RSA

- Salakirjoitus- ja allekirjoitusalgoritmi
- Julkinen ja yksityinen avain
- Periaate yksinkertainen
- Turvallisuuden oletetaan riippuvan kokonaislukujen tekijöihin jakamisen vaikeudesta.

  (Yksityisen avaimen ratkaiseminen on oletettavasti yhtä vaikeaa kuin tekijöihin jako. Ei tosin tiedetä varmasti, voisiko selväkielen ratkaista selvittämättä yksityistä avainta.)
- De facto-standardi
- Hidas, DES on raudalla 1000 ja ohjelmallisesti 100 kertaa nopeampi kuin RSA.

  Vaatii suurten kokonaislukujen tehokasta käsittelyä.

RSA-algoritmi

- Avainten generointi
  1. Valitaan satunnaisesti kaksi suurta, yhtä pitkää alkulukua p ja q. Luvut pidettävä salassa.
  2. $n = p\,q$. $\quad \phi(n) = (p-1)(q-1)$.
  3. Valitaan e $(1 < e < \phi(n))$ siten, että $\gcd(e, \phi(n)) = 1$.
  4. Lasketaan $d = e^{-1} \bmod \phi(n)$.
  5. Julkinen avain on (e, n). Yksityinen avain on (d, n). Unohdetaan p, q ja $\phi(n)$.
- Salaus ja avaus $(2 \le m \le n-2)$

$$\boxed{\begin{array}{c} c = m^e \bmod n. \\ m = c^d \bmod n. \end{array}}$$

- Toimii, koska

$$\begin{aligned} c^d &= m^{ed} \\ &= m^{k\phi(n)+1} \\ &= m\,(m^{\phi(n)})^k \\ &\equiv m \cdot 1^k \pmod n \\ &= m, \end{aligned}$$

  (Mitä tapahtuu, kun $p|m$ tai $q|m$?)
- Modulin n pitää olla vähintää 768 bittiä (suositeltava 1024–2048) ja salaisen avaimen d suurin piirtein yhtä pitkä. Julkinen avain voi olla lyhyt. (Äärimmäisestä lyhyydestä aiheutuu joitain heikkouksia.)

RSA-esimerkki

Valitaan p ja q.

$$\begin{aligned} &p = 53,\ q = 79. \\ &n = p\,q = 53 \cdot 79 = 4187. \\ &\phi(n) = (p-1)(q-1) = 52 \cdot 78 = 4056. \end{aligned}$$

Valitaan satunnainen e siten, että $\gcd(e, 4056) = 1$.
(Jos valinta epäonnistuu, valitaan satunnaisesti uusi e.)

$$\begin{aligned} &e = 49. \\ &d = 49^{-1} \bmod 4056 = 745. \end{aligned}$$

Julkistetaan $(49, 4187)$, pidetään salaisena $(745, 4187)$ ja unohdetaan p, q ja $\phi(n)$.

Salataan julkisella avaimella sanoma.

$$\begin{aligned} &m = 1234. \\ &c = 1234^{49} \bmod 4187 = 3738. \end{aligned}$$

Avataan yksityisellä avaimella

$$\begin{aligned} &m = 3738. \\ &c = 3738^{745} \bmod 4187 = 1234. \end{aligned}$$

RSA:n turvallisuus

- RSA:n ratkaiseminen luultavasti vaatii n jakamisen tekijöihin. Tätä **ei ole todistettu**.

  Jos p ja q tunnetaan, on helppo laskea d.

  Jos d tunnetaan, on melko helppo laskea p ja q.
- Kannattaa valita **pieni julkinen eksponentti** e ja suuri d, jotta salaus (ja allekirjoituksen tarkistus) on nopeaa.

  Voidaan valita aina $e = 3,\ 17,\ 2^{16}+1$ tms.
- **Vaaroja**, jos julkinen avain pieni (esim. 3):
  - $2^3 = 8,\ 3^3 = 27,\ 4^3 = 64, \ldots$ eivät redusoidu $\bmod n$.
  - Jos kolmen henkilön julkiset avaimet ovat $(3, n_1),\ (3, n_2),\ (3, n_3)$ ja kaikki salaavat sanoman m,

$$c_1 = m^3 \bmod n_1,\ c_2 = m^3 \bmod n_2,\ c_3 = m^3 \bmod n_3.$$

    Ratkaistaan Gaussin algoritmilla kongruenssit

$$\begin{cases} x \equiv c_1 \pmod{n_1} \\ x \equiv c_2 \pmod{n_2} \\ x \equiv c_3 \pmod{n_3}. \end{cases}$$

    Koska $x = m^3$ $(< n_1 n_2 n_3)$. Ratkaistaan $m = \sqrt[3]{x}$.

