8. poglavje

Upravljanje ključev

- **Distribucija ključev**
  (Blomova shema, Diffie-Hellmanova shema)
- **Certifikati**
  (avtentikacijska drevesa, certifikatna agencija, infrastruktura javnih ključev, proces certifikacije, modeli zaupanja)
- **Uskladitev ključev**
  (Kerberos, Diffie-Hellmanova shema, MTI protokoli, Giraultova shema)
- **Internetne aplikacije**
  (Internet, IPsec: Virtual Private Networks, Secure Sockets Layer, varna e-pošta)

Vprašanja

- Od kje dobimo ključe?
- Zakaj zaupamo ključem?
- Kako vemo čigav ključ imamo?
- Kako omejiti uporabo ključev?
- Kaj se zgodi, če je kompromitiran (izgubljen) zasebni ali tajni ključ? Kdo je odgovoren?
- Kako preklicati ključ?
- Kako lahko obnovimo ključ?
- Kako omogočimo servis preprečitve zanikanja?

Ta vprašanja veljajo tako za simetrične (tajne) ključe kakor tudi za javne in zasebne ključe.

Upravljanje ključev je množica tehnik in postopkov, ki podpirajo dogovor in vzdrževanje relacij ključev med pooblaščenimi strankami/sogovorniki.

Infrastruktura javnih ključev (PKI): podporni servisi (tehnološki, pravni, komercialni, itd.), ki so potrebni, da lahko tehnologijo javnih ključev uporabimo za večje projekte.

Sistemi z javnimi ključi imajo prednost pred sistemi s tajnimi ključi, saj za izmenjavo tajnih ključev ne potrebujejo varnega kanala.

Večina sistemov z javnimi ključi (npr. RSA) je tudi do 100-krat počasnejša od simetričnih sistemov (npr. DES). Zato v praksi uporabljamo za šifriranje *daljših* besedil simetrične sisteme.

Obravnavali bomo več različnih protokolov za tajne ključe. Razlikovali bomo med *distribucijo ključev* in *uskladitvijo ključev*.

Sistem distribucije ključev je mehanizem, kjer na začetni stopnji verodostojna agencija generira in distribuira tajne podatke uporabnikom tako, da lahko vsak par uporabnikov kasneje izračuna ključ, ki je nepoznan ostalim.

Uskladitev ključev označuje protokol, kjer dva ali več uporabnikov sestavjo skupen tajni ključ, s komunikacijo po javnem kanalu. Vrednost ključa je določena s funkcijo vhodnih podatkov.

Obstaja potreba po zaščiti pred potencialnimi nasprotniki, tako pasivnimi kot tudi aktivnimi.

Pasivni sovražnik je osredotočen na prisluškovanje sporočilom, ki se pretakajo po kanalu.
Več nevšečnosti nam lahko naredi *aktivni* sovražnik:

- spreminjanje sporočil,
- shranjevanje sporočil za kasnejšo uporabo,
- maskiranje v uporabnika omrežja.

Cilj *aktivnega* sovražnika uporabnikov U in V je lahko:

- prelisičiti U in V tako, da sprejmeta neveljaven ključ kot veljaven,
- prepričati U in V, da sta si izmenjala ključ, čeprav si ga v resnici nista.

Center zaupanja

V omrežju, ki ni varno, se v nekaterih shemah pojavi agencija, ki je odgovorna za

- potrjevanje identitete,
- izbiro in prenos ključev
- itd.

Rekli ji bomo **center zaupanja** ali **verodostojna agencija** (angl. Trusted Authority – TA ali Trusted Third Party – TTP). Uporabljali bomo oznako **TA**.

Distribucija ključev

- "Point-to-point" distribucija po varnem kanalu:
  - zaupni kurir,
  - enkratna registracija uporabnikov,
  - prenos po telefonu.
- Neposreden dostop do overjene javne datoteke:
  - avtentična drevesa,
  - digitalno podpisana datoteka.
- Uporaba "on-line" zaupnih strežnikov,
- "Off-line" certifikatna agencija (CA).

Point-to-point

Predpostavimo, da imamo

– omrežje z n uporabniki,
– agencija TA generira in preda enolično določen ključ vsakemu paru uporabnikov omrežja.

Če imamo varen kanal med TA in vsakim uporabnikom omrežja, potem dobi vsak posameznik $n-1$ ključev, zahtevnost problema pa je vsaj $\mathcal{O}(n^2)$.

Ta rešitev ni praktična celo za relativno majhne n.

Želimo si boljšo rešitev, npr. z zahtevnostjo $\mathcal{O}(1)$.

Blomova shema

Naj bo javno p praštevilo večje od danega $n \in \mathbb{N}$ in naj bo $k \in \mathbb{N}$ za katerega velja $k \leq n-2$.

TA pošlje po varnem kanalu $k+1$ elementov $\mathbb{Z}_p$ vsaki osebi in nato si lahko vsak par $\{U, V\}$ izračuna svoj ključ $K_{U,V} = K_{V,U}$.

Število k je velikost največje koalicije, proti kateri bo shema še vedno varna.

Paul R. Halmos

"...the source of all great mathematics is the special case, the concrete example. It is frequent in mathematics that every instance of a concept of seemingly great generality is in essence the same as a small and concrete special case."

I Want to be a Mathematician, Washington: MAA Spectrum, 1985

???

" Sometimes a research is a lot of hard work in looking for the easy way."

David Hilbert (-1900)

"The art of doing mathematics consists in finding that special case which contains all the germs of generality."

Najprej opišimo shemo v primeru, ko je $k = 1$.

- *Izberemo javno praštevilo p.*
- *TA izbere tri naključne elemente $a, b, c \in \mathbb{Z}_p$ (ne nujno različne) in oblikuje polinom*
$$f(x, y) = a + b(x + y) + c\,xy \mod p.$$
- *Za vsakega uporabnika U izbere TA javni $r_U \in \mathbb{Z}_p$, tako da so le-ti medseboj različni.*

- *Za vsakega uporabnika U izračuna TA polinom*
$$g_U(x) = f(x, r_U) \bmod p$$
in mu ga pošlje po varnem kanalu.

