Tanúsítási jelentés

HUNG-TJ-002-1-2003 amely a
HUNG-E-002-1-2003 számí értékelési jelentésen alapul.

1. A vizsgált eszköz, szoftver meghatározása

A vizsgálat az **IBM Corp.** által előállított és forgalmazott

IBM 4758-002 PCI kriptográfiai modul (co-processor)

2-es modell, hardver, Miniboot 0: A verzió, Miniboot 1: A verzió

2. A vizsgálat célja

Jelen Tanúsítási jelentés a HUNG-T-002-2003. számú tanúsítvány érvényességi feltételeinek kiterjesztésére vonatkozó vizsgálat eredményeinek összegzését tartalmazza. A vizsgálat alapja, hogy Megbízó a HUNG-T-002-2003. számú tanúsítvány kiadása után további dokumentációkat nyújtott be, kérve, hogy az előző tanúsítványban szereplő DSA aláíró algoritmus mellett a modulra az RSA aláíró algoritmus biztonságos alkalmazhatósága is igazolható legyen. A Tanúsítási jelentés a **HUNG-E-002-1-2003** számú értékelési jelentésen alapul.

2.1 A vizsgálat peremfeltételei

Az IBM 4758-002 egy olyan beavatkozásra reagáló, programozható, kriptográfiai PCI kártya, mely általános célú számítástechnikai környezetet és nagy hatékonyságú kriptográfiai támogatást biztosít. Kriptográfiai funkciók széles választékának megvalósítását támogatja, speciális tervezésű, hardverben megvalósított algoritmusok elérhetővé tételével. Képes szoftvert befogadni, futtatni, egyben megvédeni a betöltött szoftvert és annak titkos adatait, magas támadó potenciállal rendelkező támadók legkülönbözőbb logikai és fizikai támadásával szemben.

A vizsgálat tárgyát képező eszköz a következő fő komponensekből áll:

- hardver /benne: véletlen zaj-generátor, SHA-1-t számító és hatványozó célhardverek, beavatkozást érzékelő, s erre reagáló áramkörök, hardver zárak/,
- Miniboot szoftver /az IBM 4758-002 alapját képező két réteg (0. és 1.), mely az egész eszköz biztonságát és konfigurációját felügyeli/,
- magasabb rendszerszoftver és alkalmazási rétegek (2. és 3. rétegek) kialakításának lehetősége.

Az IBM 4758-002 PCI kriptográfiai koprocesszor hardvere, valamint a 4 egymásra épülő rétegre betölthető szoftver/förmver rendszer alsó két rétege (Miniboot Layer 0, 1) tanúsítvánnyal igazoltan, a legmagasabb 4-es biztonsági szinten kielégíti a FIPS 140-1 követelményeit.

Az utólag az eszközbe tölthető rendszerszoftver és alkalmazás védelmét a hardver és az alsó két réteg támogatja (a 4-es biztonsági szinten tanúsított eszköz biztonságos platformot nyújt biztonságos rendszerszoftverek és alkalmazások védett tárolására és futtatására), amennyiben

az alsó két förmver réteg és hardver ezt támogató biztonsági mechanizmusait helyesen alkalmazzák.

A betölthető rendszerszoftverre és alkalmazásra a FIPS tanúsítvány nem vonatkozik. A FIPS 140-1 tanúsítással rendelkező alapkiépítés (hardver, és a szoftver 0. és 1. rétege) önmagában működésképtelen. A 2. és 3. rétegekre töltött szoftverek felelőssége, hogy az alapjukat képező biztonságos platform szolgáltatásait helyesen hívják meg, illetve szabványos, kriptográfia szempontból korrekt, magas szintű interfészt biztosítsanak az IBM 4758-002 eszközt kívülről meghívó alkalmazások számára.

Amennyiben az IBM 4758-002 kriptográfiai modult egy minősített hitelesítés-szolgáltató kívánja felhasználni biztonságkritikus tevékenységeihez (az általa kibocsátott tanúsítványok aláírására, időbélyeg válaszai aláírására, aláírói kulcspárok generálására, stb.), további követelményeknek kell megfelelni kiegészítő feltételek betartását követelve meg.

