Luku II: Kryptografian perusteita

Tässä toisessa luvussa esitellään muutamia peruskäsitteita ja -tekniikoita

- symmetrisestä salauksesta,
- julkisen avaimen salauksesta eli epäsymmetrisestä salauksesta,
- kryptografisista tiivistefunktioista,
- digitaalisista allekirjoituksista.

Tietokonesalaus

Ehdot symmetriselle (osapuolilla sama salainen avain) tietokonesalaukselle:

- Avain on kohtuullisen kokoinen (< 250 bittiä kaupallisissa sovelluksissa, sotilaallisissa voi olla suurempikin).
- Avainta on voitava käyttää useaan kertaan.
- Salaus on nopeaa ja salauspiirit halpoja.
- Salauksella tulee olla hyvät *diffuusio-ominaisuudet* eli yksi selvätekstin bitti vaikuttaa moneen salatekstin bittiin niin, että selvätekstin tilastollinen rakenne häviää.
- Lisäksi salauksella tulee olla hyvät *sekoitusominaisuudet* eli on vaikeaa päätellä, miten salatekstin tilastollinen rakenne riippuu selvätekstin tilastollisesta rakenteesta.

Salausmenetelmien luokittelu I

Symmetrinen tietokonesalaus on joko jonosalausta tai lohkosalausta.

- Jonosalaus tapahtuu merkki merkiltä: selvätekstin merkkiä muutetaan ja se ketjutetaan lähettävään salamerkkien virtaan. Tunnetuin jonosalaaja on RC4 ja sen uusi versio RC5.
- Jonosalausta käytetään tilanteissa, joissa salauksen on oltava nopeaa ja reaaliaikaista tiedon synnyn kanssa. Esimerkiksi GSM-puheluissa on käytetty jonosalausta A5/1, A5/2.
- Jonosalausta ei ole kuitenkaan pidetty täysin turvallisena, josta syystä siitä ei ole edes standardeja.
- Uusimpia jonosalaajia ovat Salsa ja Sosemanuk.

Salausmenetelmien luokittelu II

- Lohkosalauksessa selväteksti jaetaan lohkoihin (128 b, 256 b, 512 b), jokainen lohko salataan samalla salaisella avaimella ja lohkot lähetetään joko sellaisenaan tai ketjutettuna. Avaimen pituus on sama kuin lohkon pituus. Tällä hetkellä vallitseva standardi on AES (Advanced Encryption Standard, alunperin Rijndael). Toinen paljon käytetty on 3DES eli vanha DES kolminkertaisena kolmella avaimella.
- **Suorituskykyvertailu** (nopeus MB/sec)

