EP2500 Networked Systems Security
Homework 2

Emonet Camille
Fleitz Pierre
04/01/2016

Work distribution:

Camille 65% - Pierre 35%

1. Unauthorized Access

Exercise 1 *Firewalls*

1) – Stateless firewall: the firewall examines each packet independently of the previous packets and compares it with a list of preconfigured rules.
These firewalls need little memory and are very fast, but they cannot make a complex decision when it is about an established communication between two hosts.

   – Stateful firewall: the firewall maintains context about active sessions. If a packet does not match an executing connection, it will be evaluated according to the ruleset for a new connection.

   The Stateful Firewall is better, and for instance in our case we use SSH to externally manage the internal web server, so it's better to have a firewall which remember the connection establishment.

2) Since we choose a Stateful firewall we know that we only need to set up rules for NEW connection excepted for the Facebook connection for which we need NEW and ESTABLISH.
Moreover, we have an ALLOW-ALL firewall and the *rules (7) "No other traffic is allowed"*, we only need to specify the banned traffic (authorization rules are useless with the allow-all policy).
However, we can notice that *InSecure AB* company choose a wrong policy for its firewall, an ALLOW-NOTHING would be a better solution in this case…

Rule N°	Direction	Source	Destination	Protocol	Src Port	Dest Port	State	Action
(1)	IN and OUT	*ANY	*ANY	ICMP			NEW	DROP
(2)	IN	207.46.130.0/24	*ANY	HTTPS			NEW	REJECT
(2)	IN	207.46.130.0/24	*ANY	SSH			NEW	REJECT
(3)	IN	*ANY	*ANY excepted 17.0.0.1	HTTPS			NEW	REJECT
(4)	IN	*ANY	*ANY excepted 17.0.0.1	SSH			NEW	REJECT
(4)	OUT	*ANY excepted 17.0.0.1	*ANY	SSH			NEW	REJECT
(5)	IN	*ANY	*ANY	HTTP			NEW	REJECT
(6)	OUT	17.0.0.0/8	69.171.239.12	HTTP			NEW	REJECT
(6)	OUT	17.0.0.0/8	69.171.239.12	HTTP			ESTABLISH	REJECT
(6)	OUT	17.0.0.0/8	69.171.239.12	HTTPS			NEW	REJECT
(6)	OUT	17.0.0.0/8	69.171.239.12	HTTPS			ESTABLISH	REJECT
(1)–(2)	IN and OUT	*ANY	*ANY	*ANY excepted SSH ICMP HTTP HTTPS			NEW	REJECT

(1) Obvious

(2) Since we reject all connection excepted with the protocols SSH, HTTP and HTTPS we need to create rules for them.

- SSH: obvious
- HTTP: we reject all incoming packet (it is not a problem with the rule (5) since we have a Stateful firewall which allow the HTTP establish connection)
- HTTPS: obvious

(3) We reject all HTTPS incoming packet excepted the one with the internal server as destination.

(4) We reject all SSH packet (incoming and outgoing) excepted when the communication is with the internal server.

(6) We need to reject HTTP/HTTPS both new and establish state because if a previous connection is established with another web site the Stateful firewall will not check if the new destination is Facebook or not.

3) – REJECT: prohibit a packet from passing and send a response in case of failure.
"connection established" VS "connection reject"
– DROP: prohibit a packet from passing but with no response in case of failure.
"connection established" VS "connection time out"

With a reject action we know right now if the connection is reject or not, while with a drop action we need to wait for the connection to time-out.

One could think that a drop is better because an attacker will need to wait a time-out at each attack attempt, but the waste of time is minimal for an attacker while a user need to know if its connection is reject or out-of-time.
So, reject is a better way.

4) Cache poisoning attack:

- Aim of the firewall policy: to prevent an intrusion
- Who can go in with our policy? external admin with SSH
- Attack: the attacker waits for an external admin SSH connection to the internal web server. He/She steals the SSH key and spoofs his/her address to look alike the external admin. As we have a Stateful firewall, the attacker would be able to talk with the internal server in SSH and it is the end.

Exercise 2 *Public Key Infrastructure*

1) If a British car receives a certificate from a car from Zurich, the chain of trust would be like that:
< message, signature of the Swiss car, cert $_{CA\ CAR}$*, cert* $_{CA\ ZURICH}$*, cert* $_{CA\ SWITZERLAND}$ *>*
The car in Britain sees that the certificate of the Swiss car is certified by the CA of Zurich, but it does not trust this CA, so it continues to examine the chain and sees that the certificate of CA Zurich is certified by CA Switzerland which is cross certified by CA Europe.
Since CA Europe is the trust-anchor for the British car, it will trust the Swiss car certificate.
So, a car from Zurich is able to travel in UK.

2) If the German CA gets compromised, the first thing to do is to invalidate the certificate of Germany.
For that we use the Certificate Revocation Lists (CRLs) and add into it the serial number of Germany. It is the CA Europe which have to publish this CRL.
Since the other car will check CRL of CA Europe before to decide to talk to a German car, they will know that German cars are compromised.

3) In this scenario a car from Zurich will not be able to communicate with a car from Munich.
Indeed, when the car from Munich will want to respond to the car from Zurich, the trust chain will be like that:
< message, signature of the German car, cert $_{CA\ CAR}$*, cert* $_{CA\ MUNICH}$*, cert* $_{CA\ GERMANY}$*, cert* $_{CA\ EUROPE}$*>*
Since the car from Zurich does not trust neither CA of the German car nor CA Munich and CA Germany, it will ask to CA Europe and discover that the German

serial number is on the CRL and then, not accept to communicate with the German car.

4) Before to trust a message (and use the Key, Signature, etc. to decode it) a car needs to question the CA it trusts and certificate the other element about the CRL. The car needs to verify is the serial number of the element is or not in the list of revocation.
   If yes, the car will not trust the message et reject it.
   If no, the car will decode the message.

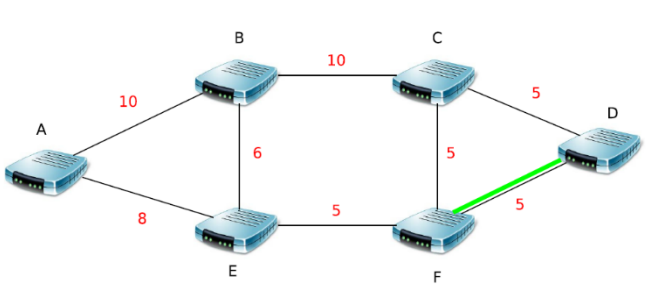
   f = 4 Hz
   T = 0.25 sec
   So, 1 message each 0.25 sec

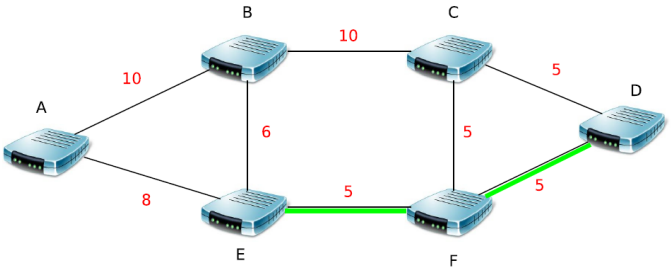
   Then, after 250 sec V_{CH1} sent 1000 messages (with ideal conditions).
   The CRL need to be publish (refresh) every 250 sec, approximately every 4 min.

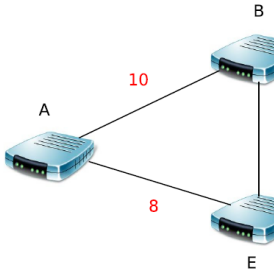
2.Secure Routing

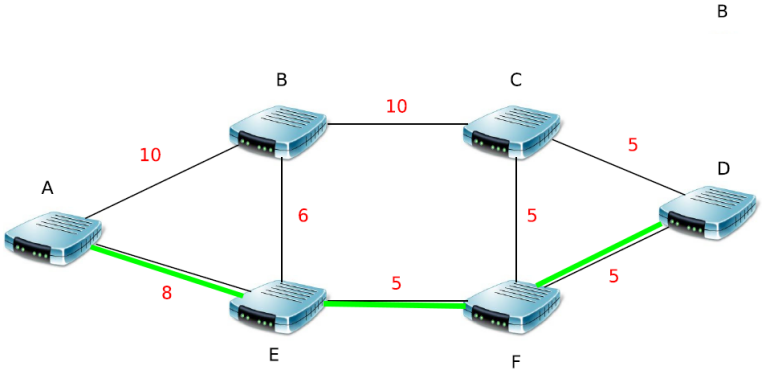
Exercise 3 *Secure Link-State Routing*

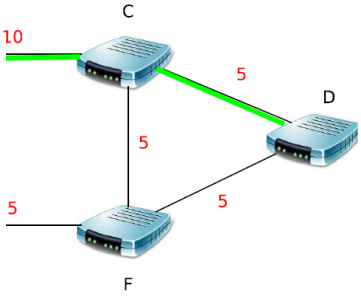
a) Shortest paths:











- D -> C
  (D, C) =5
- D -> F
  (D, F) =5
- D -> B
  (D, C, B) =15
- D -> E
  (D, F, E) =10
- D -> A
  (D, F, E, A) =18

b) B sends the erroneous LSU to A, E and C.
A and C ignore the erroneous LSU but E retransmits it to F.
F transmits the erroneous LSU to D and D updates its routing table.

Before the erroneous LSU:

- D -> C
  (D, C) =5

After the erroneous LSU:

D -> C
(D, C) =5

• D -> F (D, F) =5	D -> F (D, F) =5
• D -> B (D, C, B) =15	D -> B (D, C, B) =15
• D -> E (D, F, E) =10	D -> E (D, F, E) =10
• D -> A (D, F, E, A) =18	**D -> A** **(D, C, B, A) =7**

Then, after the update the data originating D and destined to A will take the wrong way (pass through the faulty router).

c) If C and A transmit again their LSU after the erroneous LSU of B, the routers B, F and D will update their routing table with the rights paths and costs.
And the path from D to A will be again **(D, F, E, A) =18**. Then, even if the cost seems to increase in comparison with the erroneous LSU from B, it is a positive thing.

d) To make this possible the routers need a **private key** to be able to sign their LSU and then, the other router would be able to verify this signature by using the public key of the sender. Since we need public/private keys, we also need a **third authorit**y to manage and give these keys.

e) To verify the freshness of a LSU we can add a field with an ID sequence which increase by +1 for each new LSU sent (each update). Then, if ID sequence in the router memory is lower than the ID sequence of the received LSU that means it is a fresh information.

And of course, we can also add a TTL (Time To Live) to prevent receiving too old LSU.

Thanks to the digital signature, in addition to verify the authentication we also assure the non-repudiation of a LSU. An attacker cannot send fake information and hides his identity.

f) Signed LSU cannot prevent the effect of the malfunction in our example. However, with the digital signature we can be certain that the malfunctioning router B sends the erroneous LSU and if we discover the problem we can add a rule that ask to reject all the LSU from B.

g) If B sends the fake LSA cost(B,F)=1 to its neighbors after a little time all the routers will update their routing table and D will have:

- D -> C
  (D, C) =5

- D -> F
  (D, F) =5
- D -> B
  (D, F, B) =6
- D -> E
  (D, F, E) =10
  D -> A
  (D, F, B, A) =16

h) This do not solve the problem; the attacker B can send the fake LSU as many times as he wants.
However, since we have the digital signature, we can find the identity of the attacker and say to the other to not trust the LSA / LSU from B. That is equivalent to create a List with the identity of the non-trusted router.

3.Data Plane Attacks

Exercise 4 *CASTOR*

1) H = efe50337
b_k = 50d02858
f_k = (c7e2b366; 23f140ad; 95294aec; a4745acb)
x1= c7e2b366
x2= 23f140ad
x3= 95294aec
x4= a4745acb

h(h(h(h(h(b_k)||x1)||x2) ||x3) ||x4) =? H

➔ h(b_k)= h(50d02858)= 045afeae$_{f6d4b1b351aa35d50c84d424}$
➔ h(b_k) || x1= 045afeaec7e2b366
➔ h(h(b_k) || x1)= a7626801$_{30cabd8feb7fc68e50515a61}$
➔ h(h(b_k) || x1)||x2 = a762680123f140ad
➔ h(h(h(b_k) || x1)||x2) = 1461c8e3$_{718070f827086b6f541131ea}$
➔ h(h(h(b_k) || x1)||x2)||x3= 1461c8e395294aec
➔ h(h(h(h(b_k) || x1)||x2)||x3)= 9cb546a7$_{ff135ddd4643191c3dbbefc8}$
➔ h(h(h(h(b_k) || x1)||x2)||x3)||x4=9cb546a7a4745acb
➔ h(h(h(h(h(b_k) || x1)||x2)||x3)||x4)= efe50337$_{37345c91f2453871b3a1528b}$

So, the result is equal to H, then Bob will forward the packet.

2) H= 12328e72
b_k = 07fd01d5
fk = (5a9d0480; 93aded1d; d97d3412; 414833e5)

x1= 5a9d0480
x2= 93aded1d
x3= d97d3412
x4= 414833e5

h(h(h(h($h(b_k)$||x1)||x2) ||x3) ||x4) =? H

➔ $h(b_k)$= h(07fd01d5)= 0561a9ca$_{01d3ba79ece8055b0d523147}$
➔ $h(b_k)$ || x1= 0561a9ca5a9d0480
➔ h($h(b_k)$ || x1)= 833f7176$_{33a2f7bb70c9d95403da966c}$
➔ h($h(b_k)$ || x1)||x2 = 833f717693aded1d
➔ h(h($h(b_k)$ || x1)||x2) = c30c554b$_{b3796b7e7c73906e26086adc}$
➔ h(h($h(b_k)$ || x1)||x2)||x3= c30c554b d97d3412
➔ h(h(h($h(b_k)$ || x1)||x2)||x3)= ff4b94a7$_{0314866b12760c4ff35f8e6b}$
➔ h(h(h($h(b_k)$ || x1)||x2)||x3)||x4= ff4b94a7414833e5
➔ h(h(h(h($h(b_k)$ || x1)||x2)||x3)||x4)= 5475804c$_{bfb33b741a563ce97cc048c0}$

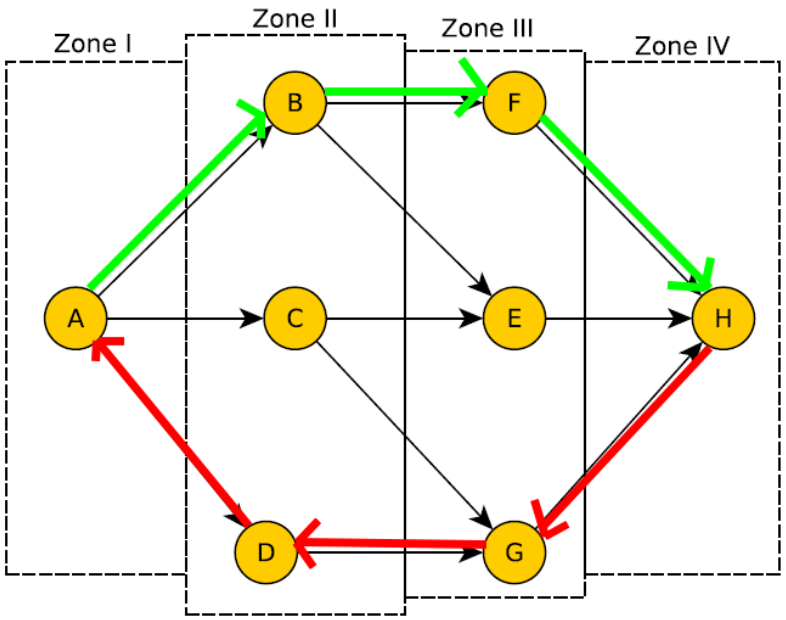
H is different than the result, then Alice will not forward the packet.

Exercise 5 *Data Plane Attacks*

1) If node C is captured, the probability that a packet sent from A to H is dropped is **p=1/3**

2) If node E is captured, the probability that a packet sent from A to H is dropped is
p = p(pass through B then E) + p(pass through C then E)
= (1/3)*(1/2) + (1/3)*(1/2)
= 1/3

3)



The probability that an ACK is dropped if node E is captured is
p = p(original packet not dropped) * p(ACK pass through E)
= 2/3 * 1/3
= 2/9

4) If the adversary has a resource with value 0.5, the best technique for him is to capture the node G.
Indeed, if A send its packet to D, the packet has 100% of chance to be dropped and if A send its packet to C, the packet has 50% of chance to be dropped.
And from H to A, all the nodes in the zone III are equals.
5) If the adversary has a ressource with value 2, the best technique for him is to capture all the nodes from the zone 3 : F,E,G. Then the total cost for him will be 1,5, but at the end, the probability for a packet to be dropped is 100%.
6) If the attacker decides to capture the node G, the probability to dropped a packet from A to H is
p = p(pass through C then G) + p(pass through D)
= 1/3*1/2 + 1/3 * 1
= ½
Then, if A send 1bytes/minute, 10*1= 10bytes in 10 minutes.
So 0.5*10=5 bytes dropped.

4.Domain Name System (DNS) Security

Exercise 6 *Birthdays*

1) Birthday paradox:
The birthday paradox is a probabilistic estimation of the number of people that must meet to have more than 50% of chances that two people in the group have their birthday on the same day.
It turns out that this number is only 23, which shocks a little the human intuition.
That means, from a group of 57 people, the probability is higher than 99%!

2) The BIND birthday attacks is so much more effective than the conventional spoofing because with the conventional spoofing an attacker send **n** spoofed replies for only one query (so, the probability of success is n/X), while with the BIND birthday attack we send **n** number of spoofed replies for **n** queries. Then, the probability of a collision (that is a success) is much higher with the BIND birthday rather than the conventional spoofing attack.

3) According to the article's formula, we have:

$$p = 1 - \left(1 - \frac{1}{t}\right)^{\frac{n(n-1)}{2}}$$

$\Leftrightarrow$

$$0.25 = 1 - \left(1 - \frac{1}{65535}\right)^{\frac{n(n-1)}{2}}$$

$\Leftrightarrow$

$$n = 195$$

Exercise 7 DNSSEC

1) The main difference between DNS and DNSSEC is that DNSSEC try to overcome every problem related to security that DNS couldn't fight. DNSSEC try to do that by signing all response data fields. It provides authentication, data integrity and authenticated denial of existence but not confidentiality.
2) DNSSEC solves several problems such as cache poisoning, man in the middle attacks and false zones also.
3) Because even if DNSSEC and SSL certificates both use the same principle for securing data, they serve a different purpose. SSL certificate encrypts the transmitted data while DNSSEC authenticates the data. Therefore we still gonna need DNSSEC in order to authenticate the data.
4) We want authenticated denial of existence replies in order to protect against false zones.

5.Wireless Security

Exercise 8 *Securing Web Services*

1) In order to authenticate herself to the KTH Library Web Server, Alice is going to use her Electronic ID. Alice sends to the KTH Library web server a message containing: <ID-Alice , CA-Alice , TTL=10 , CRC>.
   ID-Alice and CA-Alice, are used in order to be able to ensure the fact that Alice is speaking and not someone trying to impersonate her.
   To this message we can see that a TTL "TTL=10" is added. We use this in order to make sure of the freshness of the message, every time the message is passed the TTL is decreased, and once the TTL is equal to 0 the message should be discarded.
   At the end of the message we have a CRC. This CRC is used to ensure the integrity of the message. The CRC will be compute using the values of the ID-Alice and CA-Alice,

but not the TTL, as the value of TTL is made to change.
When KTH Library Web Server receives this message it will be able to make sure that Alice is the one she claims to be and then processing the application.

2) In order to augment our protocol and to add confidentiality we could encrypt the communication using a key known by Alice the KTH Library Web Server. This key could be a value calculated with ID-Alice and CA-Alice using a function f(ID-Alice , CA-Alice). Since Alice is supposed to be authenticated in the previous part the KTH Library Web Server can compute the key and then decrypt the message containing Alice's choice.
3) The vulnerability is clearly that if someone found his wallet and his smartcard, this person could use these informations to log in the KTH Library Web Server in the name of Bob and could borrow books or purchase them with Bob's money or Bob's responsibility, putting him in a bad position.
   To revoke his previous smartcard Bob will have to authenticate itself to KTH Administration, in order to be able to ask for a new smartcard, we must be sure that Bob really lost his card and that no one is trying to ask for a new one in the name of Bob.
   To do that we could simply use a student ID certificate or student ID card. Once we are sure that Bob actually lost his card, we need to revoke his old one. To do that we can send an update to the KTH Library Web Server saying that previous CA-Bob is not legit anymore, and delete it to the database of CA associated to ID-Bob and put a new CA.
   In the protocol previously described, since the old CA-Bob would be incorrect if someone tries to log in with the old smartcard the KTH Library Web Server will discard the connection and ask for a new one, that could only be accepted with the new CA-Bob.
4) In order to strengthen the protocol we could add a third part to the connection. Not only the student is going to need his ID-Student and CA-Student to log in but also to answer a question for example. When the first connection has been done, the system could ask the student to choose a question in a list of question and give an answer. Then even if the student looses his smartcard, nobody could log in using his identity, as he would miss information that only the student knows.
   After 3 attempts the KTH Library Web Server could automatically discard the connection and revoke the CA associated to the smartcard and send a mail to the student advertising him that someone tried, and failed to log in the web service using his smartcard.
5) They need to legitimize the CA delivered by the other school: CA-SU must advertise that CA-KTH can be trusted, and CA-KTH must advertise that CA-SU can be trusted. Then when someone from SU, for example, wants to log in KTH Library Web Server, KTH Library Web Server will check its CA and will see "CA-SU" and will be able to trust it.
6) If Alice wants to authenticate herself to SU e-library web server she will send: <ID-Alice , CA-Alice , TTL=10 , CRC>.
   SU e-library web server will lookup CA-Alice and see: ((CA-Alice)CA-KTH). Then SU e-library web server will understand that Alice is a KTH student under the authority of KTH Library Web Service, and that she can be trusted.

7) The problem is that in our protocol we need a third party to trust the person that tries to connect, the CA-KTH or CA-SU. Here we don't have this, if the database get hacked and someone get access to student's usernames and passwords they could use them and impersonate them. Plus, we could add that in our protocol explained before the book's choice is encrypted using a key computed from ID-Student and CA-Student or here we don't have this.
   We could address them by using a new function g, computing the key by using the username and the password: g(username,password). The third party could be a web server established by the Erasmus organization or Erasmus Sweden that could provide a CA for every Erasmus student and that every school in the area could trust, for example CA-ErasmusStockholm could be trusted by CA-KTH, CA-SU, etc.
8) There is a problem with the string "sql", such a protocol connection could be weak against any type of SQL Injection. A SQL injection attack consists of insertion or injection of a SQL query via the input data from the client to the application. In order to prevent this type of attack we could use stored procedures and prepared statements. (SQLi).
9) /

6.Web Security and Privacy

Exercise 9 *k-Anonymity*

First Name	Family Name	Age	Goals
Jimmy	*	28	12
*	Larsson	*	*
Jimmy	*	28	12
*	Larsson	*	*