  Ratkaisu:
  - käytetään eksponenttia $2^{16}+1$
  - lisätään sanomaan **suola**, 64 satunnaista bittiä.

- Pieni yksityinen eksponentti d nopeuttaisi salauksen avaamista ja allekirjoitusta. Ongelmia:
  - *brute force*-haku, kun $|d| < 80$
  - on olemassa tehokas algoritmi d:n ratkaisemiseen, kun $|d| \lesssim 0.25|n|$.

  Yksityisen eksponentin d oltava samaa suuruusluokkaa kuin n. Satunnainen valinta on ok.

- **Multiplikatiivisuus**:

$$(m_1 m_2)^e \equiv m_1^e m_2^e \equiv c_1 c_2 \pmod{n}.$$

  Ongelmia:
  - On mahdollista havaita salakielestä, jos $m_3 = m_1 m_2$.
  - Jos avataan (allekirjoitetaan) automaattisesti satunnaisen näköisiä sanomia $\bar{c}$, tulee helposti avattua mitä tahansa sanomia c. Valitaan $\bar{c} = c \cdot x$:

$$\bar{m} \equiv \bar{c}^d \equiv c^d (x^e)^d \equiv mx \pmod{n}.$$

    (adaptiivinen valitun salakielen hyökkäys)

  Ratkaisu:
  - jos on tarpeen avata vastustajan valitsemia sanomia, lisätään ennen salausta selväkieleen redundanssia (esim. 64 vakiobittiä) ja palautetaan vain oikean muotoiset sanomat.

- **Forward search attack**: jos mahdollisia selväkieliä on liian vähän, on mahdollista kokeilla ne kaikki läpi.

  Ratkaisu:
  - lisätään sanomaan **suola**, 64 satunnaista bittiä.
- Eri avainpareissa käytettävä eri modulia n.

  Jos käytetään yhteistä modulia,
  - d ja (e, n) riittävät tekijöiden p ja q laskemiseen, joten toisten yksityiset avaimet ratkeavat
  - jos salataan sama sanoma eri eksponenteilla, selviää sen sisältö ulkopuolisille (Schneier s. 472).
- **Alkulukujen generointiin käytettävä luotettavaa algoritmia, syötteenä satunnaisia bittejä.**
- Uusien tekijöihinjakoalgoritmien ja kasvaneiden avainpituuksien (>1024 bittiä) takia kirjallisuudessa esitetyt erityisvaatimukset RSA:n p:lle ja q:lle (vahvoja alkulukuja jne.) voidaan unohtaa.

Tekijöihin jako

- Prosessorien nopeuden kasvu ei ole uhka RSA:lle. Lisätään vain avaimen pituutta.

  Algoritmien kehitys saattaa tuottaa yllätyksiä.
- Julkisen avaimen salakirjoitusten avaimet ovat huomattavasti pitempiä kuin lohkosalakirjoituksien. *Brute force*-hyökkäys ei koskaan kannata.
- Pisimmät eri vuosina tekijöihin jaetut luvut

vuosi	desimaaleja	bittejä
1983	71	236
1985	80	266
1988	90	299
1989	100	332
1993	120	399
1994	129	425
1996	130	429

- Parannuksia 1996:
  - mukana suuri määrä koneita internetissä
  - *Multiple polynomial quadratic number sieve* $\longrightarrow$ *General number field sieve.*

Alkulukujen generointi

- RSA-avainta varten tarvitaan satunnaisia, suuria alkulukuja.
- Propabilistinen alkulukutesti kertoo tietyllä todennäköisyydellä, onko luku alkuluku.

  Miller-Rabin-testi paljastaa jaollisen luvun todennäköisyydellä $3/4$. Iteroidaan testiä haluttu määrä kertoja.

  Huom: testejä ei voi käyttää luvun jakamiseen tekijöihin.
- Valitaan satunnainen luku halutulta alueelta ja etsitään kokeilemalla seuraava suurempi alkuluku.

  Alkulukuja on paljon, $TN(n \text{ on alkuluku}) \approx 1/\ln(n)$. Kokeiltavien lukujen määrä on lineaarinen suhteessa n:n pituuteen.

Allekirjoitus

- Julkisen avaimen allekirjoituksen tavoitteet
  - **Alkuperän todennus**: vain yksityisen avaimen haltija voi allekirjoittaa. Kuka tahansa voi tarkistaa aitouden julkisella avaimella.
  - **Eheys**: muut kuin allekirjoittaja eivät voi muuttaa allekirjoitettua sanomaa huomaamatta.
  - **Kiistämättömyys**: allekirjoittaja ei voi myöhemmin kieltää allekirjoittaneensa.
- Allekirjoituksen tarkistus on yleisin kryptografinen operaatio.

