UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

INFR10067 COMPUTER SECURITY

Tuesday 15th May 2018

14:30 to 16:30

INSTRUCTIONS TO CANDIDATES

Answer any TWO of the three questions. If more than two questions are answered, only QUESTION 1 and QUESTION 2 will be marked.

All questions carry equal weight.

CALCULATORS MAY NOT BE USED IN THIS EXAMINATION

Year 3 Courses

Convener: C. Stirling
External Examiners: S.Rogers, A. Donaldson, S. Kalvala

THIS EXAMINATION WILL BE MARKED ANONYMOUSLY

1. **Casing a network**

One of the first steps an attacker (or new admin) takes is called "casing". The attacker looks around at easy-to-get information to get a sense of what a network looks like and where the likely weak points are. In this question you will be using a provided network capture file to do an initial analysis of a network.

(a) Draw the logical network that is producing the traffic in Figure 1. Figure 2 shows an example of the type of network drawing we are looking for. Your drawing should contain the following information:

- IP address of each machine sending traffic.
- What services, if any, each machine is most likely running.
- What ports they are offering those services on.
- Which machines are communicating with each other.

Write a brief justification of your drawing. Focus on less obvious decisions you may have made and make use of the line numbers in Figure 1 to help justify decisions. [*10 marks*]

(b) One of the computers in Figure 1 is clearly running a firewall. Which computer is it and why? [*2 marks*]

(c) Someone carelessly left the old IPTables setup of the computer in question 1b on an unprotected server (Figure 3). The file is from September and likely out of date. Use the content of Figure 1 to update the firewall rules to their most likely current state.
Write out the full current firewall policy, similar to how Figure 3 is displayed. You may omit comment lines and the line numbers from your answer. Justify your answer. [*7 marks*]

(d) The 192.168.7.2 computer is a laptop owned by Bob who often travels with it. Imagine Bob connects his computer to a hotel wifi and engages in the type of interactions visible in Figure 1. He does not use a VPN. You can assume that all the computers involved will accept connections from Bob's laptop as if he was on site and they will see the same IP from him that he normally uses on site (Gateways, NATs, etc. have all been properly setup and all necessary technology "magic" has happened).

List the three primary CIA principles that define a system as being "secure". For each principle, state if that principle holds or not in the above scenario and justify your answer. [*6 marks*]

	Source	Destination	Protocol	Info
1	192.168.7.2	192.168.7.8	TCP	69710→22 [SYN]
2	192.168.7.8	192.168.7.2	TCP	22→69710 [SYN,ACK]
3	192.168.7.2	192.168.7.8	TCP	69710→22 [ACK]
4	192.168.7.2	192.168.7.8	SSHv2	Client: Protocol (SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.2)
5	192.168.7.8	192.168.7.2	TCP	22→69710 [ACK]
6	192.168.7.8	192.168.7.2	SSHv2	Server: Protocol (SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.2)
7	192.168.7.2	192.168.7.8	TCP	69710→22 [ACK]
8	192.168.7.2	192.168.7.8	SSHv2	Client: Key Exchange Init
9	192.168.7.8	192.168.7.4	SSHv2	Server: Key Exchange Init
10	192.168.7.2	192.168.7.8	SSHv2	Client: Diffie-Hellman Key Exchange Init
11	192.168.7.8	192.168.7.4	SSHv2	Server: Diffie-Hellman Key Exchange Reply, Keys, Encrypted
12	192.168.7.2	192.168.7.8	TCP	69710→22 [ACK]
13	192.168.7.2	192.168.7.8	SSHv2	New Keys
14	192.168.7.8	192.168.7.2	TCP	22→69710 [ACK]
15	192.168.7.2	192.168.7.8	SSHv2	Client: Encrypted packet
16	192.168.7.8	192.168.7.2	TCP	22→69710 [ACK]
17	192.168.7.2	192.168.7.8	SSHv2	Server: Encrypted packet
18	192.168.7.2	192.168.7.8	TCP	69710→22 [ACK]
19	192.168.7.2	192.168.7.8	TCP	69710→22 [FIN,ACK]
20	192.168.7.8	192.168.7.2	TCP	22→69710 [FIN,ACK]
21	192.168.7.4	192.168.7.13	TCP	45264→80 [SYN]
22	192.168.7.4	192.168.7.13	FTP	49876→80 [SYN]
23	192.168.7.13	192.168.7.4	TCP	80→45264 [SYN,ACK]
24	192.168.7.4	192.168.7.13	TCP	45264→80 [ACK]
25	192.168.7.4	192.168.7.13	HTTP	GET /?login=bob HTTP1.1
26	192.168.7.13	192.168.7.4	TCP	80→45264 [ACK]
27	192.168.7.13	192.168.7.4	HTTP	HTTP/1.0 200 OK (password=BestPasswordEver!)
28	192.168.7.4	192.168.7.13	TCP	45264→80 [FIN,ACK]
29	192.168.7.13	192.168.7.4	TCP	80→45264 [ACK]
30	192.168.7.50	192.168.7.13	TCP	34456→80 [SYN]
31	192.168 7.3	192.168.7.13	TCP	35111→21 [SYN]
32	192.168.7.13	192.168 7.3	TCP	21→35111 [SYN,ACK]
33	192.168 7.3	192.168.7.13	TCP	35111→21 [ACK]
34	192.168 7.3	192.168.7.13	FTP	Request: USER zoe
35	192.168.7.13	192.168 7.3	FTP	Response: 331 Enter password.
36	192.168 7.3	192.168.7.13	TCP	35111→21 [PSH, ACK]
37	192.168 7.3	192.168.7.13	FTP	Response: PASS Password1
38	192.168.7.13	192.168 7.3	TCP	21→35111 [PSH,ACK]
39	192.168.7.2	192.168.7.13	TCP	49876→22 [SYN]
40	192.168.7.2	192.168.7.13	TCP	63472→80 [SYN]
41	192.168.7.13	192.168.7.2	TCP	80→63472 [SYN,ACK]
42	192.168.7.2	192.168.7.13	TCP	63472→80 [ACK]
43	192.168.7.2	192.168.7.13	HTTP	GET /index.html?id=2845
44	192.168.7.13	192.168.7.2	TCP	80→63472 [ACK]
45	192.168.7.13	192.168.7.2	HTTP	HTTP/1.0 200 OK
46	192.168.7.2	192.168.7.13	TCP	63472→80 [FIN,ACK]
47	192.168.7.13	192.168.7.2	TCP	80→63472 [ACK]

