Adatbiztonság a gazdaságinformatikában ZH
2015. december 16.

Név:

Neptun kód:

Pontozás: 1: 0-19, 2: 20-26, 3: 27-34, 4: 35-42, 5: 43-51

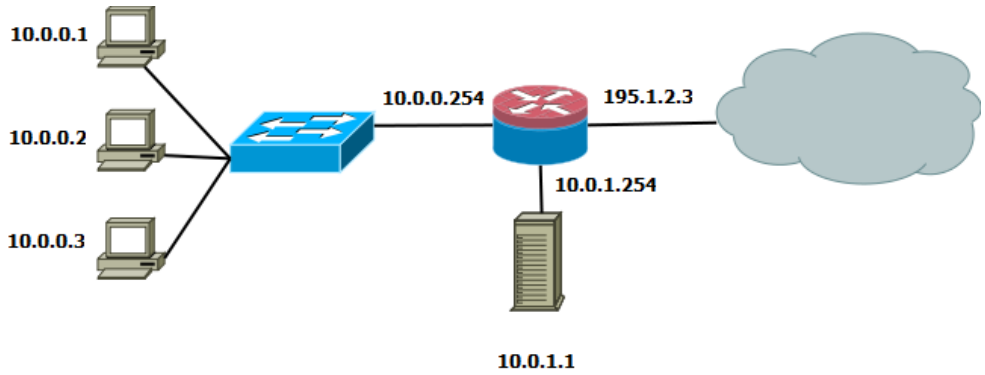
1. RSA algoritmus esetén a kulcsok előállításához p=101, q=113 prímekből indultunk ki.

a) Adja meg a lehető legkisebb kódoló kulcsként használható exponenst! (**3p**)
b) Határozza meg a dekódoló kulcsot! (A részletszámítások során indokolja, mit, miért végez el.) (**7p**)

Megoldás:
a.) $p_1=101$, $p_2=113$
$m = p_1*p_2 = 101*113 = 11413$
$fi(m) = (p_1-1)*(p_2-1) = 100*120 = 11200$
Az exponenssel kapcsolatos feltétel, hogy relatív prím legyen $fi(m)=2^6 5^2 7$ *-hez. A legkisebb ilyen szám a 3.*
Tehát e = 3.

*b.) euklideszi algoritmussal (egy lépésben): 11200=3733*3+1, ezért d= -3733= 7467.*

2. Tűzfal feladat



Az ábrán látható elrendezésben szeretnénk a Linux alapú tűzfalat bekonfigurálni. A belső hálózaton a gépek a 10.0.0.0/24 hálózatban vannak. A DMZ-ben (10.0.1.0/24) egy emailszerver működik a 10.0.1.1-es címen. A tűzfal lábai a következők: eth0 kifelé, eth1 befelé, eth2 a DMZ irányába. Írjon iptables parancsokat a következő formátumban a részfeladatok megoldásához (a paraméterek sorrendjét lehetőleg ne változtassa meg, az alapértelmezett szabályokra ne alapozzon):

```
iptables [-t TÁBLANÉV] -A LÁNC [-p PROTOCOL] [-i INIF] [-o OUTIF]
[-s SOURCE] [--sport SPORT] [-d DESTINATION] [--dport DPORT][--to
ADD:PORT] -j ACTION
```

1. A belső hálózaton lévő gépek elérhetik az emailszerver IMAP portját (143) a DMZ-ben, és az válaszolhat is, ha az interfészek és a címek megfelelőek (állapotmentes megoldást írjon, 2 parancs) (**4 pont**).
2. A tűzfalon futó SSH szerver (22-es port) csak a belső hálózatból kaphasson csomagot, és a tűzfal mint feladó általában is (nem csak SSH tekintetében) csak a belső hálózatnak küldhessen csomagot (ügyeljen a megfelelő lánc választásra, az interfészeket nem kell megadni, 4 parancs) (**6 pont**).

3. Az emailszerver a külső hálózatból is elérhető legyen a nyilvános cím megfelelő portjain (IMAP és IMAP over SSL (993) forgalom is lehetséges legyen, 2 parancs) (**6 pont**)
4. A tűzfalon áthaladó bármilyen TELNET szervernek címzett forgalom (23-as port) logolva legyen (1 parancs). (**2 pont**)
5. A tűzfalon fut egy Snort is. Mire lehet jó a következő szabály: (**2 pont**)

Alert tcp 10.0.0.0/24 any -> any 23 (content:"|anonymous|";)

Megoldás

1.
   a. iptables –A FORWARD –p tcp –i eth1 –o eth2 –s 10.0.0.0/24 –d 10.0.1.1 –dport 143 –j ACCEPT
   b. iptables –A FORWARD –p tcp –i eth2 –o eth1 –s 10.0.1.1 –sport 143 –d 10.0.0.0/24 –j ACCEPT
2.
   a. iptables –A INPUT –p tcp –s 10.0.0.0/24 –dport 22 –J ACCEPT
   b. iptables –A INPUT –p tcp –dport 22 –J DROP
   c. iptables –A OUTPUT –d 10.0.0.0/24 –J ACCEPT
   d. iptables –A OUTPUT –J DROP
3.
   a. iptables –t NAT –A PREROUTING –p tcp –d 195.1.2.3 –dport 143 –to 10.0.1.1:143 –J DNAT
   b. iptables –A PREROUTING –p tcp –d 195.1.2.3 –dport 993 –to 10.0.1.1:993 –J DNAT
4.
   a. iptables –A FORWARD –dport 23 –j LOG
5.
   a. Akkor küld jelzést, ha egy felhasználó anonymous-ként próbál meg belépni egy telnet szerverre, vagy valahol máshol a kommunikáció során átküldi ezt a sztringet (csak akkor működik feltétlenül jól, ha a telnet line-módban fut és nem karakter módban).

