

- ✓ Symmetrisestä avaimesta sopiminen
- ✓ Tiiviste- ja MAC –funktiot
- ✓ Digitaalinen allekirjoitus

# Symmetrisestä avaimesta sopimisen protokollat

## Key Exchange Protocols:

RSA Exchange

DH Exchange

ECDH Exchange

# Mitä ovat Key Exchange protokollat?

TLS- yhteyksissä ja salatussa sähköpostiliikenteessä data salataan nopealla symmetrisellä salauksella, joka on lähes poikkeuksetta AES.

Ennen AES - salausta pitää sopia istuntoavaimesta. Tähän on kolme vaihtoehtoista menetelmää:

- **RSA Exchange** on ollut yleisin viime vuosiin saakka
- **ECDHE** on korvannut RSA Exchange:n mm. verkkopankeissa
- **DH Exchange** on protokolla, jota on käytetty mm. monenkeskisissä AES:lla suojatuissa videoneuvotteluissa

# RSA key exchange

- Standardi TLS – yhteyksissä vielä 2017 lopulla

## **Periaate:**

**Yhteyden jompikumpi osapuoli luo satunnaislukugeneraattorilla symmetrisen AES –avaimen ja lähettää sen toiselle osapuolelle RSA:lla salattuna**

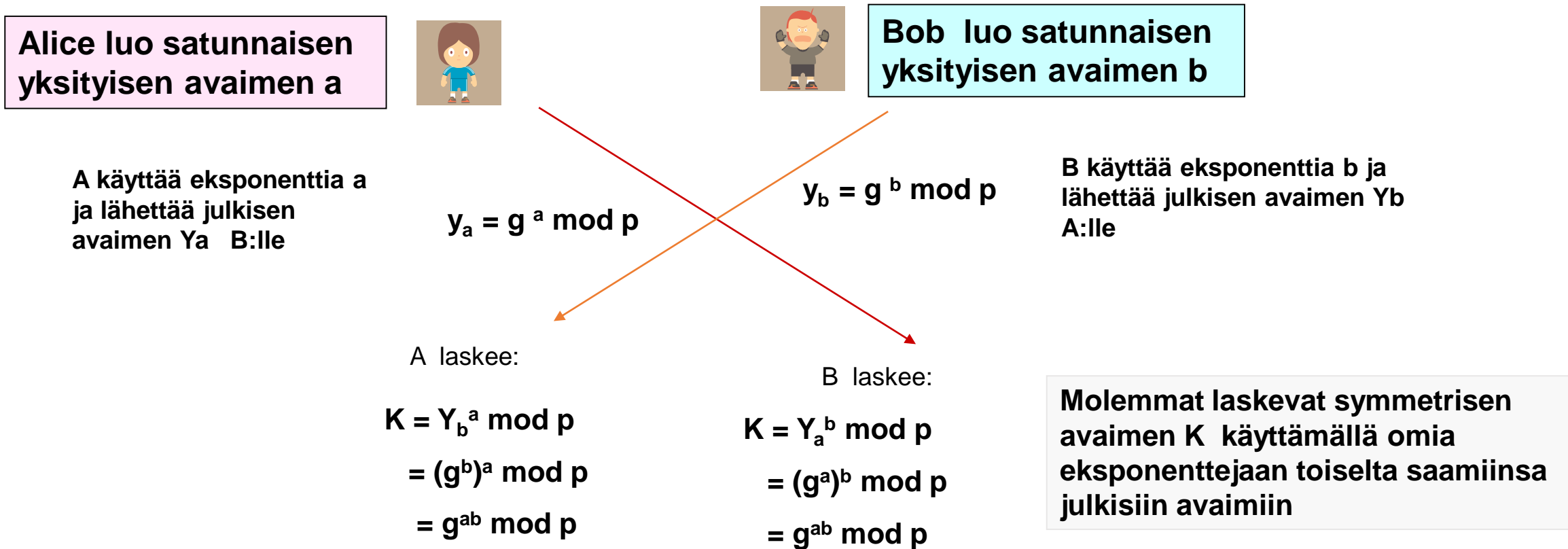
**RSA – key exchange protokollaan liittyvä turvallisuusnäkökohtia**

