9 Identifieringsmetoder

S. Lindström. Det som inte tål att skämtas med, det förtjänar sällan tas på allvar.

I detta kapitel behandlas metoder för att uppnå användarautenticitet. Först kommer några enkla metoder, därefter beskrivs protokoll som är avsedda för smarta kort.

9.1 Allmänt

9.1.1 Nomenklatur

Identifiering kan ske via tre principiellt olika komponenter:

- Egenskap, t ex röst och utseende.
- Egendom, t ex id-kort och pass.
- Kunskap, t ex lösenord, nyckel, pin och diskret logaritm.

I datoriserade sammanhang utnyttjas ofta kunskap.

Men då lösenord eller pin sänds i klartext över datornät uppstår osäkerhetsfaktorer. Om varje "terminal" kan utföra beräkningar och lagra nycklar på ett säkert sätt, så kan kryptografiska funktioner användas för att undvika att exponera privata kunskaper i onödan.

i. Termer. Om skillnad mellan signering och identifiering:

- Vid digital signering är det endast den signerande parten som kan generera godkända transaktioner medan alla kan verifiera signaturen. Signaturen binds till ett dokument.

- Vid identifiering råder förutsättningen "ömsesidig misstänksamhet" eftersom den som ska verifiera identiteten t ex kan tänkas "luras" genom att konstruera påhittade transaktioner eller stjäla id-information. Vidare ska en avlyssnare inte kunna imitera via återspelning.

- Skilj också på förbindelse eller åtagande ('commitment') och identifiering.

> Ett anonymt brev har ingetdera.
> Ett företagsbrev identifieras m h a en logotype e dyl.
> En check innebär ett åtagande även om identifiering saknas.

A. Fiat och A. Shamir använder följande nomenklatur. Men observera att det förekommer andra definitioner också.

Alla metoder uppfyller:	A kan bevisa för B att hon är A
i. Autenticeringsmetoder uppfyller också:	men ingen annan kan bevisa att han är A
ii. Identifieringsmetoder uppfyller också:	men B kan inte bevisa för någon annan att han är A
iii. Digitala signaturer uppfyller också:	men B kan inte bevisa ens för sig själv att han är A

Härav följer att i. är användbara endast då A och B samarbetar mot externa faror. Oftast skyddar de sig via en delad hemlig nyckel. Skillnaden mellan ii. och iii. ligger väsentligen i att iii. erbjuder möjlighet för en tredje part att efteråt bedöma processens äkthet. Detta kan vara eller inte vara oväsentligt om verifieringen sker i realtid.

De flesta metoder för identifiering bygger på 'challenge-response'-protokoll.

Exempel

0. A besitter kunskap om k	-- 'witness'
1. B väljer ett x ('challenge') och skickar detta till A.	-- 'challenge'
2. A beräknar $y = e_k(x)$ och skickar y till B.	-- 'response'
3. B beräknar $z = e_k(x)$ och verifierar att $z \equiv y$.	-- verifiering

Detta protokoll kräver att A och B i förväg delar en privat nyckel.

Not `År 1960 uppfattades uppstigande månen som missiler då den inte kunde svara på jaktflygets 'challenge'.`

Några protokoll som beskrivs nedan (t ex Schnorr, Guillou-Quisquater, Feige-Fiat-Shamir) bygger på PKS kombinerat med certifikat varvid identifieringen kan sammanfattas:

1. A → B: certifikat, öppen nyckel ('commitment, 'witness')

2. B → A: "utmaning".

3. A → B: "svar" baserat på privat nyckel samt att B verifierar.

Certifikatet är nödvändigt för att den som verifierar identiteten ska vara övertygad om att paret <namn, öppen nyckel> stämmer överens. Certifikat kan också bildas med blind signering för de fall en central 'trusted/certification authority', TA/CA, är olämplig.

ii. Smarta kort. Dessa är kort i betalkortsformat bestyckade med processor och minnen; RAM, ROM, och EEPROM. Många innehåller dessutom ett filsystem. Strömförsörjning sker externt via kontakter och speciella kortläsare.

Användningsområden är (grovt)

- Identifiering
- Åtkomstkontroll.
- Signering.
- Informationslagring, t ex journaler, lexikon.
- Betalningsmedel.
- Symmetrisk kryptering för konfidentialitet.

Programvaran på kortet brukar faktiskt kallas operativsystem.

Några exempel på data: Här gäller det Philips DX Smart Card: fem pinnar; *clk*, *rst*, *vcc*, *i/o*, *gnd*, 8051 processor, 256 byte RAM; 'run time' minne, 2K EEPROM; data, program, 6K ROM; OS och RSA-enhet.

Ett annat exempel: SIM ('subscriber identity module') används i mobiltelefoner (ME = 'moblie equipment'). Det tre väsentligaste funktionerna som dessa erbjuer är

- Anonymitet för abonnent via tempoärara identiteter
- Meddelandekonfidentialitet via ett symmetriskt flödeschiffer (A5, A8)
- Användarautenticitet via ett 'challenge-response'-förfarande (A3)

Kortet som är 25 x 15 mm^2 innehåller ett ROM om 20K, ett RAM om 0.5K och ett EEPROM om 16K och klarar på serieporten nominellt 9600 bits/sek. ROM innehåller

algoritmerna A3 0ch A8 medan däremot A5 ligger i ME. A8 är nyckelgeneratorn för flödeschiffret. Access till kortet innefattar pin, pin2 och puk ('pin unblocking key').

Man uppskattar att det 1994 fanns ca 400 M kort. Prognosen lyder på 4 G kort fram till år 2000.