A vizsgálat célja az, hogy az értékelendő eszközben implementált, de eddig nem vizsgált RSA-algoritmus minősített aláírásra való használatát lehetővé tegye amellett, hogy a szoftverbetöltésnél alkalmazott digitális aláírás továbbra is csak DSA-alapú lehet.

Tehát feladatunk, hogy támaszkodva az eddigi bevizsgálási eredményekre az RSA alkalmazásával előálló különbségeket vizsgáljuk, az RSA alkalmazásából adódó különbségeket értékeljük biztonsági szempontból.

Az eszközben implementált RSA algoritmus vizsgálata a következő alapokon nyugszik.

- A Miniboot szoftver 1 és 2 szintjének, valamint a DSA-alapú aláírásnak a bevizsgált minősítése garanciát ad a szoftver alaprendszer, a véletlenszám-generátor, valamint a nagypontosságú aritmetika megfelelő működésére.
- Az RSA algoritmus megfelelő működését az IBM Corp. nyilatkozata garantálja.
- A véletlenszám-generátor működését statisztikai tesztekkel megvizsgáljuk.
- Nagymennyiségű generált RSA-kulcshármasok vizsgálatával az algoritmus-szintű, és a kulcsválasztáshoz kapcsolódó protokoll-szintű hibák kizárása.

3. A véletlen generátor értékelése

Az RSA kulcsok generálásához használt véletlen generátort kiterjedt statisztikai próbáknak vetettük alá. Ennek oka az, hogy a véletlenszám-generátor hibás működése az egyik leggyakoribb oka a nem kielégítő szabadsági fokkal rendelkező RSA-kulcsok generálásának. Szélsőségesen alacsony szabadsági fok esetén a támadó számba tudja venni a szóbajöhető prímeket és így egy részhalmaz teljes kipróbálásával fel tudja törni a generált RSA-kulcsokat.

A véletlen generátor értékeléséhez 100000 byte nagyságú véletlen byte-ot kaptunk base64 formában kódolva, amelyet az Értékelési jelentés mellé csatolt tesztjegyzőkönyv szerint a vizsgált HSM generált.
Kiindulásként byte-szinten végeztünk teszteket. Ezek eredménye a következő:

- log2(256) = 8.000
- Entrópia = 7.998
- khi2=254.75 (255 szabadsági fokkal)

Az adatcsomag 256 különböző elemet tartalmaz. Az elemfrekvenciát részletesen az Értékelési jelentésben mutatjuk be.

A véletlenszerűség további megerősítésére, illetve esetleges lokális egyenlőtlenségek kimutatására további teszteket is elvégeztünk. E tesztek bitszinten dolgoznak az adatokkal.
A teljes halmaz tesztelése mellett a teszteket lefuttattuk úgy is, hogy a 100000 byte-ot öt különálló darabra vágtuk, s a darabokat külön-külön teszteltük.

A következő teszteket végeztük el.

- Bitfrekvencia teszt
- Futamhossz teszt
- Diszkrét Fourier Transzformációs teszt
- Maurer-féle univerzális entrópia teszt (csak a teljes anyagra a megkívánt minimális hossz miatt)
- Sorteszt 5-ös és 8-s blokkmérettel
- Kumulatív összeg teszt előre, illetve visszairányba.

A rand.txt tartalmazza a teljes sorozatot, míg a rand1.txt…rand5.txt a teljes sorozat egyötödét. Az elvégzett statisztikai próbák 1%-os szinten bizonyítják, hogy a vizsgált sorozatok véletlenszerűek.

A próba neve	Rand.txt	Rand1.txt	Rand2.txt	Rand3.txt	Rand4.txt	Rand5.txt
Bitfrekvencia	0.1768284	0.1970507	0.0827408	0.0836303	0.3575728	0.9402147
Futamhossz	0.6087462	0.5726411	0.8006707	0.6873236	0.3260401	0.6030734
Spektr. DFT	0.2733112	0.7702877	0.4653924	0.0915836	0.8203382	0.9741178
Maurer univ.	0.8558085	**nem elég hosszú**	**nem elég hosszú**	**nem elég hosszú**	**nem elég hosszú**	**nem elég hosszú**
Sor1, blokk=5	0.9862275	0.7571147	0.6800986	0.2611547	0.7291777	0.6333265
Sor2, blokk=5	0.9227472	0.8854387	0.8581505	0.0542936	0.3984472	0.4329428
Sor1, blokk=8	0.0436764	0.1076985	0.6408255	0.0403513	0.1605897	0.3955422
Sor2, blokk=8	0.0268308	0.0216303	0.6455786	0.0344624	0.2246045	0.0461857
Kum.összeg előre	0.1334565	0.2180873	0.1084590	0.1611011	0.5280558	0.6246556
Kum.összeg vissza	0.0773742	0.0812181	0.0817099	0.1300753	0.0837020	0.5578944