RC4	126
Salsa20/12	643
Sosemanuk	727
3DES	13
AES-128	109

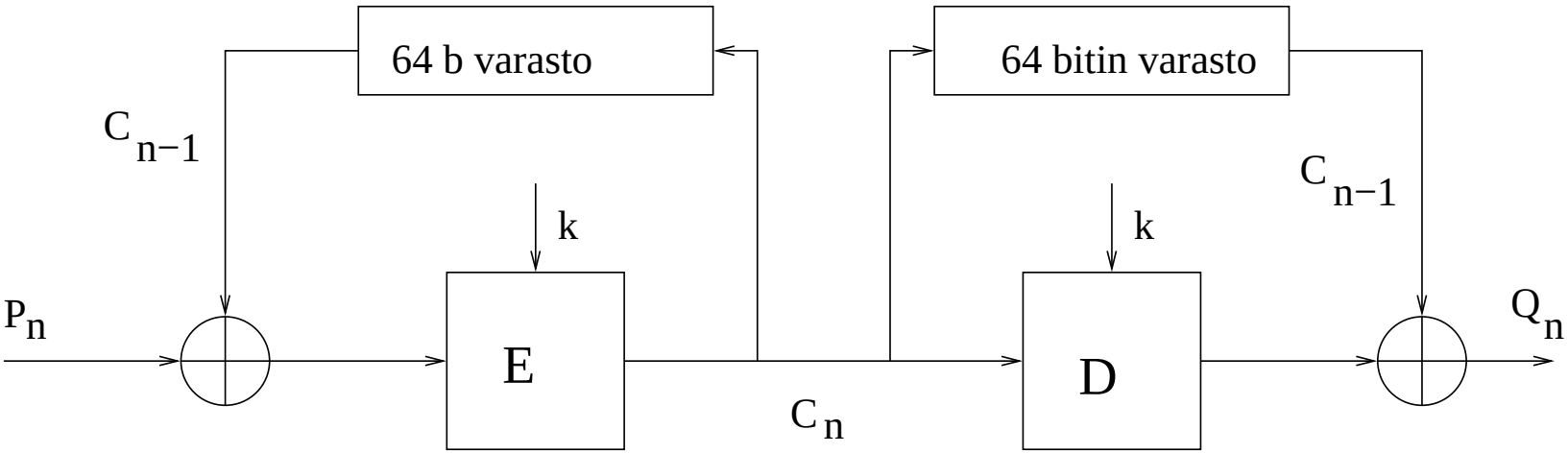
Symmetrinen salaus käytännössä: ECB I

- Yksinkertaisimillaan salausta käytetään siten, että selväteksti jaetaan esim. 128:n (AES) bitin lohkoihin, jokainen lohko salataan erikseen ja lohkot lähetetään vastaanottajalle.
- Kysymyksessä on ns. elektroninen koodikirja, ECB.
- Tällä menetelmällä on kuitenkin huonoja puolia:
  - Säännöllisyydet saattavat näkyä salatekstissä.
  - Esimerkiksi rahansiirrossa summa saattaa olla aina samalla paikalla. Suurista summista voi olla tietoa jne.
- Jotta esitetyiltä ongelmilta vältyttäisiin, pyritään salalohkot ketjuttamaan niin, että yksi salalohko vaikuttaa kaikkien muiden seuraavaksi tulevien koodaukseen.
- Näin sekoitusominaisuudet paranevat, eikä vakio-osia ole enää mahdollista paikantaa. Näitä ns. *ketjutustekniikoita* voidaan käyttää minkä tahansa symmetrisen lohkosalausmenetelmän kanssa.

Salalohkojen ketjutus I

- Salalohkojen ketjutuksessa (engl. cipher block chaining, CBC) selväteksti jaetaan lohkoiksi, joita ruvetaan salaamaan järjestyksessä.
- Nyt kuitenkin käytetään apuna yhteenlaskua modulo 2. Aina kun lohko on salattu, salattu lohko lasketaan yhteen modulo 2 seuraavan selvätekstilohkon kanssa, joka salataan vasta tämän jälkeen.
- Kaaviona salaus- ja purkuprosessi ovat seuraavan kuvan mukaisia (lohkon pituus kuvioissa 64, todellisuudessa esim. 128).

Salalohkojen ketjutus II



Kuva: CBC

Vastaavasti kaavoina:

Salalohkojen ketjutus III

$$\begin{aligned} C_n &= E_K(P_n \oplus C_{n-1}), \\ Q_n &= D_K(C_n) \oplus C_{n-1}, \\ D_K(C_n) &= P_n \oplus C_{n-1}, \\ Q_n &= P_n \oplus C_{n-1} \oplus C_{n-1} = P_n. \end{aligned}$$

- Ensimmäisen ja viimeisen lohkon käsittely vaatii erikoiskäsittelyä. Otetaan käyttöön 64 bitin *alustusmuuttuja* I, jota käytetään ensimmäisen selvälohkon salauksessa:

$$C_1 = E_K(P_1 \oplus I), \quad Q_1 = D_K(C_1) \oplus I.$$

- Yleensä alustusmuuttuja I on salainen. Viimeinen selvälohko on täydennettävä 64-bittiseksi. Tämä voidaan tehdä lisäämällä nollia tai satunnainen bittijono.

Ketjutuksessa tiedonsiirtovirheet leviävät laajemmalle kuin elektronisen koodikirjan tapauksessa. Oletetaan, että n. salalohkossa tapahtuu yhden bitin tiedonsiirtovirhe. Merkitään

- C_n virheetön salalohko,
- C'_n yhden bitin virheen sisältävä salalohko,
- Q_n selvälohko purkamisen jälkeen,
- $\tilde{Q}_n$ täysin virheellinen lohko purkamisen jälkeen,
- Q'_n yhden bitin virheen sisältävä selvälohko.

