Turvallisuustavoitteet

- **Salaisten** avainten on pysyttävä salaisena.

  Istuntoavaimen pitää olla **tuore**.

  Joko luotettu osapuoli määrää avaimen, toinen osapuoli määrää avaimen tai **molemmat vaikuttavat avaimen arvoon**.

- **Autentikointi** on aina kiinteä osa avaintenjakelua. Sekä julkisten että salaisten avainten pitää olla aitoja. Kenen kanssa yhteinen istuntoavain luodaan?

  Autentikointi voi olla **yksi- tai kaksisuuntainen** (keskinäinen, *unilateral or mutual*).

  Kuinka yhdistetään autentikointi ja avainten jakelu?

- **Jakelun onnistumisesta** voidaan varmistua. (*key confirmation*)

  Implisiittisessä autentikoinnissa tunnistus tapahtuu vasta avainta käytettäessä, **eksplisiittisessä** protokollan aikana.

- **Kiistämättömyys** on tarpeen ainakin, jos yhteydestä laskutetaan.

Avaintenjakelun toimintaympäristö

- Kryptografiset algoritmit oletetaan luotettaviksi.
- Hyökkääjä (*adversary, intruder, opponent, enemy, attacker, eavesdopper, impersonator*) yrittää
  - saada istuntoavaimen selville
  - vaihtaa istuntoavaimen omakseen tai vanhaksi
  - esiintyä väärällä nimellä
  - estää muuten turvallisuustavoitteiden saavuttamisen
  - käyttää avaintenvaihtoprotokollaa hyväkseen muissa hyökkäyksissä.
- Hyökkääjä voi olla myös laillinen protokollaosapuoli tai useampien **salaliitto**.
- Hyökkäys on **passiivinen** tai **aktiivinen**.

  Passiivinen hyökkääjä osaa kuunnella kommunikaatiokanavia.

  Aktiivinen hyökkääjä osaa manipuloida (poistaa, muuttaa, lisätä) sanomia.

- **Oletetaan hyökkääjän olevan laaja salaliitto, joka pystyy vapaasti manipuloimaan liikennettä kaikissa epäluotettavissa kanavissa.**

Avaintenhallinnan kustannukset

- **Yhteyksien määrä** eri osapuolien välillä.
- Lähetettyjen **sanomien määrä** (*round-trips*)
- Kommunikaation **kaistanleveys** eri kanavissa, sanomien bittien määrä
- **Laskennan vaativuus** asiakkaan, palvelimen ja luotetun osapuolen päätelaitteissa
- **On-line** -laskenta
- Satunnaislukujen generointi
- **Luotettu osapuoli** *on-line / off-line*

  (*trusted third party TTP, authentication server, key distribution center KDC, key translation center KTC, certification authority CA*)
- Tarkojen **kellojen** tarve
- Kustannuksiin vaikuttaa merkittävästi, käytetäänkö salaisen vai julkisen avaimen algoritmeja.

Hyökkäykset

- Useimmat hyökkäykset ovat **toistohyökkäyksiä**.

  Toistohyökkäyksessä hyökkääjä nauhoittaa sanoman ja kopioi sen **protokollaan, sanomaan, sanoman osaan tai protokolla-ajoon**, jonne se ei kuulu.

  Toistohyökkäykset estetään sitomalla sanoma sen oikeaan yhteyteen.
- Hyökkäystyyppejä:
  - tunnistusongelma
  - tuoreusongelma (*freshness problem*)
  - *man-in-the-middle*
  - oraakkelihyökkäys (*oracle attack*)
  - lomitushyökkäys (*interleaving attack*) .

Tunnistettavuusongelma

Aidon sanoman erottaminen satunnaisesta

- Mikä tahansa bittijono näyttää salakieleltä. Sillä on jokin (satunnainen) selväkielinen vastine.
- Istuntoavain K_{SES} on satunnainen. Avainta katsomalla ei voi päätellä, onko avain aito vai ei.

  Bob ei pysty erottamaan aitoa sanomaa $E_{AB}(K_{SES})$ väärästä.
- Hyökkäys: Mallory lähettää Bobille satunnaisen bittijonon. Bob luulee saaneensa uuden avaimen. Mallory ei tiedä avainta, mutta pystyy näin häiritsemään Bobia.
- Lisätään selväkieleen vähintään 64 bittiä redundanssia, jotta Bob **tunnistaa** oikeat ja väärät sanomat.