A statisztikai próbák mindegyike 0,01-nál nagyobb értéket ad vissza a véletlenszerű esetben, s ez minden esetben teljesül. Azaz megállapítható, hogy a véletlenszám-generátor minden empirikus tesztet sikeresen teljesített.

4. Az RSA- specifikus hibák kizárására irányuló vizsgálatok

Az RSA-algoritmussal szemben általános szinten a következő támadási lehetőségek vethetők fel:

I. A modulus faktorizálása lehetségessé válik, mert a modulust túl kicsinek választják, s az a mai leggyorsabb NFS (Number Field Sieve) algoritmussal, vagy más módon tényezőkre bontható.
II. A modulus tényezőkre bontása lehetővé válik a prímek speciális tulajdonságai miatt
III. A nyílt szöveg a kulcs visszaállítása nélkül lehetővé válik.
IV. Egyéb ismert marginális vagy ismeretlen támadási lehetőségek

4.1 Az RSA elleni támadások rövid történeti áttekintése

Az RSA-rendszerben a titkos **d** kulcs azért maradhat titokban az **e** nyilvános kulcsot ismerők előtt, mert bár **e*d**=1 modulo φ(m), vagyis a **d** az **e** multiplikatív inverze, a φ(m)=(p-1)*(q-1) modulus nem ismert a támadó előtt. Ha a támadó meg tudja ismerni p és q értékét, akkor **d** is meghatározható az euklideszi algoritmussal. (Illetve m,e,d ismeretében a prímek is meghatározhatók, amint azt látni fogjuk.) Az elsődleges támadási irány tehát a modulus tényezőkre bontása, faktorizálása.

4.1.1 Általános faktorizálás

A prímtényezőkre bontás feladatára általános esetben a szakirodalom nem ismer olyan algoritmust, amely futásideje polinomiális módon függne az input méretétől. (Tegyük fel, hogy a faktorizálandó szám, -éppúgy mint az RSA-modulus- két nagy prím szorzata.) Ez azt jelenti, hogy a faktorizálandó szám méretével exponenciálisan nő a futásidő. A mai legjobb faktorizáló algoritmusok lépésszáma kb.

$$L(x)=\exp(\sqrt{[a\ln(x) * \ln(\ln(x))]}) \quad a\sim 1.$$

Ez a mai technikai színvonalon azt jelenti, hogy kb. 100-155 digites (332-512 bit) számok viszonylag könnyen faktorizálhatók. Az RSA cég összetett számokból álló feladványokat tesz közzé, így tesztelve a faktorizálási módszerek fejlődését.

A legfrissebb eredmény szerint már az RSA-155 jelű, 155 digitből álló számot is sikerült faktorizálni. Az alábbi táblázat mutatja a faktorizálási fordulópontokat. Az alkalmazott algoritmus korábban minden ezetben az úgynevezett kvadratikus szita módszer volt, míg ma már a sokkal hatékonyabb NFS (Number Field Sieve) algoritmust használják. Fontos, hogy ez az algoritmus bár nagyságrendekkel gyorsabb mint a QS, nem tudja kihasználni a faktorizálandó szám speciális tulajdonságait.

A mai elvárások az RSA cég ajánlásai szerint általános célokra 1024 bit körüli modulust tartanak megfelelőnek.

