Final exam in

Web Security EITF05

Department of Electrical and Information Technology
Lund University

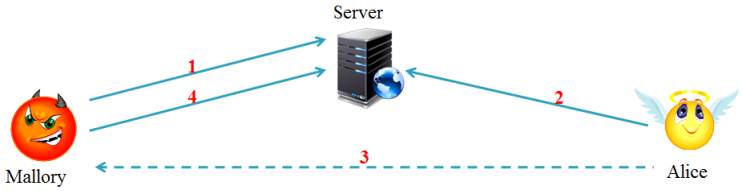
October 30th, 2015, 14.00–19.00

- 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!

Paul

Problem 1. Consider the following illustration of an XSS attack.



a) Explain how an XSS attack works, detailing where JavaScript(s) reside and are executed. You may refer to the picture.

b) Does TLS protect against XSS attacks? Motivate.

c) Does the same-origin policy protect against XSS attacks? Motivate.

Answer

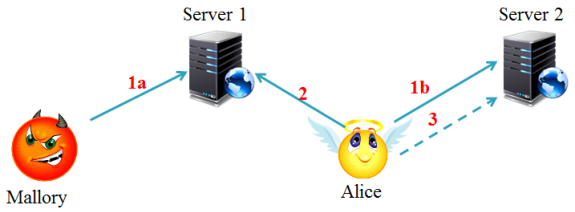
a) 1: Mallory puts a JavaScript on webpage on Server.
2: Alice retrieves the webpage from Server, including Mallory's JavaScript.
3: The JavaScript is executed in Alice's browser, sending Alice's session cookie to Mallory.
4: Mallory can use Alice's cookie to hijack her browsing session.
In particular, the JavaScript resides on Server and is executed in Alice's browser.

b) No. SSL operates on session level in the OSI model, while script injection is applied on application level. SSL protects transport of content, but it does not look at the content itself. Injected scripts reside in (are part of) the website content that is stored on the server. This content is interpreted by the victims browser after transport.

c) No. The main HTML page and the script will be from the same origin, and sending cookies or other data to any other origin is permitted.

(3 points)

Problem 2. Consider the following illustration of a CSRF attack.



a) Explain how a CSRF attack works. You may refer to the picture.

b) Explain how CSRF protection with synchronizer token pattern works.

c) CSRF attacks are possible even when Alice has disabled JavaScript in her browser. Describe one feasible scenario.

Answer

a) 1a: Mallory puts a script on webpage on Server 1.
1b: Alice logs in to Server 2 and obtains a session cookie.
The order of steps 1a and 1b is irrelevant.
2: Alice visits webpage on Server 1.
3: Mallory's script is executed on Alice's computer, sending a request to perform some action on Server 2.

b) See secret validation token in lecture notes.

c) See img-link technique in lecture slides.

(3 points)

Problem 3. 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+JEuv9KUrZO7aiypvf/6Y1N2iIatvLvdzwOnZX
  /W6Kxyx6Z4Ybuk8Dqk/vNTIE7Jpy+GQUUHFvM0NFtmZo1CbGRvo8DdHnXRBB/qWw
  lV+Z6wxw/mq7lNuJknVprOAaTLws5mwcZ+AWL8KwHg0=
```

a) How can the client verify that she has obtained the correct public key (for signature verification)?

b) Can a domain use different keys for different users or groups of users? Motivate.

c) How does DKIM provide integrity protection for the message part of the email? (What is signed, and how?)

Answer

a) She can if the DNS server is DNSSEC-enabled, but generally she cannot.

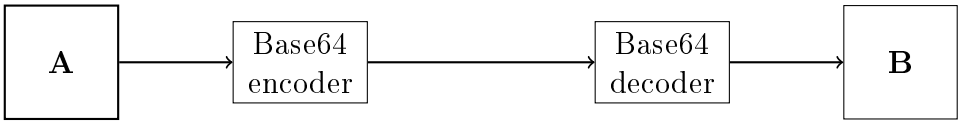
b) Yes, by using different selectors 's' for different users or groups of users.

c) bh is a hash of the body. The DKIM header (including bh) is included in h, even though it is not explicitly listed. Thus, the signature in b covers bh as well.

(3 points)

Problem 4. A company has a problem with a communication channel, and you have been hired to resolve the issue. The problem is as follows.

Device A sends single control bytes to device B. A uni-directional channel that requires printable characters to be sent is used, so each byte is Base64-encoded before it is sent from A, and Base64-decoded before it is interpreted at B.



Recently, the Base64 decoder was replaced with a new and improved implementation. However, after this upgrade, only a small fraction of the messages are delivered – most are discarded by the decoder.

One particular message that could be transmitted before but not after the upgrade is "wd==".

a) Explain what the most likely problem is, and how it should be fixed.
b) What exact fraction of the messages does the new decoder discard? Assume uniform distribution where applicable.

Hint 1: Symbol w is 0x30 and d is 0x1D in Base64.

Hint 2: The following section is from RFC4648 titled "The Base16, Base32, and Base64 Data Encodings".