Smarta kort gör identifieringen säkrare. Notera dock att det är kortet som identifierar sig för tjänstestället och att man som person måste identifiera sig för kortet separat.

Icke desto mindre kan identifikation göras betydligt säkrare med kort än utan.

I resten av detta avsnitt presenteras några metoder baserade på lösenord eller etablerade nycklar för symmetriska chiffer.

9.1.2 Lösenordshantering

Lösenord lagras i en fil i operativsystem. Denna kan skyddas med vanlig åtkomstkontroll. Många system (t ex Unix) vidtar ytterligare säkerhetsåtgärder genom att (envägs)kryptera (*EV*) lösenordsfilen.

Användare	Krypterat lösenord
A	*EV* (*password A*)
B	*EV* (*password B*)
…	…

När en användare X gör 'log in' transformerar 'log in'-programmet omedelbart angivet lösenord och jämför detta med versionen i tabellen och accepterar X endast vid överensstämmelse.

Eftersom lösenorden aldrig behöver återtransformeras till klartext behöver *EV* inte ha någon invers (därav namnet). Ofta används ett blockchiffer e_k och varvid

$$EV(\mathrm{X}) = e_{password\ X}(m), \quad \text{-- } passwordX \text{ lagras inte i systemet,}$$

där *m* är ett fixt meddelande tex *m* = "PASSWORD".

Observera att ett en-vägschiffer INTE kan implementeras med ett flödeschiffer där m_0 = nyckeln och *m* = "klartextsträngen"; det gäller ju då att $m_0 = c$ **xor** *m*.

Ofta kompletteras varje lösenord av operativsystemet med ett 'salt'. Detta är en publik slumpbitsträng vars syfte är att försvåra en process att generera slumpmässiga gissningar av lösenord i akt och mening att finna "matchning" över hela lösenordsfilen. Det skyddar inte mot gissning av en specifik individs lösenord.

Ingen teknik i världen kan heller kompensera för mediokert etablerade lösenord eller pin-koder.

9.1.3 Inloggning med signaturer och lösenord

Följande protokoll, tabell 9.1, kan användas för inloggning över avlyssnade förbindelser. Transformationerna *d* är privata och *e*-transformationerna är publika.

Här undviks alltså att skicka lösenordet i klartext genom att använda PKS. En terminal i ett sådant system måste alltså ha viss beräkningskapacitet, endera i en generell processor eller i ett smart kort.

Steg	*Meddelande*	*Kommentar*
1. A → S	*A*	As identitet till systemet S
2. S → A	$X = d_S(T)$	T = tid, d = Ss privata sign.
3. A → S	$Y = d_A(T)$ $Z = e_S(T, P)$	A tar fram T med e_S P = lösenordet
4. S	--	Validerar nu autenticitet av A via T och P

Tabell 9.1. Inlogging

Transformationerna *d* kan ses som signaturer och *e* som en verifiering.

9.2 Transaktionsautomater

Här skisseras huvuddragen av protokollet för kommunikation mellan en betalautomat och kortutfärdarens dator. Kommunikationen går vanligtvis över hyrda eller helt privata linjer.

i. Förutsättningar. Både transaktionsautomaten (terminalen) och datorn innehåller varsin välskyddad 'security module'. Till terminalen associeras en terminalnyckel *T* skyddad i denna.

Varje transaktion kräver ett betalkort som innehåller ett *PAN* ('Primary Account Number'). Varje användare har en (hemlig) *PIN*-kod ('Personal Identification Number').

I säkerhetsmodulen i bankdatorn finns två skyddade tabeller:

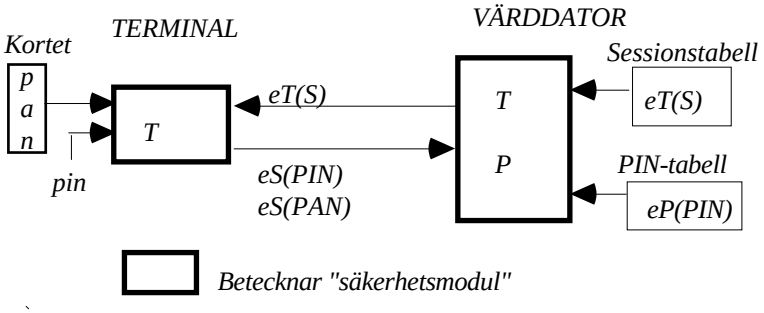
- Sessionstabell.

- *PIN*-tabell.

Sessionstabellen innehåller poster av typen $e_T(S)$; alltså en krypterad version av förestående sessions krypteringsnyckel.

PIN-tabellen indexeras av *PAN* och innehåller poster av typ $e_P(PIN)$, dvs krypterade versioner av *PIN*. Nyckeln *P* är en '*PIN* master key'.

ii. Protokoll. Följande dialog genomförs. En säkerhetsmodul är en fysiskt skyddad enhet.



Figur 9.1. PIN- och PAN- hantering

1. Kortet sätts in i automaten och den fysiska förbindelsen etableras.

2. Bankdatorn översänder $e_T(S)$ till terminalen.

3. Terminalen läser kortets *PAN* och frågar efter användarens *PIN*.

4. Terminalen skickar över paret $<e_S(PIN), e_S(PAN)>$ till datorn.

5. Datorn kontrollerar *PIN/PAN*-överenstämmelse via *PIN*-tabellen

6. och svarar *OK* eller '*REJECT*' till terminalen.