Nyt tiedonsiirtovirheen vaikutus selviä seuraavista kaavoista:

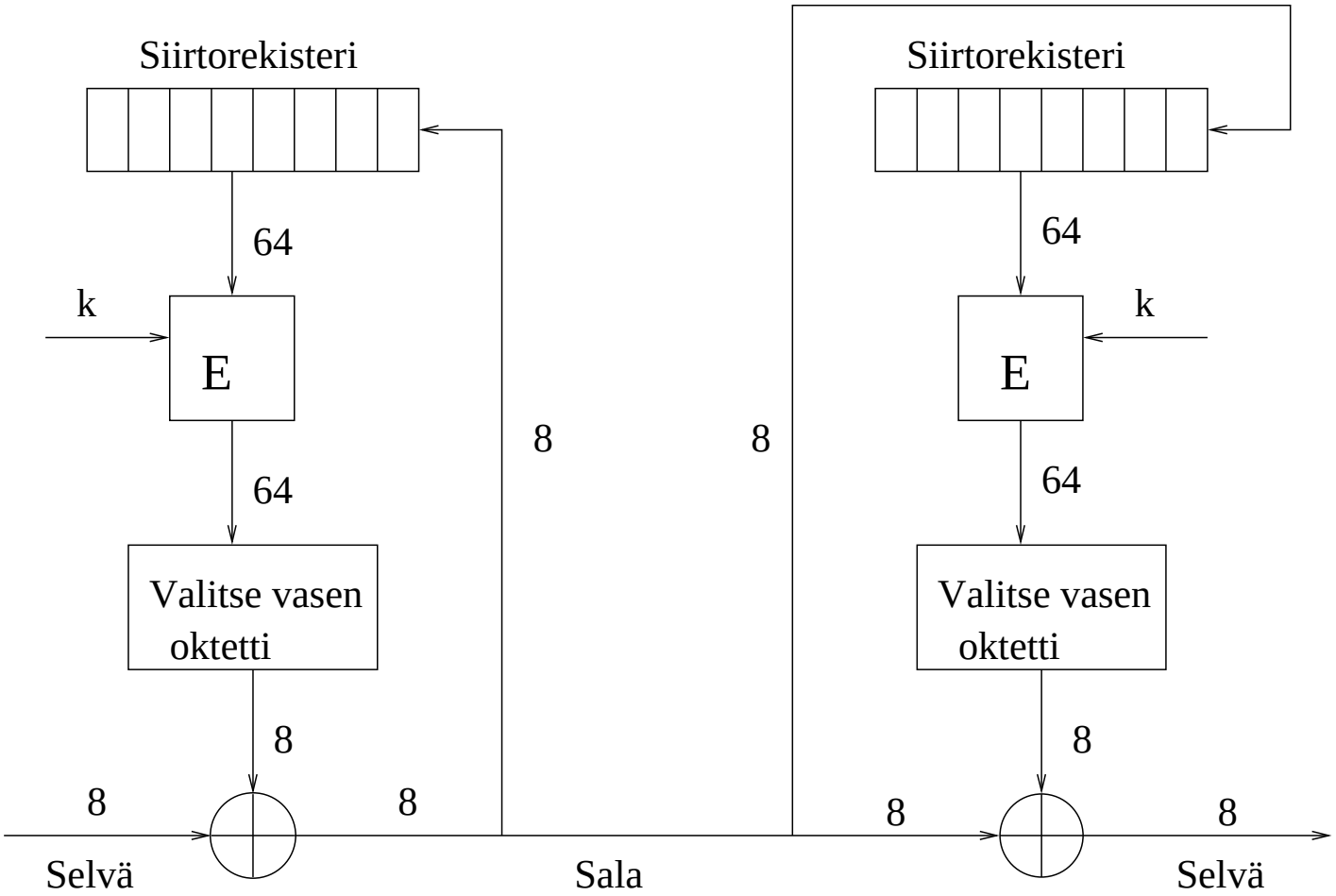
$$\begin{aligned} D_K(C'_n) \oplus C_{n-1} &= \tilde{Q}_n, \\ D_K(C_{n+1}) \oplus C'_n &= Q'_{n+1}, \\ D_K(C_{n+2}) \oplus C'_{n+1} &= Q_{n+2}. \end{aligned}$$

Eli yksi selvälohko tuhoutuu täysin ja yhdessä on yhden bitin virhe. Muut lohkot selviävät vaurioitta.

Salauksen takaisinkytkentä I

Salauksen takaisinkytkentää (engl. cipher feedback chaining, CFB) käytetään, kun salaus tapahtuu merkki merkiltä tai bitti bitiltä. Oletetaan, että merkin pituus on m bittiä. Yleensä $m = 8$. Menetelmän idea käy selville seuraavasta kaaviosta.