Opozorimo, da je $g_U(x)$ linearen polinom, tako da ga lahko zapišemo v naslednji obliki

$$g_U(x) = a_U + b_U x,$$

kjer je

$$a_U = a + br_U \bmod p \quad \text{in} \quad b_U = b + cr_U \bmod p.$$

- *Za medsebojno komunikacijo osebi U in V uporabita ključ*
$$\begin{aligned} K_{U,V} = K_{V,U} &= f(r_U, r_V) \\ &= a + b(r_U + r_V) + c\,r_U r_V \bmod p. \end{aligned}$$

Uporabnika U in V izračunata svoja ključa $K_{U,V}$ in $K_{U,V}$ zaporedoma s

$$f(r_U, r_V) = g_U(r_V) \quad \text{in} \quad f(r_U, r_V) = g_V(r_U).$$

Izrek 1. *Blomova shema za $k = 1$ je brezpogojno varna pred posameznimi uporabniki.*

Dokaz: Recimo, da želi uporabnik W izračunati ključ

$$K_{U,V} = a + b(r_U + r_V) + c\,r_U r_V \bmod p.$$

Vrednosti r_U in r_V so javne, a, b in c pa ne.
Oseba W pozna vrednosti

$$a_W = a + br_W \bmod p \quad \text{in} \quad b_W = b + cr_W \bmod p,$$

ker sta to koeficienta polinoma $g_W(x)$, ki ju je dobila od agencije TA.

Pokažimo, da je informacija, poznana osebi W, konsistentna s poljubno vrednostjo $\ell \in \mathbb{Z}_p$ za ključ $K_{U,V}$, tj. W ne more izločiti nobene vrednosti za $K_{U,V}$.

Poglejmo si naslednjo matrično enačbo v $(\mathbb{Z}_p)$:

$$\begin{pmatrix} 1 & r_U + r_V & r_U r_V \\ 1 & r_W & 0 \\ 0 & 1 & r_W \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} \ell \\ a_W \\ b_W \end{pmatrix}.$$

Prva enačba vsebuje hipotezo, da je $K_{U,V} = \ell$, drugi dve enačbi pa sledita iz definicije števil a_W in b_W.

Determinanta zgornje matrike je

$$r_W^2 + r_U r_V - (r_U + r_V) r_W = (r_W - r_U)(r_W - r_V).$$

Iz $r_W \neq r_U$ in $r_W \neq r_V$ sledi, da je determinanta različna od nič in zato ima zgornji sistem enolično rešitev za a, b in c.

Koalicija uporabnikov $\{W, X\}$ pa ima štiri enačbe ter tri neznanke in od tod zlahka izračuna a, b in c ter končno še polinom $f(x, y)$, s katerim dobi vsak ključ. ■

Posplošitev

Za splošno shemo (tj. shemo, ki je varna pred koalicijo velikosti k) je potrebna ena sama sprememba. Pri drugem koraku TA uporablja polinom $f(x, y)$ naslednje oblike

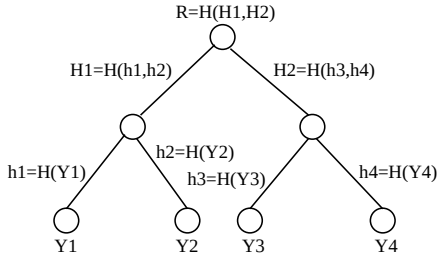
$$f(x, y) = \sum_{i=0}^{k} \sum_{j=0}^{k} a_{ij}\, x^i y^j \mod p,$$

kjer je $a_{ij} \in \mathbb{Z}_p$ za $0 \leq i, j \leq k$ in $a_{ij} = a_{ji}$ za vsak i, j. Ostali del protokola se ne spremeni.

Avtentična drevesa

- Merkle, 1979.
- metoda za hranjenje javno dostopnih in preverljivo overjenih podatkov
- Uporaba:
  - avtentičnost velike datoteke javnih ključev,
  - servis časovnih oznak (Timestamping).

Primer: H je zgoščevalna funkcija brez trčenj.



Vzdržujemo avtentičnost korenske vrednosti R (npr. s podpisom agencije TA).

Za avtenticiranje javne vrednosti Y_2:

- sledi (natanko določeno) pot od Y_2 do korena,
- pridobi vrednosti h_1, H_2, R,
- preveri avtentičnost R,
- preveri $R = H(H(h_1, H(Y_2)), H_2)$.

Če ima drevo n javnih vrednosti, je dolžina avtenticiranja kvečjemu $\lceil \log_2 n \rceil$.

Slaba stran: dodajanje in brisanje javnih vrednosti je lahko precej zamudna.

Diffie-Hellmanova distribucija ključev

Zaradi enostavnosti bomo delali v obsegu $\mathbb{Z}_p$, kjer je p praštevilo in α generator grupe $\mathbb{Z}_p^*$.

Naj bo $\mathrm{ID}(U)$ oznaka za določeno informacijo, ki enolično identificira osebo U (npr. ime, e-pošta, telefonska številka itd).

Vsak uporabnik si izbere tajni/zasebni $a_U \in \{0, 1, \ldots, p-2\}$, in naj bo

$$b_U = \alpha^{a_U} \mod p.$$

Agancija TA si izbere shemo za digitalni podpis z javnim algoritmom za preverjanje podpisov $\mathrm{ver}_{\mathrm{TA}}$ in tajnim algoritmom za podpisovanje $\mathrm{sig}_{\mathrm{TA}}$.

Nazadnje privzemimo še, da so vse informacije zgoščene z javno zgoščevalno funkcijo, preden jih podpišemo, vendar pa zaradi estetskih razlogov ne bomo omenjali zgoščevalne funkcije pri opisu protokolov.

Za osebo U bo agancija TA izdala naslednji **certifikat**:

$$C(U) = (\,\mathrm{ID}(U),\, b_U, \mathrm{sig}_{\mathrm{TA}}(\mathrm{ID}(U),\, b_U)\,)$$

(TA ne potrebuje zasebne vrednosti a_U).