Szám	Időpont	MIPS-év	Alg.
RSA-100	**1991 04**	**7**	**QS**
RSA-110	**1992 04**	**75**	**QS**
RSA-120	**1993 06**	**830**	**QS**
RSA-129	**1994 04**	**5000**	**QS**
RSA-130	**1996 04**	**500**	**NFS**
RSA-140	**1999 02**	**?**	**NFS**
RSA-155	**1999 09**	**?**	**NFS**

4.1.2 Faktorizálás speciális esetben

Vannak olyan faktorizáló algoritmusok, amelyek bizonyos speciális esetekben lényegesen gyorsabbak, mint az általános QS algoritmus. Messze a legfontosabb ilyen speciális eset, amikor a p-1 számnak csak kicsi prímosztói vannak. Természetesen a módszer ugyanúgy működik, ha a q-1 számnak vannak csak kicsi prímosztói. Ez a támadási mód tehát elkerülhető, ha a választott p, q prímekre a p-1, q-1 számoknak is van legalább egy 'nagy' prímosztója.

Ez a feltétel egy másik lehetséges támadástól is megvédi az RSA rendszert. Az $y=x^{\mathbf{d}}$ egyenlet diszkét logaritmus feladatnak is tekinthető. Ha feltesszük, hogy ismerünk egy (x,y) rejtjeles párt, akkor a d titkos kulcs megkapható a $\mathbf{d}=\log_x y$ egyenletbôl. Ennek a feladatnak a általános esetben hasonló a komplexitása mint a faktorizálásé, de ha a p-1 (és a q-1) számnak csak kicsi prímosztói vannak akkor az ún. Pohlig módszer sikeres lehet.

A p-1-hez hasonló feltételt fogalmazhatunk meg a p+1 prímosztóira is. Ezt a feltételt azonban általában nem követelik meg olyan szigorúan. Kimutatható, hogy a p+1 faktorizálási módszer komplexitása hasonló Lenstra elliptikus görbéken alapuló faktorizálási módszeréhez, vagyis ha a választott prím elég nagy, hogy ellenálljon Lenstra módszerének, akkor a p+1 módszer sem lesz sikeres. A biztonságot növelendő, célszerű a generált prímekre a p+1 értéket megvizsgálni, próbafaktorizálást végezni, s ha csak kicsi prímosztói vannak (ennek kicsi az esélye), akkor új prímet célszerű generálni

4.1.3 Iterációs támadás

Itt kell megemlíteni az úgynevezett iterációs támadást, amely az egyik legérdekesebb fenyegetés az RSA rendszer ellen. Ha az RSA rendszert nem digitális aláírás képzésére, hanem rejtjelzésre használjuk, akkor a módszer megkísérli megtalálni a nyílt szöveget anélkül, hogy a modulus prímfelbontását megkapná. Ha n a nyílt, r a rejtjeles blokk, akkor $r=n^e$.

A módszer lényege, hogy az r rejtjelest emeljük többször e. hatványra. Mivel a hatványozás egy-egy értelmű leképezés, előbb-utóbb (k lépés után) visszakapjuk az r értéket. (Ez amiatt igaz, mert véges halmazon dolgozunk, tehát a hatványok sorozatában előbb-utóbb megjelenik egy olyan szám, ami már szerepelt korábban. Ha ez nem a kiinduló r érték lenne, hanem egy másik szám, akkor ez a szám két különböző szám e. hatványaként is előállna, ami lehetetlen, mert a hatványozás invertálható leképezés.)

Ekkor

$$r=r^{k*e} \text{ modulo } m.$$

Ekkor nyilván

$$n=r^{(k-1)*e} \text{ modulo } m$$

, vagyis megkaptuk a nyíltat. (A kulcsot viszont nem fejtettük meg.) A módszer gyakorlati használhatósága attól függ, hogy a hatványozás által létrehozott operációnak milyen az állapottere, milyen hosszú ciklusok vannak benne. Rivest[1] analizálta az állapotteret, s azt találta, hogy ha a p'-1 számnak is van legalább egy nagy prímosztója, (p' a p-1 nagy prímosztója) akkor az állapottér elegendően nagy köröket tartalmaz ahhoz, hogy a támadás a gyakorlatban ne legyen végrehajtható. Később U. Maurer ennél könnyebben teljesíthető feltételt is adott.[2] Ma azonban a faktorizáló algoritmusok viharos fejlődése miatt sokkal nagyobb prímeket kell használni mint korábban, s ilyen nagy számok esetében a fenti támadás nem kivitelezhető.