- 1. Jos ”HAKKERI” onnistuu faktoroimaan palvelimen julkisen avaimen  $n$  muotoon  $p \cdot q$ , hän voi tästä eteenpäin avata kaikki avaimen sisältävät viestit ja siten salakuunnella viestiliikennettä.**
- 2. VIRANOMAISET voivat autoritäärisissä maissa vaatia esim. pankkia luovuttamaan serverin yksityisen avaimen valtiolle**
- 3. SATUNNAISLUKUIHIN LIITTYVÄT RISKIT. On ilmennyt tapauksia, että satunnaislukugeneraattorissa on takaportti, sen tuottamat avaimet eivät olekaan satunnaisesti jakautuneita**

# Diffie – Hellman key exchange

Järjestelmäparametrit: **Suuri** (2048 bitin) **alkuluku**  $p$

ja **generoiva alkio**  $g$  (kantaluku potenssilaskuille) **annettu**



DH on tietyssä mielessä turvallisempi kuin RSA key exchange, sillä joka istunnossa luodaan uudet yksityiset avaimet  $a$  ja  $b$ . Jos yhden istunnon salauksen onnistuisi murtamaan, seuraavaa ei.

# Diffie Hellman exchange :n turvallisuus

Hyökkääjä voi napata Alicen ja Bob:n toisilleen lähettämät julkiset avaimet

$$Y_a = g^a \bmod p \text{ ja } Y_b = g^b \bmod p$$

mutta ei kykene ratkaisemaan yksityisiä avaimia  $a$  ja  $b$ , joista ainakin toinen tarvittaisiin symmetrisen avaimen  $K$  laskemiseen.

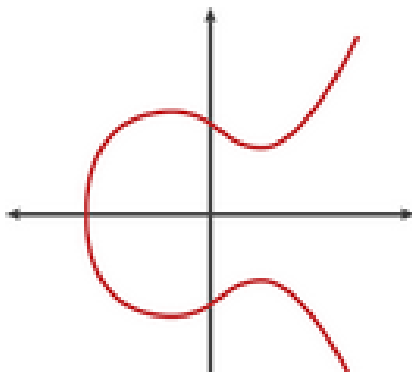
**DLP = "Discrete Logarithm Problem"** (diskreetin logaritmin ongelma) on tämän matemaattisesti erittäin vaikeana pidetyn tehtävän nimitys:

**DLP:** Ratkaistava kokonaislukueksponentti  $x$  yhtälöstä  $y = g^x \bmod p$  missä  $y$ ,  $g$  ja  $p$  on annettu.

**Turvallisena pidetty modulus  $p$  on suuruudeltaan vähintään 2048 bittiä**

# ECDHE Elliptic Curve Diffie Hellman Exchange

Mm. Verkkopankit ja monet muut palvelimet ovat siirtyneet v.2018 alussa käyttämään ECDHE avaimesta sopimisen menetelmää, joka on korvannut aiemman standardin RSA exchange



Elliptiset käyrät joita käytetään salauksessa ovat muotoa

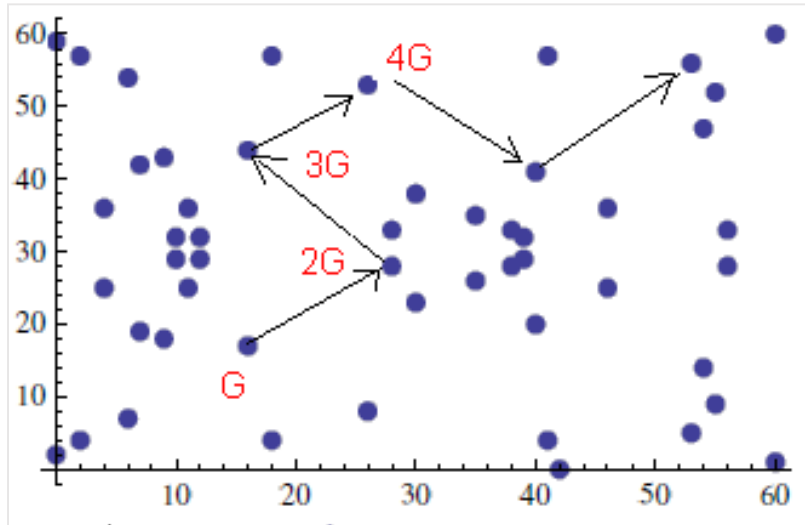
$$y^2 = x^3 + a x + b$$

Käyrän pisteiden P ja Q välille on mahdollista määritellä yhteenlasku, joka antaa kolmannen käyrän pisteen, jota sanotaan summaksi  $P + Q$