```
3.5.  Canonical Encoding

   The padding step in base 64 and base 32 encoding can, if improperly
   implemented, lead to non-significant alterations of the encoded data.
   For example, if the input is only one octet for a base 64 encoding,
   then all six bits of the first symbol are used, but only the first
   two bits of the next symbol are used.  These pad bits MUST be set to
   zero by conforming encoders, which is described in the descriptions
   on padding below.  If this property do not hold, there is no
   canonical representation of base-encoded data, and multiple base-
   encoded strings can be decoded to the same binary data.  If this
   property (and others discussed in this document) holds, a canonical
   encoding is guaranteed.

   In some environments, the alteration is critical and therefore
   decoders MAY chose to reject an encoding if the pad bits have not
   been set to zero.  The specification referring to this may mandate a
   specific behaviour.
```

Answer

a) The decoder discards non-canonical encodings. Improve the encoder to produce conforming encodings only.
b) There are four unused symbol bits in the padding part, so $\frac{1}{2^4} = \frac{1}{16}$ of the messages will pass though, while $1 - \frac{1}{16} = \frac{15}{16}$ of them will be discarded.

(2+1 points)

Problem 5. Consider a Hashcash solution in which a string

$$ver : bits : date : resource : rand : counter$$

is hashed using SHA-1, where

ver is version number (currently 1),
bits indicates how costly the function is for sender,
date gives current date,
resource is recipients email address,
rand is a random number.

a) Explain the principles of Hashcash, and detail the relationship between the *counter* and *bits* parameters.
b) How can HashCash be misused if the *rand* parameter is removed from the protocol?

Answer

a) The HashCash header is valid if the hashed string has *bits* initial zeros. You can create a valid HashCash header as follows. Initially construct the string with the *counter* part set to zero. While that string does not hash to a value with *bits* initial zeros (while it is not a valid HashCash string), increase *counter*.
b) The *rand* parameter separates different senders, so different senders can collaborate and reuse the same HshCash header in this case.

(2+1 points)

Problem 6. An IPv4 address is a group of four numbers from 0 to 255 (inclusive) separated by dots, as in 127.0.0.1, 130.235.202.25, and so on.

Write regular expressions for matching the following.

a) All IPv4 addresses, with no additional restriction on the numbers.
b) All IPv4 addresses, restricting all four numbers to $0, \ldots, 255$.

Hint 1: You may disregard leading zeros.

Answer
One possibility is
`(<num>\.){3}<num>`, where
`<num>=(25[0-5]|2[0-4][0-9]|[01]?[0-9]?[0-9])`. (1.5+1.5 points)

Problem 7. Consider Domain Name System Security Extensions (DNSSEC).

a) Explain how the correctness of the public key in DNSKEY is verified. Compare the procedure to that of verifying a certificate.
b) How can NSEC and NSEC3 provide precomputed answers for *all* requests? (What trick is used?)
c) Describe one very significant problem with DNSSEC.

Answer

a) Explained during lecture, see lecture slides. Or lecture notes. The public key for a zone is signed by the parent zone. More specifically, the DS record type contains the hash of the public key and is stored in the parent zone together with a corresponding RRsig record. This will achieve the same functionality as that of certificates.
b) The (precomputed) response is an *interval*. It is sufficient that the queried entry is in this interval.