Salauksen takaisinkytkentä II



Kuva: CFC

Salauksen takaisinkytkentä III

Ensimmäisen oktetin kohdalla käytetään alustusmuuttujaa I, jonka pituus on sama kuin salausjärjestelmän lohkon pituus. Alustusmuuttuja sijoitetaan valmiiksi siirtorekisteriin. Se on luonnollisesti vaihdettava tarpeeksi usein. Edellisten merkintöjen lisäksi merkitään:

- S_n siirtorekisterin sisältö n. kierroksella,
- L_n on $E_K(S_n)$:n m vasenta bittiä.

Analysoidaan taas tiedonsiirtovirheen vaikutusta. Oletetaan, että n. kierroksella tapahtuu yhden bitin virhe salalohkossa. Tällöin kun $M = 8$:

- S_n ja L_n kunnossa, mutta $C'_n \oplus L_n = Q'_n$ eli selvälohkoon tulee n. kierroksella yhden bitin virhe;
- kierroksilla $n+1, \cdots, n+8$ tilanne on S'_i ja $\tilde{L}_i$, $i = n+1, \cdots, n+8$;
- Siten

$$C_i \oplus \tilde{L}_i = \tilde{Q}_i,$$

$i = n+1, \cdots, n+8$;

Salauksen takaisinkytkentä IV

- kierroksella $n + 9$ kaikki on kunnossa.

Siis virhe vaikuttaa 9 oktettiin.

Laskurimoodi I

Laskurimoodi eli CTR on saavuttanut suosiota viime aikoina, joskin se on vanha ehdotus. Salaus tapahtuu nyt muodossa

$$C_i = P_i \oplus E(K, L_i),$$

missä

- P_i on i.s selvälohko,
- L_i on laskurin arvo i. kierroksella,
- K on salainen, symmetrinen avain ja
- $\oplus$ on XOR-operaatio.

Tyypillisesti laskurilla on jokin sovittu alkuarvo, joka kasvatetaan joka kierroksella yhdellä. Mitään ketjutusta ei ole käytössä.
Menetelmällä on etuja:

- Laitteistotason tehokkuus.

Laskurimoodi II

- Ohjelmallinen tehokkuus.
- Esiprosessointi mahdollista (salaus).
- Lohkot voidaan prosessoida satunnaisessa järjestyksessä.
- CTR:n voidaan näyttää olevan ainakin yhtä vahva kuin muut ketjutusmenetelmät, jotka ovat olleet esillä.
- Tarvitaan vain salauksen toteutus, ei purun toteutusta.

Julkisen avaimen salaus RSA I

- Julkisen avaimen salauksen eli *epäsymmetrisen salauksen* edellytyksenä on, että löytyy salausmenetelmä, jossa on mahdotonta saada selville salaista purkuavainta D_K, vaikka salausavain E_K tunnetaan.
- Julkisen avaimen salauksen etuna on, että kuka tahansa voi lähettää salakirjoitetun viestin vastaanottajalle ilman, että molempien täytyy sopia etukäteen salaisesta avaimesta. Vastaanottaja on kuitenkin ainoa, joka pystyy purkamaan viestin salaisella avaimellaan.
- Julkisen avaimen salauksen idean julkaisivat ensimmäisinä Diffie ja Hellman vuonna 1976. Joissakin lähteissä mainitaan myös Merkle samanaikaisena keksijänä.
- Heidän ehdottamansa menetelmä oli kuitenkin lähinnä teoreettinen ja varsin epäkäytännöllinen.

Julkisen avaimen salaus RSA II

- Ensimmäinen julkisen avaimen julkinen käytännöllinen menetelmä oli RSA, jonka kehittivät Rivest, Shamir ja Adleman 1977. RSA on edelleenkin laajimmin käytössä oleva julkisen avaimen järjestelmä.
- Vuonna 1997 CESG (brittiläinen kryptografian osasto) julkaisi dokumentteja, jotka osoittivat, että James Ellis oli jo vuonna 1970 keksinyt julkisen avaimen salauksen.
- Samoin Clifford Clocks oli 1973 kuvannut RSA:n version, missä salauseksponentti e oli sama kuin modulus n.
- RSAn jälkeen on ehdotettu monia muita, mutta ne eivät ole saaneet samanlaista suosiota. Nykyään elliptisiin käyriin perustuvat menetelmät ovat kuitenkin nostamassa suosiotaan, koska niissä voidaan käyttää lyhyempää avainta (1500-2000 b -¿ 300-600 b).