- *Izberemo javno praštevilo p in javen primitivni element $\alpha \in \mathbb{Z}_p^*$.*
- *Oseba V izračuna*
$$K_{U,V} = \alpha^{a_U a_V} \bmod p \;=\; b_U^{a_V} \bmod p,$$
z uporabo javne vrednosti b_U iz certifikata osebe U in s svojo zasebno vrednostjo a_V.
- *Oseba U izračuna*
$$K_{U,V} = \alpha^{a_U a_V} \bmod p \;=\; b_V^{a_U} \bmod p,$$
z uporabo javne vrednosti b_V iz certifikata osebe V in s svojo zasebno vrednostjo a_U.

Podpis agencije TA preprečuje osebi W, da spreminja certifikate, torej je dovolj preprečiti pasivne napade.

Ali lahko oseba W izračuna $K_{U,V}$, če je $W \neq U, V$, tj, če poznamo $\alpha^{a_U} \bmod p$ in $\alpha^{a_V} \bmod p$ ne pa tudi a_U ali a_V, ali je mogoče izračunati $\alpha^{a_U a_V} \bmod p$?

To bomo imenovali **Diffie-Hellmanov** problem.

Očitno je **Diffie-Hellmanova distribucija ključev** varna natanko tedaj, ko je varen **Diffie-Hellmanov** problem.

Izrek 2. *Razbitje ElGamalovega kriptosistema je ekvivalentno reševanju Diffie-Hellmanovega problema.*

Dokaz: Spomnimo se, kako potekata ElGamalovo šifriranje in odšifriranje. Ključ je $K = (p, \alpha, a, \beta)$, kjer $\beta = \alpha^a \bmod p$ (a je tajni in p, α in β so javni). Za tajno naključno število $k \in \mathbb{Z}_{p-1}$ je
$$e_K(x, k) = (y_1, y_2),$$
kjer $y_1 = \alpha^k \bmod p$ in $y_2 = x\beta^k \bmod p$.

Za $y_1, y_2 \in \mathbb{Z}_p^*$ je $d_K(y_1, y_2) = y_2(y_1^a)^{-1} \bmod p$.

Predpostavimo, da imamo algoritem A, ki reši Diffie-Hellmanov problem in podano ElGamalovo šifriranje (y_1, y_2). Z uporabo algoritma A na podatkih p, α, y_1 in β dobimo vrednost
$$A(p, \alpha, y_1, \beta) = A(p, \alpha, \alpha^k, \alpha^a) =$$
$$= \alpha^{ka} \bmod p = \beta^k \bmod p.$$
Potem odšifriranje (y_1, y_2) lahko enostavno izračunamo:
$$x = y_2(\beta^k)^{-1} \bmod p.$$

Predpostavimo, da imamo še algoritem B, ki izvrši ElGamalovo odšifriranje. Torej B vzame podatke p, α, β, y_1 in y_2 in izračuna
$$x = y_2(y_1^{\log_\alpha \beta})^{-1} \bmod p.$$
Naj bodo p, α, β in γ podatki Diffie-Hellmanovega problema. Torej je $\beta = \alpha^b$ in $\gamma = \alpha^c$ za neka $b, c \in \mathbb{N}$, ki nista poznana, pa vendar lahko izračunamo
$$(B(p, \alpha, \beta, \gamma, 1))^{-1} = (1(\gamma^{\log_\alpha \beta})^{-1})^{-1} \bmod p =$$
$$= \gamma^{\log_\alpha \beta} \bmod p = \alpha^{c \cdot b} \bmod p,$$
torej DH-ključ, kar smo tudi želeli. ■

Certifikati

Certifikatna agencija (CA) izda certifikat $\boldsymbol{C(U)}$, ki poveže uporabnika U z njegovim javnim ključem.

Sestavljen je iz:

- **podatkovnega dela $\boldsymbol{D(U)}$**:
uporabnikova identifikacija, njegov javni ključ in druge informacije kot npr. veljavnost,
- **podpisanega dela $\boldsymbol{\mathrm{sig}_{\mathrm{CA}}(D(U))}$**:
CA-jev podpis podatkovnega dela.

B pridobi avtentično kopijo A-jevega javnega ključa na naslednji način:

- pridobi avtentično kopijo javnega ključa CA (npr. dobljenega z brskalnikom ali operacijskim sistemom),
- pridobi $C(U)$ (preko nezavarovanega kanala),
- preveri podpis $\mathrm{sig}_{\mathrm{CA}}(D(U))$.

Opombe: 1. CA ni potrebno zaupati uporabniških zasebnih ključev.

2. CA moramo zaupati, da ne bo izdajala ponarejenih certifikatov.

Infrastruktura javnih ključev (PKI)

Nekatere komponente:

- format certifikata,
- proces certificiranja,
- razdeljenanje certifikatov,
- modeli zaupanja,
- preklic certifikatov,
- politika certificiranja: podrobnosti o namenu in obsegu uporabe določenega certifikata.
- Izjava o prakticiranju certificiranja (CPS) (postopki in politike CA).

Format certifikata: X.509 Ver.3

- X.509 originalno predlagan za podporo X.500, ki omogoča servis imenikov na velikih računalniških mrežah.
- Ver. 1 izide leta '88;
  Ver. 2 leta '93;
  Ver. 3 pa leta '97.
- Najnovejši PKI produkti uporabljajo Ver.3.
- Dopušča precejšnjo fleksibilnost.

Podatkovna polja zajemajo:

- verzijo številke certifikata,
- certifikatovo serijsko številko,
- CA-jev podpisni algoritem ID,
- CA-jevo ime v X.500,
- rok veljave,
- uporabnikovo X.500 ime,
- uporabnikova informacija o javnem ključu,
  - algoritmov ID, vrednost javnega ključa,
- Ext. polja: omogočajo vključevanje poljubnega števila dodatnih polj. Primeri:
  - politika certifikata in politika prirejanja, pot certificiranja, omejitve.

Proces certifikacije

1. Generiranje para ključev za CA-jev podpis:
   - varnost zasebnega ključa CA je osrednja,
   - po možnosti opravljena v nepropustni napravi,
   - deljenje delov zasebnega ključa večim modulom, tako da certifikat ne more biti izdan s strani posameznega modula.
2. Generiranje para ključev osebe A:
   - bodisi s stani osebe A ali CA.
3. Zahteva za A-jev certifikat:
   - lahko, da bo CA kasneje potrebovala to zahtevo,
   - avtentičnost zahteve je potrebna.