c) DNSSEC responses are larger due to the signatures, which can be useful for a DNS amplification attack.

(3 points)

Problem 8. Explain how Content Security Policy (CSP) works.

Answer
A web server hosting a webpage that only loads JavaScript from its own domain could just tell the user-agent that scripts can not be loaded from other domains. Similarly, the web server could provide the user-agent with a whitelist of domains allowed to host JavaScript. The content security policy (CSP) is a way to do this, and it also generalizes the idea even further to include other types of sources. Thus, CSP is a mechanism that can be used to mitigate a large number of content injection vulnerabilities.

The information is sent in a HTTP header named Content-Security-Policy. The information given in the header consists of a directive name and a directive value. The name controls what to be restricted and the value gives the actual restrictions. The following are some examples of directive names and their meanings.

- default-src. If a name is not explicitly given, the values given in the default-src directive are used.
- script-src. Restricts the scripts that can be executed.
- object-src. Restricts from where plugins can be loaded.
- img-src. Restricts from where images can be loaded.
- style-src. Restricts from where stylesheets can be loaded.
- report-uri. Specifies the URI to which the client can send reports.

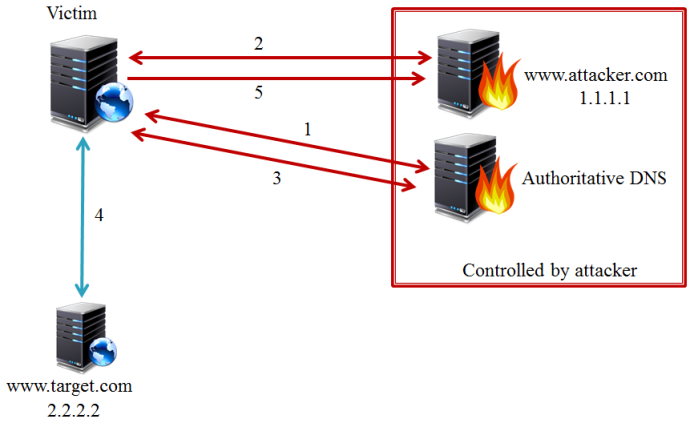
about policy violations. The values can be domains hosting resources, but there are also some special values that can be used. The 'self' value is used to restrict sources to the origin the document was fetched from. (3 points)

Problem 9. Consider the following illustration of a DNS rebinding attack.

a) Referring to the picture below, explain how JavaScript is used in a DNS rebinding attack.

b) Why will the attack fail if the attacker instructs the script to go directly to www.target.com? Motivate. (Which protection mechanism is triggered in this case?)

c) What can the target server (www.target.com) do to avoid serving content in DNS rebinding attacks? (How can it detect DNS rebinding attacks?)



Answer

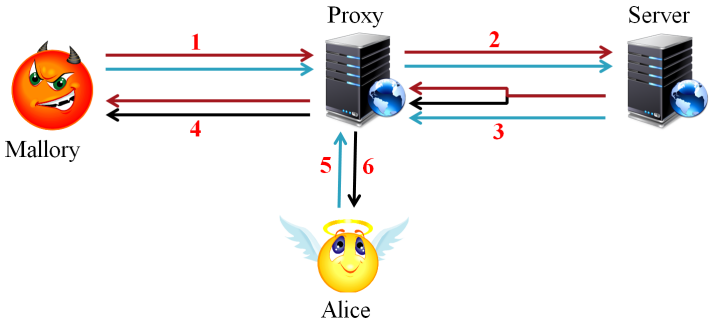
a) At step 2, the page retrieved from `www.attacker.com` contains a JavaScript. When this script is interpreted in the victims browser, it requests another page from `www.attacker.com`, resulting in steps 3 and 4. The response in step 4 is also handled by the JavaScript, relaying the response page from `www.target.com` in step 4 to `www.attacker.com` in step 5.

b) Same-origin policy in the victim's browser, since the attacker's domain will not have been re-bound in this case. The script will fetch the page from www.target.com, but it will not send it along to www.attacker.com.

c) Check the HTTP Host header, even if the server does not support virtual hosts.

(3 points)

Problem 10. Consider the following illustration of an HTTP response splitting attack.



a) Referring to the picture, briefly explain how an HTTP response splitting attack works.

b) Write some PHP code that enables the attack. Your code does not need to be syntactically correct, but it should show a feasible case.

c) What would happen if the proxy was not a *caching* proxy?

Answer

a)
1. Mallory sends two requests to server via proxy (second is for /index.php).
2. Proxy forwards requests to server.
3. Server responds to both queries (proxy sees them as three responses).
4. Proxy forwards (and caches) the first two responses (third thrown away).
5. Alice requests /index.php.
6. Proxy delivers Mallory's cached page.

b)
```
$x=_GET['language']
header("Location: http://www.example.com/index_lang.php?language=$x");

