Julkisen avaimen salausjärjestelmät

Public key encryption

Julkisen avaimen salaus: public key encryption

- Idean julkisen avaimen salauksesta esittivät v. 1977 **Diffie ja Hellman**

  Periaate:
  Jokaisella käyttäjällä X on kaksi avainta:
  - **julkinen avain**, jolla X:lle lähetettävät viestit salataan
  - **yksityinen avain**, jolla X purkaa saamansa viestit

  Viestit salataan käyttämällä viestin **vastaanottajan julkista avainta**. Julkinen avain saadaan tavallisesti avainpalvelimelta (CA), joka pitää yllä julkisten avainten rekisteriä. Viestin vastaanottaja purkaa salauksen käyttämällä yksityistä avaintaan.

  Avainparia voidaan käyttää myös toisinpäin: yksityisellä avaimella salattu viesti voidaan purkaa julkisella avaimella. **Autentikoinnissa eli käyttäjän todennuksessa** A varmistuu B:n identiteetistä lähettämällä tälle satunnaisluvun R, johon B vastaa lähettämällä luvun salattuna yksityisellä avaimellaan. A purkaa saamansa vastauksen B:n julkisella avaimella. Jos purettu vastaus sisältää alkuperäisen satunnaisluvun R, B:n aitous on varmistettu.

RSA – salausalgoritmi

Ensimmäisen toimivan julkisen avaimen salausmenetelmän esittivät

Rivest, Shamir ja Adlemann v. 1977. RSA – salaus on tätä kirjoitettaessa (toukokuussa 2019) edelleen käytössä osana mm. Suomen verkkopankkien salausta.

Rivest, Shamir ja Adlemann 1970 -luvulla

RSA:n avaimet

Jokaisella käyttäjällä on kaksi avainta:

<u>Julkinen avain</u> sisältää kaksi kokonaislukua

modulus $n = p*q$, missä p ja q ovat alkulukuja

eksponentti e = 65537 (TLS:ssä sama kaikille käyttäjille *)

<u>Yksityinen avain</u> $d = e^{-1} \mod (p-1)(q-1)$

ts. d = eksponentin e käänteisluku modulo $(p-1)(q-1)$

Huom! Matematiikkaosassa on käsitelty Eulerin funktiota ϕ. Kun n on kahden alkuluvun p ja q tulo, niin $\phi(n) = (p-1)(q-1)$

*) Yleisessä tapauksessa eksponentti e voi olla käyttäjäkohtainen, sen ei siis tarvitse olla vakio 65537. Ainoa ehto on, että e:llä pitää olla käänteisluku mod $(p-1)(q-1)$, mikä toteutuu jos $GCD(e, (p-1)(q-1))=1$

RSA-avainten generointi WolframAlpha Online laskimella

1. Luodaan 2 alkulukua p ja q, joiden pituus on tässä esimerkissä 15 bittiä. Lasketaan julkinen avain kaavalla n = p q

```
p=RandomPrime[{2^15,2^16}]; q=RandomPrime[{2^15,2^16}]; n=p*q
```

$p = 59\,023\,,\quad q = 43\,313\,,\ n = 2\,556\,463\,199$

2. Lasketaan yksityinen avain d kaavalla $d = e^{-1}$ mod (p-1)(q-1), missä e=65537

```
p=59023; q=43313;  65537^-1 mod (p-1)(q-1)
```

d = $1\,626\,331\,841$

Avainpari on siten: Julkinen avain 2 556 463 199, yksityinen avain 1 626 331 841

- **Viestin salaus ja purku RSA:lla**

Viesti esitetään kokonaislukuina m

Salaus tapahtuu käyttämällä vastaanottajan julkisia avaimia

$$c = m^e \mod n$$

Vastaanottaja purkaa salakirjoituksen c yksityisellä avaimellaan d

$$m = c^d \mod n$$

Salaus ja purku laskuesimerkki

Bobin RSA avaimet: Julkinen avain n= 2 556 463 199, yksityinen d = 1 626 331 841
Laske Bob:lle lähetettävän viestin m = 12345 salakirjoitus. Näytä miten Bob purkaa sen.