4.1.4 Azonos nagyságrendű prímek

Azért hogy a modulus faktorizálását megnehezítsék, a fejlesztők általában két nagyjából azonos méretű prím szorzataként állítják elő a modulust. Ennek oka az, hogy Lenstra 1987-ben publikált, elliptikus görbéken alapuló faktorizálási módszere jól működik olyan összetett számokra, ahol a második legnagyobb prímtényező lényegesen kisebb, mint a legnagyobb[3]. Azonban az is potenciális támadás forrása lehet, ha a két prím túl közel van egymáshoz A támadás lényege:

$$\left(\frac{p+q}{2}\right)^2 - n = \left(\frac{p-q}{2}\right)^2$$

Ekkor ha p és q nagyjából egyenlő, akkor $\left(\frac{p+q}{2}\right) \approx \sqrt{n}$. A támadás úgy folyik, hogy n gyökétől a nagyobb számok felé indulva keresünk olyan k számokat, amelyre k^2-n négyzetszám. Ezt a támadást elkerülendő, kívánatos, hogy az egyik prím legalább négyszer-nyolcszor nagyobb legyen mint a másik. A prímek kiválasztásakor pedig az azonos nagyság helyett célszerűbb úgy fogalmazni, hogy mindkét prím legyen nagyobb mint egy adott korlát.

4.2 A speciális esetek

A 4.1.2 ,4.1.3, 4.1.4 pontokban vázolt támadások legtöbbje úgy védhető ki, hogy a p, q prímekre feltételeket állapítunk meg. Azonban az itt leírt támadási módszereknek csak történeti érdekessége van, abból az időből származnak, amikor 512 bites modulus volt a gyakorlatban használt maximális méret. A faktorizáló algoritmusok időközben hatalmas fejlődésen mentek át, főleg azok a faktorizáló algoritmusok fejlődtek, amelyek a faktorizálandó szám tulajdonságaitól függetlenül működnek, mint arra az NFS algoritmusnál utaltunk. Így az a helyzet állt elő, hogy egy általánosan választott szám faktorizálása ugyanaddig tart mint egy speciális tulajdonságokkal rendelkező számé. A következmény az lett, hogy az RSA modulusok mérete jelentősen megemelkedett, és fent leírt támadások ekkora méretek mellett reálisan nem hajthatók végre, csak elméleti érdekességük van.[4]

---

[1]Rivest, R. L Remarks on a proposed cryptoanalytic attack on the MIT pks, Cryptologia, No1 1978 jan

[2]U. Maurer , Fast generation of secure RSA-moduli with almost maximal diversity, EUROCRYPT '89

[3]H.W. Lenstra, Jr., 'Factoring integers with elliptic curves' Ann. Math. vol 126, 1987

[4] Burt Kaliski,Matt Robshaw, The secure use of RSA, Cryptobytes, 1995 vol 1, num 3

Fentiek alapján a biztonságos RSA kulcsgeneráláshoz az IBM által tanúsított, (e,d)-re vonatkozó szintaktikai feltételek mellett az alábbiakat kell ellenőrizni.
A prímeket megfelelő szabadsági fokkal kell generálni. Ehhez az szükséges, hogy a prímkeresés kiinduló pontjaként szolgáló véletlenszám-generátor megfelelően működjön, illetve, hogy a prímkeresésre használt algoritmus elegendően nagy intervallumban keressen egyenletes eloszlás alapján prímeket. A módszer megfelelő, ha elegendően nagy számú (legalább 2^{100}) prím jöhet szóba megközelítőleg egyenlő valószínűséggel.

4.3 Támadási lehetőségek RSA algoritmust használó protokollok ellen, ezek kivédése

Az előzőekben vázolt támadási irányok magát az RSA algoritmust vették célba. Ha azonban az RSA algoritmust egy kriptográfiai hálózatban alkalmazzák egyéb támadási lehetőségek is felmerülhetnek.

A. A hálózat egy részénél az **e** nyilvános kulcs azonos és kicsi. A támadás bemutatásához legyen **e**=3. Ha egy adó három helyre is elküldi ugyanazt a nyílt blokkot. akkor

$$y_1=x^3 \text{ modulo } m_1$$
$$y_2=x^3 \text{ modulo } m_2$$
$$y_3=x^3 \text{ modulo } m_3$$

Ekkor a kínai maradéktétel alapján az x üzenet megkapható. A támadás kivédésének két útja van: ne válasszunk a hálózat csomópontszámával összemérhető nyilvános kulcsot, és/vagy gondoskodjunk arról, hogy ne legyen két azonos üzenet. Ez történhet például egy néhány byte-nyi pszeudovéletlen blokk beszúrásával.