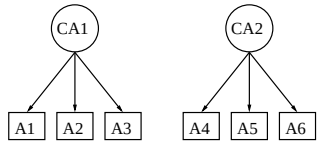
4. Identiteta osebe A je preverjena:
   - to je lahko zamudno in drago v praksi,
   - preložiti to delo na Registration Authority (RA); npr. pošto ali banko,
   - RA generira registracijski certifikat in ga prosledi CA za izdajo certifikata.
5. A-jev par ključev je preverjen:
   - CA preveri, da je javni ključ veljaven, tj. zasebni ključ logično obstaja,
   - A dokaže, da ima zasebni ključ.
6. CA naredi A-jev certifikat.
7. A preveri, da je certifikat izpraven:
   - CA lahko zahteva od A še potrdilo od prejemu.

Primer: Verisignov digitalni ID

- www.verisign.com/client/index.html
- Certifikat za javno podpisovanje in javno šifriranje.
- Certifikati so hranjeni v brskalniku ali e-poštni programski opremi.
- Brezplačni certifikati za 60-dnevno preiskusno dobo.
- Trije razredi certifikatov:
  - odgovornost prevzema Verisign (US $100, $5,000, $100,000),
  - potrditev identitete,
  - zaščita CA-jevega zasebnega ključa,
  - zaščita posameznih uporabnikovih zasebnih ključev.
- `www.versign.com/repository/index.html`

Model zaupanja

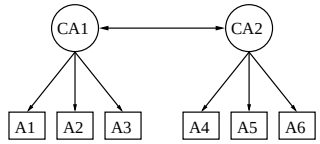
- strukturiran odnos med številnimi CA-ji.



- Stranke dobijo avtentične kopije CA-jevega javnega ključa (zunaj tekočega obsega - out-of-band, npr. med certifikacijo).
- Kako lahko A_1 preveri podpis sporočila osebe A_5? Tj. kako lahko dobi overjeno kopijo javnega ključa od A_1?
- A_1 potrebuje overjeno kopijo javnega ključa od CA_2.

Navzkrižna certifikacija

- CA-ji si lahko medsebojno overijo javne ključe

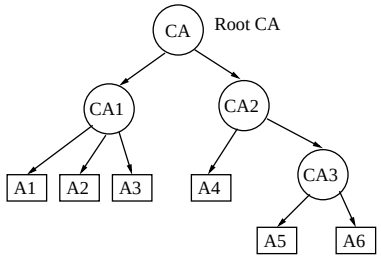


- A_1 pridobi A_5-jev overjeni javni ključ:
  - Pridobitev certifikatov CA_2 in A_5 z javnega (nezaščitenega, ne-overjenega) imenika.
  - Preveri od CA_1 podpisan certifikat CA_2 (s tem dobi overjeno kopijo javnega ključa CA_2).
  - Preveri od CA_2 podpisan certifikat A_5 (s tem dobi overjeno kopijo javnega ključa A_5).

Pomisleki glede navskrižnega certificiranja

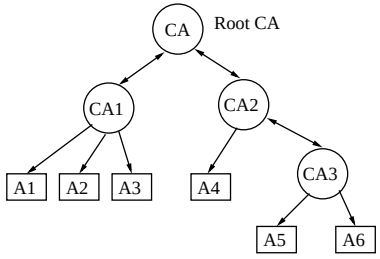
- Ali je CA_1 odgovoren osebi A_1 za varnostne probleme v domeni CA_2?
  - Potencialni problemi so lahko omejeni z izjavo v politiki CA_1 za CA_2 certifikate.
  - CA_1 mora previdno preveriti CA_2jev CPS.
  - Neodvisni pregled politike CA_2 bo pomagal.
- Ali je CA_1 odgovoren osebam iz CA_2 domene za varnostne probleme v svoji domeni?
- Vprašanje: ali bodo problemi navskrižnega certificiranja za obsežnejše aplikacije *kdaj* rešeni?

Strogo hierarhičen model



- Vsi vipis začenjajo z overjeno kopijo korenskega javnega ključa.
- Zadrški:
  - vse zaupanje je odvisno od korenskega CA,
    - rešitev: razdeli dele zasebnega ključa;
  - Certifikatne verige lahko postanejo predolge,
    - rešitev: nekatere certifikate spravimo v cache.
  - Certifikatne verige zahtevane celo za osebe znotraj iste CA,
    - rešitev: nekatere certifikate spravimo v cache.

Povratni hierarhičen model



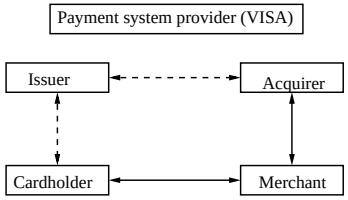
- CA lahko preveri javni ključ starševskega CA.
- Vsaka oseba prične z overjenim javnim ključem svojega CA.
- Najkrajša veriga zaupanja med A in B je pot od A do najmlajšega skupnega prednika od A in B, in nato navzdol do B.

Secure Electronic Transaction (SET)

- Standard, ki sta ga predlagala Visa in MasterCard (Feb 1996).
- Glej www.setco.org
- Cilj: varne transakcije s kreditnimi karticami prko Interneta.

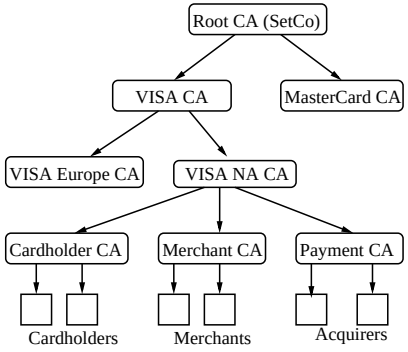
- Sodelujoči pri transakciji s kreditno kartico:
  - *Izdajatelj*: finančno podjetje, ki izdaja kreditne kartice.
  - *Lastnik kartice*: Nepooblaščen imetnik kreditne kartice holder of a credit card who is registered with the corresponding issuer.
  - *Prodajalec*: trgovec, services, or information, who accepts payment electronically.
  - *Dobavitelj*: finančna inštitucija, ki podpira prodajalca s tem, da ponuja servis za procesiranje transakcij z bančnimi karticami.