Salaus (kaava $c=m^e \bmod n$)

12345^65537 mod 2556463199

1 706 161 508

Purku (kaava $n=c^d \bmod n$)

1706161508^1626331841 mod 2556463199

12 345

- **Webpalvelimen autentikointi RSA:lla**

TLS protokollassa RSA:n tärkein tehtävä liittyy palvelimen todennukseen, jossa palvelin todistaa aitoutensa käyttämällä yksityistä RSA- avaintaan - siis päinvastoin kuin tavanomaisessa viestin salauksessa, jossa salaukseen käytetään julkista avainta.

Oletetaan, että palvelimen julkinen avain on n, yksityinen avain on d ja vakioeksponentti e = 65537

1. Asiakkaan selain lähettää satunnaisluvun R (ns. Haasteluku) palvelimelle.

2. Palvelin lähettää vastauksen RES = R^d mod n
   ts. palvelin salaa saamansa haasteluvun R yksityisellä avaimellaan d

3. Asiakas lukee palvelimen sertifikaatista palvelimen julkisen avaimen n ja purkaa vastauksen RES potenssiin korotuksella RES^e mod n.
   Mikäli tulos on R, palvelin on todennettu

• Palvelimen autentikointi RSA:lla: esimerkki

Palvelimen avaimet: $e = 65537$, $d = 1\,626\,331\,841$ ja $n = 2\,556\,463\,199$

1. Asiakas lähettää haasteluvun R = 112233

2. Palvelin lähettää vastauksen RES = R^d mod n

112233^1626331841 mod 2556463199

2017034810

3. Asiakas purkaa vastauksen: RES^e mod n ja vertaa tulosta lähettämäänsä haastelukuun.

2017034810^65537 mod 2556463199

112233

Tulos täsmää lähetetyn haasteluvun kanssa => palvelin on autentikoitu

RSA:n tarvitsemaa matematiikkaa

Useimmat listan aiheista on yksityiskohtaisesti käyty läpi kurssin 1. osassa

1. Eulerin teoreema (RSA:n kaavojen matemaattinen todistus)
2. Lukujärjestelmämuunnokset (viestin koodauksessa luvuiksi)
3. Nopea potenssiin korotus (”powermod” - algoritmi)
4. Käänteisluvun laskeminen mod n (extendedGCD)
5. Satunnaislukujen generointi
6. Alkulukutestit (tarvitaan julkisen avaimen luomisessa)
7. Turvallisten avainpituuksien taustatieto

1. RSA: todistus

Olkoon viesti m kokonaisluku, jonka salakirjoitus $c = m^e \mod n$, missä n on viestin vastaanottajan julkinen avain ja e = julkinen eksponentti. Olkoon d vastaanottajan yksityinen avain. Osoitetaan, että kaava $c^d \mod n$ palauttaa viestin m.

$c^d \mod n = (m^e)^d \mod n = m^{ed} \mod n$

Koska $d = e^{-1} \mod (p-1)(q-1)$, niin

$ed = 1 \mod (p-1)(q-1)$

=> $ed = 1 + k*(p-1)(q.1)$ missä k on kokonaisluku

=> $ed = 1 + k*\phi(n)$

Siten $m^{ed} \mod n = m^{1+k*\phi(n)} = m^1 * m^{k*\phi(n)} = m*(m^{\phi(n)})^k$

Eulerin teoreeman mukaan $a^{\phi(n)} = 1 \mod n$ kaikille a

=> $m*(m^{\phi(n)})^k = m*1^k = m$

Tulos: Purkukaava palauttaa alkuperäisen viestin m

<u>Todistuksessa käytettiin:</u>

Potenssilaskusääntöä:

$\mathbf{(a^x)^y = a^{xy} = (a^y)^x}$

Eulerin teoreemaa:

$\mathbf{a^{\phi(n)} = 1 \pmod n}$

Eulerin ϕ funktio:

$\phi(n)$ = number of integers $1 \le a \le n-1$ for which gdc(a,n) =1

Jos n on p*q, missä p,q ovat alkulukuja

$\mathbf{\phi(n) = (p-1)(q-1)}$

2. Tekstilohkon koodaus kokonaisluvuiksi

ASCII codes of English letters

a	97	A	65
b	98	B	66
c	99	C	67
d	100	D	68
e	101	E	69
f	102	F	70
g	103	G	71
h	104	H	72
i	105	I	73
j	106	J	74
k	107	K	75
l	108	L	76
m	109	M	77
n	110	N	78
o	111	O	79
p	112	P	80
q	113	Q	81
r	114	R	82
s	115	S	83
t	116	T	84
u	117	U	85
v	118	V	86
w	119	W	87
x	120	X	88
y	121	Y	89
z	122	Z	90

Periaate: Merkkien ASCII – koodit muodostavat 256-kantaisen luvun, joka muunnetaan kokonaisluvuksi käyttämällä koodeja kantaluvun 256 potenssien kertoimina.

Viesti ”Helsinki” koodataan luvuksi $(72,101,108,115,105,110,107,105)_{256}$

Muunnetaan luku 10- järjestelmään:

$72*256^7 + 101*256^6+108*256^5+115*256^4+105*256^3+110*256^2+107*256+105$

$= 5216694986324470633$

Dekoodauksessa edetään päinvastaisessa järjestyksessä:

$5216694986324470633_{10} =(72,101,108,115,105,110,107,105)_{256}$

Luvut ovat ASCII koodeja tekstille ”Helsinki”

WolframAlphassa on valmisfunktiot lukujärjestelmuunnoksille:

FromDigits[ToCharacterCode["Helsinki"],256] result : 5216694986324470633

FromCharacterCode[IntegerDigits[5216694986324470633,256]] result : Helsinki

3. Nopea potenssiinkorotus ("Powermod")

Algoritmi potenssin a^b mod n laskemiseksi on esitetty kurssin matematiikkaosuudessa.

Voidaan osoittaa, että muistin tarve potenssin a^b mod n laskemiseksi on $n^2 + 3n$
Yleisin TLS:n julkinen avain $n = 2^{2048} \Rightarrow n^2 + 3n \approx 2^{4100}$ = 4100 bittinen luku.
=> RSA:n muistin tarve on vain n. 4100 bittiä = 512 Tavua

4. Käänteisluvun mod n laskeminen

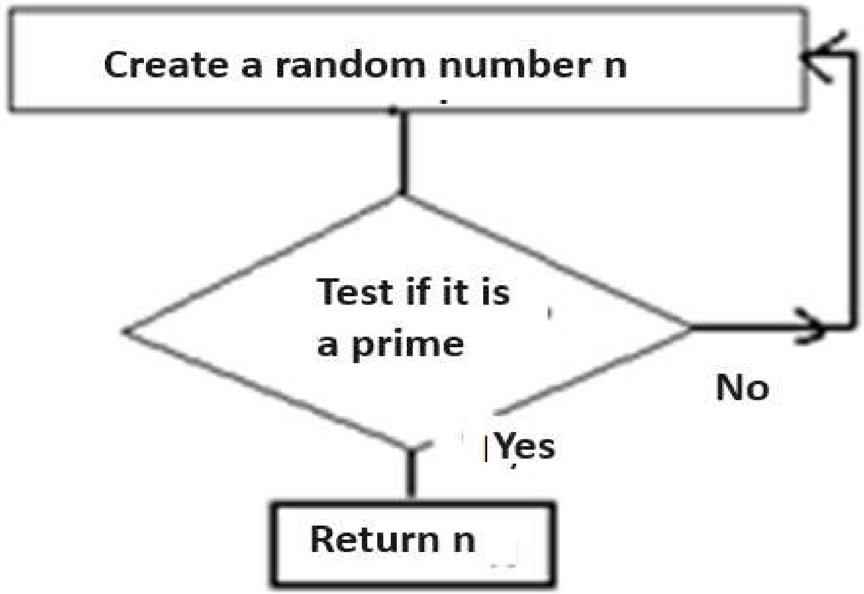
Yksityisen avaimen $d = e^{-1}$ mod (p-1)(q-1) laskemissa käytetään ExtendedGDC -algoritmia, joka on esitetty kurssin matematiikkaosuudessa.