B. Átgondolandó veszélyforrást jelent az RSA-leképezés művelettartása is. Mind a rejtjelzés, mind a digitális aláírás művelettartó a moduláris szorzásra nézve. Vagyis

$$f(x_1)*f(x_2)=f(x_1*x_2)$$

Ez azt jelenti, hogy ha ismerjük az x_1 x_2 üzenetekhez tartozó digitális aláírást, akkor egyszerű szorzással (a titkos kulcs ismerete nélkül is) képezhetjük a nyílt üzenetek szorzatához tartozó digitális aláírást. Ha azonban az eredeti nyílt szövegek strukturáltak, szerkezetük van, akkor gyakorlatilag nincs valószínűsége, hogy a szorzat üzenet is ilyen lesz, vagyis a szorzat nem lesz egy lehetséges nyílt, s ekkor ennek aláírása is haszontalan a támadó szempontjából.

C. Létezik két olyan felhasználó, hogy az általuk használt modulus azonos. Ekkor mindkét felhasználó ki tudja számítani a másik titkos **d** kulcsát.[5] Tehát nem szabad, hogy a hálózatban azonos modulusokat használjanak

[5]J: H: Moore, 'Protocol failures in cryptosystems', Proc. IEEE Vol 76, No. 5 1988 május

D. M. Wiener eredménye szerint[6] a titkos d exponens visszaállítható, ha a d exponens elegendően kicsi. Ez a gyakorlatban azt jelenti, hogy a rendszer támadható, ha $d<n^{1/4}$. Annak, hogy ez véletlenül választott d, illetve rögzített e és véletlenül választott m mellett előforduljon , elhanyagolható a valószínűsége. Ezért ez a támadás akkor hatékony csak, ha a d értékét számdékosan kicsire választják.

5. Vizsgálati eredmények az RSA-ra vonatkozóan

Az előző fejezet megállapításait röviden és némileg pongyolán úgy lehet összefoglalni, hogy megfelelő nagyságú, véletlenül választott prímeket kell használnunk, illetve több kulcshármas használata esetén be kell tartani néhány további, RSA-protokollhiba elkerülését szolgáló szabályt.

A vizsgálat alapja kétszer 100 darab RSA kulcshármas. Az első csomag (rsakeyfixed-e.txt) 100 olyan kulcshármast tartalmaz, amelynél a nyilvános exponens minden esetben 65537. A második csomagban (rsakey128.txt) az exponensre nincs megkötés. Az input adatok az Értékelési jelentés mellékletét képezik.

5.1 Az RSA kulcsok mérete

A vizsgált RSA hármasok áttekintése azt mutatja, hogy minden modulus nagyobb, mint 2^{1020}, azaz az RSA kulcsok megfelelő méretűek ahhoz, hogy az általános faktorizáláson alapuló támadást kizárják.

5.2 A prímek vizsgálata

Megfelelően nagynak választott modulus esetén is támadható az RSA rendszer, ha a prímgenerálás módja olyan, hogy a lehetséges prímek csak egy kis, kipróbálható részhalmaza jöhet szóba RSA-kulcsként.
Másrészt láttuk, hogy az elegendően nagyra, és egyenletesen választott prímek védelmet jelentenek a 4.1.2 ,4.1.3, 4.1.4 pontokban vázolt támadások ellen is.

5.2.1 Véletlenszám-generátor

A prímgenerálás kiindulópontjaként szolgáló véletlen–generátor vizsgálatát a fentiek miatt végeztük el a 3 fejezetben. . A megkapott 100000 byte sokoldalú statisztikai vizsgálata azt mutatja, hogy a véletlengenerátor megfelelően működik, a generált sorozat statisztikai módszerekkel nem különböztethető meg egy valódi véletlen forrásból származó sorozattól.

5.2.2 A prímvisszaállítás módszere

A vizsgálat következő részében az (e,d,N) kulcshármas ismeretében visszaállítjuk a prímeket, és megállapításokat teszünk ezek eloszlására.

A prímek visszaállításának elméleti alapja a következő:
Jelölje $\phi(n)$ az n-hez relatív prímek számát. Az RSA algoritmus alapegyenletei szerint:

[6] M.J. Wiener. Cryptoanalisys of short RSA secret exponents. IEEE Trans. on Information Theory, 36: 553-558, 1990

$$p*q=n$$
$$e*d\equiv 1 \text{ modulo } \phi(n)$$

ahol p, q azonos nagyságrendű prímek,
e, d relatív prím n-hez,
$\phi(n)$ az n-hez relatív prímek száma, jelen esetben $\phi(n)=(p-1)*(q-1)$

Az e,d,n hármas ismeretében az n faktorizálható, azaz p és q értéke visszaállítható az alábbi módon:

- Számítsuk ki a k=d*e-1 értéket. k a definíció szerint többszöröse a $\phi(n)$-nek.
- Mivel $\phi(n)$ páratlan, írhatjuk $k=2^t *r$ alakban, ahol r páratlan, és $t\geq 1$.
- k definíciója szerint minden g n-hez relatív prím számra $g^k=1$ az n-hez ralatív prím számokat tartalmazó $\mathbf{Z}^*_n$ csoportban.
- Mivel k páros, ezért $g^{k/2}$ létezik, és az előbbiek miatt az 1 négyzetgyöke. A kínai maradéktétel miatt az 1-nek négy négyzetgyöke van a csoportban, ebből kettő (1, -1) triviális. A másik kettőt jelöljük x-szel, illetve –x-szel.
- Igaz, hogy $x\equiv 1$ modulo p és $x\equiv -1$ modulo q. Az x ismeretében a prímek már könnyen meghatározhatók. Mivel x-1 osztható p-vel és nem osztható q-val, x és n legnagyobb közös osztója megadja p, és így közvetve q értékét is. A legnagyobb közös osztó kiszámítását a hatékony Euklideszi algoritmussal el tudjuk végezni.
- Szükségünk van még egy nemtriviális négyzetgyökre az egységelemre vonatkozóan.
- Bizonyítható, hogy a $\mathbf{Z}^*_n$ csoportból véletlenül választott g esetén legalább ½ a valószínűsége annak, hogy az $g^{k/2}$, $g^{k/4}$,… $g^{k/2\exp(t)}$ modulo n sorozatban lesz egy nemtriviális gyöke az 1-nek.

Ezzel a faktorizálás polinomiális időben elvégezhető.

5.2.3 A prímvisszaállítás eredménye

A kapott RSA-kulcshármasokból kiszámított prímek adatai a primlist.txt fájl tartalmazza. Ebben a következő adatokat találhatjuk meg:

- Az e,d,n kulcshármas
- A p,q prímek
- A prímek helye, ha az $1-2^{512}$ intervallumot arányosan leképezzük a [0,l] intervallumra
- A két prím különbségének kettes alapú logaritmusa.

A prímek kiszámításán túl minden prímre sikeresen elvégeztük a Rabin-Miller prímtesztet is 20-szoros iterációval, azaz biztos, hogy a kapott értékek prímek.
A lista elsődleges áttekintése is mutatja, hogy a prímek nem egyenletesen helyezkednek el a 2^{511}-2^{512} intervallumban. Megfigyelhető, hogy minden prímnél igaz, hogy nemcsak a legnagyobb, hanem a második legnagyobb helyiértékű bit is egyes. Az előbb említett intervallum fölső felére vonatkoztatva a prímek már egyenletesen elhelyezkedőnek látszanak. Ez a tér még mindig bőven elegendő a szükséges szabadsági fok biztosításához, ezért ez biztonsági problémát nem jelent.

A táblázat tartalmazza a prímek különbségét is, pontosabban annak kettes alapú logaritmusát. Ebből látható, hogy a prímek elég messze vannak ahhoz, hogy a 4.1.4 pontban említett támadás ne legyen megvalósítható. Nincs nyoma annak, hogy a prímek választása a során olyan szabályt követnének, amely csökkenti a szabadsági fokot.

5.3 A prímek eloszlásának vizsgálata

A prímek eloszlását statisztikai módszerekkel is megvizsgáltuk. Az általunk ismert 400 prím eloszlását vizsgáltuk khi-négyzet próbával. A könnyebb kezelés miatt a $1\text{-}2^{512}$ intervallumot leképeztük a [0,1] intervallumra .

5.3.1 Az alkalmazott statisztika elméleti háttere

Feladat:

Döntsük el, hogy egy adott **I** intervallumból származó *m* darab egész szám véletlenszerű mintavételnek tekinthető-e.

Megoldás:

Osszuk fel az **I** intervallumot $k=m/20$ darab egyenlő hosszú diszjunkt részre. Jelölje ezeket $\mathbf{I_1}$, $\mathbf{I_2}$, … , $\mathbf{I_k}$. Jelölje a_i az *m* darab szám közül azok számát, melyek az $\mathbf{I_i}$ intervallumba esnek. Egyenletes esetben, egy adott részintervallumba esés valószínűsége $1/k$.
Alkalmazzuk a χ^2-próbát:

$$stat = \sum_{i=1}^{k} \frac{(a_i - m \cdot 1/k)^2}{m \cdot 1/k} = \sum_{i=1}^{k} \frac{(a_i - 20)^2}{20}$$

Egyenletes esetben *stat* értéke k-1 szabadsági fokú χ^2 eloszlásból származó értéknek tekinthető. $k \leq 30$ esetén a megfelelő küszöbértéket táblázatolva találhatjuk meg, $k > 30$ esetén *stat* értékét standardizáljuk és a standard normális eloszlás táblázatát használjuk, azaz:

$$statnorm = \frac{stat - (k-1)}{\sqrt{2(k-1)}}$$

5.3.2 Az elvégzett teszt eredménye

Ha az $1\text{-}2^{512}$ intervallumot arányosan leképezzük a [0,l] intervallumra, akkor a prímek [0.75,1.00] intervallumba esnek. Itt vizsgáljuk az egyenletességet. Az elméleti leírásra visszautalva esetünkben m=400, és az intervallumot k=20 részre osztottuk fel. A kiszámolt statisztika értéke 0.048666 a [0.75,1.00] intervallumban. A 19 szabadsági fokú χ^2 próba küszöbértéke 1%-os szinten 36.2, azaz a prímek egyenletes eloszlásról való eltérését nem tudtuk kimutatni.

5.4 RSA-protokollhibák

Meg kell említeni, hogy a kapott kulcsok egy részében a más rendszerekben is előszeretettel használt 65537-et állítják be nyilvános exponensnek. Nincs nyoma annak, hogy a titkos exponenst kicsinek választanák. A protokollhibák ellen, illetve a rögzített exponensből adódó kockázatok kivédésére az alábbi elvárásokat tesszük.

- A nyilvános exponens úgy válasszák meg, hogy annak a mérete legalább 65537 legyen.
- Az RSA müveletek során az output formátumaként használják az egyébként is szokásos PKCS#1 legfrissebb verzióját, amely alapvetően a random padding segítségével elkerüli az említett kockázatokat.

A második feltétel azonban nem a hitelesítés-szolgáltató kompetenciája, hanem általános biztonsági feltétel RSA-operációk esetén, ezért ezt a tanúsítványban nem kell feltételként megemlíteni.

6. Összegzés

Munkánk során megvizsgáltuk az az 1. pontban meghatározott eszközt abból a szempontból, hogy a tanúsításában külön nem szereplő RSA algoritmus alkalmazása jelent-e kockázatot. Megállapítottuk, hogy

- Az alkalmazott véletlenszám-generátor megfelelő.
- A generált kulcsok mérete megfelelő
- A generált prímek eloszlása véletlenszerű az $[2^{511}+2^{510},2^{512}]$ intervallumban.
- Nem mutattunk ki a prímgenerálás szabadsági fokát lényegesen csökkentő tényezőt.
- Nem mutattunk ki RSA-protokollhibát.

A fentiek alapján az RSA algoritmus az 1. pontban meghatározott termékben minősített digitális aláírás létrehozására használható a HUNG-T-002-1/2003 tanúsítási jelentésben leírt feltételek mellett.
A vizsgálatokhoz kapott anyagok, valamint a futtatási és statisztikai eredmények az Értékelési jelentés részét képezik.