  Eheyttä ja alkuperän autentikointia tarvitaan useammin kuin salausta.
- Jotta allekirjoitus voidaan tarkastaa, pitää allekirjoitetussa sanomassa olla riittävästi **redundanssia**. Pitää olla jotain mitä verrataan, jotta aito sanoma erotetaan väärästä.
- Yleensä allekirjoitetaan vain sormenjälki ja liitetään se sanoman perään.

  (Allekirjoitus on kuin MAC, mutta tarkistus tapahtuu julkisella eikä salaisella avaimella.)

- Yleensä allekirjoitus tarkistetaan erityisellä tarkistufunktiolla, joka palauttaa arvon *true* tai *false*.

  Allekirjoittaminen:

  1. Lasketaan sanomasta sormenjälki $h(m)$.
  2. Allekirjoitetaan sormenjälki A:n yksityisellä avaimella
  $$S_A(h(m)).$$
  3. Liitetään allekirjoitus sanoman perään,
  $$m,\ S_A(h(m)).$$

  Tarkistaminen:

  1. Lasketaan sanomasta sormenjälki $h(m)$.
  2. Verifioidaan allekirjoitus A:n julkisella avaimella:
  $$V_A(h(m),\ S_A(h(m))\ =\ \text{true}\,?$$

- Joillakin allekirjoituksilla on käänteisfunktio, joka voidaan laskea julkisen avaimen avulla.

  Tällöin allekirjoitusta ei tarvitse liittää sanoman perään, vaan riittää, että allekirjoitetussa sanomassa on tarpeeksi redundanssia aidon sanoman tunnistamiseen (esim. 64 vakiobittiä).

  Allekirjoittaminen:

  - Liitetään sanomaan redundanssia, $m, 0$.
  - Allekirjoitetaan sanoma A:n yksityisellä avaimella
  $$S_A(m, 0).$$

  Tarkistaminen:

  1. Avataan allekirjoitus julkisella avaimella,
  $$V_A(S_A(m, 0)) = m, 0.$$
  2. Jos avatussa sanomassa on tunnistettava redundanssi jäljellä, hyväksytään allekirjoitus.

- Allekirjoituksessa voidaan käyttää satunnaista parametria, jotta allekirjoitus ei paljasta sanoman sisältöä.

  Joissain allekirjoitusmenetelmissä parametri tarvitaan aina.

  Satunnaisuutta voidaan lisätä esim. laskemalla sormenjälki sanomasta ja 64-bittisestä (pseudo)satunnaisluvusta,
  $$h(m, r).$$
  Satunnainen parametri pitää tällöin liittää allekirjoitukseen sen tarkistamista varten,
  $$E_B(m),\ S_A(h(m, r)), r.$$
- Kiistämättömyys on myös juridinen ongelma.

  Allekirjoitus pitää osoittaa aidoksi kolmansille osapuolille.

  Allekirjoittaja voi väittää yksityisen avaimensa paljastuneen ja kiistää allekirjoituksen. Joutuuko hän vastaamaan allekirjoituksestaan?

RSA-allekirjoitus

- Allekirjoitus ja tarkistus

$$s = m^d \pmod n.$$
$$m = s^e \pmod n.$$

RSA-allekirjoituksella on siis käänteisfunktio.

- Modulin n pitää olla vähintää 768 bittiä (suositeltava 1024–2048) ja salaisen avaimen d suurin piirtein yhtä pitkä. Julkinen avain voi olla lyhyt. (Ei tunneta mitään lyhyydestä aiheutuvia heikkouksia.)
- Kun allekirjoittaminen on sama asia kuin salauksen avaaminen, mitä tahansa ei pidä allekirjoittaa.

  Varovaisuuskaan ei auta. RSA säilyttää kertolaskun,

  $$(x\,m)^d \bmod n \;=\; (x^d \bmod n)(m^d \bmod n).$$

  Jos c on salakielinen teksti, valitaan mielivaltainen x, lasketaan $x^e\,c \mod n$ ja annetaan se allekirjoitettavaksi. Saadaan $(x^e\,c)^d = x\,m$. Tästä ratkaistaan helposti m.
- Opetus: Salaukseen ja allekirjoitukseen pitää käyttää eri funktoita.
  - Mielellään eri avaimet salaukseen ja allekirjoitukseen.
  - Lisätään sanomaan redundanssia ennen allekirjoitusta, esim. vakio-otsake.
  - Lasketaan sormenjälki ja allekirjoitetaan vain se.