5. Turvallisten pseudosatunnaislukujen generointi

Symmetriset avaimet generoidaan käyttämällä CSPRNG generaattoreita (**cryptographically safe pseudorandomnumber generators).** (ks. chapter 2). Avainten satunnaisuus on kriittinen turvallisuuden kannalta

6. Alkulukujen generointi. Alkulukutestit

RSA:n julkinen avain n on kahden alkuluvun p ja q tulo. Alkulukuja löydetään alkulukutestin avulla satunnaisluvuista. 2000 bitin alkuluvun löytäminen voi kestää jonkin aikaa.



Alkulukutestejä

1. Rabin – Miller testi

2. Fermat'n testi

Wolfram Alpha creates a 1000 bit prime in following way:

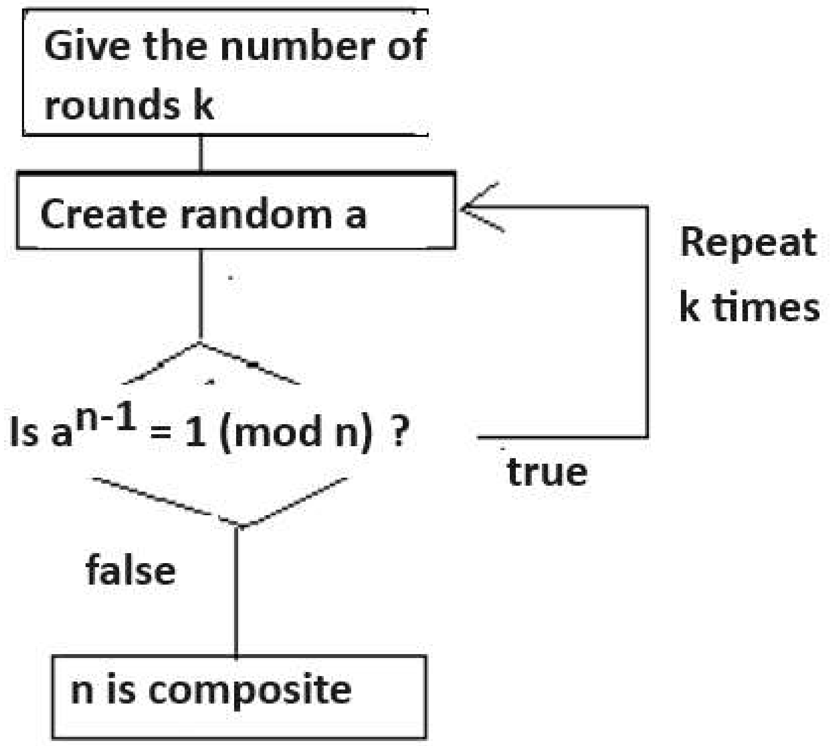
RandomPrime[{2^1000, 2^1001}]

Fermat'n testi perustuu Fermat'n teoreemaan:

Famous PGP encryption uses Fermat's test

Fermat's Theorem: Jos p on alkuluku, niin $a^{p-1} = 1 \pmod p$ for all $0 < a \leq p-1$

Fermat's primality test



Testi on probabilistinen, joka tarkoittaa että

Jos $a^{n-1} \neq 1 \pmod n$ jollekin kantaluvulle a, niin n on varmasti yhdistetty luku (ei alkuluku)

Jos $a^{n-1} = 1 \pmod n$ jollekin kantaluvulle a, se ei vielä todista että n on alkuluku. *Toinen a:n valinta voi antaa eri tuloksen* **.**

Kryptologiassa testi toistetaan esim. kymmenellä satunnaisella kantaluvulla a. Jos testi menee läpi jokaisesta, todennäköisyys sille, että n on alkuluku, on suuri. *(Tarkkaa todennäköisyyttä ei tunneta.)*

Esim: Todista Fermat'n testillä, että a) 4763 b) 561 ovat yhdistettyjä lukuja

Test number 4763

2^4762 mod 4763

Result:

158

$\Rightarrow$ **4763 is composite**

Test number 561

2 ^560 mod 561

Result:

1

test passed

5 ^560 mod 561

Result:

1

test passed

3^560 mod 561

Result:

375

test failed =>
561 is composite

Luku 561 on yksi ns. Charmichaelin luvuista, jotka läpäisevät Fermat'sn testin useilla ***a***:n arvoilla.

Esim. arvoilla a = 2 ja a = 5 testi menee läpi, mutta arvolla a = 3 ei mene. 561 on siten yhdistetty luku.

(Numbers **a** like 2 and 5 above, which give a false result of primality for a composite number are called "Fermat's liars". Charmichael numbers have lots of Fermat's liars)

Rabin Miller testi on yleisesti käytetty alkulukutesti

Perustuu faktaan, että jos p on alkuluku, luvun 1 neliöjuuri mod p voi olla vain 1 tai -1 (eli p-1)

Rabin Miller testi, yksi kierros

p = testattava luku
1. Valitaan satunnainen kantaluku a
a. Suoritetaan Fermat'n testi :
Jos $a^{p-1} \bmod p = 1$
b. Otetaan neliöjuuri vasemmasta puolesta:
ts. lasketaan

$$a^{(p-1)/2} \bmod p$$

Jos tulos = p- 1 , luku p läpäisee testin
Jos tulos = 1, jatketaan taas ottamalla neliöjuuri ts. Lasketaan

$$a^{(p-1)/4} \bmod p$$

Jos tulos = p- 1 testi on läpi, jos tulos = 1, otetaan taas neliöjuuri. Eksponenttia puolitetaan, kunnes se on pariton. Mikä tahansa muu tulos kuin 1 tai p-1 merkitsee yhdistettyä lukua.

Tämäkin testi on probabilistinen:

Jos testi ei mene läpi jollakin kantaluvulla, luku on varmasti yhdistetty luku

Jos ehdokas n läpäisee testin k:lla satunnaisella kantaluvulla a, todennäköisyys sille, että n on alkuluku P > $1 - 1/4^k$

Tässä testissä 10 läpäistyä kierrosta antaa varmuuden 99.9999% siitä, että luku on alkuluku.

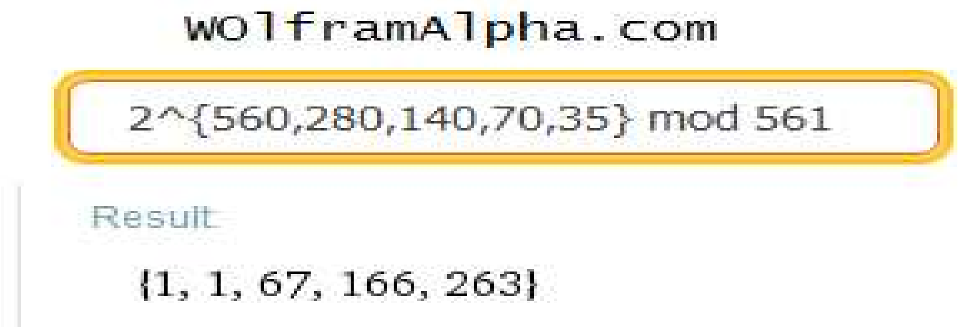
Esim: Testaa onko luku 561 alkuluku Rabin Miller testillä

Tutkitaan, montako kertaa luvun p – 1 voi puolittaa:

560 = $2 \cdot 280 = 2^{2} \cdot 140 = 2^{3} \cdot 70$ **= $2^{4} \cdot 35$**

Valitaan kantaluvuksi a = 2:
Suoritetaan laskut
2^{560} mod 561 = 1
2^{280} mod 561 = 1
2^{140} mod 561 = 67 **=> fail**
2^{70} mod 561 =
2^{35} mod 561 =

WolframAlphassa on hyödyllinen ominaisuus: Kantaluvun 2 voi korottaa samalla käskyllä useampaa potenssiin kirjoittamalla potenssilista kaarisulkuihin.



Koska 3. potenssiin korotus antaa 67 (joka ei ole 1 eikä 560) , 561 on yhdistetty luku

Testaa ”Rabin Miller testillä” luku 1973

Askel 1: **p-1 = 1973-1 = 1972 = 2^2*493** (voidaan puolittaa kahdesti)

Askel 2: Suoritetaan Rabin Miller testi neljällä satunnaisella kantaluvulla: 2 , 35 , 854 ja 114

Kantaluku 2:
2^{1972} mod 1973 = 1
2^{986} mod 1973 = 1972
Testi läpäisty

Kantaluku 35:
35^{1972} mod 1973 =1
35^{986} mod 1973 =1
35^{493} mod 1973 = 1
Testi läpäisty

Kantaluku 854:
854^{1972} mod 1973 =1
854^{986} mod 1973 =1972
Testi läpäisty

Kantaluku 114:
114^{1972} mod 1973 =1
114^{986} mod 1973 =1
114^{493} mod 1973 = 1
Testi läpäisty

Todennäköisyys että 1973 on alkuluku on
P > $1 - 4^{-k} = 1 - 4^{-4} = 0.996 = 99.6\%$

Alkulukugenerointi WolframAlphassa

Ohjelmointikielissä, joilla salausohjelmia kirjoitetaan on yleensä funktio, jolla voi generoida halutun bittimäärän pituisia alkulukuja

Esim: Luo 100 –bittinen alkuluku

```
RandomPrime[{2^100,2^101}]
```

2 422 530 443 145 414 600 337 950 658 763

Esim: Luo 512 –bittinen alkuluku

```
RandomPrime[{2^512,2^513}]
```

15 787 372 807 814 935 269 337 946 439 168 767 722 475 175 033 260 767 647 751 154 530 ⋱
333 820 130 747 181 919 458 745 158 006 634 759 921 476 598 518 589 413 311 054 638 ⋱
013 394 363 696 097 684 553 327 539

RSA:n turvallisuus perustuu suurten alkulukujen tekijöihin jaon vaikeuteen

Suurten kokonaislukujen tekijöihinjako kuuluu matematiikan ”hard problems” luokkaan. Nopeimmat faktorointimetodit ovat ”Guadratic Number Field Sieve” and GNFS (General Number Field Sieve).

Suurin faktoroitu RSA:n julkinen avain (n = p*q) on RSA-768 , (768 - bittinen luku). Menetelmä oli GNFS. Murtaminen kesti 2 vuotta satojen koneiden verkolla.

RSA-768 = 1230186684530117755130494958384962720772853569595334792197322452151726400507263657518745202199786469389956474942774063845925192557326303453731548268 507917026
1221429134616704292143116022212404792747377940806653514195974598569021434 13

Turvallisen julkisen RSA-avaimen minimipituus

Jos hyökkääjä kykenee jakamaan palvelimen julkisen avaimen n sen tekijöihin p ja q, hän kykenee laskemaan helposti palvelimen yksityisen avain d. Tällöin palvelimen tietoliikenteen salaus on murrettu.