Figure 1: Wireshark-type capture of all traffic passing between several computers on a network. You can assume that this is a valild packet capture, minor formating changes were made to fit the capture onto an exam.

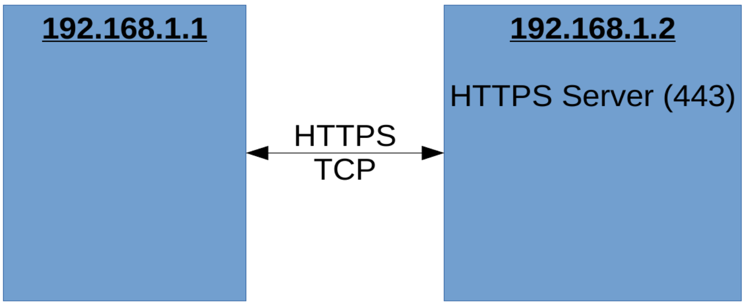

Figure 2: Example network drawing depicting a network with two computers that communicate using HTTPS over TCP with communications going in both directions. (192.168.1.1 connects to 192.168.1.2 which sends messages back.)

```
# Generated by iptables-save v1.6.0
# Generated on Sept 18 2017
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -p tcp -m tcp -s 192.168.7.2 --dport 443 -j ACCEPT
-A INPUT -p tcp -m tcp -s 192.168.7.2  -j DROP
-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A INPUT -j DROP

-A OUTPUT -m conntrack -ctstate RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -j DROP
COMMIT
```

Figure 3: Old firewall rules for a computer on the network. Lines that begin with a '#' are comments and do not need to appear in your answer.

2. **TLS protocol**

In class we discussed the TLS handshake protocol:

- The handshake begins when a client connects to a TLS-enabled server and sends a `client-hello` message requesting a secure connection and presenting the list of supported ciphers and hash functions.
- From this list, the server selects a cipher and a hash function that it also supports and notifies the client of the decision with the `server-hello` message.
- The server then provides identification in the form of a digital certificate. The certificate contains the server name, the trusted certificate authority (`T`) that vouches for the authenticity of the certificate, and the server's public encryption key. The certificate is signed under `T`'s signing key.
- The client confirms the validity of the certificate before proceeding (recall that `T`'s verification key is installed on the client's computer).
- To generate the session keys used for the secure connection, the client then encrypts a random number with the server's public key and sends the result to the server in the `client-key-exchange` message (which only the server can decrypt with its private key); both parties then use the random number to generate a unique session key for subsequent encryption and decryption of data during the session.
- The server sends a `finish` message which contains the hash of all messages sent in steps 1-5.
- The client compares the received hash and its own hash on his view of the messages sent in steps 1-5. If the hashes are different the client aborts the session, otherwise the client sends a `finish` message which contains the hash of all messages sent in steps 1-6. The session is now established.

(a) Consider a company network that connects to the Internet through a border gateway. The administrator wants to spy on the company's employees and decrypt all their HTTPS communications (which goes through the gateway). To this aim, the administrator (`A`) generates a CA signing and verification key, and installs the verification key on all employees' computers. All employees' browsers will now trust any certificate signed with `A`'s signing key. Explain how the administrator can now decrypt any HTTPS connection established by an employee's browser, to an external HTTPS website. Explain this man-in-the-middle attack at each step of the protocol. [*5 marks*]

(b) Assume an employee of the company connects to `acmebook.com`, and that his connection is being intercepted as described in question 2a. Can Acmebook's server detect this attack? Explain your answer. [*2 marks*]

QUESTION CONTINUES ON NEXT PAGE

(c) Assume an employee of the company connects to `acmebook.com`, and that his connection is being intercepted as described in question 2a. Can the user detect this attack? Explain your answer. [*3 marks*]

(d) To mount the man-in-the-middle attack you described in your answer to question 2a, the border gateway issues and signs a certificate under `A`'s signing key for the external domain (for example for `acmebook.com`). Suppose the certificate issued by the gateway is identical to the one received by `acembook.com` in step 3, except for the CA's name, the public key of `acamebook.com`, and the CA's signature. Suppose also that the border gateway does not do any checks on the certificate sent by `acmebook.com`. Explain how an attacker outside the company can now mount man-in-the-middle attacks on the company's employees. Describe your attack at each step of the protocol. [*5 marks*]

(e) A previous version of the TLS protocol (called SSL2) is subject to an attack called ROBOT: an attacker can iteratively query a server running SSL2 to decrypt previously captured sessions of the SSL/TLS handshake protocol. But many servers support both SSL2 and more recent versions of TLS, to allow connections from older clients. Explain how an active attacker can exploit this, to learn the session key established through the more recent TLS protocol. Describe your attack. [*5 marks*]

(f) Suppose a remote server that supports both SSL2 and the more recent TLS 1.2, to allow to older clients which only implement SSL2 to connect to it. How could the remote server defend against the attack from question 2e? [*5 marks*]

3. **CSRF attacks and defenses**

(a) In class we discussed Cross Site Request Forgery (CSRF) attacks. Explain what a CSRF attack is. *[4 marks]*

(b) Can HTTPS prevent CSRF attacks? Explain your answer. *[2 marks]*

(c) Can session cookies solely prevent CSRF attacks? Explain your answer. *[3 marks]*

(d) The `Referer` HTTP header contains the URL of the previous page visited. If you click on a link on a page, a GET/POST request is issued with the URL of this page as the value for the referer header. Explain how the `Referer` HTTP header can be used to prevent CSRF. *[4 marks]*

(e) A common defense against CSRF attacks is the introduction of anti-CSRF tokens in the DOM of every page (often as a hidden form elements) in addition to the cookies. An HTTP request is accepted by the server only if it contains both a valid cookie and a valid anti-CSRF token in the POST parameters. Why and when does this prevent CSRF attacks? Explain your answer. *[4 marks]*

(f) One way of choosing the anti-CSRF token is to choose it as a fixed random string. The same random string is used as the anti-CSRF token in all HTTP responses from the server. Does this prevent CSRF attacks? If your answer is yes, explain why. Otherwise describe an attack. *[4 marks]*

QUESTION CONTINUES ON NEXT PAGE

(g) Suppose `bank.com` has a 'Transfer money' link which links to the following page:

```
<p> Transfer details:
<form action="/transfer" method="post">
   <input type="hidden" name="user" value="{{username}}">
   <input type="hidden" name="CSRFToken" value="OWY4Nm...4ODRjN2Q2">
   Recipient's account: <input type="input" name="account"/><br>
   Amount to transfer: <input type="input" name="amount"/><br>
   <input type="submit" value="Transfer now"/>
</form>
```

The web server replaces {{`username`}} with the username of the logged-in user who wants to do the transfer. The implementation of /transfer is given by the following pseudocode:

```
if validate_login_cookie(request.parameters['user'],
                         request.cookies['login_cookie'])
then transfer_money(request.parameters['user'],
                    request.parameter['account'],
                    request.parameters['amount']);
     return '<p>Money successfully transfered</p>'
else return '<p>Sorry, ' + request.parameters['user']
                                         + ', an error occurred.</p>'
```

where `validate_login_cookie()` checks that the cookie sent by the browser is authentic and was issued to the specified username. Assume that `login_cookie` is tied to the user's account and difficult to guess. Is `bank.com` vulnerable to a CSRF attacks? [*4 marks*]