7. Transaktionen inleds med att begärd åtgärd översänds till datorn i formen

< Diverse klartextinformation,
e_S(åtgärd, belopp, etc),
'*message authentication code*' >

8. varvid denna kan svara på lämpligt vis.

Tabell 9.2. Att ta ut pengar på bankomat

iii. Anmärkningar. Oftast används idag DES som krypteringsalgoritm i t ex 'cipher feedback mode'. RC4 är också vanlig.

Det finns lite olika standarder för kodning av plastkort men alla innehåller *PAN* (upp till 19 tecken), landskod, giltighetstid för kortet, kontobalans eller kreditgräns, checksumma och oftast också för- och efternamn.

Säkrare kort kan framställas om dessa också innehåller en personlig nyckel *PK*, vald av utfärdaren.

Chiffret $e_{PK \mathbf{xor} PIN}(PAN)$ kan nu sändas till utfärdaren för verifiering.

9.3 Användarautenticering enligt ISO

i. 9594-9. Dessa s k X.509-protokoll (ITU-I, som för ISO heter 9594-8) bygger på PKS. RSA (transformationernna *e* och *d* nedan) är ett uppenbart val, men protokollet kan användas med andra signaturmetoder. ISO använder benämningen autenticering i st f identifiering.

De tre protokollstegen bygger på *certifikat e*ller *kreditiv* ('credentials') C_X som utfärdas av en 'certification authority' (CA) eller 'trusted authority' (TA).

Notera också att metoden kombinerar signaturer och data-chiffrering för konfidentialitet (för godtyckliga meddelanden eller speciellt nycklar).

CERTIFIKAT

Certifikaten innehåller användarens namn, en giltighetstid,
publik nyckel och eventuellt en signatur

$C(A) = <id(A), t, key, sig_{CA}(id(A), key)>$

Det enkelriktade protokollet (steg 1) avser att verifiera As identitet och åstadkomma meddelandeintegritet. Det förhindrar falsk återspelning via engångs-slumptal.

Not. Att både slumptal och tidsstämplar finns med förklaras av att de förra är olämpliga vid 'connection-less' kommunikation (slumptal kräver ju 'challenge-respons') medan de senare är olämpliga vid 'connection-oriented' kommunikation (synkronisering kan vara svårt).

Det dubbelriktade (steg 1 och steg 2) garanterar att "respondern" inte är en 'active wiretapper'.

Det tredje steget eliminerar behovet av tidstämplar.

Steg 1

1. A: Skapar ett slumptal R_a och $m = <T_a, R_a, I_b, data>$,

där T_a är As tidstämpel, I_b är Bs identitet. Kvantiteten *data* kan chiffreras med Bs publika nyckel e_B om konfidentialitet önskas.

2. A $\rightarrow$ B: $<C(A), d_A(m)>$

eller

$$<C(A), m, sig_A(m)>$$

om signatur med appendix används.

3. B: Utför följande.

- Verifierar $C(A)$ och erhåller e_A och kontrollerar att giltighetstiden är OK.

- Använder e_A för att dekryptera $d_A(m)$. Detta verifierar dels As signatur, dels integriteten hos signerad information m.

- Kontrollerar I_b och T_a.

- Eventuellt kan B också kontrollera R_a för att övertyga sig om att ingen återspelning skett.

Steg 2

4. B skapar ett slumptal R_b och $m' = <T_b, R_b, I_a R_a, data>$,

där T_b är Bs tidstämpel, I_a är As identitet. Kvantiteten *data* kan chiffreras med As publika nyckel d_A om konfidentialitet önskas.

5. B $\rightarrow$ A: $d_B(m')$. Alternativt: $<C(B), m', sig_B(m')>$ i appendixfallet

6. A: Utför följande

- Använder e_B för att dekryptera $d_B(m')$. Detta verifierar dels Bs signatur, dels integriteten av signerad information m'.

- Kontrollerar I_a och T_b i m'.

- Eventuellt kan A också kontrollera R_b mot en befintlig databas för att övertyga sig om att ingen återspelning skett.

Steg 3

Med $T_a = T_b = 0$ i steg 1 och steg 2 fortsätter det tredje protokollet med följande steg.

7. A kontrollerar mottagen version av R_a mot den version som sändes.

8. A $\rightarrow$ B: $d_A(R_b, I_b)$. Alternativt: $<R_b, I_b, sig_A(R_b, I_b)>$ i appendixfallet

9. B utför följande

- Använder e_A för att dekryptera $d_A(R_b)$. Detta verifierar dels As signatur, dels R_b.
- Kontrollerar mottagen version av R_b med den version B sände till A.
- Not. Kvantiteten I_b behövs för att förhindra en 'man-in-the-middle replay-attack'.

ii. 9798-8. Dessa tre protokoll är renodlat för identifiering.

1. Envägsidentifiering med tidsstämpel (t).

A$\rightarrow$ B: C(A), t_A, B, $sig_A(t_A, B)$

2. Envägsidentifiering med slumptal (r).

B $\rightarrow$ A: r_B
A $\rightarrow$ B: C(A), r_A, B, $sig_A(r_A, r_B, B)$

3. Ömsesidig (tvåvägs) identifiering med slumptal (r).

Börjar som 2. men innehåller även

B $\rightarrow$ A: C(B), A, $sig_B(r_A, r_B, A)$

9.4 Schnorrs identifikationsmetod

Denna metod bygger som flera andra (9.5, 9.6) på följande idéer.

A väljer ett slumpmässigt element från en förutbestämd mängd som privat 'commitment'; han gör privata slantsinglingar. Från erhållet värde skapas sedan ett publikt vittne.

Man kan beskriva detta som en definition av en mängd frågor som A förbinder sig kunna svara på. Genom konstruktionen av protokollet så är det A och bara A som med kunskap om sin [långlivade] privata nyckel kan svara på alla frågor/utmaningar och utan att ge ut någon information om sin privata nyckel.

B's utmaning väljer ut en av dessa frågor. A svarar och B kontrollerar om svaret är korrekt. Om så behövs upprepas protokollet för att ytterligare minska sannolikheten för bedrägeri/imitation.

Dessa protokoll kombinerar i själva verket 'challenge-response'-tekniken med vad som skulle kunna kallas 'cut-and-choose' (jämför hur två barn delar på en tårta; en skär upp bitarna, det andra väljer bit först).

Centralt är att A bara svarar på en fråga/utmaning för givet vittne och att ett vittne inte återanvänds. I många verkliga protokoll skulle i så fall [den långlivade] privata nyckeln kompromettеras.

Scnorrs metod förutsätter en 'trusted authority' TA som bl a skapar certifikat för användare. Användarna representeras av smarta kort.

Schnorrs metod är betydligt snabbare än RSA vid likvärdig säkerhetsnivå.

Metodens säkerhet bygger på svårigheten att beräkna den diskreta logaritmen.

TA definierar systemets grundparametrar.

i. Grundparametrar. Dessa är som följer

1. p ett stort primtal, $p \geq 2^{512}$, för att göra problemet med den diskreta logaritmen i Z_p^* svårt.

2. q en stor primtalsdivisor, $q \geq 2^{140}$, till $p - 1$ (t ex $p = 2q + 1$).

3. $\alpha \in Z_p^*$ är av ordning q (en q:te rot till 1). Utgå från ett primitivt element α_0 och beräkna $\alpha = \alpha_0^{(p-1)/q}$.

4. t är en säkerhetsnivå, tex $t = 40$, sådan att $q > 2^t$.

5. TA fastställer en signaturmetod med en publik ver_{TA} och en privat sig_{TA}.

6. Slutligen behövs en säker 'hash'-funktion h som preparandsteg före signering. (Den utelämnas dock i nedanstående beskrivningar.)

Talen p, q och α och funktionerna ver_{TA} och h är publika.

ii. Certifikat. Utgående från dessa parametrar framställs certifikat $C(A)$ mm till en användare A på följande vis.

Detta använder sedermera A när A vill bevisa sin identitet för B, C, ...

1. TA fastställer As identitet manuellt (id-kort edyl) och formar strängen $id(A)$.

2. A väljer en *privat* och slumpmässig exponent a, $0 \leq a \leq q - 1$.

3. A $\rightarrow$ TA: $v = \alpha^{-a} \bmod p$. (Den publika nyckeln)

4. TA $\rightarrow$ A: $C(A) = \langle id(A), v, sig_{TA}(id(A), v) \rangle$ -- certifikatet

iii. Identifieringen visas i tabell 9.3.

Syftet med t är att definiera en säkerhetsnivå som förhindrar för en inkräktare från att gissa sig till Bs 'challenge' r; se steg 4.

Om nämligen denne kunde gissa sig till r så kunde hon välja ett godtyckligt y och med följande γ

$$\gamma \equiv \alpha^y v^r \ (mod p)$$

via steg 1 i identifieringsprotokollet lura B genom att efter mottagandet av r (steg 4.) returnera detta y till B.

Steg	*Meddelande*	*Kommentarer*
1. A	$\gamma = \alpha^k \bmod p$	$k \in [0, q - 1]$, väljs slumpmässigt och som en-gångs-'commitment'
2. A → B	$<C(A), \gamma>$	Certifikat och vittne
3. B	$ver_{TA}(id(A), v, sig_{TA}(id(A), v)) = true$	Verifiera $C(A)$. Paret $(id(A), v)$ 'hash'-as i praktiken
4. B → A	r	Ett 'challenge' sådant att $1 \le r \le 2^t$
5. A → B	$y = k + ar \bmod q$	Svaret från A
6. B	$\gamma \equiv \alpha^y v^r \pmod p$	Verifieringstest

Tabell 9.3. Identifiering enligt Schnorr

Notera att sannolikheten för att en imitatör ska gissa rätt r är $\approx 2^{-t}$, men att B måste välja ett nytt r vid varje identifiering.

Observera att B verifierar signaturen för A som är utställd av TA.

Värdet a har i princip samma funktion som en *PIN*, dock att a inte exponeras. Vad As smarta kort gör är att det utan att ge ut a ändå visar via värdet y att det vet värdet a ('proof of knowledge').

Det vore naturligtvis möjligt att som publik nyckel välja $v = \alpha^{+a}$. Hur skulle detta påverka resten av protokollet?

Notera att svaret y måste innehålla privat information (k, a) utan att denna avslöjas, att y ska kunna beräknas så effektivt som möjligt och att svaret måste möjliggöra verifiering med endast publikt tillgänglig information. Dessutoom måste naturligtvis utmaningen r ingå.

Det är i skenet av dessa villkor som den enkla konstruktionen $y = k + ar$ ska ses.

iv. Bevis av fullständighet ('completeness') är enkelt och som följer.

$$\alpha^y v^r \equiv \alpha^{k + ar} v^r \equiv \alpha^{k + ar} \alpha^{-ar} \equiv \alpha^k \equiv \gamma \quad \text{(alla led } mod\ p\text{)}$$

Nå, kan nu någon ändå imitera A?

1. Hon kan försöka med ett förfalskat certifikat $C'(A) = <id(A), v', s'>$. Men om signeringen är säker så kommer inte B att verifiera s'.

2. Hon kan göra ett försök med det korrekta certifikatet $C(A) <id(A), v, s>$ eftersom det ju sänds i klartext. Men det går ändå inte eftersom hon måste veta a för att i steg 5. i protokollet kunna svara med rätt y, men y beror av a och att beräkna a från v innebär att beräkna den diskreta logaritmen.

v. Bevis av sundhet ('soundness'). Den egenskap som illustreras av följande sats brukar kallas sundhet.

vi. Sats. Om det existerar värde γ för vilket sannolikheten för imitation via givet protokoll är $\varepsilon \geq 1/2^{t-1}$ så kan a bestämmas inom polynomiell tid.

Detta innebär att om a inte kan bestämmas inom polynoniell tid så är sannolikheten för imitation mindre än ε.

Bevis.

För delen ε av de 2^t möjliga r-värdena kan förfalskaren beräkna ett y som accepteras av B i steg 6.

Eftersom $2^t\varepsilon \geq 2$ så kan två par y_1, y_2, och r_1, r_2 beräknas sådana att

$$y_1 \neq y_2 \ (mod\ q)$$

och

$$\gamma \equiv \alpha^{y_1} v^{r_1} \equiv \alpha^{y_2} v^{r_2} \ (mod\ p).$$

Eftersom $v = \alpha^{-a}$ så gäller då att

$$y_1 - y_2 \equiv a\ (r_1 - r_2)\ (mod\ q). \qquad (*)$$

Men nu är $0 < |\ r_2 - r_1\ | < 2^t$ och $q > 2^t$ ett primtal.

Alltså är $gcd\ (r_1 - r_2, q\ ) = 1$ och a kan lösas ur (*).

vii. Kommentarer

1. Notera att sundhet och fullständighet dock inte innebär "säkerhet". Även om A avslöjade sitt a skulle protokollet fortfarande vara sunt och fullständigt; dock "fullständigt osäkert". Säkerhet är ett mångfacetterat begrepp.

2. Protokollet är som sagt utvecklat för att kunna användas av smarta kort med begränsad beräkningskraft och begränsad lagringsförmåga. Vidare bör det datautbyte som så sker vara av begränsad omfattning.

Ett numeriskt räkneexempel:

> Om *i*d(*A*) och v omfattar 512 bitar och signaturen 320 bitar (som vid tex DSA)
> så blir totala omfattningen av en *C*(*A*) 1344 bitar att lagra på kortet.

Steg 1 och steg 5 omfattar enkla beräkningar modulo m. Antalet bitar som överförs är 1936 fördelade på följande steg:

> steg 2: 1344 + 512 = 1856, steg 4: 40, steg 5: 140.

viii. Digitala signaturer. Som de flesta andra indentifieringsmetoder kan Schnorrs metod transformeras till ett system för digital signering.

Idén är att ersätta verifieraren B med en publik 'hash'-funktion. Så vid signering 'hash'-as ett meddelande m som en *integrerad* del av algoritmen.

Följande tabell visar hur idén utvecklas i detta fall.

Parametrar	**Alfabet**
p ett 512 bits primtal sådant att problemet diskreta logaritmen i Z_p är svårt q ett 160 bits primtal som delar p - 1 $\alpha \in Z_p^*$ vara en q:te rot till 1 modulo p h en hashfunktion med värdeförråd Z_q	$P = Z_p^* \qquad A = Z_p^* \times Z_q$ $K = \{ <p, q, a, \alpha, \beta>:$ $\beta \equiv \alpha^{-a} \pmod p \}$ Publik nyckel: p, q, β och α Privat nyckel: $a, k \in Z_q^*$ $K \in K$
Signering	**Verifiering**
$sig_K(x, k) = <\gamma, \delta>$ $\gamma = \alpha^k \bmod p$ $\delta = k + a\, h(x, \gamma) \bmod q$	$x, \gamma \in Z_p^* \qquad \delta \in Z_q$ $ver_K(x, \gamma, \delta) = true$ $\Leftrightarrow$ $\gamma \equiv \alpha^{\delta}\beta^{h(x, \gamma)} \pmod p$

Tabell 9.4. Schnorrs signaturmetod

I själva verket är detta en generell princip att omvandla ett system för identifiering till ett signatursystem:

Låt en publik 'hash'-funktion ersätta verifieraren.

Notera dock att 'hash'-vädet bör omfatta fler bitar (säg 160) än vad en 'challenge' behöver omfatta (säg 40 bitar). Detta eftersom vi vill undvika att en inkräktare finner kollisioner för 'hash'-funktionen.

För identifiering används ju ett 'challenge' bara för att förhindra en bedragare att i förväg finna ett bra svar.

9.5 Guillous och Quisquaters metod

Denna metod bygger på RSA och består av nedanstående steg. Metodens säkerhet bygger på svårigheten att beräkna en b:te rot modulo ett tal n vars faktorisering är okänd.

i. Grundparametrar. De är här precis som i RSA: $n = p\ q$ och en publik primtals- och publik RSA- exponent b som också fungerar som säkerhetsparameter. Vidare behövs ett signatursystem och en 'hash'-funktion.

ii. Certifikat. Certifikatet $C(A)$ mm till en användare A framställs på följande vis.

1. TA fastställer As identitet manuellt (id-kort edyl) och formar strängen $id(A)$.
2. A väljer en privat och slumpmässigt talet u, $0 \le u \le n$ -1.
3. A $\rightarrow$ TA: $v = (u^{-1})^b \bmod n$. (Den publika nyckeln)
4. TA $\rightarrow$ A: $C(A) = <id(A), v, sig_{TA}(id(A), v)>$ -- certifikatet

iii. Identifieringen (av A inför B) består av följande steg; tabell 9.5.

Steg	*Meddelande*	*Kommentarer*
1. A	$\gamma = k^b \ mod\ n$	$k \in [0, n - 1]$, väljs slumpmässigt och som engångs-'commitment'
2. A → B	$<C(A), \gamma>$	Certifikat och vittne
3. B	$ver_{TA}(id(A), v, sig_{TA}(id(A), v) = true$	Verifiera $C(A)$
4. B → A	r	Ett slumptal sådant att $1 \leq r \leq b - 1$
5. A → B	$y = ku^r \ mod\ n$	Svaret från A
6. B	$\gamma \equiv v^r y^b \ (mod\ n)$	Verifieringstest

Tabell 9.5. Identifiering enligt Guillou-Quisquater

iv. Bevis av fullständighet ('completeness') är (som vanligt) enkelt och som följer.

$$v^r y^b \equiv (u^{-b})^r (ku^r)^b \equiv u^{-br} k^b u^{br} \equiv k^b \equiv \gamma \quad ; \text{alla led } (mod\ n)$$

v. Sats. (Sundhet). Om det existerar värde γ för vilket sannolikheten för imitation via givet protokoll är $\varepsilon \geq 1/b$ så kan u bestämmas inom polynomiell tid.

Bevis.

För något γ kan imitatören beräkna y_1, y_2, r_1 och r_2 med $r_1 \neq r_2$ sådana att

$$\gamma \equiv v^{r_1} y_1^b \equiv v^{r_2} y_2^b \ (mod\ n).$$

Antag, wlog ('without loss of generality'), att $r_1 > r_2$.

Då gäller

$$v^{r_1 - r_2} \equiv (y_2 y_1^{-1})^b \ (mod\ n).$$

Eftersom $0 < r_1 - r_2 < b$ och b är prima så existerar $t = (r_1 - r_2)^{-1}$ och kan beräknas med Euklides algoritm i polynomiell tid.

Vidare är $t(r_1 - r_2) = s\,b + 1$ för något s, varför

$$v^{sb+1} \equiv (y_2 y_1^{-1})^{bt} \ (mod\ n)$$

och

$$v \equiv (y_2 y_1^{-1})^{bt} (v^{-1})^{sb} \ (mod\ n).$$

Genom att upphöja båda sidor med b^{-1} *mod* $\phi(n)$ erhålls

$$u^{-1} \equiv (y_2 y_1^{-1})^t \, (v^{-1})^s \; (mod \; n)$$

och alltså

$$u \equiv (y_1 y_2^{-1})^t \, v^s \; (mod \; n).$$

vi. Identitetsbaserad identifiering. Det intressanta med denna metod är att den kan användas utan certifikat $C(A)$.

I stället för att beräkna ett certifikat tar TA fram följande kvantitet, där a är TA's privata RSA-exponent,

$$u = (h \, (id(A))^{-1})^a \; mod \; n,$$

som returneras till A. Identifiering tillgår nu som följer; observera att $(h^{-ab})^r \equiv h^{-r}$.

Steg	*Meddelande*	*Kommentarer*
1. A	$\gamma = k^b \; mod \; n$	$k \in [0, n - 1]$, väljs slumpmässigt
2. A $\rightarrow$ B	$<id(A), \gamma>$	Identitet och vittne
3. B	$v = h(id(A))$	Beräkna ett 'hash'-värde ($\neq 0$ annars kan C välja $k = 0$ och imitera A)
4. B $\rightarrow$ A	r	Ett slumptal sådant att $1 \leq r \leq b - 1$
5. A $\rightarrow$ B	$y = ku^r \; mod \; n$	Svaret fån A
6. B	$\gamma \equiv v^r y^b \; (mod \; n)$	Verifieringstest

Tabell 9.6. Alternativ identifiering enligt Guillou-Quisquater

Som synes beräknas värdet v ur As *id* och en publik h. Men A (eller en imitatör) måste veta värdet u för att identifieringen ska lyckas.

Detta u kan emellertid endast beräknas av TA (under förutsättningen att RSA är säkert).

Bevisen för sundhet och fullständighet är analoga med tidigare bevis.

vii. Digital signatur. En sådan är också i detta fall möjlig att generera.

viii. Kommentar. Detta system kan (enkelt) generaliseras till tillämpningar med multipla signaturer.

Varje undertecknare A, B, ... tillverkar egna γ_A, γ_B, ... och egna y_A, y_B, ... och protokollet använder

$y = y_A\, y_B$ ... (*mod n*) ,
$\gamma = \gamma_A\, \gamma_B$... (*mod n*) och
id = *id*(*A*) *id*(*B*) ... (*mod n*).

9.6 Feiges, Fiats och Shamirs protokoll

Här antas också att något pålitligt centrum TA står för kortutfärdandet, men att ingen övrig interaktion med detta sedermera krävs.

Metoden bygger på svårigheten att beräkna kvadratrötter modulo ett sammansatt tal vars faktorer är publikt okända.

Metoden bygger på en teknik föreslagen av Goldwasser, Micali och Rackoff, i samband med 'zero knowledge proofs'; se avsnitt 7.6.

Många andra användare A', A'', ... ska lätt kunna anslutas till systemet.

(En tankeväckande anekdot, hämtad ur verkliga livet, som avser 'zero-knowledge-proofs', där man alltså vill bevisa att man besitter en viss kunskap utan att för den skull yppa den, kan härledas från Tartaglia (1499 - 1557).

Historien är den att han hade upptäckt en metod att lösa tredjegradsekvationer. Han ville emellertid inte avslöja metoden för någon. Ett 'challange' bestod i att ge honom en (slumpmässig) tredjegradsekvation. De rötter han svarade med kunde ju enkelt kontrolleras av alla.

Genom att upprepa med många ekvationer, tvingades till slut alla att tro på att Tartaglia verkligen hade en metod. Cardano (1501 - 1576) lyckades till slut övertala Tartaglia att avslöja metoden.

Cardano publicerde den 1545, varvid han hederligt nog angav Tartaglia som upphovsman. Med den ironi som är så vanlig i vetenskapen kallas metoden idag för Cardanos formler.

De har också lite grand att göra med elliptiska kurvor som använts som bas för PKS.)

i. Grundsystemet. TA väljer nu två hemliga primtal p och q och ger ut $n = p\, q$ som publikt och gemensamt för alla kort. Vidare antas TA ha tillgång till en ('hash'-) funktion f som avbildar strängar på intervallet $[0, n)$, $n > 2^{512}$.

ii. Id-strängen I. När en användare A ansöker om ett kort sammanställs uppgifter som namn, adress, behörighet, kortets giltighetstid, id nummer, mm kodade som en sträng $I = id(A)$.

iii. Kortinnehållet. Kortet laddas med I och paren $<s_i, i>$, $i = 1, ...,k$.

Dessa tas fram med följande procedur.

a. $u_j = f(I, j)$ beräknas för ett antal små $j = 1, 2, 3, ...$.

b. k (ca 20) stycken olika j för vilka u_j är en kvadratisk residu modulo n väljs ut. Dessa antas vara (permutera index) $j = 1, 2, ..., k$.

c. Talen s_j tas fram som minsta roten ur u_j^{-1} (*mod n*).

Talen u_j ges ut publikt, medan talen s_j är privata för A[s kort].

iv. Verifiering. Processen sköts av en enhet som innehåller f och n. Idén är att A bevisar kunskap om s_i-värdena för B utan att för den skull avslöja dem.

Följande tabell 8.7 visar stegen. Stegen utförs t gånger med $i = 1, ..., t$.

Bs roll är tämligen passiv men vital: Det är slumpmässigheten i **B** som förhindrar A att fuska vid identifieringen.

Steg	*Meddelande*	*Kommentar*
1. A → B	$y_i = v_i^2 \ mod \ n$ I	v_i slumpade i Z_n^* $I = id(A)$
2. B → A	$\mathbf{B} = b_{i1}, ..., b_{ik}$	slumpbitvektor
3. A → B	$z_i = v_i \Pi_{\{bij=1\}} s_j \ (mod \ n)$	Dolda s_j
4. B kontrollerar	$y_i = z_i^2 \Pi_{\{bij=1\}} u_j \ (mod \ n)$	$u_j = f(I, j)$ (kända av B)

T*abell 9.7. Identifiering enligt Feige-Fiat-Shamir*

v. Bevis av 'completeness'. Om A och B följer protokollet så är kontrollen ok, ty

$$z_i^2 \Pi_{\{bij=1\}} u_j \equiv v_i^2 \Pi_{\{bij=1\}} s_i^2 u_j \equiv v_i^2 \equiv y_i \quad (mod \ n).$$

vi. 'Soundness'. Det gäller att sannolikheten för att B ska utföra identifieringen fel är högst 2^{-kt}.

Det handlar ju om att A försöker sig på bedrägeri genom att gissa v_i eller s_j, dvs k stycken 2-valsmöjligheter i t varv.

Om A inte vet s_j kan hon välja r_i så att hon lurar B om B skickar $e_{ij} = 0$ eller så kan hon välja ett r_i så att hon kan lura B om B sänder $e_{ij} = 1$. A kan inte göra bådadera.

Hon lämnas med chansen 1/2 per komponent per varv. Principen kallas ibland för ett 'cut-and-choose'-protokoll.

vii. Exempel. Låt I = (101 111 101 000 111 00) och n = 35 = 5 * 7 och låt f vara definierad via följande tabell, som också innehåller övriga aktuella värden.

Markeringen * anger om f-värdet är kvadratisk residu modulo 35 eller ej (j/n).

j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
u	2	3	5	6	2	11	4	29	18	5	11	1	10	12	18
*	n	n	n	n	n	j	j	j	n	n	j	j	n	n	n
u^{-1}						16	9	29			16	1			
s						4	3	8			4	1			

På kortet skrivs då följande information: I - 4,6 - 3,7 - 8,8 - 4,11 - 1,12.

Ett exempel på ett varv av sekvensen 1, ..., 4 är:

- A väljer $v_1 = 6$ och skickar därför $y_1 = 1$ till B.
- B skickar $\mathbf{B} = 10111$ till A.
- A returnerar $z_1 = ... = 33$ (mod 35) till B.
- B kontrollerar att $y_1 = 1 = z_1^2 \Pi_B u_j$ (mod 35).

viii. Digital signatur. Protokollet kan modifieras för att åstadkomma en digital signatur. Man ersätter återigen den roll som B spelar med funktionen *h*.

Följande visar kort hur det går till.

