ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Family Name: _____

First Name: _____

Section: _____

Cryptography and Security Course (Security Part)

Midterm Correction

December 15th, 2006

This document consists of 7 pages.

Instructions

Books and lecture notes are *not allowed.*

Electronic devices are *not allowed.*

Answers must be written on the exercises sheet.

Answers can be written either in French or in English.

Questions of any kind will certainly *not* be answered.
Potential errors in these sheets are part of the exam.

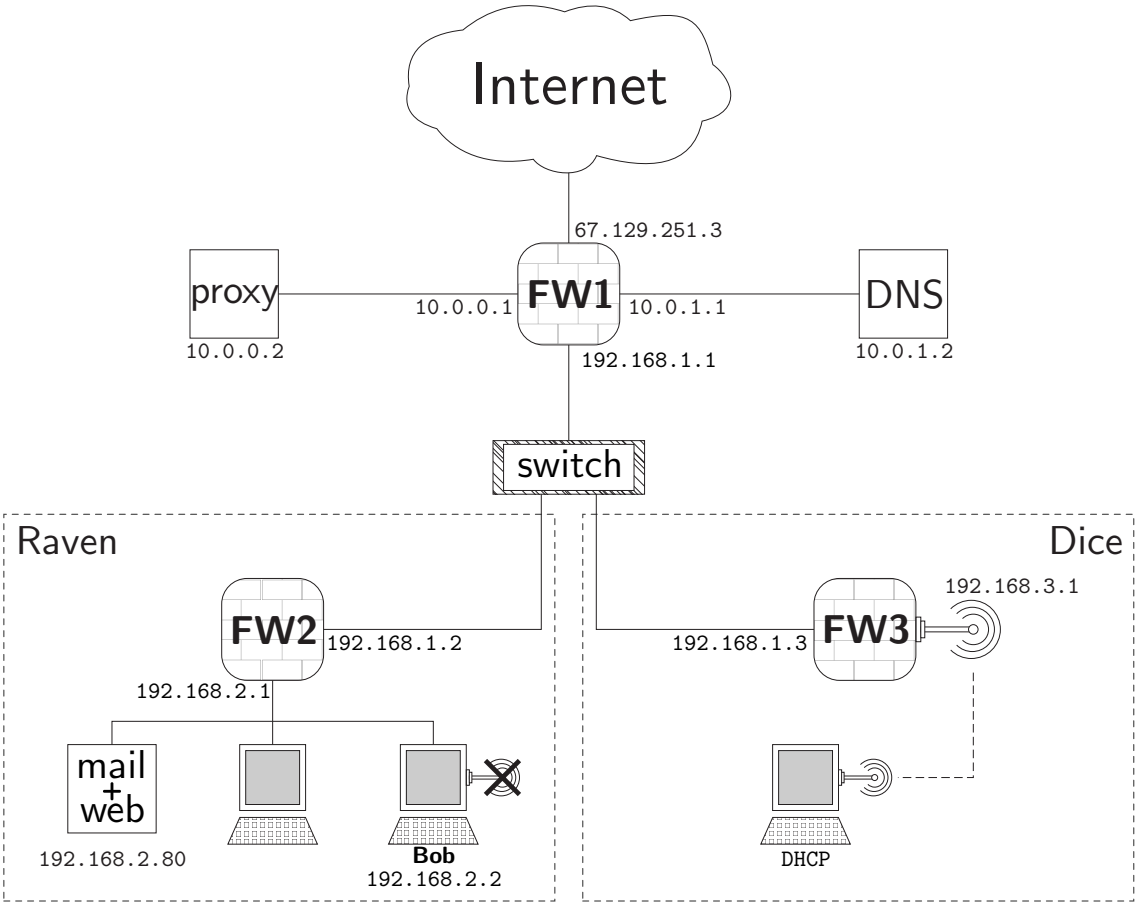

Figure 1: Map of the network infrastructure of 12 long street.

At the number 12 of long street there is an office building. In this building, companies who rent an office are offered an internet access through the infrastructure, that is pre-installed in the building. The infrastructure is represented in Figure 1 and is composed of:

- the main firewall **FW1** filtering all incoming/outgoing connections
- a proxy (`10.0.0.2`) ensuring many roles:
  - HTTP and HTTPS proxy for anyone inside the building, using ports `8080` and `8443`.
  - mail relay: it checks *incoming* mail for viruses and relays it to the mail servers of the companies. Anyone inside the building can also use it to *send* mail.
  - HTTP inverse proxy: it relays (and filters) incoming connections to the web servers of the companies.
- a DNS server (`10.0.1.2`) usable by anyone inside the building and responsible for the domain names of companies in the building (thus accessible from the internet too).
- the main switch used to connect **FW1** to all the companies sub-networks.

We focus on two companies in particular: Raven and Dice. Raven has quite a few employees (including **Bob**) and has its own mail+web server, behind a firewall **FW2**. Dice is much smaller and has little money to invest on infrastructures. They simply own a Linksys WRT54G wireless router **FW3**, have no servers, and connect their machines by WiFi, using DHCP.

We first focus on the configuration of **FW1**. It is a stateful firewall doing both static and dynamic NAT. Anyone behind the switch (which means, inside the companies networks) should be able to open a network connection towards any machine on the internet. And both the proxy and the DNS server should be able to fulfil their roles.

1. Shortly explain why **FW1** needs to do both static and dynamic NAT.

Static NAT: makes the internal web servers accessible from the Internet (through the proxy).

Dynamic NAT: makes it possible for internal machines to connect to the Internet.

(4 points)

2. Complete the static NAT Table 1 below so that the proxy and DNS server work properly. For info, HTTP, HTTPS, and SMTP respectively use port 80, 443, and 25. DNS uses port 53 and both UDP protocol and TCP protocol (for zone transfer).

Table 1: Static NAT table for **FW1**.

External		Internal		
source	port	destination	port	protocol
67.129.251.3	80	10.0.0.1	80	tcp
67.129.251.3	25	10.0.0.1	25	tcp
67.129.251.3	53	10.0.1.2	53	any

(4 points)

3. When **Bob** wants to visit (for the first time) the web site `www.bitsoup.org`, using the proxy, many connections have to be established (including DNS requests). List all these connections in chronological order, each time stating which protocol it uses.

1. **Bob** $\xrightarrow{\texttt{tcp}}$ proxy : request for the web page `www.bitsoup.org`
2. proxy $\xrightarrow{\texttt{udp}}$ DNS : request to resolve name `www.bitsoup.org`
3. DNS $\xrightarrow{\texttt{udp}}$ external DNS server : request to resolve name `www.bitsoup.org`
4. proxy $\xrightarrow{\texttt{tcp}}$ `www.bitsoup.org` : HTTP request for the page.

(4 points)

4. Now it is time to write the firewall filtering rules for **FW1**. For simplicity, 3 zones have been defined: `internal` corresponding to the IPs of the firewalls of the companies (`192.168.1.0/24`), `dmz_DNS` corresponding to the DMZ containing the DNS server (`10.0.1.0/24`), and `dmz_proxy` corresponding to the DMZ containing the proxy (`10.0.0.0/24`). As explained in the intro, many services have to be available both for the `internal` users and for anyone on the internet, and as always when configuring a firewall, only the necessary ports should be accessible in each zone.

Complete the Table 2 on next page with the filtering rules that a *good* system administrator would write. Remember that anyone behind the switch (which means, inside the companies networks) should be able to open a network connection towards any machine on the internet, and that both the proxy and the DNS server should be able to fulfil their roles described in the intro.

Note: the way rules are ordered and grouped will be taken into account when grading your answer. Please follow the recommendations of the lecture notes. Even if your rules work as intended, a bad organisation will not give you the maximum amount of points.

Table 2: Filtering rules for **FW1**.

source	port	destination	port	protocol	action
any	any	dmz_DNS	53	any	allow
any	any	dmz_DNS	any	any	deny
dmz_DNS	any	internet	53	udp	allow
dmz_DNS	any	any	any	any	deny
any	any	dmz_proxy	25	tcp	allow
any or internet	any	dmz_proxy	80	tcp	allow
internal	any	dmz_proxy	8080	tcp	allow
internal	any	dmz_proxy	8443	tcp	allow
any	any	dmz_proxy	any	any	deny
dmz_proxy	any	any	25	tcp	allow
dmz_proxy	any	any	80	tcp	allow
dmz_proxy	any	internet	443	tcp	allow
dmz_proxy	any	any	any	any	deny
any	any	internal	any	any	deny
internal	any	any	any	any	allow
any	any	any	any	any	deny,log

(8 points)

Now that the global infrastructure of the building is well configured we can go on with the configuration of the different companies internal networks. The two companies have very different security policies. On one hand, Raven wants to block as much traffic as possible with **FW2** in order to secure its network: the only incoming traffic allowed should be towards the mail+web server, and the only outgoing traffic allowed should be to send mails (still through the mail+web server) and to browse the web (both HTTP and HTTPS). The configuration of **FW2** is given in Table 3 below, where `Raven` designates the IP addresses `192.168.2.0/24`.

On the other hand, Dice does not want to loose any time with security considerations and entirely relies on the security that the building's infrastructure and firewall offer: **FW3** is configured to allow any outgoing connection. Moreover, **FW3** uses un-encrypted WiFi, with a simple MAC address filter to control the computers which can access their network.

Table 3: Filtering rules for **FW2**.

source	port	destination	port	protocol	action
10.0.0.2	any	192.168.2.80	80	tcp	allow
10.0.0.2	any	192.168.2.80	25	tcp	allow
any	any	Raven	any	any	deny
192.168.2.80	any	10.0.0.2	25	tcp	allow
Raven	any	10.0.0.2	8080	tcp	allow
Raven	any	10.0.0.2	8443	tcp	allow
Raven	any	10.0.1.2	53	udp	allow
Raven	any	any	any	any	deny
any	any	any	any	any	deny,log

5. First, one question remains: what is the use of an HTTP and HTTPS proxy if all the connections towards the internet are allowed by **FW1**? Of course it can be used as a filter or a cache (as any proxy), but in the present configuration it also allows much more precise filtering in **FW2**. In what sense? Explain why this is important for the security policy of Raven's network.

Without the proxy, allowing to browse the web from Raven's network would require **FW2** to allow any outgoing connection towards the internet on ports 80 and 443. Going through the proxy makes sure that these connections are really HTTP and HTTPS connections, and not any other protocol using port 80. The firewall can do filtering at the IP level, but the proxy does filtering at the application level.

(3 points)

6. Allowing access to an HTTPS proxy however opens possibilities to bypass the restrictions imposed by **FW2**. Explain how **Bob** could connect to an SSH server on his home machine (outside the building at 12 long street) even though **FW2** should normally prevent it.

HTTPS proxy are very similar to SOCKS proxy. As the content they relay is encrypted they cannot do any *application level* filtering. If **Bob** configures is home SSH server to run on port 443 and issues the HTTPS proxy a `connect` request towards his home machine he can establish a sort of tunnel through which he can connect to his SSH server normally.

(3 points)

7. **Bob** is a big fan of peer-to-peer downloads and mostly uses BitTorrent and eMule. He noticed that the bandwidth available at work is much larger than what he can afford for his home internet connection. He decides to use the same trick as for his SSH connection but here some problems occur: he cannot connect to the `udp`-based Kad network used by eMule, and gets very bad transfer rates for both BitTorrent and eMule with error messages stating that he (**Bob**'s machine) is "not connectable". Explain briefly what "not connectable" could mean and why this happens.

In peer-to-peer network a peer who wants to download data from another has to open a direct network connection to this machine. When using NAT, if no static entry is present to redirect this incoming connection then it cannot be established. "not connectable" thus means that the other peers are not able to connect to **Bob**'s machine. This happens for many reasons: no static NAT entries in neither **FW1** nor **FW2** will redirect such a connection correctly, but also no rule will allow such incoming traffic!
Concerning `udp`, there are also many reasons why this will not work: first the proxy will not relay `udp` packets, then neither **FW1** nor **FW2** will let `udp` packets through.

(3 points)

8. Being a big addict of James Bond, **Bob** decides that he cannot take it any longer: his life cannot go on if he cannot download a screener of Casino Royale. Going against all the security rules at Raven, he decides to activate the wireless network card on his computer and looks for open WiFi networks. Give the main reason why Raven forbids the use of wireless network cards in its internal network.

Having a WiFi card active in Raven's network will break the *choke point* rule. In a network, all traffic should go through a single choke point where all the necessary security measures can be taken. This is the case with **FW2**. If someone uses a WiFi connection, it will not go through **FW2** and traffic will not be filtered. Such a secondary connection would be a weakness in Raven's network.

(3 points)

9. After activating his wireless network card, **Bob** detects the WiFi network of Dice but is unable to connect to it because of the filter on the MAC addresses. Willing to do anything to bypass this protection, he puts his wireless card in `monitor` mode and starts sniffing Dice's network. Here is what he was able to sniff:

```
Source            Destination       Protocol Info
...
192.168.3.100     10.10.1.2         DNS    Standard query A lasecwww.epfl.ch
10.10.1.2         192.168.3.100     DNS    Standard query response CNAME lasecpc28.epfl.ch A 128.178.73.88
192.168.3.100     128.178.73.88     TCP    3978 > http [SYN] Seq=0 Len=0 MSS=1460
128.178.73.88     192.168.3.100     TCP    http > 3978 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1380
192.168.3.100     128.178.73.88     TCP    3978 > http [ACK] Seq=1 Ack=1 Win=16560 Len=0
192.168.3.100     128.178.73.88     HTTP   GET /cs2007/ HTTP/1.1
128.178.73.88     192.168.3.100     TCP    http > 3978 [ACK] Seq=1 Ack=532 Win=6432 Len=0
128.178.73.88     192.168.3.100     HTTP   HTTP/1.1 302 Found (text/html)
192.168.3.100     128.178.2.9       TCP    3980 > https [SYN] Seq=0 Len=0 MSS=1460
128.178.2.9       192.168.3.100     TCP    https > 3980 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1380
192.168.3.100     128.178.2.9       TCP    3980 > https [ACK] Seq=1 Ack=1 Win=16560 Len=0
192.168.3.100     128.178.2.9       TLSv1  Client Hello
128.178.2.9       192.168.3.100     TCP    https > 3980 [ACK] Seq=1 Ack=151 Win=5840 Len=0
128.178.2.9       192.168.3.100     TLSv1  Server Hello,
128.178.2.9       192.168.3.100     TLSv1  Certificate
192.168.3.100     128.178.2.9       TCP    3980 > https [ACK] Seq=151 Ack=2290 Win=16560 Len=0
192.168.3.100     128.178.2.9       TLSv1  Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
128.178.2.9       192.168.3.100     TLSv1  Change Cipher Spec, Encrypted Handshake Message
...
192.168.3.100     128.178.2.9       TLSv1  Application Data, Application Data, Application Data
128.178.2.9       192.168.3.100     TCP    https > 3990 [ACK] Seq=139 Ack=1078 Win=8112 Len=0
00:0f:66:c9:34:37 00:0f:66:74:0e:41 ARP    Who has 192.168.3.100?  Tell 192.168.3.1
00:0f:66:74:0e:41 00:0f:66:c9:34:37 ARP    192.168.3.100 is at 00:0f:66:74:0e:41
128.178.2.9       192.168.3.100     TLSv1  Application Data, Application Data
192.168.3.100     128.178.2.9       TLSv1  Encrypted Alert
192.168.3.100     128.178.2.9       TCP    3990 > https [FIN, ACK] Seq=1115 Ack=885 Win=15676 Len=0
128.178.2.9       192.168.3.100     TLSv1  Encrypted Alert
192.168.3.100     128.178.2.9       TCP    3990 > https [RST, ACK] Seq=1116 Ack=922 Win=0 Len=0
128.178.2.9       192.168.3.100     TCP    https > 3990 [FIN, ACK] Seq=922 Ack=1078 Win=8112 Len=0
192.168.3.104     192.168.3.1       TCP    4778 > http [SYN] Seq=0 Len=0 MSS=1460
192.168.3.1       192.168.3.104     TCP    http > 4778 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
192.168.3.104     192.168.3.1       TCP    4778 > http [ACK] Seq=1 Ack=1 Win=17520 Len=0
192.168.3.104     192.168.3.1       HTTP   GET / HTTP/1.1
192.168.3.1       192.168.3.104     TCP    http > 4778 [ACK] Seq=1 Ack=407 Win=6432 Len=0
192.168.3.1       192.168.3.104     TCP    [TCP segment of a reassembled PDU]
192.168.3.1       192.168.3.104     HTTP   HTTP/1.0 401 Unauthorized (text/html) (WWW-Authenticate: Basic realm="WRT54G")
192.168.3.104     192.168.3.1       TCP    4778 > http [ACK] Seq=407 Ack=307 Win=17215 Len=0
192.168.3.104     192.168.3.1       TCP    4778 > http [FIN, ACK] Seq=407 Ack=307 Win=17215 Len=0
192.168.3.1       192.168.3.104     TCP    http > 4778 [ACK] Seq=307 Ack=408 Win=6432 Len=0
192.168.3.100     128.178.73.88     HTTP   GET /cs2007/?key=dz525wk2iqaaq1esft3eajlc4w81jm3y HTTP/1.1
128.178.2.9       192.168.3.100     TCP    https > 3990 [ACK] Seq=923 Ack=1115 Win=8112 Len=0
128.178.2.9       192.168.3.100     TCP    https > 3990 [ACK] Seq=923 Ack=1116 Win=8112 Len=0
128.178.73.88     192.168.3.100     TCP    http > 3978 [ACK] Seq=637 Ack=987 Win=7504 Len=0
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
192.168.3.100     128.178.73.88     TCP    3978 > http [ACK] Seq=987 Ack=2381 Win=16560 Len=0
...
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
192.168.3.100     128.178.73.88     TCP    3978 > http [ACK] Seq=987 Ack=21701 Win=16560 Len=0
128.178.73.88     192.168.3.100     HTTP   HTTP/1.1 200 OK (text/html)
192.168.3.100     128.178.73.88     HTTP   GET /cs2007/updates.rss HTTP/1.1
128.178.73.88     192.168.3.100     TCP    http > 3978 [ACK] Seq=22369 Ack=1502 Win=8576 Len=0
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
192.168.3.100     128.178.73.88     TCP    3978 > http [ACK] Seq=1502 Ack=25129 Win=16560 Len=0
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
192.168.3.100     128.178.73.88     TCP    3978 > http [ACK] Seq=1502 Ack=27889 Win=16560 Len=0
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
192.168.3.100     128.178.73.88     TCP    3978 > http [ACK] Seq=1502 Ack=30649 Win=16560 Len=0
128.178.73.88     192.168.3.100     TCP    [TCP segment of a reassembled PDU]
192.168.3.104     192.168.3.1       TCP    4793 > http [SYN] Seq=0 Len=0 MSS=1460
192.168.3.1       192.168.3.104     TCP    http > 4793 [SYN, ACK] Seq=0 Ack=1 Win=5840 Len=0 MSS=1460
192.168.3.104     192.168.3.1       TCP    4793 > http [ACK] Seq=1 Ack=1 Win=17520 Len=0
192.168.3.104     192.168.3.1       HTTP   GET / HTTP/1.1 (Authorization: Basic T3VjaCEgTXk6YXNzIGh1cnRzLg==)
192.168.3.1       192.168.3.104     TCP    http > 4793 [ACK] Seq=1 Ack=458 Win=6432 Len=0
192.168.3.1       192.168.3.104     TCP    [TCP segment of a reassembled PDU]
192.168.3.1       192.168.3.104     TCP    [TCP segment of a reassembled PDU]
192.168.3.104     192.168.3.1       TCP    4793 > http [ACK] Seq=458 Ack=307 Win=17215 Len=0
128.178.73.88     192.168.3.100     HTTP   HTTP/1.1 200 OK (application/rss+xml)
192.168.3.100     128.178.73.88     TCP    3978 > http [ACK] Seq=1502 Ack=32499 Win=16560 Len=0
...
```

What information in this trace can he use to connect to Dice's network?

The WiFi network at Dice is only secured with a MAC address filter. In order to connect **Bob** simply needs to learn the MAC address of a machine which is allowed to connect to the network, wait for this machine to disconnect, and use its MAC address in his own WiFi card. Of course he needs a card for which it is possible to change the initial MAC address.
From what he was able to sniff the interesting part is the ARP request where he learns the MAC address of the access point (`00:0f:66:c9:34:37`) which is pretty useless and the mac address of one computer allowed to connect (`00:0f:66:74:0e:41`) which he will have to use.

(4 points)

What other "useful" information is visible in this trace?

There are two other useful pieces of information in the trace. First we see an HTTP request:
`GET /cs2007/?key=dz525wk2iqaaq1esft3eajlc4w81jm3y HTTP/1.1`
This request contains a tocken which can be used to access the page `/cs2007/` on the lasec web server. This tocken has a short validity period but could be used by **Bob** to do some HTTP hijacking and access the page without being allowed to do so. If we look at the trace more precisely, we see first a request `GET /cs2007/ HTTP/1.1` to which the server answers by a 302 error message: this indicate a temporary redirection. The user is then redirected to the tequila server where an HTTPS connection is established and where the user can authenticate using Gaspar. At the end, after the authentication the user is redirected to the `/cs2007/` page with a valid token allowing him to connect. This token is not encrypted and can be stolen by **Bob**.
The second interesting piece of information concerns another user (with IP address `192.168.3.104`). This user uses HTTP to connect to the IP `192.168.3.1` which is the IP of the **FW3**. If first gets a 401 error message indicating that he has to authenticate for the realm "WRT54G" before being able to see the page. Luckily for **Bob**, this uses Basic authentication over HTTP (thus non encrypted password based authentication). When he issues the request:
`GET / HTTP/1.1 (Authorization: Basic T3VjaCEgTXk6YXNzIGh1cnRzLg==)`
this user is sending the login/password combination granting him access to the configuration page of the the WRT54G wireless router. He can either replay this packet to gain access to the configuration page, or simply read the log/password which are simply base64 encoded in the string `T3VjaCEgTXk6YXNzIGh1cnRzLg==`. He can them easily add his own MAC address to the list of allowed MAC addresses and will no longer need to use the MAC address of the other computer.

(4 points)

Security part

Any attempt to look at
the content of these pages
before the signal
will be severly punished.

Please be patient.

Security part