**Diskreetin elliptisen käyrän muodostavat kokonaislukuparit  $(x,y)$  jotka toteuttavat yhtälön**  
 **$y^2 = x^3 + a x + b \pmod{q}$  , missä modulus  $q = \text{alkuluku}$**

# Diskreetti Elliptinen käyrä

Esim  $y^2 = x^3 + 2x + 4 \pmod{61}$ , missä  $0 < x, y < 60$



Pisteet joille  $y^2 = x^3 + 2x + 4 \pmod{61}$

Käyrän pisteille voidaan määritellä yhteenlasku

$$k = (y_2 - y_1) \cdot (x_2 - x_1)^{-1} \pmod{q}$$

$$x_{P+Q} = k^2 - x_1 - x_2 \pmod{q}$$

$$y_{P+Q} = y_1 - k(x_{P+Q} - x_1) \pmod{q}$$

Käyrän pisteiden välille voidaan määritellä yhteenlasku, joka muodosta ryhmärakenteen. Yhteenlaskun avulla voidaan määritellä pisteen monikerta.

Pisteistä osa on ns. Generoivia alkioita, joiden monikerrat kattavat kaikki käyrän pisteet.

Esimerkkikäyrällä eräs generoiva piste on

$$\mathbf{G} = (2, 4)$$

Generaattorin G monikerrat antavat kaikki ryhmän alkiot:

$$\{G, 2G, 3G, \dots, 60G, O\}$$



# ECDHE:n periaate

Järjestelmäparametrit ovat käyrä ja generaattori  $G$

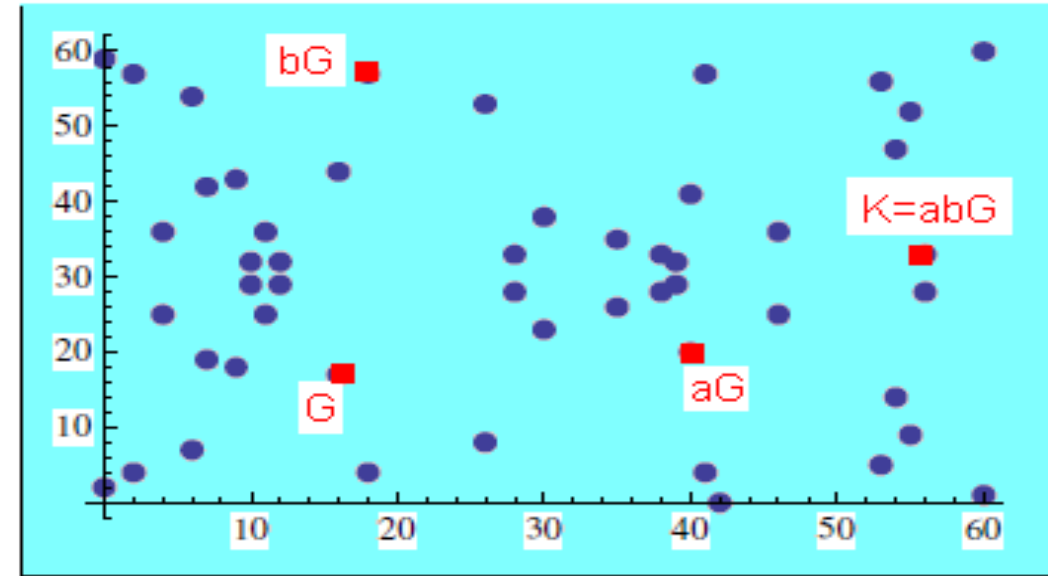
1. Alice valitsee **yksityisen avaimen**  $a$  ja lähettää Bobille käyrän pisteen  $aG$

2. Bob valitsee **yksityisen avaimen**  $b$  ja lähettää Alicelle käyrän pisteen  $bG$

3. Molemmat laskevat symmetrisen avaimen (= piste  $abG$ )

Alice:  $K = a(bG)$

Bob:  $K = b(aG)$



**TLS:ssä** käytetään standardoituja käyriä, joissa **modulus  $q$**  ja **pisteiden määrä on n. 250 – bittinen kokonaisluku**

**Turvallisuus:** Hyökkääjä voi napata A:n ja B:n toisilleen lähettämät viestit, jotka sisältävä pisteet  $aG$  ja  $bG$ . Mikäli käyrän pisteiden määrä on riittävän suuri, hyökkääjä ei kykene laskemaan Alicen ja Bob:n yksityisiä avaimia  $a$  ja  $b$  näistä viesteistä esim. 100 vuoden ajassa.

**ECDLP = "Elliptic curve Discrete Logarithm Problem"** on yllä kuvatun matematiikan "Hard Problem" luokkaan kuuluvan tehtävän nimitys

# SANASTOA:

## **Protokolla eli yhteyskäytäntö**

on käytäntö tai [standardi](#), joka määrittelee tai mahdollistaa laitteiden tai ohjelmien väliset yhteydet. Toinen osapuoli lähettää viestin toiselle, tämä reagoi siihen ja mahdollisesti vastaa toisella viestillä protokollan mukaisesti.

Abstraktimmin tämä voidaan nähdä tietokoneissa olevan [tilakoneen](#) tilan vaihtoina toisen koneen viestien perusteella.

## **Avaimenvaihtoprotokolla**

= standardimenettely, jolla voidaan sopia symmetrisestä salausavaimesta

**DLP** = matemaattisesti vaikea diskreetin logaritmin ongelma, johon DH protokollan turvallisuus perustuu

**ECDLP** = matemaattisesti vaikea diskreetin logaritmin ongelmaa vastaava ongelma elliptisellä käyrällä, johon ECDHE protokollan turvallisuus perustuu

- Tiivisteeet ja MAC:it
- Digitaalinen allekirjoitus

# Tiivistefunktiot (hash -functions)

**Määritelmä:** Tiivistefunktiot ovat yksisuuntaisia funktioita, jotka tuottavat viestistä määrämittaisen tiivisteeseen (tarkistussumman)

**Tiivisteiden käyttötarkoituksia:**

1. Tiiviste varmistaa tiedonsiirron eheyden (ts. että tieto ei ole muuttunut siirron aikana).
2. Palvelimien salasanatiedostoihin ei tallenneta salasanoja vaan salasanatiivisteitä
3. Digitaalisen allekirjoituksen laskemisessa käytetään tiivisteitä.
4. Tietoturvaohjelmien tarkistussummat kiintolevyn kansioista ovat tiivisteitä

# TIIVISTEFUNKTIOILTA vaadittavia tietoturvaominaisuuksia

Vuonna 2005 useita siihen asti käytössä olevia tiivisteitä murrettiin, voidaan jopa sanoa niiden olleen tuolloin salausohjelmistojen heikko kohta.

## Hyvän tiivistefunktion vaatimukset:

Seuraavassa merkitään tiivistefunktiot  $h(m)$ , missä  $m$  on viesti, josta tiiviste lasketaan,

1.  $h(m)$  on **yksisuuntainen funktio**. Tiivisteestä ei ole mahdollista laskea taaksepäin itse viestiä.
2. Kun tunnetaan yhden viestin tiiviste  $h(m)$ , on siitä mahdotonta generoida toista viestiä jolla olisi sama tiiviste. **(collision resistance 1)**
3. On mahdotonta generoida kahta viestiä  $m_1$  ja  $m_2$  joilla olisi sama tiiviste. **(collision resistance 2)**

# Tunnettuja tiivisteitä



"Chinese Xiaoyun Wang broke in 2005 almost all hash functions in sight - and paradoxally the community of cryptographers loved it"

Kiinalaiset tutkijat osoittivat v. 2005 useiden siihen asti käytössä olevien tiivisteiden haavoittuvuuden. Käytännössä he demonstroivat miten annetulle tiivisteelle  $h$  voidaan generoida viesti  $m$ , jolla on sama tiiviste.

- MD5 - ei suositella, murrettu
- SHA-1 (160 b) - ei suositella, murrettu
- Seuraavia tiivisteitä ei ole murrettu ja niitä pidetään turvallisina
- SHA256
- SHA384
- SHA512

Esimerkki TLS:ssä yleisestä SHA256 tiivisteestä.

```
Hash["Tämä on koeviesti, josta lasketaan tiiviste","SHA256"]  
32134986657396586487192643852384396035054468584117070131813446773863278924851
```

# Turvallisen tiiviste bittimäärä ja "birthday paradox"

Turvallisen tiiviste bittimäärän tulee olla 2 x lohkosalaimen turvallinen avainpituus, eli nykyisin 2 x 128 bittiä = 256 bittiä.

Taustalla on ns. "syntymäpäiväparadoksi" (birthday paradox).

Mikä on sellainen koululuokan vähimmäiskoko, että luokassa on yli 50% todennäköisyydellä kaksi henkilöä jolla on sama syntymäpäivä ? Vastaus: 23

Tulos perustuu  
todennäköisyslaskentaan

$$P = 1 - \prod_{k=1}^{23} \frac{365 - k + 1}{365} = 0.51$$

Syntymäpäiväparadoksi voidaan yleistää tiivisteille seuraavasti:

Jos tiivisteavaruuden koko on  $n$ , noin  $\sqrt{n}$  tiivisteiden joukossa on yli 50% todennäköisyydellä "törmäyksiä"  
=> 256 bittisiä tiivisteitä on  $2^{256}$  kpl. Jo  $2^{128}$  :n tiivisteiden joukossa on yli 50% todennäköisyydellä väh. 2 samaa tiivistettä.

# MAC -funktiot

MAC = message authentication code

- MAC = tiiviste, jonka syötteenä on viestin m lisäksi symmetrinen salausavain
  - MAC:llä voidaan taata paitsi viestin muuttumat-tomuus, myös lähettäjä (koska avain tiivistetään viestin mukana)
- \* MAC tiivistettä voidaan käyttää digitaalisen allekirjoituksen tapaan tiedonsiirron pakettien perässä samaan takaamaan pakettien muuttumattomuus ja lähettäjän oikeellisuus.



# HMAC-SHA

= tavallisin MAC: **keyed hash message authentication code**  
lasketaan viestistä  $m$  ja symmetrisestä avaimesta seuraavasti

$$\text{HMAC}(K, m) = \text{sha}(K \oplus \text{opad} || \text{sha}(K \oplus \text{ipad} || m))$$

$\oplus$  = XOR summa (esim. 1101 + 0101 = 1000)

$K$  = symmetrinen avain

Ipad ja Opad ovat 64 bitin mittaisia vakioita. Alla vakiot heksadesimaalimuodossa

Ipad = 5c5c5c5c , Opad = 36363636

$||$  liittäminen (concatenation) (esim. "auto"  $||$  "mat" = "automat" )

**sha-HMAC** – funktiota käytetään yleisesti pseudosatunnaisluku-  
generaattorina laitteissa, jotka tuottavat kertakäyttösalasanoja  
esim. verkkopankkisovelluksiin. Tästä lisää kurssin viimeisessä  
luennossa.



# Digitaalinen allekirjoitus (digital signature)

- Digitaalinen allekirjoitus mahdollistaa turvallisen asioinnin verkossa. Juridisesti digitaalisesti allekirjoitettu asiakirja vastaa todistajien oikeasti todistamaa asiakirjaa.

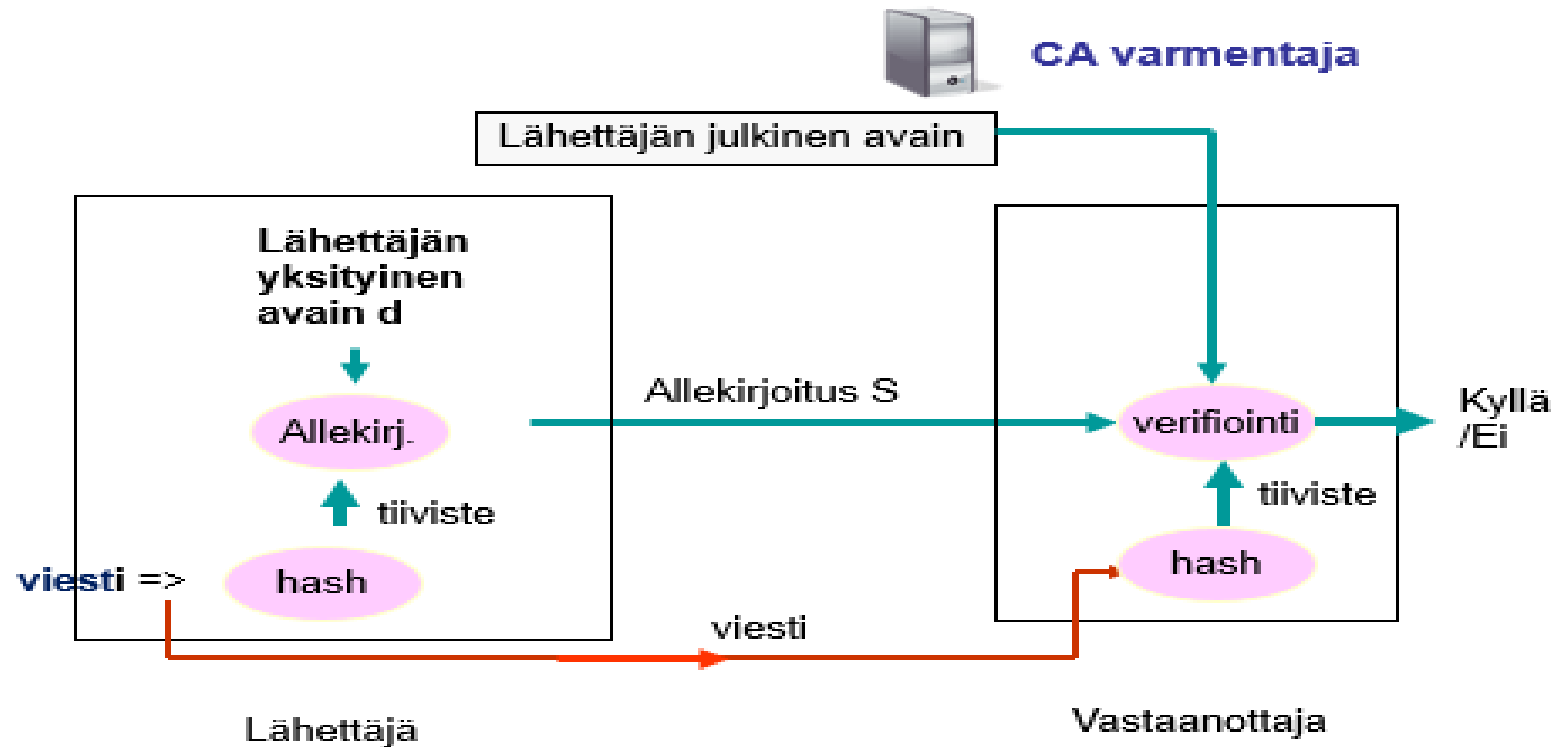
Tarkoitus on varmistaa, että

- 1) viesti on muuttumaton
- 2) lähettäjä on todennettu

Digitaalisessa allekirjoituksessa käytetään tiivistettä ja julkisen avaimen salausta

Tyypillinen yhdistelmä on RSA ja SHA –tiiviste. Tällainen digitaalinen allekirjoitus on **sha256RSA**, joka on yleinen TLS –yhteyksissä. (uudempi ECDSA on osittain korvannut sen)

# Digitaalinen allekirjoitus kaaviona



**Digitaalinen allekirjoitus on luku, joka lasketaan viestin tiivisteestä salaamalla se lähettäjän yksityisellä avaimella**

**Vastaanottaja verifioi seuraavasti:**

- 1) purkaa digitaalisen allekirjoituksen lähettäjän julkisella avaimella ja saa näin viestin tiiviste.
- 2) Hän laskee itse viestistä toisen tiiviste. Jos tiivisteet täsmäävät, lähettäjä on varmennettu ja viesti muuttumaton

# Digitaalinen allekirjoitus sha256RSA

Allekirjoituksen  
laskukaava:

$$S = sha256(m)^d \bmod n$$

sha256(m) on viestin m SHA-tiiviste

d = viestin lähettäjän yksityinen RSA avain

n = viestin lähettäjän julkinen RSA avain

Esimerkki. **Lasketaan digitaalinen allekirjoitus shaRSA** viestille **"Tämä on koeviesti"**. Viestin lähettäjän RSA- avaimet ovat  
Julkinen avain  $n = 308911$  , Yksityinen avain  $d = 133073$

Viesti  $m = \text{"Tämä on koeviesti"}$ .

Käytetään SHA tiivistettä, josta kuitenkin otetaan vain 4 numeroa

WolframAlpha:

SHA "Tämä on koeviesti"

antaa 9126...

Lasketaan **digitaalinen allekirjoitus** tiivisteestä lähettäjän yksityistä avainta käyttäen

$$S = sha(m)^d \bmod n = 9126^{133073} \bmod 308911 = 298954$$

Allekirjoitus on siten 298954

## Näytetään, miten vastaanottaja verifioi allekirjoituksen?

Vastaanottaja purkaa allekirjoituksesta viestin tiivisteen lähettäjän julkisella avaimella  $n$ .

$$S^e \bmod n = 298954^{65537} \bmod 308911 = 9126$$

Vastaanottaja laskee viestiosasta tiivisteen.

SHA "Tämä on koeviesti" antaa **9126**...

Vastaanottaja vertaa edellisten kohtien 1 ja 2 tuloksia

TIIVISTEET TÄSMÄÄVÄT => viesti ei ole muuttunut ja lähettäjä on varmennettu