- Plačilo s kreditno kartico:



- Po Internetu: $C \longleftrightarrow M$ in $M \longleftrightarrow A$.
- Šifriranje se uporabi za zaščito številk kreditnih kartic med prenosom po Internetu; številke niso razkrite prodajalcu.
- Digitalni podpisi se uporabljajo za celovitost podatkov in overjanje udeleženih strank.

SET-ov hierarhični PKI



Preklic certifikata

- Razlogi za preklic certifikata:
  - kompromitiran ključ (redko).
  - Lastnik zapusti organizacijo.
  - Lastnik spremeni vlogo v organizaciji.
- Primer: Scotiabank tele-banking PKI:
  - Čez 90,026 certifikatov izdanih do aprila 21, 1999.
  - Čez 19,000 certifikatov preklicanih.
- Uporabnik naj bi preveril veljavnost certifikata pred njegovo uporabo.
- Preklic je enostaven v primeru on-line CA.

Certifikatne preklicne liste (CRL)

- Lista preklicanih certifikatov, ki je podpisana in periodično izdana od CA.
- Uporabnik preveri CRL predno uporabi certifikat.

Problemi z CRLs

- časovna prerioda CRL
  - Čas med preklicom in obnovitvijo CRL.
- velikost CRL
  - Delta CRL: vključuje le zadnje preklicane certifikate.
  - Groupiraj razloge za preklic.
  - Delitvene točke: revocation data is split into buckets; each certificate contains data that determines the bucket it should be placed in (patent: Entrust Technologies).
  - Uporabi avtentikacijska drevesa (komercializacija: Valicert).

Kerberos

Doslej smo spoznali sisteme, kjer vsak par uporabnikov izračuna fiksen ključ, ki se ne spreminja.
Zaradi tega je preveč izpostavljen nasprotnikom.

Zato bomo vpeljali tako imenovan sejni ključ, ki se oblikuje brž, ko se pojavita dva, ki želita komunicirati.

Tak sistem, ki uporablja simetrične sisteme, je Kerberos. Slabost tega sistema pa je zahteva po sinhronizaciji ur uporabnikov omrežja.

Določena časovna variacija je dovoljena.

Predpostavimo, da vsak uporabnik deli z agencijo TA tajni DES ključ K_U. Tako kot prej imejmo tudi $\mathrm{ID}(U)$.

Ko dobi agencija TA zahtevo po novem sejnem ključu, si TA izbere naključni ključ K, zabeleži časovno oznako T (timestamp), določi življenjsko dobo L (lifetime) za ključ K ter vse skupaj pošlje uporabnikoma U in V.

Prenos sejnega ključa z uporabo Kerberosa

- *Uporabnik U zahteva od agencije TA sejni ključ za komunikacijo z uporabnikom V.*
- *Agancija TA izbere naključni sejni ključ K, časovno oznako T in življenjsko dobo L.*
- *TA izračuna* $m_1 = e_{K_U}(K, \mathrm{ID}(V), T, L)$ *in* $m_2 = e_{K_V}(K, \mathrm{ID}(U), T, L)$ *ter ju pošlje uporabniku U.*
- *U uporabi odšifrirno funkcijo d_{K_U}, da dobi iz m_1 K, T, L in $\mathrm{ID}(V)$. Potem izračuna $m_3 = e_K(\mathrm{ID}(U), T)$ in ga pošlje osebi V skupaj s sporočilom m_2, ki ga je dobil od agencije TA.*

- *V uporabi odšifrirno funkcijo d_{K_V}, da dobi iz m_2 K, T, L in $\mathrm{ID}(U)$. Potem uporabi d_K, da dobi T in $\mathrm{ID}(U)$ iz m_3. Preveri, da sta tako dobljeni vrednosti za T in $\mathrm{ID}(U)$ enaki prejšnjim. Če je tako, potem izračuna še*

$$m_4 = e_K(T + 1)$$

  in ga pošlje uporabniku U.
- *U odšifrira m_4 z uporabo e_K in preveri, ali je rezultat enak $T + 1$.*

V tem protokolu se prenašajo različne funkcije sporočil.

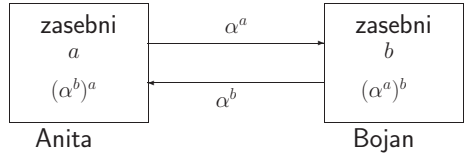
Sporočili m_1 in m_2 poskrbita za tajnost pri prenosu sejnega ključa K.

Sporočili m_3 in m_4 se uporabljata kot potrdilo sejnega ključa K tako, da se U in V prepričata, da imata res isti sejni ključ K.

Diffie-Hellmanova uskladitev ključev

Naj bo p praštevilo in α generator multiplikativne grupe $\mathbb{Z}_p^*$. Naj bosta oba javno poznana (ali pa naj ju oseba U sporoči osebi V).

1. *Oseba U izbere naključen a_U, $0 \le a_U \le p-2$, izračuna $\alpha^{a_U} \bmod p$ in ga pošlje osebi V.*
2. *Oseba V izbere naključen a_V, $0 \le a_V \le p-2$, izračuna $\alpha^{a_V} \bmod p$ in ga pošlje osebi U.*
3. *Osebi U in V izračunata zaporedoma $K = (\alpha^{a_V})^{a_U} \bmod p$ in $K = (\alpha^{a_U})^{a_V} \bmod p$.*



Anita in Bojan si delita skupni element grupe:

$$(\alpha^a)^b = (\alpha^b)^a = \alpha^{ab}.$$

Edina razlika med tem protokolom in pa Diffie-Hellmanovim protokolom za distribucijo ključev je, da si izberemo nova eksponenta a_U in a_V uporabnikov U in V zaporedoma vsakič, ko poženemo ta protokol.