GET /redirect.php?language=swedish%0d%0aContent-Length:%200
%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-Length:%2032%0d%0a
%0d%0a<html>MallorysPage</html> HTTP/1.1
Host: www.example.com
...

HTTP/1.1 302 Moved Temporarily
...
Location: http://www.example.com/index_lang.php?language=swedish
Content-Length: 0

HTTP/1.1 200 OK
Content-Length: 32
<html>MallorysPage</html>
...
```

c) The attack would be pointless, as the specially crafted response would not be stored at the proxy for later retrieval by Alice.

(3 points)

Problem 11. HTTP digest authentication (RFC2617) is a challenge response protocol in which the client calculates the digest (the response) according to

$$\text{MD5}(\ \text{MD5}(A1) : \mathit{nonce} : \mathit{nc} : \mathit{cnonce} : \mathit{qop} : \text{MD5}(A2)\ ),$$

with

$$\begin{aligned} A1 &= \mathit{username} : \mathit{realm} : \mathit{password}, \\ A2 &= \begin{cases} \mathit{method} : \mathit{URI} & \text{if } \mathit{qop} = \mathit{auth}, \\ \mathit{method} : \mathit{URI} : \text{MD5}(\mathit{entity\text{-}body}) & \text{if } \mathit{qop} = \mathit{auth\text{-}int}. \end{cases} \end{aligned}$$

a) Explain the usage and purpose of the *qop* parameter?

b) Explain the usage and purpose of the *nonce* parameter.

c) In which form are the credentials stored on the server?

d) Both *nonce* and *cnonce* are randomly selected strings, so why is it that the *cnonce* parameter protects against a MITM that can modify messages and has TMTO/Rainbow capabilities, when the *nonce* parameter does not?

Answer

a) Quality of protection. Optional integrity protection.

b) A nonce that the server chooses. The challenge.

c) $MD5(A1)$ is stored on the server (essentially to be regarded as the password).

d) *cnonce* is chosen by the client, and it is revealed to the MITM *after* the digest has been calculated. The MITM therefore cannot replace this value with one that he chooses – one that his TMTO/Rainbow tables have been precomputed for. Contrary to this, the MITM can easily replace the *nonce* parameter, since it is sent via the MITM *before* the client calculates the digest.

(1+1+1+2 points)

Problem 12. You have successfully stolen a database from a popular website using an SQL injection. Passwords in this database have been salted with a site-wide salt, same for all passwords, and then hashed with SHA3-512. You have full access to the database, and the salt is known to you. You want to recover as many passwords as possible, so you are considering a TMTO attack with parameters t, n and ℓ. That is, the TMTO attack uses t tables, each with n start-/endpoint pairs (SP/EP-pairs), with chains of length ℓ.

a) Why will you not find pre-built tables online that you can use?
b) Explain how chains are constructed using the hash- and reduction functions.
c) Explain why and how hash table(s) are used during password recovery.
d) In terms of t, n and ℓ, what is the expected time complexity for recovery of *one* password? (How many hash table lookups?)

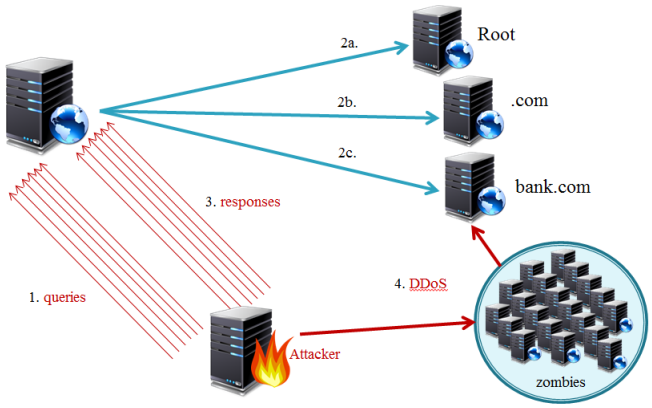