Julkisen avaimen salaus RSA III

- Julkisen avaimen menetelmät eivät voi taata turvallisuutta joka tilanteessa. Jos vastapuolella nimittäin on salateksti, hän voi salata jokaisen mahdollisen selvätekstin ja verrata tulosta salatekstiin. Mikäli tulos on sama, on selväteksti paljastunut. Siten mahdollisia selvätekstejä on oltava suunnaton määrä.
- Lisäksi julkisen avaimen salaus on paljon hitaampaa kuin symmetrisen avaimen salaus, joten sitä ei yleensä käytetä datan salaamiseen. Sen sijaan julkisella menetelmällä salataan tavallisesti symmetrisen salauksen avaimia, joita näin välitetään osapuolelta toiselle.
- Julkisen avaimen salauksella toteutetaan useimmiten myös digitaalisen allekirjoitus, jota tarvitaan osapuolten todentamisessa.

RSA I

RSA:ssa avaimet ovat muodoltaan seuraavia:

- julkinen avain on lukupari (e, n);
- salainen avain on lukupari (d, n);
- selväteksti jaetaan lohkoihin, lohkon koon binäärilukuna tulee olla alle n eli lohko käsittää korkeintaan $\log_2(n)$ bittiä.

Salaus tapahtuu kaavalla

$$C = M^e \mod n.$$

Purkuun käytetään kaavaa

$$M = C^d \mod n = (M^e)^d \mod n = M^{ed} \mod n.$$

Ennenkuin systeemi toimii,

- on löydettävä sopivat luvut e, d ja n,

RSA II

- on laskettava tehokkaasti M^e ja C^d kaikilla $M < n$,
- d ei saa olla helposti laskettavissa e:stä ja n:stä.

Sivuutamme kuitenkin nämä tarkastelut, jotka vaativat hieman lukuteorian ja moduloaritmetiikan tuntemusta.

Digitaalinen allekirjoitus RSA:n avulla I

- Esimerkiksi elektronisessa kaupankäynnissä ja varmenteiden käytössä täytyy kyetä osoittamaan, että tietty taho on viestin lähettäjä. *Digitaalinen allekirjoitus* tähtää siihen, että se osoittaa kiistatta lähettäjän ja lähetyksen ajankohdan sekä todentaa sanoman.
- Lisäksi kolmannen osapuolen tulee kyetä verifioimaan allekirjoitus. Todentaminen tässä yhteydessä tarkoittaa, että allekirjoitettua viestiä ei voi muuttaa vaikuttamatta allekirjoitukseen.
- *Suora digitaalinen allekirjoitus* perustuu julkisen avaimen salaukseen. Edellytyksenä on, että salausmenetelmälle pätee ehto

$$E_{K_p}(D_{K_s}(M)) = M.$$

- Esimerkiksi RSA toteuttaa yllä olevan kaavan. Jos ehto pätee, sanoma allekirjoitetaan "salaamalla"se lähettäjän salaisella avaimella D_{K_s}.

Digitaalinen allekirjoitus RSA:n avulla II

- Vastaanottaja purkaa salauksen lähettäjän julkisella avaimella ja toteaa, että tuloksena on selväkielinen viesti.
- Jos lähettäjä haluaa, ettei viestiä voi lukea muut kuin vastaanottaja, viesti voidaan vielä salata vastaanottajan julkisella avaimella.
- Usein allekirjoituksen yhteydessä käytetään myös tiivistefunktiota (määritelmä hieman myöhemmin). Tiiviste allekirjoitetaan, ei alkuperäistä lähdettä.
- *Digitaalisen allekirjoituksen standardi* on muunnos ELGamalin allekirjoituksesta. Siinä allekirjoitus on kohtuullisen kokoinen päinvastoin kuin ElGamalissa. Se kehitettiin osaksi sen vuoksi, että RSA:n käyttöä säätelivät patentit.
- Näiden perustekniikoiden lisäksi on lukuisia muita, joita on lueteltu teoksessa Menezes, van Oorschot, Vanstone: *Handbook of Applied Cryptography,* joka on saatavissa ilmaiseksi verkosta.

Tiedon eheyden käsite ja eheystekniikoita I

- On tärkeää, että tieto ei muutu tai että tietoa ei muuteta, jos sen on tarkoitus pysyä vakiona. Tiedon eheyteen joudutaan kiinnittämään huomiota erityisesti tietoliikenteessä, mutta myös muistissa olevan tiedon varmistaminen tulee joskus kysymykseen.
- Tavalliset tarkistussummat eivät riitä tietoturvatarkasteluissa, koska ne eivät suojaa tiedon tahalliselta muuttamiselta (hyökkääjä voi muuttaa myös tarkistussummaa). Tarvitaan menetelmä, joka paljastaa tahattomat ja tahallaan tehdyt muutokset.
- Usein eheyteen liitetään vielä todennus: tiedon vastaanottaja pystyy varmasti näkemään, kuka tiedon on lähettänyt. Hyvä todennus toteuttaa lisäksi ehdon, ettei vastaanottaja voi itse väärentää sanomaa ja väittää, että se on tullut toiselta osapuolelta.

Tarkastellaan seuraavia menetelmiä, joilla toteutetaan eheys ja todennus:

- salaus,
- tiivistefunktiot ja digitaalinen allekirjoitus,
- MAC-funktiot.

Tiedon salaus eheystekniikkana I

- Jos selvätekstiä sisältävä tieto salataan ja salattua tietoa muutetaan, salauksen purku tuottaa mielivaltaisen bittijonon, joka ei ole selvätekstin esitystä. Näin salauksen purku paljastaa muutoksen ja salaus toimii eheyden takaajana. Lisäksi salaus todentaa lähettäjän, jos salausavain on yhteisesti sovittu, eikä se ole vuotanut ulkopuolisille.
- Voi olla kuitenkin vaikeaa automaattisesti päätellä, onko salauksen purun lopputulos oikeanlaatuista, jos alkuperäinen tieto on esimerkiksi binäärikoodia, röntgenkuvia tms. Tällaisissa tilanteissa tietoon voidaan liittää tarkistussumma (esimerkiksi CRC) ja vasta sitten salata. Toinen mahdollisuus on strukturoida tieto siten, että struktuuri on helppo havaita automaattisesti.

Tiedon salaus eheystekniikkana II

- Salaus ei kuitenkaan estä vastaanottajaa väärentämästä sanomaa, salaamasta sitä ja väittämästä, että se on tullut toiselta. Usein tietokoneverkkosovelluksissa osapuolilla ei ole valmiina yhteisiä salaisia avaimia, vaan niistä pitää sopia aluksi. Tämä hankaloittaa salauksen käyttöä eheyden takaajana.
- Myöskään aina ei tarvita luottamuksellisuutta, vaan tieto voitaisiin lähettää selväkielisenä perille, jos vain se menisi vastaanottajalle ehyenä. Sen tähden on hyvä erottaa luottamuksellisuus ja eheys/todennus toisistaan ja käsitellä niitä eri proseduureilla.

Tiivistefunktiot I

- Tiivistefunktio (engl. cryptographic hash function) on kuvaus h, joka laskee sanomasta M tarkistuskentän $h(M)$.
- M voi yleensä olla vaihtuvamittainen, sen sijaan $h(M)$ on kiinteämittainen. Normaalisti $h(M)$:n pituus on paljon lyhyempi kuin M:n.
- Tiivistefunktion h arvon laskemisessa ei tarvita salaisia avaimia. Funktio on kuitenkin yksisuuntainen eli $h^{-1}(X)$ on vaikea laskea, vaikka h ja X tunnetaan.
- Tiivistefunktioita voidaan käyttää eheyden ja todennuksen yhteydessä. Tyypillisesti sanomasta lasketaan ensin tiivistefunktiolla tiiviste. Sen jälkeen tiiviste allekirjoitetaan digitaalisesti. Lähtevä sanoma koostuu selväkielisestä sanomasta plus allekirjoituksesta.

Tiivistefunktiot II

- Vastaanottaja laskee sanomasta myös tiivisteen ja sen jälkeen verifioi allekirjoituksen. Verifiointi tarkoittaa yleensä sitä, että allekirjoituksesta lasketaan alkuperäinen tiiviste. Nyt voidaan verrata itse laskettua tiivistettä ja allekirjoituksesta saatua tiivistettä. Jos nämä ovat samoja, tieto on tullut eheänä perille ja allekirjoitus vielä vahvistaa lähettäjän. Myöskään vastaanottaja ei voi väärentää allekirjoitusta, joten lähettäjän ei tarvitse pelätä väärennöksiä.
- Tarkemmin kuvattuna allekirjoituksen verifiointi tapahtuu seuraavasti:
  1. Verifioija tarvitsee allekirjoittajan julkisen RSA-avaimen. Koska ei ole mitään paikkaa, josta sen voisi hakea, allekirjoittaja lähettäessään viestinsä liittää pakettiin myös varmenteensa.
  2. Varmenne sisältää allekirjoittajan julkisen RSA-avaimen. Lisäksi varmenteessa on varmenteen myöntäneen varmenneviranomaisen (certification authority) tunnus ja viranomaisen allekirjoitus.

Tiivistefunktiot III

3. Vastaanottaja eli verifioija "purkaa" allekirjoituksen käyttämällä lähettäjän julkista avainta, joka siis löytyy varmenteesta. Hän laskee sanoman tiivisteen ja vertaa purkamaansa arvoa laskemaansa tiivisteehen. Jos molemmat ovat identtiset, verifiointi jatkuu. Muuten allekirjoitus ei ole pätevä.
4. Seuraavaksi on tarkistettava, että varmenteessa oleva julkinen avain on todella aito. Eli että varmenne todella liittää oikein identiteetin ja julkisen avaimen. Ensiksi tarkistetaan, ettei varmennetta ole peruutettu. Tämä tehdään joko hakemalla peruutuslista varmenneviranomaiselta tai käyttämällä erityistä protokollaa, esimerkiksi OCSP:tä (Online Certificate Status Protocol). OCSP:lle annetaan varmenne ja se hoitaa tarkistuksen palauttaen tiedon, onko varmenne ok vai ei. Peruutuslistat voivat olla suuria ja niitä saatetaan päivittää muutaman minuutin välein.

Tiivistefunktiot IV

5. Vielä on tarkistettava varmenneviranomaisen allekirjoitus. Tätä varten tarvitaan viranomaisen julkinen avain, jonka lähettäjä saattaa myös lähettää paketin yhteydessä. Verifioijan tulisi tuntea varmenneviranomainen. Jos näin ei ole, täytyy etsiä toinen viranomainen, joka on varmentanut ensimmäisen ja jonka verifioija tuntee. Tällainen ketju saattaa olla varsin pitkä.

- Mikäli luottamuksellisuutta ei tarvita, salausta yritetään välttää useista syistä:
  - Salausohjelmat ovat hitaita.
  - Piiritason AES- yms. toteutukset ovat melko tehokkaita, mutta näidenkin kustannukset tuntuvat, jos salausta tarvitaan kaikissa verkon solmuissa.
  - Salauslaitteisto on optimoitu suurten datamäärien salaukseen. Jos salattavana on pieni lohko, suurin osa ajasta menee alustukseen.
  - Salausalgoritmi voi olla patentoitu kuten oli RSA:n tapauksessa. Tämä lisää kustannuksia.

Tiivistefunktiot V

- Salausalgoritmen vientiä säädellään (nykyään paljon vähemmän kuin esim. 1980-luvulla).

Tiivistefunktioilta vaadittavat ominaisuudet I

Seuraava luettelo listaa tiivistefunktioilta vaadittavat ominaisuudet:

1. $h(M)$ voidaan laskea minkä pituiselle M tahansa.
2. $h(M)$ on kiinteän pituinen.
3. $h(M)$ on helppo laskea.
4. Jos annetaan y, ei ole helppoa löytää sellaista M:ää, että $h(M) = y$. Eli h on *alkukuvaresistentti* (h has preimage resistance).
5. Jos on annettu y ja M_1, ei ole helppoa löytää sellaista $M_2 \neq M_1$, että $h(M_1) = h(M_2)$. Eli h on *injektiotyyppinen* (h has second preimage resistance).
6. On vaikeaa löytää mitään paria (M, M'), jolle $h(M) = h(M')$. Eli h on *törmäysresistentti* (collision resistance).

Käytännön tiivistefunktio toteutetaan samaan tapaan kuin symmetrinen salaus. Erona on, että lopputulosta ei tarvitse enää purkaa niinkuin salauksessa. Siten teksti jaetaan lohkoihin, lohkoja yhdistellään binäärioperaatioilla, välillä suoritetaan rekisterien sivuttaissiirtoja jne.

Tiivistefunktioita

- Tilanne tiivistefunktioiden kanssa on tällä hetkellä sekava. Kiinalainen tutkimusryhmä kehitti noin 7 vuotta sitten uudenlaisen tekniikan murtaa tiivistefunktioita, ja tämä johti siihen, että perinteellisiä tiivistefunktioita MD5, SHA-1, SHA-2, RIPEMD-160 ei pidetä enää turvallisina.
- Paraikaa ollaan kehittämässä SHA-3:ta, ja sen piti tulla markkinoille viime vuonna.
- MD5 (pituus 128 bittiä) on ehdottomasti vanhentunut, SHA-1:tä (160 b) ei myöskään ole suositeltu enää vähään aikaan. Sen sijaan SHA-2 -versiot (224, 256, 384, 512 bittiä) toiminevat vielä käytännössä jonkin aikaa.

MAC-funktiot I

- MAC-koodi (Message Authentication Code) on tiiviste, joka liitetään sanomaan, jotta vastaanottaja voisi varmistaa alkuperäisen lähettäjän. Tiivisteen laskemisessa käytetään salaista parametria.
- Ensimmäiset MAC-koodit generoitiin salauksen avulla. Tällöin vastaanottajalla ja lähettäjällä täytyi olla sama salainen avain. Salalohkot laskettiin vielä yhteen, joten lopputulos oli salalohkojen summa.
- Toinen tapa muodostaa MAC on käyttää tavallista tiivistefunktiota salaisen avaimen kanssa. Toisin sanoen sanomasta lasketaan tiiviste, mutta laskennassa käytetään myös salaista avainta pelkän sanoman lisäksi. Esimerkiksi MD5:ta ja SHA:ta on käytetty tällä tavoin MAC:in muodostamiseen.
- Sitten on vielä esimerkiksi MAA-algoritmi, joka on ISO-standardi 8731-2. Sen laskenta muistuttaa tiivisteen laskemista salaisen avaimen kera.

MAC-funktiot II

- MAC-menetelmän avulla tiedon eheys voidaan varmistaa, sillä kukaan kuin salaisen avaimen omistajat eivät pysty laskemaan tiivistettä. Toisaalta MAC ei estä vastaanottajaa väärentämästä sanomaa ja MAC arvoa. Siten MAC-arvoa ei voi käyttää, jos halutaan toteuttaa kiistämättömyys. Toisaalta MAC on nopeampi laskea kuin allekirjoitus. Siten protokollissa käytetään usein allekirjoitusta alkuvaiheessa, kun sovitaan avaimista. Sen jälkeen eheys varmistetaan MAC-arvolla.

Vaatimukset avaimille I

Tällä hetkellä vaatimukset avaimille ovat:

- n:n on oltava välillä $1024 - 2048$ bittiä,
- p:n ja q:n tulee olla "lähellä"toisiaan, välillä $10^{75} - 100^{100}$,
- $p-1$:n ja $q-1$:n tulee sisältää suuri alkulukutekijä ja
- $\mathrm{syt}(p-1, q-1)$:n tulee olla pieni.

Tarkastellaan vielä avainten pituuksia. Koska laitteistot kehittyvät koko ajan, täytyy avaimen pituuksia kasvattaa aika ajoin. Seuraavassa taulukossa on sekä symmetrisiltä että epäsymmetrisiltä salausmenetelmiltä vaadittavia avainpituuksia tuleville vuosille. Arviot perustuvat ECRYPT II -suosituksiin vuodelta 2009. Vastaavia suosituksia on NIST:ltä, NSA:lta ja monilta muilta sekä yksittäisiltä tutkijoilta että organisaatioilta. Taulukossa DL tarkoittaa diskreettiin logaritmiiin perustuvia tekniikoita ja EC elliptisiin käyriin perustuvia. Käytämme myös seuraavia lyhenteitä:

Vaatimukset avaimille II

A	Attacks in real-time by individuals
V	Very short-term protection against small organizations
Sh	Short-term protection against medium organizations
Sm	Smallest general-purpose level 2009-2012
Le	Legacy standard level 2009-2020
M	Medium-term protection 2009-2030
Lo	Long-term protection 2009-2040
F	Foreseeable future, against quantum computers
DL	Discrete logarithm
EC	Elliptic curve

Vaatimukset avaimille III

Taso	Suoja	Symm.	Epäsymm.	DL	EC	Tiiviste
1	A	32				
2	V	64	816	128	128	128
3	Sm	72	1008	144	144	144
4	Sh	80	1248	160	160	160
5	Le	96	1776	192	192	192
6	M	112	2432	224	224	224
7	Lo	128	3248	256	256	256
8	F	256	15424	512	512	512