Parametrar	**Alfabet**
t, k är tal för säkerhetsnivå $n = p\ q$, där p och q primtal h en 'hash'-funktion med värdeförråd Z_n I en identitetssträng $\in Z_n$ $first_z$ en funktion som tar de första z bitarna	$P = Z_n$ $A = Z_n \times Z_n \times (Z_2)^{t \times k} \times (Z_n)^t$ $K = (Z_n)^k \times (Z_n)^k$ Privat nyckel: $s_1, ..., s_k$ Publik nyckel: $u_1, ..., u_k$ s_j = minsta kvadratroten till u_j^{-1} modulo n
Signering	**Verifiering**
$sig_K(x, \mathbf{v}) = <I, x, \mathbf{B}, \mathbf{z}> = s$ $x \in P$ $y_i = v_i^2 \ mod\ n,\ i = 1, ..., t$ $b_{ij} = first_{t \times k}(h(x, y_1, ..., y_t))$ $z_i = v_i \prod_{bij = 1} s_j$	$ver_K(x, s) = true$ $\Leftrightarrow$ $b_{ij} = first_{t \times k}(h(x, w_1, ..., w_t)$ $w_i = z_i^2 \prod_{bij = 1} u_j \ (mod\ n)$ $u_j = h(I, j)\ j = 1, ..., k$

Tabell 9.8. Feige-Fiat-Shamir-signering

Noter

Fiats och Shamirs nomenklatur är från [FS86]. Schnorrs metod finns beskriven i [Sch91]. Guillous och Quisquaters metod är från [GQ88]och FFS metoden finns i [FFS88].

Övningar

9.1. Många system (inklusive Unix) krypterar lösenord med en envägsfunktion *f* och lagrar *f*(*password*) i stället för *password* i lösenordsfilen. Vid 'log in' beräknas *f* av inmatat lösenord och resultatet jämförs med den lagrade versionen. Mer specifikt konkateneras lösenordet med ett användarspecifikt 12 bitars sk 'salt' *X* så att det i själva verket är *f*(*password* || *X*) som är lagrat. Vad tillför detta *X* i fråga om "säkerhet". Beskriv noga !

9.2. Visa att Guillou-Quisquater-protokollet uppfyller "sundhet" i följande bemärkelse:

Om en imitatör känner ett värde γ för vilket hon har sannolikhet $\varepsilon > 1/b$ att imitera den rätta användaren, så är det möjligt att inom polynomiell tid beräkna det privata värdet *a*.

9.3. I Feiges, Fiats och Shamirs metod bör gälla att $gcd(y_i, n) > 1$. Varför?
Ledning. Betrakta vad som händer om $y_i \equiv 0$.

9.4. Konstruera ett signatursystem utgående från dels från Guillous och Quisquaters identifieringsmetod och dels från Feige-Fiat-Shamirs identifieringsmetod.

9.5. Vilka är nackdelarna med följande "uppenbara" identifieringsmetod baserad på RSA ?

A identifierar sig för B

A → B: e_A e_A är As öppna nyckel

B → A: $e_A(r)$ r ett slumpvärde

A → B: r efter att ha använt sin privata nyckel d_A

9.6. Visa hur identifiering enligt både Feige-Fiat-Shamir och Guillou-Quisquater kan betraktas som specialfall av Ong-Schnorr-identifiering enligt nedan.

Visa också hur metoden kan omvandlas till en signaturmetod.

Ong-Schnorr-identifiering

Parametrar: $n = p\,q$, där p och q är stora primtal.
n är publikt, p och q ges inte ut.
A har privat nyckel $\mathbf{s} = \langle s_1, ..., s_k \rangle$, där $s_j \in Z_n^*$
Publik nyckel är $\mathbf{v} = \langle v_1, ..., v_k \rangle$, där $v_j^{-1} = s_j^m$, $m = 2^t$.

Protokoll:
A → B: $x = r^m$ där $r \in Z_n^*$ och $m = 2^t$.
B → A: $\mathbf{e} = \langle e_1, ..., e_k \rangle$ $\mathbf{e} \in [0, 2^t)^k$, ett 'challenge'
A → B: $y = r \prod_j s_j^{e_j}$
B verifierar att $x = y^m \prod_j v_j^{e_j}$

9.7. Visa hur det är möjligt att "göra en maskerad" vid Schnorr- och vid GQ- identifiering om samma k (eller γ) används mer än en gång av A.

9.8. Betrakta följande tänkbara identifieringsmetod.

A har två privata primtal p och q som uppfyller $p \equiv 3 \pmod 4$ och $q \equiv 3 \pmod 4$ och sätter $n = p\,q$.

Värdet n och id(A) signeras av en TA i As certifikat.

För identifiering av A skickar B en slumpmässig kvadratisk residu modulo n; talet x.

A beräknar och returnerar till B en av de fyra kvadratrötterna y till x. B verifierar att A vet faktoriseringen av n genom att kontroller att $y^2 \equiv x \pmod n$.

Vilken förutsättning bygger metoden på och vilken svaghet är den behäftad med?

9.9. I Okamotos identifieringsmetod som liknar Schnorrs metod används två tal α_1 och α_2 av ordning q och två slumptal k_1 och k_2.

1. A beräknar och skickar $\gamma = \alpha_1^{k1}\ \alpha_2^{\ k2}\ mod\ p$.

2. B utmanar med r.

3. A svarar med

$$y_1 = k_1 + \alpha_1\ r\ mod\ q$$

och

$$y_2 = k_2 + \alpha_2\ r\ mod\ q.$$

a. Hur verifierar B detta?

b. Visa sundheten.