Hint 1: Alternatively, you may answer the same question for a rainbow attack setting with parameters n and ℓ as above (one table). If you do so, please state this clearly.
Hint 2: Hash tables can provide $O(1)$ lookup time. That is, lookups into a hash table can be performed in constant time, regardless of how many entries that are stored.

Answer

a) The salt prevents this.
b) Randomly generate one potential password SP= v_0. Apply hash function h and reduction function r to obtain the next potential password from the previous one; $v_{i+1} = r(h(v_i))$. Set $v_{\ell-1}$ =EP.
c) One password lookup in a TMTO table breaks down into several $O(1)$ hash tables lookups into a hash table storage. SP/EP-pairs are stored in this hash table, indexed by the EP's.
When looking up a password hash x, we actually want to find the corresponding password v_j (for some j if it exists in the TMTO table) such that $h(v_j) = x$. We cannot invert h, but we can travel to the next potential password in the chain as shown above; $v_{j+1} = r(x)$. We traverse this chain step by step, and for every step we check if we have reached an EP (one hash table lookup per step). When so, we go to the corresponding SP= v_0 and traverse the v's step by step until we reach v_j (no hash table lookups required for this, why?).
d) Traversing (or reconstructing) a chain takes $\frac{\ell}{2}$ hash table lookups on average, or $O(\ell)$. We have t tables, so total expected time complexity is $\frac{t\ell}{2}$ or $O(t\ell)$.

(1+1+2+1 points)

Problem 13. Consider the following illustration of a DNS cache poisoning attack.



a) Why is DNS spoofing (DNS cache poisoning) trivial for an adversary that can observe outgoing traffic (2a, 2b and 2c above) from the DNS server?

b) Modern DNS implementations do not only randomize transaction IDs. What else do they randomize?

c) How would usage of TCP (instead of UDP) protect against DNS cache poisoning attacks?

d) Make a reasonable estimate of how long it would take for an attacker to succeed with a DNS cache poisoning attack for a moderately popular website. State your assumptions, estimates and approximations along with your calculation.

Answer

a) The adversary can then read the transaction ID and port number of the request sent from the DNS and provide a spoofed answer that will be accepted by the DNS.

b) The port number.

c) Very well, since an adversary cannot easily spoof a TCP connection.

d) Be creative. There is no standard solution.

(1+1+1+2 points)

Problem 14. Briefly explain the following terms.

a) Birthday paradox
b) SPF
c) CORS
d) Greylisting
e) Idempotent HTTP method

Answer

a) A probability problem that is applicable to hash functions (collision counting).
b) Sender Policy Framework, spam protection technique that enables the receiving MTA to check if the sending MTA is authorized to send mail from a given domain.
c) Cross-Origin Resource Sharing, a technique that allows several domains to share content with one another. Can be used to create mash-ups.
d) A spam protection method in which a first-time MTA is forced to resend.
e) An HTTP method that has no additional server side-effects when called many times.

(5 points)
