Tik-79.159 Tiedon salaus ja suojaus	
Tentin 12.5.1997 ratkaisut	Tuomas Aura

1. (a) Autentikointilogiikat ovat autentikointi- ja avaimenvaihtoprotokollien osapuolien tietämystä tai uskomuksia kuvaavia tietämyslogiikkoja (formaalia kieliä päättelysääntöineen ja aksioomineen). Logiikkaa käytetään kryptografisten protokollien ominaisuuksien todistamiseen.

   (b) Salauksessa konfuusion tehtävä on häivyttää havaittavat yhteydet selväkielisen ja salakielisen tekstin välillä ja estää selväkielisen sanoman redundanssin havaitseminen salakielisessä. Lohkosalauksessa konfuusio saadaan useimmiten aikaan substituutiolla.

   (c) Subliminaalinen kanava tarkoittaa viestintätapaa, jota käytettäessä ulkopuolinen taho ei voi edes tietää, että viestejä lähetetään. Sanomat siis piilotetaan.

   (d) CBC (*Cipher Block Chaining*) on lohkosalausalgoritmin käyttötapa, jossa selväkielinen lohko xorrataan edellisen salakielisen lohkon kanssa ennen salaamista algoritmilla. Ensimmäinen lohko xorrataan satunnaisesti valitun alustuvektorin kanssa.

   $$C_0 = IV; \quad C_i = E_K(P_i \oplus C_{i-1}), i = 1 \ldots$$

   Hyöty on, että sama sanoma kaksi kertaa salattuna näyttää erilaiselta eikä sanomassa toistuvia osia voida havaita. (Kuva olis kiva ylläri.)

   (e) Sertifikaatti on luotetun tahon allekirjoitettama todistus jostain asiasta, esim. avaimen kuulumisesta tietylle henkilölle.

   (f) Nollatietotodistuksessa todistaja todistaa verifioijalle tuntevansa salaisuuden paljastamatta itse salaisuutta tai muuta informaatiota. Verifoija ei edes voi jälkikäteen vakuuttaa muita todistuksen aitoudesta, vaikka olisi siitä itse vakuuttunut.

2. (a) *One-time-pad* -salauksen avain on yhtä pitkä kuin sanoma joten salakielestä saadaan mikä tahansa selväkieli avaamalla se sopivalla avaimella. Niinpä salakieli ei ilman avainta sisällä mitään tietoa selväkielestä, eikä *brute-force* -hyökkäyskään ole mahdollinen. Koska avainta käytetään vain kerran, ei yhden sanoman tai avaimen tunteminen myöskään auta muiden murtamisessa.

   (b) Hybridisalauksessa salataan varsinainen sanoma symmetrisellä menetelmällä käyttäen satunnaista istuntoavainta ja istuntoavain salataan vastaanottajan julkisella avaimella. *Forward-search* -hyökkäyksessä julkisen avaimen salausta vastaan hyökkääjä salaa mahdollisena pitämiään selväkielisiä sanomia julkisella avaimella, kunnes salakielet täsmäävät. Hyökkäys onnistuu, jos mahdollisia selväkielisanomia on vähän. Hybridisalauksessa mahdollisia selväkieliä (istuntoavaimia) on paljon, joten oikeaa ei löydy tällä tavoin kokeilemalla. (Schneier ei – kumma kyllä – mainitse koko hyökkäystä. Luennolla ja laskuharjoituksissa siitä on puhuttu.)

   (c) Syntymäpäivähyökkäys: kun tietokannassa on $\sqrt{2^{128}} = 2^{64}$ hajautusarvoa, löytyy kaksi samaa arvoa todennäköisyydellä, joka on suuruusluokkaa 50%. Niinpä hajautusfunktiolle voidaan löytää törmäyksiä, eikä funktio enää ole vahvasti törmäyksetön. Siksi se ei sovellu allekirjoituksiin. (Yksisuuntaisuus ja heikko törmäyksettömyys säilyvät eli halutulle hajautusarvolle ei vieläkään ole helppo löytää toista sanomaa.)

3. (a) $p = 41$, $q = 47$. $\phi(n) = (41-1) \cdot (47-1) = 1840$. Julkinen eksponentti $d = 3$. Yksityinen eksponentti $e = d^{-1} \bmod \phi(n) = 3^{-1} \bmod 1840$. Koska $1840 =$

$613 \cdot 3 + 1$, saadaan $1 = -613 \cdot 3 + 1840$ eli $e = -613 \bmod 1840 = 1227$. Yksityinen avain on $(1227, 1927)$.

(b) $1732^3 \bmod 1927 = 1412 \cdot 1832 \bmod 1927 = 221$. Allekirjoitus on siis ok. Täytyy muistaa, että luvut ovat tässä esimerkissä aivan liian lyhyitä ja oikeasti RSA-allekirjoitettavaan lukuun sisällytettäisiin sormenjäljen lisäksi muuta redundanttia dataa (ks. esim. PKCS).

4. Laaditaan LaGrange-(3,5)-kynnysrakenne, jossa laskutoimitukset tehdään modulo 65537 (alkuluku ja suurempi kuin suurin mahdollinen salaisuus). Arvotaan toisen asteen polynomin kertoimet väliltä $0 \ldots 65536$.

$$P(x) = (4212 \cdot x^2 + 11834 \cdot x + 1997) \bmod 65537.$$

Lasketaan viisi varjoa:

$P(1) = 4212 \cdot 1 + 11834 \cdot 1 + 1997 \bmod 65537 = 18043.$
$P(2) = 4212 \cdot 4 + 11834 \cdot 2 + 1997 \bmod 65537 = 42513.$
$P(3) = 4212 \cdot 9 + 11834 \cdot 3 + 1997 \bmod 65537 = 9870.$
$P(4) = 4212 \cdot 16 + 11834 \cdot 4 + 1997 \bmod 65537 = 51188.$
$P(5) = 4212 \cdot 25 + 11834 \cdot 5 + 1997 \bmod 65537 = 35393.$

Varjot ovat siis (1,18043), (2,42513), (3,9870), (4,5118), (5,3539). Varjojen lisäksi pitää tietää myös moduli 65537.

LaGrange-kynnysrakennetta laatiessa tulee muistaa seuraavat asiat: modulin (jos sellaista käytetään) pitää olla suurempi kuin suurin mahdollinen salaisuus. Siis jos tehtävässä valitaan moduli luokkaa ~ 2000, pitää perustella, ettei salaisuus millään voi olla tuota suurempi. Polynomin kertoimet pitää valita satunnaisesti (modulia n käytettäessä väliltä $0 \ldots n-1$), joten 3 ja 10 eivät ole uskottava valinta, ellei erikseen kerrota miten ne on arvottu. x:n arvot 1,2,3,4 ja 5 kelpaavat hyvin. Ei ole mitään syytä valita satunnaisia suuria lukuja.

Salaisuus selviää kolmesta varjosta muodostamalla lineaarinen yhtälöryhmä ja ratkaiselmalla polynomin kertoimet. Kun polynomi on selvillä, salaisuus on $P(0)$. Varjojen lisääminen tapahtuu laskemalla polynomilta uusia pisteitä.

Jaskalle ja Reiskalle voidaan antaa oikeus yhdessä ratkaista salaisuus esimerkiksi antamalle heille erillinen (2,2)-kynnnysrakenne (Jaskalle satunnaisluku r ja Reiskalle $r \oplus 1997$), tai luomalla (6,10)-kynnysrakenne, josta Jaskalle ja Reiskalle annetaan 3 varjoa ja muille 2.

On huomattava, että Jaskalle ei voida antaa kahta (5,3)-rakenteen varjoa, koska hän pystyisi tuolloin avaamaan salaisuuden kenen tahansa muunkin kuin Reiskan kanssa. Varjon puolittaminen ei myöskään onnistu antamalle toiselle x ja toiselle $P(x)$, sillä samaa x:n arvoa ei käytetä kahta kertaa ja x:n epävarmuus on siten pienempi kuin $P(X)$:n.

5. (a) Lisätään sertifikaatteihin aikaleimat: $T_T, B, K_B, S_T(T_T, B, K_B)$ ja $T_T, A, K_A, S_T(T_T, A, K_A)$.

(b) Normaalin protokolla-ajon jälkeen Bob jatkaa:

$$\begin{array}{ll} B \to T: & B, C \\ T \to B: & C, K_C, S_T(C, K_C),\ B, K_B, S_T(B, K_B) \\ B(A) \to C: & E_C(T_A, K_{SES}, S_A(T_A, K_{SES})), \\ & \quad C, K_C, S_T(C, K_C),\ A, K_A, S_T(A, K_A) \end{array}$$

Eli kun Alice ottaa yhteyttä Bobiin, voi Bob forwardoida autentikoinnin kolmannelle osapuolelle ja esiintyä tälle Alicena.

(c) Hyökkäys onnistui, koska Alicen allekirjoittamaa sanomaa ei oltu sidottu tiettyyn vastaanottajaan, vaan Bob saattoi lähettää sen edelleen C:lle. Lisätään sanomaan vastaanottajan nimi: $E_B(T_A, B, K_{SES}, S_A(T_A, B, K_{SES}))$.

(d) Kaikki tieto, josta sanoman lähettäjän ja vastaanottajan pitäisi olla yhtä mieltä, kannattaa sisällyttää eksplisiittisesti sanomaan. Protokollaa ei saa optimoida poistamalla liikaa redundanssia sanomista.

(e) B tunnistaa A:n eksplisiittisesti, koska vastaanottaa B:ltä sanoman, jonka aikaleima osoittaa tuoreeksi. B ei lähetä yhtään sanomaa, joten A voi olla varma B:n osallistumisesta vasta salatun liikenteen alettua. Niinpä A tunnistaa B:n implisiittisesti.

6. Unix-salasanojen toteutusyksityiskohtien lisäksi olisi hyvä ymmärtää seuraavat periaatteet:

- Salasanoista lasketaan yksisuuntaisella funktiolla sormenjälki, joka talletetaan julkiseen tiedostoon. Salakirjoitusta ei käytetä eikä salausavainta tarvitse säilyttää.
- Salasanoja voidaan murtaa *brute-force* -hyökkäyksellä ja arvaamalla sanakirjan avulla todennäköisiä salasanoja. Koska hyökkäys tapahtuu vertaamalla arvauksista laskettuja ja julkisesta tiedostosta haettuja yksisuuntaisen funktion arvoja, voidaan se toteuttaa *off-line* ja kokeiltavia sormenjälkiä voidaan laskea etukäteen esim. CD-levyille.
- Tehokkaimmillaan verrataan suurta määrää salasanatietostoista haettuja sormenjälkiä suureen määrään arvauksista laskettuja sormenjälkiä, jolloin syntymäpäiväparadoksin takia samoja arvoja löytyy verrattain nopeasti. Yksittäisen salasanan murtaminen on paljon hitaampaa.
- Syntymäpäiväilmiön torjumiseksi salasanaan lisätään ennen yksisuuntaisen funktion laskemista suola eli käyttäjäkohtainen pseudosatunnaisluku. Suolan ansiosta ei kaikkien käyttäjien salasanoja voi kokeilla samanaikaisesti.
- Käytettävän yksisuuntaisen funktion pitää olla kohtuullisen hidas laskettava, sillä salasanan syöttävä käyttäjä kyllä kestää odottaa muutaman sekuntin murto-osan viiveen, mutta valtavaa salasanamäärää läpi käyvä hyökkääjä kärsii viiveestä. Unixissa tämä on toteutettu iteroimalla DES-funktiota 25 kertaa ja muuttamalla sitä epästandardiksi niin, että valmiita rautatoteutuksia ei ole mahdollista käyttää.
- Nykyisin olisi mahdollista lisätä salasanan pituutta (rajoittamattomasti) ja tehdä suolasta todella käyttäjäkohtainen (Unixissa 12 bittiä), jos yksisuuntaisena funktiona käytettäisiin esim. SHA:ta (mielellään 100 kertaa iteroituna).
- Suurin uhka järjestelmälle lienee kuitenkin, että salasanat usein siirretään salaamattomana verkossa.