  Lisätään otsikkotietoja tai maaginen numero. (Salauksen oletetaan suojaavan myös sanoman eheyttä.)

  $$\boxed{1. \quad A \longrightarrow B: \quad A, E_{AB}(\text{"\textit{uusi avain}"}, K_{SES})}$$

  Nyt Bob erottaa aidot ja väärät sanomat. Mallory ei pysty luomaan oikean näköistä sanomaa.
- Valitettavasti tunnistettavuus huomioidaan usein vasta toteutuksessa.

Klassinen toistohyökkäys

(*classic replay*)

- Vaikka vastustaja ei pysty väärentämään tai muuttamaan salattu tekstia, hän voi nauhoittaa sanoman.

  $$A \longrightarrow B: \quad A, E_{AB}(K_{SES})$$

  Mallory näkee sanoman ja tallettaa sen. Myöhemmin Alice tai Bob käy huolettomaksi ja paljastaa vanhan avaimen K_{SES}.

  Vanhojen istuntoavaimien oletetaan aina paljastuvan.

  $$M(A) \longrightarrow B: \quad A, E_{AB}(K_{SES}) \qquad \text{Toistohyökkäys!}$$

  Bob luulee, että K_{SES} on hänen ja Alicen välinen uusi avain.
- Bob voi pitää kirjaa vanhoista avaimista/viesteistä, mutta se on raskasta.

Tuoreus

- Sanoma tai avain on **tuore**, jos sitä ei ole kopioitu aiemmista protokolla-ajoista.
- Ratkaisu 1: **aikaleima**

  Liitetään sanomaan lähetysaika. Hyväksytään vain uudet sanomat.

  $$1. \quad A \longrightarrow B : \quad A, E_{AB}(T_A, \ldots, K_{SES})$$

  Ongelma 1: Mikä on riittävän tuore?

  Ongelma 2: Kellojen oltava luotettavia. Hajautetussa järjestelmässä ei ole tarkkaa globaalia aikaa.
- Ratkaisu 2: **nonssi**

  Sanoman osana pitää olla jotain uutta tietoa, jota vanhoissa sanomissa ei ole. N_A on Alicen generoima nonssi.

  $$1. \quad A \longrightarrow B : \quad E_{AB}(T_A, N_A, \ldots, K_{SES})$$

  Bob säilyttää nonssit aikaleiman voimassaoloajan ja tarkistaa, että samaa ei vastaanoteta kahdesti.

  $$\begin{array}{lll} 1. & B \longrightarrow A : & B, N_B \\ 2. & A \longrightarrow B : & E_{AB}(N_B, \ldots, K_{SES}) \end{array}$$

  Bob tietää, että sanoma on tuore, koska se sisältää hänen generoimansa tuoreen nonssin. Ei tarvitse luottaa kelloon.

Oraakkelihyökkäys

- **Oraakkeli** antaa vastauksen vaikeaan ongelmaan selittämättä, miten ratkaisu löytyi.
- Haaste–vaste eräästä protokollasta:

  $$\begin{array}{llll} 1. & A \longrightarrow B : & N_A & \text{haaste} \\ 2. & B \longrightarrow A : & E_{SES}(N_A) & \text{vaste} \end{array}$$

  Bob toimii oraakkelina, joka salaa minkä tahansa Malloryn lähettämän sanoman.

  Bob voi forwardoida autentikoinnin tai käyttää salattuja sanomia muihin tarkoituksiin.

  $$\begin{array}{lll} A \longrightarrow M(B) : & N_A & \text{haaste} \\ M(A) \longrightarrow B : & N_A & \\ B \longrightarrow M(A) : & E_{SES}(N_A) & \text{vaste} \\ M(B) \longrightarrow A : & E_{SES}(N_A) & \end{array}$$

Eksplisiittisyys

- Useimmat oraakkeli- ja lomitushyökkäykset johtuvat sanomien pituuden liiasta optimoinnista.

  Sanomiin lisää redundanttia tietoa!

- **Eksplisiittinen tyypitys**:

  Merkitään kaikkiin allekirjoitettuihin ja salattuihin sanomiin ja sanoman osiin **protokollan nimi, sanoman numero, sanoman osa tai primitiivinen tietotyyppi**.

- **Maksimaalinen informaatio** (*full information principle*) :

  Liitetään kaikkiin sanomiin kaikki protokolla-ajoon liittyvä tieto, josta lähettäjän ja vastaanottajan pitäisi olla yhtä mieltä.

- Esim. muunnettu X.509:

$$
\begin{array}{lll}
1. & A \longrightarrow B & T_A, N_A, E_B(K_{AB}), \\
 & & S_A(\text{“Msg 1 of my protocol”}, \\
 & & T_A, A, B, K_A^+, K_B^+, N_A, K_{AB}) \\
2. & B \longrightarrow A & T_B, N_B, N_A, E'_A(K_{BA}), \\
 & & S_B(\text{“Msg 2 of my protocol”}, \\
 & & T_B, A, B, K_A^+, K_B^+, N_A, N_B, K_{AB}, K_{BA}) \\
3. & A \longrightarrow B & S_A(\text{“Msg 3 of my protocol”}, \\
 & & A, B, K_A^+, K_B^+, N_A, N_B, K_{AB}, K_{BA})
\end{array}
$$

Allekirjoitetut sertifikaatit

- **Julkisen avaimen protokollissa keskeistä on julkisten avainten aitous.**

- Alice voi välittää Bobille luotetun osapuolen allekirjoittaman sertifikaatin, joka todistaa hänen julkisen avaimensa aitouden.

$$
\begin{array}{lll}
1. & A \longrightarrow T: & A, B \\
2. & T \longrightarrow A: & T_T, B, K_B^+, S_T(T_T, B, K_B^+) \\
 & & T_T, A, K_A^+, S_T(T_T, A, K_A^+) \\
3. & A \longrightarrow B: & T_T, A, K_A^+, S_T(T_T, A, K_A^+) \\
 & & E_B(T_A, B, K_{SES}, S_A(T_A, B, K_{SES}))
\end{array}
$$

Sertifikaatit voivat olla pitkäikäisiä, jolloin niitä ei tarvitse aina välittää uudestaan.

$$S_T(T_T, L, A, K_A)$$

A = käyttäjä

K_A = julkinen avain

T_T, L = luontiaika ja voimassaoloaika

Uusi istuntoavain on helppo luoda vanhojen sertifikaattien ollessa voimassa.

X.509-sertifikaatti

- CCITT/ISO 1988, ITU 1993
- Standardi määrittelee
  - autentikoinnissa käytettävän sertifikaatin muodon
  - useiden sertifiointiauktoriteettien yhteistoiminnan
  - kolme autentikointiprotokollaa
  - sertifikaattien peruuttamisen.
- Standardi suosittelee RSA:ta, mutta sallii eri salaus- ja allekirjoitusalgoritmit.
- Sertifikaatin muoto

$$CA << A >> = S_{CA}(SN, CA, T_{CA}, L, A, K_A)$$

- Sertifikaatin antaa **sertifiointiauktoriteetti CA** (*certification authority*) .
- Sertifikaatilla käyttäjä todistaa toiselle samaan CA:han luottavalle käyttäjälle julkisen avaimensa aitouden.
- Sertifikaatit voivat olla julkisessa palvelimessa kaikkien saatavilla.

X.509-CA-hierarkia

- CA:t muodostavat hierarkian, jossa jokainen sertifioija jakaa puussa ylä- ja alapuolella olevien naapureiden sertifikaatteja.

  Alice voi todistaa Bobille avaimensa aitouden, jos Alicen ja Bobin välillä on **sertifikaattipolku**.

$$CA_A <<CA_1>> CA_1 << CA_2 >>$$
$$CA_2 << CA_3 >> \ldots CA_N << B >>$$

- Alice luottaa CA_A:han ja transitiivisesti tämän yläpuolella oleviin auktoriteetteihin.

  Jokin auktoriteetti CA' on puussa sekä Alice attä Bobin yläpuolella.

  Alice luottaa CA':un ja kaikkiin tämän alapuolella puussa oleviin auktoriteetteihin.

- **Huom! Yleisessä tapauksessa luottamus ei ole transitiivista.** Kuinka syviä hierarkioita sallitaan?

X.509-sertifikaatin peruutus

(*revocation of certificates*)

- Sertifikaatti peruutetaan, kun aikaleima on vielä käypä ja
  - yksityinen avain on paljastunut
  - käyttäjään tai sertifiointiauktoriteettiin ei enää luoteta
  - CA ei enää halua sertifioida käyttäjää.
- Lista peruutetuista sertifikaateista on kaikilla CA:lla ja käyttäjillä muistissa.

  Listassa on sertifikaatin numero.
- Käyttäjä päivittää listaansa haluamassaan tahdissa.
- Peruutuslistojen ylläpito on kallista. Vaihtoehtona lyhyet voimassaoloajat ja sertifikaattien säännöllinen päivitys.

X.509-autentikointi

- Autentikoinnin vaiheet:
  1. Osapuolet hankkivat sertifikaattipolun toisiinsa.
  2. Autentikointi ja avaintenvaihto tapahtuu julkisia avaimia käyttäen.
- Yleensä Alicella ja Bobilla on muistissa sertifikaattipolku puussa itsestä ylöspäin.

  Kumpikin lähettää oman polkunsa toiselle tai sertifikaatit haetaan palvelimesta.
- **Yksisuuntainen X.509-autentikointi**

$$\boxed{\begin{array}{lll} 1. & A \longrightarrow B: & T_A, N_A, B, E_B(K_{AB}), \\ & & S_A(T_A, N_A, B, E_B(K_{AB})) \end{array}}$$

  - A lähetti sanoman ja se on tarkoitettu B:lle.
  - Sanomaa ei ole muutettu tai toistettu.
  - Vaatii globaalin kellon. Aikaleima T_A sisältää voimassaolon alkuajan ja loppuajan.
  - B voi muistaa nonssin kunnes aikaleima vanhenee, jotta sanomaa ei voi toistaa sinä aikana.

- **Heikkous 1**:

  Salatun X.509-istuntoavaimen voi kopioida sitä tuntematta.

$$
\begin{aligned}
A \longrightarrow B: \quad & T_A, N_A, B, E_B(K_{AB}), \\
& S_A(T_A, N_A, B, E_B(K_{AB})) \\
M \longrightarrow B: \quad & T_M, N_M, B, E_B(K_{AB}), \\
& S_M(T_M, N_M, B, E_B(K_{AB}))
\end{aligned}
$$

  Kun istunnoissa on sama avain, voi hyökkääjä kopioida sanomia istuntojen välillä.

  Esim: A lähettää B:lle ja M:lle saman tenttikysymyksen. M kopioi B:n vastauksen näkemättä sitä.

$$
\begin{aligned}
A \longrightarrow B: \quad & \text{"Mikä on nonssi?"} \\
B \longrightarrow A: \quad & E_{AB}(\text{"Nonssi on satunnaisluku."}) \\
M \longrightarrow A: \quad & E_{AB}(\text{"Nonssi on satunnaisluku."})
\end{aligned}
$$

- **Kaksisuuntainen X.509-autentikointi**

$$
\boxed{
\begin{aligned}
1.\quad A \longrightarrow B: \quad & T_A, N_A, B, E_B(K_{AB}) \\
& S_A(T_A, N_A, B, E_B(K_{AB})) \\
2.\quad B \longrightarrow A: \quad & T_B, N_B, A, N_A, E_A(K_{BA}) \\
& S_B(T_B, N_B, A, N_A, E_A(K_{BA}))
\end{aligned}
}
$$

  – B vastasi ja vastaus on tarkoittu A:lle.
  – Sanomaa ei ole muutettu tai toistettu.
  – T_B on tarpeeton. Sanoman 1 tuoreus riippuu edelleen globaalista kellosta.

- Istuntoavaimia on kaksi. **Liikenne eri suuntiin salataan eri avaimella.**

  Hyöty: dataa voi salata etukäteen puskuriin tai lähettää samalla avaimella salattuna useammalle. Toistohyökkäys vaikeutuu.

  Huom: Avaimen voi yhä kopioida.

  (Avainmateriaalit voidaan myös yhdistää yhdeksi avaimeksi.)