Varnost Diffie-Hellmanovega protokola

Protokol ni varen pred aktivnim napadalcem, ki prestreže sporočila in jih nadomesti s svojimi. Ta napad bomo imenovali **napad srednjega moža**.

$$U \underset{\alpha^{a'_V}}{\overset{\alpha^{a_U}}{\rightleftarrows}} W \underset{\alpha^{a_V}}{\overset{\alpha^{a'_U}}{\rightleftarrows}} V$$

Na koncu sta osebi U in V vzpostavili z napadalcem W zaporedoma ključa $\alpha^{a_U a'_V}$ in $\alpha^{a'_U a_V}$.

Tako bo zašifrirano sporočilo osebe U odšifriral napadalec W ne pa oseba V.

Uporabnika U in V bi bila rada prepričana, da ni prišlo namesto medsebojne izmenjave sporočil do izmenjave z napadalcem W.

Potrebujeta protokol za medsebojno avtentikacijo (predstavitev).

Dobro bi bilo, če bi potekala avtentikacija istočasno z uskladitvijo ključev, saj bi s tem onemogočili aktivnega napadalca.

Overjena uskladitev ključev

Diffie, Van Oorschot in Wiener so predlagali protokol **uporabnik-uporabniku** (station-to-station - STS), ki je protokol za *overjeno uskladitev kjuča* in je modifikacija Diffie-Hellmanove uskladitve ključev.

Vsak uporabnik ima **certifikat (potrdilo)**

$$C(U) = \Big(\mathrm{ID}(U), \mathrm{ver}_U, \mathrm{sig}_{\mathrm{TA}}\big(\mathrm{ID}(U), \mathrm{ver}_U\big)\Big),$$

kjer je shranjena njegova identifikacija $\mathrm{ID}(U)$.

Poenostavljen protokol uporabnik-uporabniku

1. *Oseba U izbere naključen $a_U \in \{0, \ldots, p-2\}$, izračuna $\alpha^{a_U} \bmod p$ in pošlje osebi V.*
2. *Oseba V izbere naključen $a_V \in \{0, \ldots, p-2\}$, izračuna $\alpha^{a_V} \bmod p$, $K = (\alpha^{a_U})^{a_V} \bmod p$ in $y_V = \mathrm{sig}_V(\alpha^{a_V}, \alpha^{a_U})$, ter pošlje potrdilo $(C(V), \alpha^{a_V}, y_V)$ osebi U.*

3. *Oseba U izračuna $K = (\alpha^{a_V})^{a_U} \bmod p$ ter preveri podpis y_V z uporabo ver_V in potrdilo $C(V)$ z $\mathrm{ver}_{\mathrm{TA}}$.*

   Nato izračuna $y_U = \mathrm{sig}_U(\alpha^{a_U}, \alpha^{a_V})$ in pošlje potrdilo $(C(U), y_U)$ osebi V.
4. *Oseba V preveri podpis y_U z uporabo ver_U in potrdilo $C(U)$ z uporabo $\mathrm{ver}_{\mathrm{TA}}$.*

Varnost protokola STS

Uporabnika U in V si izmenjata naslednje informacije (izpustimo potrdila):

$$
\begin{array}{ccc}
 & \xrightarrow{\alpha^{a_U}} & \\
U & \xleftarrow{\alpha^{a_V},\ \mathrm{sig}_V(\alpha^{a_V},\alpha^{a_U})} & V \\
 & \xrightarrow{\mathrm{sig}_U(\alpha^{a_U},\alpha^{a_V})} &
\end{array}
$$

Kaj lahko naredi napadalec W (mož na sredini):

$$
\begin{array}{ccccc}
 & \xrightarrow{\alpha^{a_U}} & & \xrightarrow{\alpha^{a'_U}} & \\
U & \xleftarrow{\alpha^{a'_V},\ \mathrm{sig}_V(\alpha^{a'_V},\alpha^{a_U})=?} & W & \xleftarrow{\alpha^{a_V},\ \mathrm{sig}_V(\alpha^{a_V},\alpha^{a'_U})} & V \\
 & \xrightarrow{\mathrm{sig}_U(\alpha^{a_U},\alpha^{a'_V})} & & \xrightarrow{\mathrm{sig}_U(\alpha^{a'_U},\alpha^{a_V})=?} &
\end{array}
$$

Poenostavljeni STS protokol je torej varen pred napadom srednjega moža.

Tako oblikovan protokol ne vsebuje potrditve ključa, kakor je slučaj v Kerberosovi shemi.

Protokol, v katerem je vključena potrditev ključa:

$$y_V = e_K(\mathrm{sig}_V(\alpha^{a_V},\alpha^{a_U})), \quad y_U = e_K(\mathrm{sig}_U(\alpha^{a_U},\alpha^{a_V}))$$

se imenuje STS protokol.

MTI protokoli

Matsumoto, Takashima, Imai so modificirali Diffie-Hellmanovo uskladitev ključev, tako da uporabniki U in V ne potrebujejo podpisov.

Kadar moramo izmenjati dve pošiljki, pravimo, da gre za **protokole z dvema izmenjavama**.

Predstavili bomo en njihov protokol.

Osnovne predpostavke so enake kot pri Diffie-Hellmanovi uskladitvi ključev: praštevilo p in generator α multiplikativne grupe $\mathbb{Z}_p^*$ sta javna.

Vsak uporabnik U ima svoj *zasebni* eksponent a_U $(0 \le a_U \le p-2)$ in *javno* vrednost $b_U = \alpha^{a_U} \bmod p$.

Agencija TA ima shemo za digitalni podpis, z *javnim* algoritmom $\mathrm{ver}_{\mathrm{TA}}$ in *tajnim* algoritmom $\mathrm{sig}_{\mathrm{TA}}$.

Vsak uporabnik U ima svoj certifikat:

$$C(U) = (\mathrm{ID}(U), b_U, \mathrm{sig}_{\mathrm{TA}}(\mathrm{ID}(U), b_U)).$$

1. *Oseba U izbere naključen $r_U \in \{0,\dots,p-2\}$, izračuna $s_U = \alpha^{r_U} \bmod p$ in pošlje osebi V $(C(U), s_U)$.*
2. *Oseba V izbere naključen $r_V \in \{0,\dots,p-2\}$, izračuna $s_V = \alpha^{r_V} \bmod p$ in pošlje osebi U $(C(V), s_V)$.*
3. *Osebi U in V izračunata zaporedoma*
$$K = s_V^{a_U} b_V^{r_U} \bmod p \quad \text{in} \quad K = s_U^{a_V} b_U^{r_V} \bmod p,$$
kjer sta b_V in b_U zaporedoma iz $C(V)$ in $C(U)$.

