 characters in the encoded signature (b=) is 172. This corresponds to 129 bytes. Since the last character of the encoded signature is "=" the signature is 128 bytes (1024 bits).
c) All headers defined by the h-tag are signed. In addition the body is signed and all fields except the b-tag in the DKIM-header are signed.
d) The key is obtained by asking the DNS authoritative for the domain for the key. In this case it is located in a TXT record in the domain `gamma._domainkey.gmail.com`.
e) The public key is assumed to be correct since it is located in the DNS of the domain and only the administrator has the possibility to put it there. (5 points)

---

**Problem 14.** Explain briefly the following terms

a) SPF record

b) NSEC record

c) register_globals

d) URL encoding

e) .htaccess

**Answer**
a) A DNS record stating who is allowed to send email from the domain.
b) A DNS record used to provide proof of nonexistence for a queried domain or resource records.
c) A directive in php.ini that determines if global variables can be automatically set in requests.
d) A way to encode the URL so that characters with special meaning in URLs are instead

interpreted as the actual character.
e) A configuration file for a web server that is located in the directory to which it applies. It is useful if the administrator of a particular directory of the server does not have write access to the main configuration file. (5 points)

---