3. Tekintsük az alábbi `/etc/passwd` file részletet:

```
doris:x:1001:1002::/home/doris:/bin/bash
alex:x:1002:1003::/home/alex:/bin/bash
bob:x:1003:1004::/home/bob:/bin/bash
```

Az `/etc/group` file releváns része:

```
doris:x:1002:
alex:x:1003:
bob:x:1004:
teacher:x:1001:doris,alex
```

A fájl hozzáférési jogosultságok az alábbiak:

```
root /home #: ls -al /home
drwxr-xr-x  7 root  root  4096 dec   13 12.52 .
drwxr-xr-x 17 root  root  4096 2014 nov    4 ..
drwx------  3 alex  alex  4096 nov   23 00.03 alex
drwx------  3 bob   bob   4096 nov   22 11.42 bob
drwxr-xr-x  7 doris doris 4096 nov   29 01.30 doris
drwxr-xr-x  4 root  root  4096 dec   13 12.53 kozos

root /home #: ls -al /home/kozos
drwxr-xr-x 4 root  root    4096 dec   13 12.53 .
drwxr-xr-x 7 root  root    4096 dec   13 12.52 ..
drwxrwxr-x 2 doris teacher 4096 dec   13 13.18 d1
drwxr-xr-- 2 alex  teacher 4096 dec   13 13.57 d2

root /home #: ls -al /home/kozos/d1
drwxrwxr-x 2 doris teacher 4096 dec   13 13.18 .
drwxr-xr-x 4 root  root    4096 dec   13 12.53 ..
-rw-rw---- 1 doris bob        7 dec   13 13.33 f1
-rw-rw-r-- 1 doris teacher    7 dec   13 13.18 f2

root /home #: ls -al /home/kozos/d2
drwxr-xr-- 2 alex  teacher 4096 dec   13 13.57 .
drwxr-xr-x 4 root  root    4096 dec   13 12.53 ..
-rw-rw-rw- 1 alex  bob        7 dec   13 13.56 f3
-rw-r--r-- 1 root  root       7 dec   13 13.57 f4
-rw-r--r-- 1 doris alex       7 dec   13 13.57 f5
```

Minden felhasználó a saját `home` könyvtárában található.

a) Mely felhasználók tudják olvasni az `f1` fájlt? (`cat home/kozos/d1/f1`) **(2 p)**
doris, bob (+root)

b) Mely felhasználók tudják törölni az `f1` fájlt? (`rm /home/kozos/d1/f1`) **(2 p)**
doris, alex (+root)

c) Ki tudja végrehajtani sikeresen az `f2` fájl végrehajtási jogának megadását minden csoportra? (`chmod a+x /home/kozos/d1/f2`) **(2 p)**
doris (+root)

d) A root felasználó mely fájlokat tudja törölni az `d1` alkönyvtárban (`rm /home/kozos/d1/*`)? **(2 p)**
mind

e) Ki tudja írni a d2 directoryban lévő `f3` fájlt? `(echo "AAA" >> /home/kozos/d2/f3)` **(2 p)**
doris,alex (+root)

4. Ismert a következő kódrészlet:

```
void f1(int length) {
  char s[length];
  int options2=read_options2();
  . . . .
}

int read_options2() {
  char buffer[256];
  gets( buffer  );
  . . .
}
```

a) Milyen programhibát tartalmaz a `read_options2()` függvény? (**2p**)
Túl lehet írni a buffer területet.

b) Az `f1()` függvény az ISO C99 szabványban megadott *variable length array* (VLA) tömböt tartalmaz. Azaz a tömb mérete a `length` fordítási időben nem ismert, értéke a bemenő adatok alapján futási időben dől el. Ezért a támadó pontosan nem ismeri a `buffer` kezdőcímét. A támadó ki tudja-e használni a hibát támadása kivitelezésére, és ha igen hogyan? (**2p**)
Igen. Nop csúszda használatával, vagy a processz címterében keresünk egy jmp esp utasítást, és ennek a címével írjuk felül a visszatérési címet.

c) Az ilyen jellegű hibák kihasználása ellen a fordító milyen védelmet nyújthat. Feltehetjük, hogy a nem megadott részek biztonságosan lettek megírva. (**2p**)
Kanári (stack check) fordítási opció megadásával.

5.

a) Mi a ROP (return oriented programming) lényege. (**2p**)
ret utasításra végződő utasítás sorozatok (gadget) címeit helyezzük a stack-re, így a rutinok a return utasítás hatására mindig a következő gadget-et futtatják le. A támadó a töredék kódok felhasználásával állítja össze az eljárását.

b) Mikor használja ezt a módszert a támadó. (**1p**)
Amikor a stack-en nem lehet kódot futtatni (stack szegmens lapjaira nincs futási jog megadva, DEP).

c) Hogyan védekezhetünk a támadás ellen. (**1p**)
Address space layout randomization módszerrel, azaz a dll-ek mindig más és más memória címtartományba töltődnek, így a támadó nem tudhatja a gadget-ek helyét.