Varnost protokola MTI

Ta MTI protokol je enako varen pred pasivnimi sovražniki kot Diffie-Hellmanov protokol.

Varnost pred aktivnimi sovražniki je bolj vprašljiva. Brez uporabe podpisnega algoritma nismo varni pred napadom srednjega moža.

$$
\begin{array}{ccc}
U & \xleftrightarrow[C(V),\ \alpha^{a_V} \bmod p]{C(U),\ \alpha^{r_U} \bmod p} & V
\end{array}
$$

Ključ uporabnikov, ki komunicirata, je težko izračunati, ker je v ozadju težko izračunljiv diskretni logaritem.

Tej lastnosti pravimo **implicitna overitev ključev**.

Uskladitev ključev s ključi, ki se sami overijo

Giraultova shema ne potrebuje certifikatov, saj uporabnike razlikujejo že njihovi javni ključi in identifikacije.

Vsebuje lastnosti RSA sheme in diskretnega logaritma.

Uporabnik naj ima identifikacijo $\mathrm{ID}(U)$.
Javni ključ za osebno overitev dobi od agencije TA.

Naj bo $n = p\,q$, kjer je $p = 2p_1 + 1$, $q = 2q_1 + 1$, in so p, q, p_1, q_1 velika praštevila. Potem je

$$(\mathbb{Z}_n^*, \cdot) \sim (\mathbb{Z}_p^* \times \mathbb{Z}_q^*, \cdot).$$

Največji red poljubnega elementa v $\mathbb{Z}_n^*$ je najmanjši skupni večkratnik elementov $p-1$ in $q-1$ oziroma $2p_1q_1$.

Naj bo α generator ciklične podgrupe v $\mathbb{Z}_p^*$ reda $2p_1q_1$, problem diskretnega logaritma v tej podgrupi pa naj bo računsko prezahteven za napadalca.

Javni ključ za osebno overitev

Naj bosta števili n, α javni, števila p, q, p_1, q_1 pa naj pozna samo agencija TA.

Število e je javni RSA šifrirni eksponent in ga izbere agancija TA, $d = e^{-1} \bmod \varphi(n)$ pa je tajni odšifrirni eksponent.

1. *Oseba U izbere tajni eksponent a_U, izračuna $b_U = \alpha^{a_U} \bmod n$ in izroči a_U ter b_U agenciji TA.*
2. *Agencija TA izračuna $p_U = (b_U - \mathrm{ID}(U))^d \bmod n$ ter ga izroči osebi U.*

Giraultov protokol za uskladitev ključev

1. *Oseba U izbere naključen zasebni r_U, izračuna*
$$s_U = \alpha^{r_U} \bmod n$$
ter pošlje $\mathrm{ID}(U)$, p_U in s_U osebi V.
2. *Oseba V izbere naključen zasebni r_V, izračuna*
$$s_V = \alpha^{r_V} \bmod n$$
ter pošlje $\mathrm{ID}(V)$, p_V in s_V osebi U.
3. *Osebi U in V izračunata ključ K zaporedoma z*
$$s_V^{a_U}(p_V^e + \mathrm{ID}(V))^{r_U} \bmod n, \quad s_U^{a_V}(p_U^e + \mathrm{ID}(U))^{r_V} \bmod n.$$

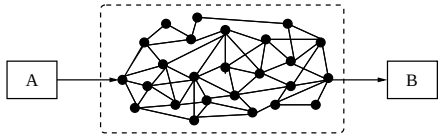
Varnost Giraultovega protokola

Ključ za osebno overitev varuje pred sovražniki.

Protokol implicitno overi ključe, zato napad srednjega moža ni možen.

Agencija TA je prepričana, da uporabnik pozna vrednost števila a predno izračuna ključ za osebno overitev.

Internetne aplikacije

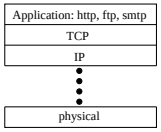


- ftp: File Transfer Protocol
- http: HyperText Transfer Protocol
- smtp: Simple Mail Transfer Protocol

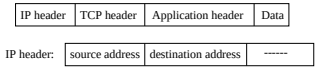
TCP – Transport Control Protocol
IP – Internet Protocol

TCP/IP

Protokolov sklad:



TCP/IP paket:



Nekateri napadi

- **IP address spoofing** (slov. ponarejanje naslovov)
  rešitev: overi glavo IP paketa
- **IP packet sniffing** (slov. vohljanje za IP paketi)
  rešitev: zašifriraj IP payload (vse kar se prenaša)
- **Traffic analysis** (slov. Analiza prometa)
  rešitev: zašifriraj pošiljateljev in prejemnikov naslov

Varnost znotraj TCP/IP

Varnostni protokoli so prisotni na različnih nivojih TCP/IP sklada.

1. IP nivo: IPsec.
2. Transportni nivo: SSL/TLS.
3. Aplikacijski nivo: PGP, S/MIME, SET, itd.

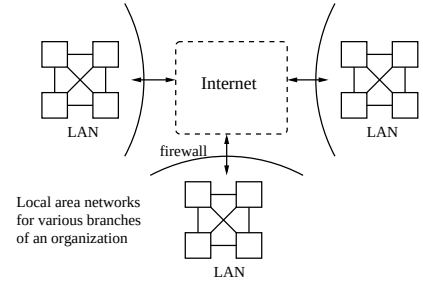
Internet Engineering Task Force (IETF)

- Sprejema standarde za razvoj Internetne arhitekture in omogoča nemoteno delovanje Interneta.
- Odprta za vse zainteresirane posameznike: `www.ietf.org`
- Delo, ki ga opravljajo delovne skupine povezane z varnostjo (Security Area) pokrivajo:

- IP Security Protocol (IPsec)
- Transport Layer Security (TLS)
- S/MIME Mail Security
- Odprto specifikacijo za PGP (OpenPGP)
- Secure Shell (secsh)
  (Nova verzija ssh protokola, ki omogoča varno prijavo na oddaljene šifre in varen prenos datotek.)
- X.509 Public-Key Infrastructure (PKIX)

IPsec: Virtual Private Networks (VPNs)

Omogočajo šifriranje in overjanje (overjanje izvora podatkov, celovitost podatkov) na IP layer.