Allekirjoitus ennen salausta!

- Hyökkäys 1:

  Alice salaa sanoman Bobin julkisella avaimella ja allekirjoittaa omalla yksityisellä avaimellaan,

  $$E_B(M),\ S_A(E_B(M)).$$

  Vaikka Carol ei saa selville sanomaa M, hän voi lähettää saman sanoman Bobille vaihtamalla allekirjoituksen,

  $$E_B(M),\ S_C(E_B(M)).$$
- Hyökkäys 2 (RSA:ta ja joitain muita vastaan):

  Alice salaa sanoman Bobin julkisella avaimella (e_B, n_B) ja allekirjoittaa omalla yksityisellä avaimella (d_A, n_A),

  $$c = (m^{e_B} \bmod n_B)^{d_A} \bmod n_A.$$

  Bob saa sanoman m, mutta haluaa väittää, että sanoma olikin m'. Bob tuntee n_B:n tekijät ja pystyy laskemaan diskreettejä logaritmeja modulo n_B. Hän etsii x:n, jolle

  $$m'^x \equiv m \pmod{n_B}.$$

  Bob julkistaa $x\,e_B$:n uutena julkisena avaimenaan ja väittää, että Alice lähetti hänelle sanoman m', sillä

  $$c = (m'^{(x\,e_B)} \bmod n_B)^{d_A} \bmod n_A.$$
- Sama hyökkäys toimii muitakin julkisen avaimen allekirjoitusmenetelmiä vastaan.

- Hyviä periaatteita:
  - Allekirjoitus tehtävä ennen salausta.
  - Allekirjoita vain sanomia, joiden sisällön ja alkuperän tunnet ja joiden tulkinta on yksikäsitteinen.
  - Lisää allekirjoitettavaan sanomaan redundanssia (= vakio-kenttä).
  - Jos allekirjoitus voi paljastaa salaista tietoa, lisää allekirjoitettavaan sanomaan satunnaisuutta.
- Jos lyhyt lohko ensin allekirjoitetaan ja sitten salataan RSA:lla, pitää salausmodulin olla suurempi kuin allekirjoitusmoduulin.

  Ratkaisu 1: Allekirjoitusavaimet erikseen ja pitempiä.

  Ratkaisu 2: pilkotaan allekirjoitus lohkoihin ja salata lohko kerrallaan.

  Ratkaisu 3: käytetään hybridi-salausta (lohkon lyhyydestä huolimatta).

Julkisten avainten aitous

- Julkisia avaimia ei tarvitse jakaa turvallista kanavaa myöten, mutta tiedätkö varmasti, kenen julkista avainta olet käyttämässä?

  Julkisen avaimen aitoudesta varmistuminen on julkisen avaimen menetelmien keskeinen ongelma.

  Ratkaisuja:
  - Luotettu taho ylläpitää avainhakemistoa, josta avaimet haetaan autentikoitua kanavaa pitkin.
  - Luotettu taho **sertifioi** avaimen eli allekirjoittaa todistuksen (**sertifikaatin**) avaimen kuulumisesta tietylle henkilölle tai kelpaamisesta tiettyyn käyttöön.
- Jos yksityinen avain paljastuu tai *peruutetaan*, julkista avainta ei enää pidä hyväksyä. Jos yksityinen avain katoaa, ei julkisella avaimella kannata enää salata. Miten varmistutaan avaimen voimassaolosta?

  Ratkaisuja:
  - vaaditaan julkisen avaimen **sertifiointia määräajoin**
  - ylläpidetään **sulkulistoja** peruutetuista sertifikaateista
  - velvoitetaan lainsäädännöllä tai sopimuksilla ihmisiä huolehtimaan avaimestaan
  - suljetaan avain älykortille, jossa se on turvassa.

Yksisuuntaiset funktiot ja salaovet

(*trapdoor one-way function*)

- **Yksisuuntainen funktio** on helppo laskea, mutta käänteisfunktio on vaikea.
- Yksisuuntaisena pidettyjä funktioita:
  - kryptografiset hajautusfunktiot
  - alkulukujen kertolasku
  - diskreetti eksponenttifunktio
  - diskreetti neliö
- Yksisuuntaisten funktioiden olemassaolo on kryptografian suurimpia avoimia kysymyksiä.
- Julkisen avaimen salakirjoitus on yksisuuntainen funktio, jos yksityinen avain unohdetaan.

  Yksityinen avain on salaovi (*trapdoor*) , jonka avulla käänteisfunktion laskeminen on helppoa.