Esimerkiksi julkinen avain n = 502560280658509 on aivan liian lyhyt. Mm. Wolfram Alpha antaa sen tekijät.

factor	502 560 280 658 509

p q

15485863 × 32452843

Luvuista p ja q on helppo laskea yksityinen avain kaavalla $d = e^{-1} \bmod (p-1)(q-1)$

TURVALLINEN RSA:N AVAINPITUUS

- RSA on vielä yleisesti käytössä TLS yhteyksissä, mm. Suomen verkkopankkien julkiset avaimet ovat RSA avaimia, joiden pituus on 2048 bittiä.
- Tätä on pidettävä turvallisen avaimenpituuten minimirajana.

ECC (Elliptic Curve Cryptosystem)

= Julkisen avain salaus, jota mm. Suomen viranomaiset suosittelevat RSA:n seuraajaksi, on ECC

ECC on jo käytössä mm. Suomen verkkopankkien TLS – yhteyksien ECDHE -protokollassa, jolla sovitaan istunnon AES -avaimesta.

Verkkopankkipalvelimien julkinen avain näyttää olevan vielä tänään (17.6.2019) 2048-bittinen RSA avain.

Elliptisten käyrien salauksesta kerrotaan tarkemmin seuraavassa luennossa, jonka aihe on avaimesta sopimisen protokollat

PK-salauksen turvalliset avainpituudet

Alla on EU:n salausmenetelmätyöryhmän arvioita RSA:n ja seuraajaksi nimetyn ECC:n turvallisuudesta eri avainpituuksilla. Taulukko osoittaa että ollakseen turvallinen, RSA tarvitsee yli 2000 bitin julkisen avaimen, kun taas ECC –salauksessa riittää 256 bittiä.

Kuvaus	RSA	ECC
Minimitason takaava turvallisuus	2432	224
Riittävä taso paitsi huippusalaisia asiakirj.	3248	256
Riittävä myös top secret asiakirjoihin	15424	512

Älykorteissa ja mobiililaitteissa on RSA – avaimia. Kasvava avainpituus on ongelma. Muisti ei riitä ja laskenta on hidasta.

Elliptic curve cryptosystem ECC on pienemmän avainpituuden vuoksi suositus tulevaisuuden julkisen avaimen salaukseksi.

Havaittuja avainpituuksia v. 2023 suomalaisissa web-palveluissa

Avainpituus	Arvio turvallisuudesta
1024 bits	**Ei turvallinen** (source: cyber security center, Finland)
2048 bits	Tavallisin avainpituus TLS:ssä (esim. Nordea)
4098 bits	Yleistyvä seuraavat taso (S pankki)

Esimerkki siitä, miten RSA murretaan

Bobin julkinen avain n = 2556463199 on liian lyhyt.
Murretaan se ja selvitetään Bobin yksityinen avain.

WolframAlpha komento factor 2556463199 antaa tekijät 43313*59023.
=> Bobin yksityinen avain d voidaan nyt laskea
d = e^-1 mod (p-1)(q-1) = 65537^-1 mod (43312*59022) = 1626331841

Ovatko RSA ja ECC ”post-quantum” turvallisia?

RSA – avain voidaan murtaa nopeasti kvanttilaskennalla. Shor’n algoritmi *) .
Tosin sellaisia kvanttitietokoneita, jotka pystyisivät tähän ei ole vielä olemassa. RSA:ta voidaan siis käyttää vielä joitakin vuosia.

Myöskään RSA:n seuraajaksi kaavailtua Elliptisten käyrien salausta ei voida käyttää. Sama koskee DH avaimesta sopimisen protokollaa.

Korvaavia järjestelmiä on kehitteillä. Käynnissä on myös kilpailu seuraajasta.

**) Shor's algorithm is a quantum algorithm for finding the prime factors of an integer. It was developed in 1994 by the American mathematician Peter Shor.*