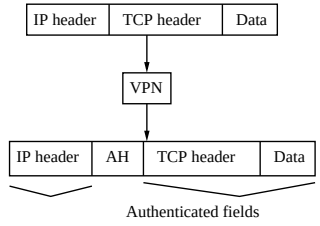


Gradniki IPsec

- Security Association (SA):
  - upravlja algoritme in ključe med sogovorniki,
  - vsaka glava IPsec se nanaša na Security Association preko Security Parameter Index (SPI).
- Upravljanje s ključi:
  - dogovor o ključu z Diffie-Hellmanovo shemo (OAKLEY),
  - kreira ključe za Security Association,
  - upravljanje z javnimi ključi, ki ni pokrito v IPsec.
- Trije načini IPsec servisov:
  - AH: overjanje,
  - ESP: šifriranje + overjanje.

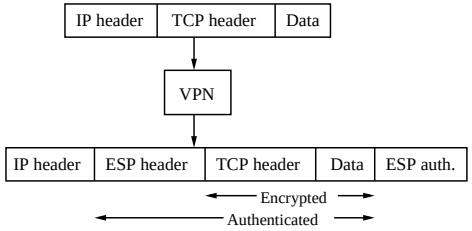
IPsec glava za overjanje (AH)

- Podpira MACs: HMAC-MD5-96, HMAC-SHA-1-96.
- Transportni način:



IPec ESP glava

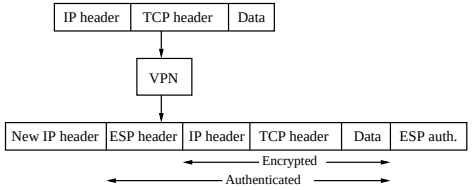
- Encapsulating Security Payload.
- Podprti šifrirni algoritmi: 3-DES, RC5, IDEA, ...
- Transportni način:



- Opomba: analiza prometa je še vedno možna (ker IP glave niso šifrirane).

ESP v tunelskem načinu

- Požarni zid vključi novo IP glavo (IP naslov pošiljateljevega požarnega zidu in IP naslov prejemnikovega požarnega zidu).
- Možna je samo zelo omejena analiza prometa.



Secure Sockets Layer (SSL)

- SSL je naredil Netscape.
- TLS (Transport Layer Security) je IETF-ova verzija SSL-a.
- SSL uporabljamo v brskalnikih (npr. Netscape) za zaščito mrežnih transakcij.
- Osnovne komponente SSL/TLS:

  handshake protocol: dopusti strežniku in klientu, da se overita in dogovorita za kriptografske ključe,

  record protocol: uporabljan za šifriranje in overjanje prenašanih podatkov.

Upravljanje z javnimi ključi v SSL/TLS

- Korenski CA ključ je vnaprej inštaliran v brskalnik.
  - Klik na "Security" in nato na "Signers", da najdete seznam ključev korenskih CA v Netscape-u.
- Mrežnim strežnikom certificirajo javne ključe z enim izmed korenskih CA-jev (seveda brezplačno).
  - Verisign-ov certification business za mrežne strežnike `www.verisign.com/server/index.html`

- Klienti (uporabniki) lahko pridobijo svoje certifikate. Večina uporabnikov trenutno nima svojih lastnih certifikatov.
  - Če klienti nimajo svojih certifikatov, potem je overjanje samo enostransko (strežnik se avtenticira klientu).
  - Obiščite varno internetno stran kot npr. webbroker1.tdwaterhouse.ca in kliknite na "padlock" v Netscapu, da si ogledate informacijo o strežnikovem certifikatu.

SSL/TLS handshake protocol

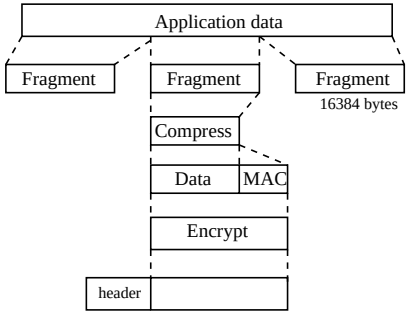
Na voljo so naslednji kriptografski algoritmi:

- MAC: HMAC-SHA-1, HMAC-MD5.
- šifriranje s simetričnimi ključi: IDEA, RC2-40, DES-40, DES, Triple-DES, RC4-40, RC4-128.
- Osnovne sheme za dogovor o ključu so:

- RSA transport ključev: deljeno skrivnost izbere klient in jo zašifrirana s strežnikovim javnim RSA ključem.
- Fixed Diffie-Hellman: strežnikov Diffie-Hellman-ov javni ključ g^x je v njegovem certifikatu. Klient ima lahko g^y v svojem certifikatu, ali generira enkratno vrednost g^y.
- Ephemeral Diffie-Hellman: Strežnik izbere enkratni Diffie-Hellman-ov javni ključ g^x in ga podpiše s svojim RSA ali DSA ključem za podpise. Klient izbere enkratni g^y in ga podpiše če in samo če ima certifikat.
- MAC in šifrirni ključi so izpeljani iz skupne skrivnosti.

SSL/TLS handshake protokol (2)

1. faza: Določi varnostne zmožnosti.
   - Verzija protokola, način kompresije, kriptografski algoritmi,...
2. faza: Strežnikovo overjanje in izmenjava ključev.
   - Strežnik pošlje svoj certifikate, in (morda še) parametre za izmenjavo ključev.
3. faza: Klientovo overjanje in izmenjava ključeve.
   - Klient pošlje svoj certifikat (če ga ima) in parametre za izmenjavo ključev.
4. faza: Zaključek.

SSL/TLS record protocol

Predpostavimo, da klient in strežnik delita MAC tajnega ključa in sejni šifrirni ključ: