Porazdeljeni informacijski sistemi in celovitost podatkov

Seminarska naloga

Digitalni podpis

Ljubljana, 16.05.2006

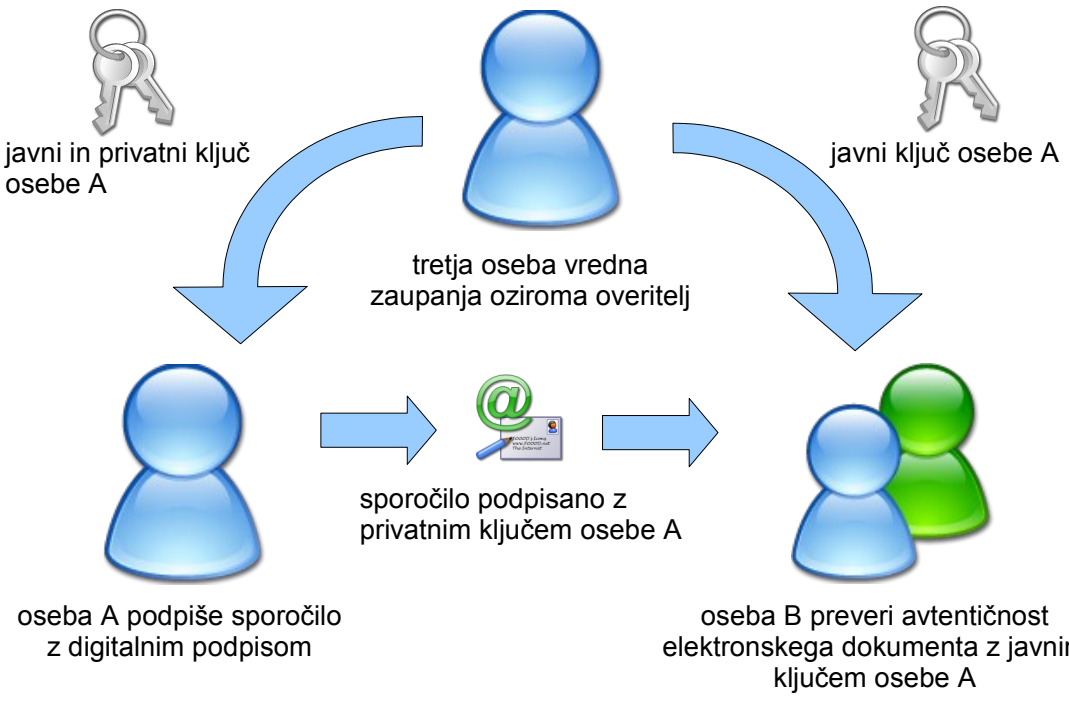
Uroš Palmin

Kazalo vsebine

Kazalo slik

1 Uvod

Šifriranje z javnimi ključi, uporaba javnega in privatnega ključa, oziroma asimetrično šifriranje se je razvilo v sredini sedemdesetih letih, pred tem se je uporabljalo zgolj simetrično šifriranje z uporabo enega samega ključa. Pri asimetričnem šifriranju je en ključ privaten oziroma tajen, drugi pa je javen oziroma javno dostopen. Sistem ne omogoča razkritje privatnega ključa na osnovi poznavanja javnega ključa. Prevzeto lahko trdimo, da je privaten ključ v lasti samo ene osebe, s katerim se ta oseba lahko enolično identificira. S pomočjo javnega ključa, pa lahko preverimo digitalni podpis osebe, ki je elektronski dokument podpisala.
Digitalni podpis lahko na grobo primerjamo z lastnoročnim podpisom s stališča enoličnosti lastnoročnega podpisa, vendar digitalni podpis izkazuje tudi istovetnost elektronskega dokumenta, zato bi lahko rekli, da je digitalni podpis bolj podoben pečatu. Po drugi strani je mogoče digitalni podpis tudi zlorabiti, vsaj za čas do preklica, če pride privatni ključ v roke nepridipravom.



Slika 1: Digitalni podpis

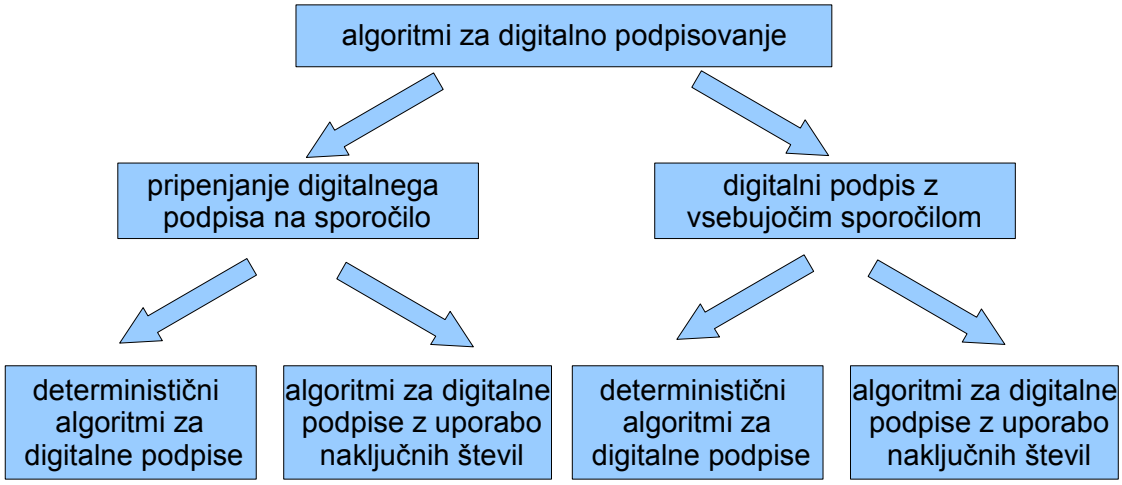
2 Algoritmi za digitalno podpisovanje

V principu ločimo dva univerzalna modela implementacije digitalnega podpisa in sicer:

- Pripenjanje digitalnega podpisa na sporočilo, kjer se za verifikacijo podpisa uporabi originalno sporočilo in pripeti digitalni podpis. Samo sporočilo v tem primeru ostane nespremenjeno.
- Digitalni podpis z vsebujočim sporočilom, kjer je sporočilo že del samega podpisa in je potrebno sporočilo le izluščit iz digitalnega podpisa.

V obeh primerih pri verifikaciji digitalnega podpisa lahko dobimo naključne podatke ali pa deterministične podatke oziroma potrditev avtentičnosti podpisa. Deterministično shemo digitalnega podpisa lahko dalje delimo na:

- enkraten podpis in
- večkratno uporaben podpis.



Slika 2: Pregled algoritmov digitalnega podpisa

2.1 Algoritem digitalnega podpisa z pripenjanjem

Shema digitalnega podpisa z pripenjanjem je najbolj uveljavljena in temelji na zgoščevalnih algoritmih. V nasprotju z shemo z vsebujočim podpisom, ki temelji na redundantnih algoritmih, za katere obstaja večja možnost za razbitje ključa.

2.1.1 Algoritem za generiranje ključev za digitalni podpis s pripenjanjem.

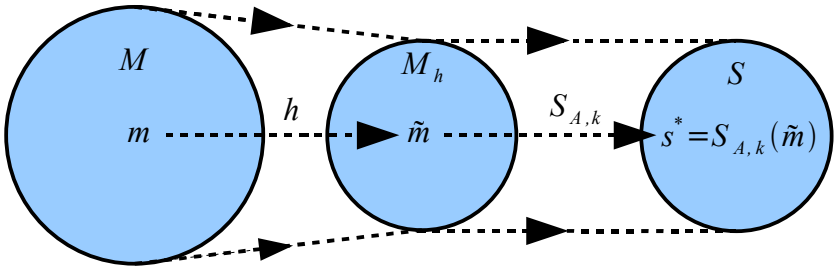
Vsaka oseba generira privaten ključ za podpisovanje sporočila, in njegov komplementarni javni ključ, ki se bo uporabil za verifikacijo digitalnega podpisa.

1. Oseba A izbere privatni ključ, ki definira transformacijo $S_A=\{S_{A,k}: k\in \mathbb{R}\}$. Vsak $S_{A,k}$ je 1-1 preslikava iz M_h v S , kar imenujemo podpisovalna transformacija.
2. S_a definira ustrezno preslikavo V_A iz $M_h \times S$ v da, ne , ki
   $$V_A(\tilde{m}, s^*)=\begin{Bmatrix} da, \check{c}e\, S_{A,k}=s^*, \\ ne,\ v\ vseh\ ostalih\ primerih \end{Bmatrix}$$
   za $\tilde{m}\in M_h, s^*\in S$; kjer je $\tilde{m}=h(m)\ za\ vse\ m\in M$, V_A se imenuje verifikacijska transformacija in je tako zasnovana, da jo je mogoče izračunati brez poznavanja privatnega ključa popisovalca.
3. Javni ključ od osebe A je V_A , privatni pa je nabor S_A

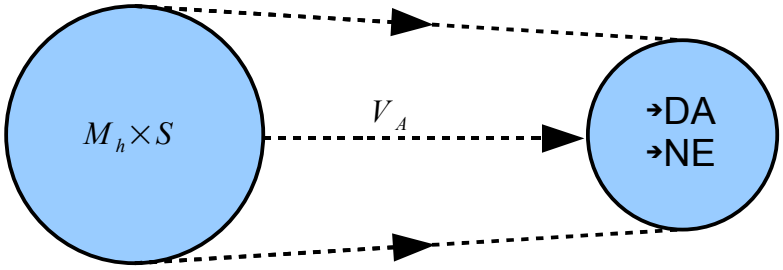
2.1.2 Algoritem digitalnega podpisa s pripenjanjem

V osnovi vsaka oseba, ki hoče digitalno podpisovati sporočila, mora najprej generirati par ključev in sicer privatni ključ s katerim bo oseba podpisovala elektronska sporočila in javni ključ s katerim bodo lahko druge oseba ta podpis preverile.

1. Oseba A, ki želi generirati digitalni podpis mora:
   - izbrati element $k\in \mathbb{R}$,
   - Izračunati $\tilde{m}=h(m)$ in $s^*=S_{A,k}(\tilde{m})$,
   - A-jev podpis za sporočilo m je s^* oba m in s^* sta na voljo, vsem osebam, ki želijo preverit digitalni podpis.
2. Oseba B, ki želi preveriti digitalni podpis mora:
   - pridobiti od osebe A avtentičen ključ V_A ,
   - izračunati $\tilde{m}=h(m)$ in $u=V_A(\tilde{m}, s^*)$,
   - sprejeti digitalni podpis in to samo v primeru, ko je $u=true$.



algoritem za podpisovanje



algoritem za preverjanje

Slika 3: Digitalni podpis z pripenjanjem

Pri algoritmu digitalnega podpisa z pripenjanjem, se uporabljajo enosmerne zgoščevalne funkcije pri tem pa samo dolžina sporočila ni pomembna. Uporaba enosmernih zgoščevalnih algoritmov je priporočljiva metoda za digitalni podpis, saj povečuje varnost algoritma digitalnega podpisa z pripenjanjem.
Na drugi strani je pri algoritmu, kjer sporočilo že vključuje digitalni podpis, omejena dolžina sporočila in že samo generiranje podpisanega sporočila je relativno počasno. Daljša sporočila se v tem primeru razbije na več delov, kar lahko predstavlja potencialno šibkost algoritma.

2.2 Algoritem digitalnega podpisa z vsebujočim sporočilom

Algoritem digitalnega podpisa z vsebujočim podpisom v principu deluje tako, da: oseba, ki sporočilo podpiše, v sporočilo zakodira digitalni podpis, oseba, ki sporočilo sprejme pa iz sporočila z vsebujočim podpisom izlušči sporočilo in preveri avtentičnost. Algoritem se v praksi uporablja za podpisovanje krajših sporočil. Primeri algoritmov so Rabinov in Nyberg-Rueppellov algoritem za digitalen podpis z parom javnega in privatnega ključa.

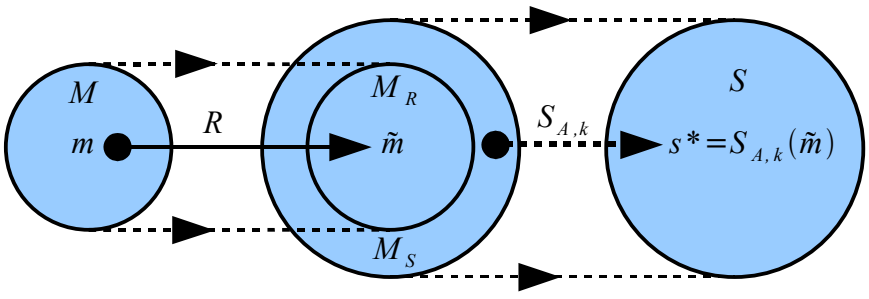
2.2.1 Generiranje ključa pri algoritmu digitalnega podpisa z vsebujočim sporočilom

Vsaka oseba generira privaten ključ za podpisovanje sporočila, in njegov komplementarni javni ključ, ki se bo uporabil za verifikacijo digitalnega podpisa.

1. Oseba A izbere privatni ključ, ki je definiran transformacijo $S_A = \{S_{A,k} : k \in \mathbb{R}\}$ Vsak $S_{A,k}$ je 1-1 preslikava iz M_h v S, kar imenujemo podpisovalna transformacija.
2. S_a, ki definira ustrezno preslikavo V_A iz $M_h \circ S$ identificira M_s za vse $k \in \mathbb{R}$. V_A imenujemo verifikacijska transformacija in je tako zasnova, da jo je mogoče izračunati brez poznavanja privatnega ključa popisovalca.
3. A-jev javni ključ je V_A, privatni pa je nabor S_A

2.2.2 Algoritem za podpisovanje in verifikacijo digitalnega podpisa z vsebovanim sporočilom

1. Oseba A, ki želi generirati digitalni podpis mora:
   - izbrati element $k \in \mathbb{R}$,
   - Izračunati $\tilde{m} = R(m)$ in $s^* = S_{A,k}(\tilde{m})$ pri tem, da je R redundančna funkcija,
   - A -jev podpis je s^* in je na voljo, vsem osebam, ki želijo preverit digitalni podpis in iz njega izluščit sporočilo.
3. Oseba B, ki želi preveriti digitalni podpis mora:
   - pridobiti od osebe A avtentičen ključ V_A,
   - izračunati $\tilde{m} = V_A(s^*)$,
   - preveriti, da je $\tilde{m} \in M_R$ v primeru, da je $\tilde{m} \notin M_R$ zavrniti podpis,
   - izluščiti m iz $\tilde{m}$ s pomočjo $R^{-1}(\tilde{m})$



Slika 4: Digitalni podpis z vsebujočim sporočilom

2.2.2.1 Redundančna funkcija

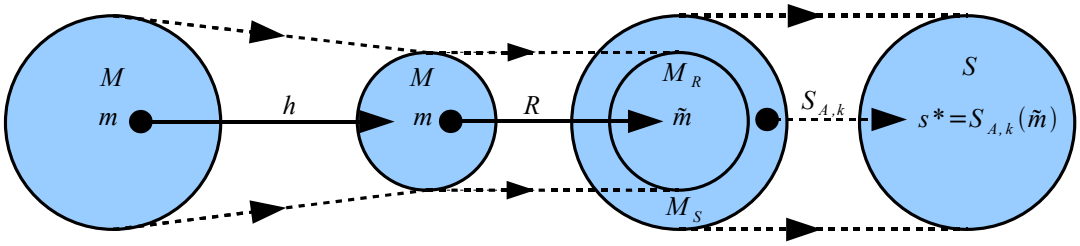
Redundančni funkciji R in R^{-1} sta splošno znani vsem uporabnikom, vendar je od izbire redundančne funkcije odvisna varnost algoritma. Z izbiro napačne redundančne funkcije lahko hitro pride do zloma algoritma oziroma možnosti ponarejanja digitalnega podpisa.
Razlaga redundančne funkcije: Predpostavimo, da velja $M_R = M_S$ in da obstajajo bijektivne preslikave za R in $S_{A,k}$ iz M in M_R ter preslikava M_S v S . Iz tega lahko sklepamo, da M in S imata enako število elementov, potem je mogoče trivialno izluščiti sporočilo m in digitalni podpis s^* , ki ga bo sprejel verifikacijski algoritem, za kateri koli $s^* \in S, V_A(s^*) \in M$.
Verifikacijski algoritem:

- Izbere se naključno število $k \in \mathbb{R}$ in naključno število $s^* \in S$,
- izračuna se $\tilde{m} = V_A(s^*)$ in
- nato se izračuna še $m = R^{-1}(\tilde{m})$

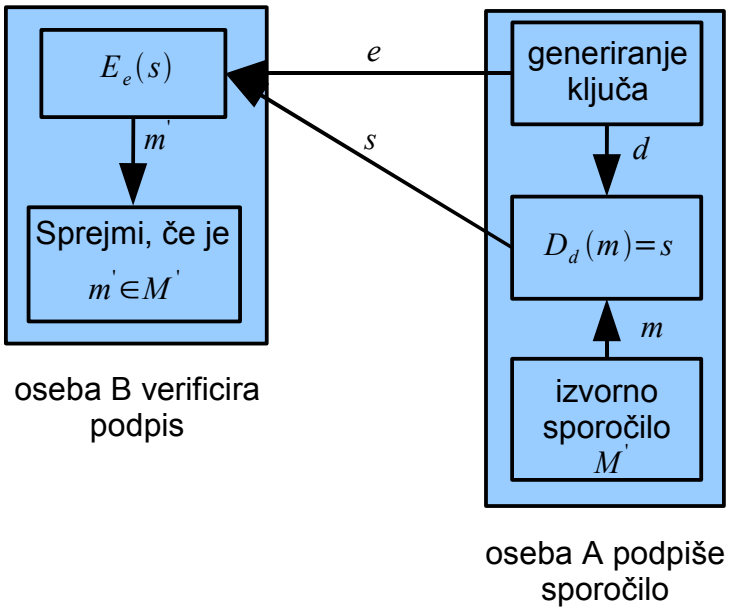
Element s^* je veljaven podpis za sporočilo m in je bil narejen brez poznavanja nabora za podpisovanje S_A .

2.2.2.2 Posebni primer algoritma digitalnega podpisa z vsebujočim sporočilom

To je digitalni podpis z reverzibilnim šifriranjem javnih ključev, uporablja se v distribuciji javnih ključev. Prednost algoritma, je v tem da overitelj pošlje samo digitalni podpis sporočilo, ki ga stranka rabi pa je že vsebovano v podpisu.



Slika 5: Posebni primer algoritma digitalnega podpisa z vsebujočim sporočilom



Slika 6: Primer uporabe digitalnega podpisa z vsebujočim sporočilom v PKI

2.3 Vrste napadov na algoritme digitalnih podpisov

Bistvo napadov na algoritme digitalnih podpisov je, ustvariti digitalni podpis, ki ga je mogoče verificirati. Kriteriji za zlom algoritma digitalnega podpisa:

- Popoln zlom: Do popolnega zloma algoritma pride, ko je napadalec sposoben izračunati privaten ključ podpisovalca ali pa ko najde ekvivalenten algoritem algoritmu podpisovalca.
- Selektiven ponaredek: Do selektivnega ponarejevanja digitalnega podpisa pri pride takrat, ko je napadalec sposoben kreirati digitalni podpis za določeno vrsto oziroma nabor sporočil, pri tem legitimen podpisovalec ne nastopa direktno v procesu.
- Pričujoč ponaredek: Napadalec je sposoben ponarediti digitalen podpis za vsaj eno sporočilo, vendar ne manipulirati s sporočilom, ki ga hoče podpisati, pri tem je legitimen podpisovalec lahko delno vpleten v zlorabo.

Obstajata dve vrsti napadov na algoritme javnih ključev

- Napad na ključ: pri tem napadalec pozna javen ključ legitimnega podpisovalca.
- Napad na sporočilo: Napadalec v tem primeru lahko analizira digitalni podpis znanega ali celo izbranega sporočila.

Napadi na sporočila se delijo še na tri pod kategorije:

- Napad na znana sporočila: Napadalec poseduje nabor sporočil, na katerega nima vpliva.
- Napad na izbrano sporočilo: Napadalec pridobi veljavne digitalne podpise za nabor izbranih sporočil z namenom zloma algoritma. Ta način napada na algoritem digitalnega podpisa, je ne adaptiven, saj napadalec izbere nabor sporočil še pred začetkom napada. Podoben napad je tekstovni napad na izbrani „chiper“ v infrakstrukturi javnih ključev.
- Adaptiven napad znanih sporočil: Napadalec lahko izkorišča legitimnega podpisovalca, katerega javni ključ pozna, za generiranje sporočil, ki jih rabi za razbijanje algoritma.

V principu je adaptiven napad znanih sporočil najtežje preprečit. Razumljivo je, da v primeru takega napada na algoritem digitalnega podpisa v dovolj velikem obsegu lahko pride do zloma algoritma. Vendar v praksi je to težko izvedljivo, vsekakor pa je priporočljivo, da že v algoritmu digitalnega podpisa preprečimo take vrste napadov.
Nivo zaščite algoritma digitalnega podpisa varira od aplikacije do aplikacije. Na primer v primeru, ko obstaja možnost, da bo napadalec izvedel napad na ključ digitalnega podpisa potem moramo pri implemetaciji algoritma paziti da preprečimo napade na izbrana sporočila.
V primeru uporabe enosmernih zgoščevalnih funkcij h v algoritmu digitalnega podpisa potem moramo preprečiti napadalcu možnost izbire ranljive zgoščevalne funkcije.

3 RSA in podobni algoritmi za digitalne podpise

To poglavje obsega RSA digitalne podpise in druge tesno povezane metode. Varnostna politika teh algoritmov v večji meri sloni na zahtevnosti faktorizacije števil. Predstavljeni algoritmi obsegajo tako principe digitalnega podpisa z pripenjanjem, kakor tudi digitalne podpise z vsebujočim sporočilom.

3.1 RSA algoritem za podpis

Zaloga vrednosti RSA šifriranja z javnim ključem je za sporočila in šifrirane podatke $\mathbb{Z}=\{0,1,2,...,n-1\}$ kjer je $n=pq$ oziroma produkt dveh naključno izbranih praštevil. Ker je preslikava bijektivna lahko naredimo digitalni podpis z zamenjavo vlog šifriranja in dešifriranja. RSA je deterministične algoritem, ki omogoča digitalne podpise z vsebujočim sporočilom. Prostor za podpisovanje M_S in prostor digitalnega podpisa S pripadata $\mathbb{Z}_n$. Redundačna funkcija $R:M\rightarrow\mathbb{Z}_n$ je izbrana izmed nabora javnih funkcij.

3.1.1 RSA Algoritem za generiranje ključev

Vsaka oseba mora kreirati svoj par javnega in privatnega ključa, oseba A naredi sledeče:

1. Naključno generira dve veliki praštevili p in q približno enakih velikosti,
2. izračuna produkt $n = pq$ in $\Phi = (p-1)(q-1)$,
3. izbere poljubno celo število e , kjer mora veljati $1 < e < \Phi$ in $gcd(e, \Phi) = 1$,
4. uporabi razširjen Euclidean algoritem za izračun unikatnega celega števila d , $1 < d < \Phi$, tako da velja $ed = 1 (mod\ \Phi)$,
5. javni ključ od osebe A je (n, e) , privatni ključ pa je d .

3.1.2 RSA algoritem za generiranje in verifikacijo digitalnega podpisa

Oseba A podpiše sporočilo $m \in M$ z privatnim ključem, katera koli druga oseba B lahko verificira A-jev podpis in izlušči sporočilo iz digitalnega podpisa spomočjo javnega ključa.

1. Za generiranje podpisa mora oseba A:
   - izračunati $\tilde{m} = R(m)$ oziroma celo število v obsegu $[0, n-1]$,
   - izračunati $s = \tilde{m}^d \bmod n$,
   - iz česar sledi, da je A-jev podpis s za sporočilo m .
2. Za verifikacijo podpisa osebe A in rekontrukcijo sporočila m mora oseba B narediti sledeče:
   - pridobiti avtentičen A-jev javni ključ (n, e) ,
   - Izračunati $\tilde{m} = ^e \bmod n$,
   - preveriti da je $\tilde{m} \in M_R$, če ni zavrne podpis,
   - izlušči sporočilo $m = R^{-1}(\tilde{m})$.

Dokaz, da verifikacija podpisa res deluje: Če je s podpis sporočila m , potem je $s \equiv \tilde{m}^d \bmod n$, kjer je $\tilde{m} = R(m)$. Ker je $ed \equiv 1 (mod\ \Phi)$, $s^e \equiv \tilde{m}^{ed} \equiv \tilde{m} (mod\ n)$ ter $R^{-1}(\tilde{m}) = R^{-1}(R(m) = m)$.

3.1.3 Možni napadi na RSA algoritem

En izmed možnih napadov na RSA algoritem je faktorizacija celih števil. Če napadalcu uspe faktorizirati javni modul n osebe A, potem napadalec lahko izračuna Φ in nato z razširjenim Euclidian algoritmom izračuna privatni ključ d iz Φ ter javni eksponent e z rešitvijo enačbe $ed \equiv 1 (mod\ \Phi)$. S tem prizadene

zlom celotnega algoritma. Da se oseba A zaščiti pred takimi napadi mora izbrati p in q tako, da je faktorizacija računsko nemogoča.

3.1.4 Multiplikativna lastnost RSA algoritma

RSA algoritem digitalnega podpisa ima sledeče multiplikativne lastnosti: Če sta $s_1 = m_1^d \bmod n$ in $s_2 = m_2^d \bmod n$ podpisa sporočil m_1 in m_2 oziroma sporočil z redundanco, potem $s = s_1 s_2 \bmod n$ ima lastnost, da $s = (m_1 m_2)^d \bmod n$. če ima $m = m_1 m_2$ zadostno redundanco (primer, da je $e \in M_R$), potem je s veljaven podpis. Pri tem je še pomembno, da redundančna funkcija R nima multiplikativnih lastnosti, predvsem za pare $a, b \in M$ mora veljati,da $R(a \cdot b) \neq R(a) R(b)$. Ne multiplikativna lastnost redundančne funkcije R je potreben pogoj za varnost ne pa tudi zadosten pogoj.

3.1.5 Računske lastnosti lastnosti RSA algoritma za generiranje in verifikacijo podpisa

Predpostavimo, da je $n = pq$ 2k-bit RSA modul, kjer sta p in q k-bitna cela praštevila. Za izračun podpisa $s = m^d \bmod n$ kjer je sporočilo m porabimo Ok^3 bitnih operacij za modularno množenje in modularno potenciranje. Ker podpisovalec pozna p in q , lahko izračuna $s_1 = m^d \bmod q$, $s_2 = m^d \bmod q$ in določi podpis s z uporabo Kitajskega teorema z ostankom. Kljub temu, da kompleksnost izračuna ostane $O(k^3)$, se učinkovitost v določenih primerih drastično poveča.

Verifikacija digitalnega podpisa je precej hitrejša od podpisa, če je javni eksponent majhno število. V takem primeru, ko je eksponent majhno število, verifikacija porabi $O(k^2)$ bitnih operacij. Predlagane vrednosti eksponenta e so v praksi 3 ali $2^{16}+1$ seveda p in q morata biti izbrana tako, da ustrezata pogoju $gcd(e, (p-1)(q-1)) = 1$.

Iz tega lahko zaključimo, da je RSA podpisovalni algoritem primernejši za situacije, kjer se v glavnem digitalni podpis večkrat preverja. Recimo v šifrirnih algoritmih z javnimi ključi, kjer se podpis enkrat generira in mnogokrat verificira s strani uporabnikov, ki želijo preveriti avtentičnost osebe oziroma sporočila.

3.2 Rabinov algoritem podpisovanja z javnim ključem.

Rabinov algoritem podpisovanja z javnim ključem je podoben RSA algoritmu, vendar uporablja tudi javni eksponent e . da poenostavimo razlago bomo predpostavili, da je eksponent $e = 2$. Zaloga vrednosti za M_S je Q_S (nabor kvadratičnih residujev po modulu n), zaloga vrednosti podpisa pa so koreni teh vrednosti. Redundačna funkcija R ki preslika M v M_S je javno poznana.

3.2.1 Rabinov algoritem za generiranje Rabinovega javnega ključa

Vsaka oseba mora generirati par javnega in privatnega ključa. Oseba A mora za generiranje ključa narediti naslednje:

1. Generirati dve naključni praštevili p in q približno enake velikosti,
2. izračunati $n = pq$,
3. javen ključ osebe A je n , privaten ključ pa je (p, q) .

3.2.2 Rabinov algoritem za generiranje Rabinovega podpisa in njegovo verifikacijo

Oseba A podpiše sporočilo $m \in M$, katera koli oseba B lahko preveri A-jev digitalni podpis in iz njega izlušči sporočilo.

1. Za generiranje podpisa, mora oseba A narediti naslednje:
   - izračunati $\tilde{m} = R(m)$,
   - izračunati kvadratni koren s od $\tilde{m} \bmod n$,
   - iz tega sledi, da je A-jev podpis s za sporočilo m .
   -
   - pridobiti A-jev javni ljluč n ,

2. Za verifikacijo A-jevega podpisa s in iz luščenje sporočila m mora oseba B:
   - izračunati $\tilde{m} = s^2 \bmod n$,
   - preveriti, da velja $\tilde{m} \in M_R$, drugače zavrniti podpis,
   - izluščiti sporočilo $m = R^{-1}$

Podobno kot pri RSA algoritmu, je tudi pri Rabin algoritmu potrebno pravilno izbrati redundančno funkcijo R , ki je ključna za varnost Rabin algoritma. Pri algoritmu digitalnega podpisa z vključenim sporočilom nastopa zaloga vrednosti M v fiksnih dolžinah, zaradi česar se izbira redundačne funkcije dodatno oteži.

3.2.3 Modificiran Rabin algoritem za generiranje podpisa in njegovo verifikacijo

Z namenom, da bi premostili problematiko dolžine sporočil in izbire prave redundančne funkcije, je bilo potrebno Rabinov algoritem prilagoditi na podoben način kot ISO/IEC9796.

3.2.4 Računske lastnosti lastnosti Rabinovega algoritma za generiranje in verifikacijo podpisa

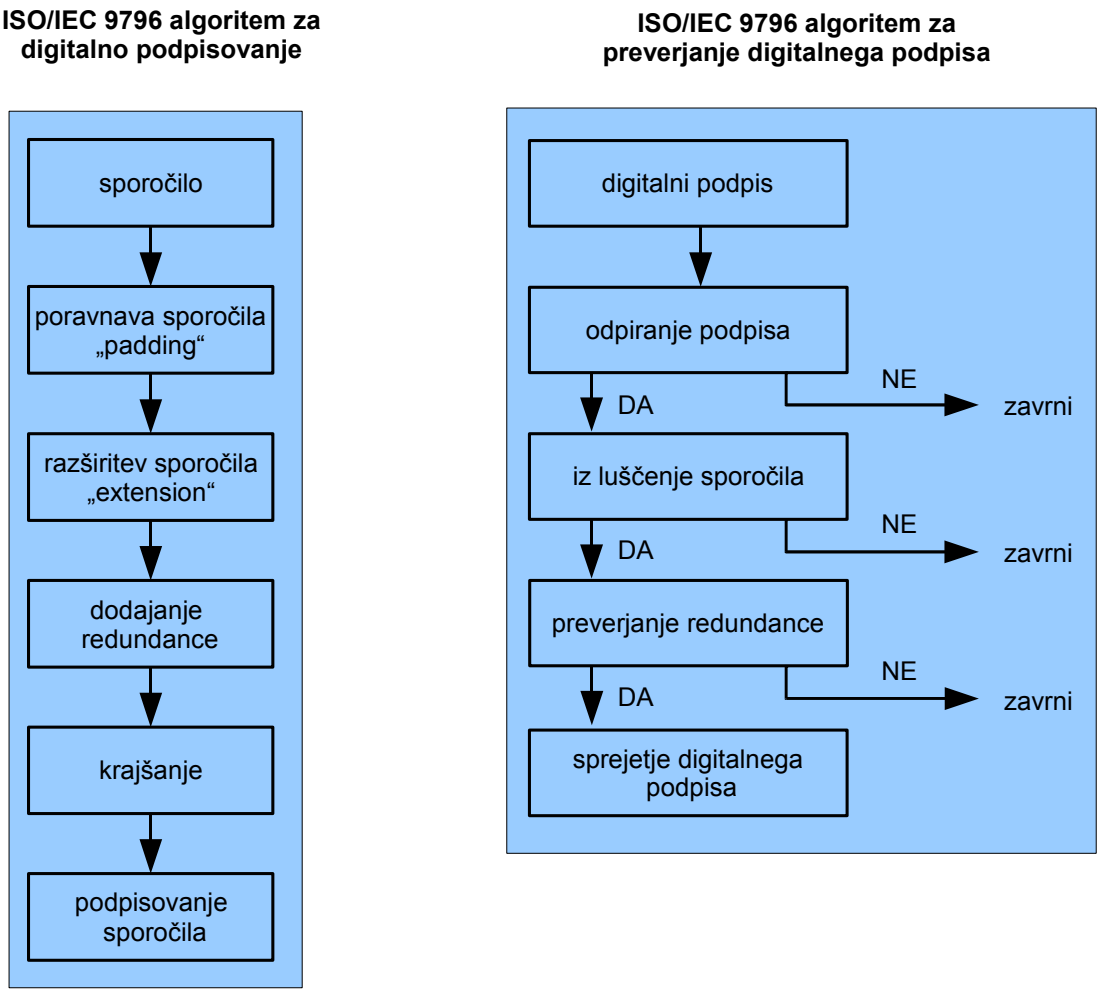
Za generiranje podpisa Rabinov algoritem je mogoče malo zahtevnejši od RSA algoritma, je pa dosti hitrejši za verifikacijo podpisa. Optimalno deluje pri eksponentu $e=2$, vendar tudi pri eksponentu $e=3$ v je primerjavi z RSA algoritmu, dosti hitrejši.

3.3 ISO/IEC 9796 standard

ISO/IEC 9796 je Mednarodna Organizacija za Standardizacijo (ISO) prvič objavila leta 1991, kot prvi standard na področju digitalnih podpisov. Standard opisuje proces digitalnega podpisovanja z sporočilom vsebovanim v podpisu.

Glavne karakteristike ISO/IEC 9796 so:

- da temelji na šifriranju z javnimi ključi,
- sam standard ne predpisuje šifrirnega algoritma, zahteva pa, da algoritem ob podpisovanju obdrži enako dolžino sporočila,
- se uporablja za podpisovanje sporočil fiksne dolžine,
- ne uporablja zgoščevalnih h funkcij,
- sporočilo se integrira v podpis,
- predpisuje poravnavo (na oktete) sporočila, kjer je to potrebno,
- vsebuje primere implementacije RSA algoritma in modificiranega Rabinovega algoritma.
- Specifične metode, ki so uporabljene za: poravnavo, redundanco in okrajševanje ISO/IEC 9796 onemogoči določene vrste napadov na algoritem.



Slika 7: Postopek podpisovanja po ISO/IEC 9769

3.3.1 Postopek podpisovanja po ISO/IEC 9796 standardu

Sam postopek podpisovanja se sestoji iz petih korakov, kakor lahko vidimo na sliki 7. Po korakih, sam podpis pa poteka takole:

1. Sporočilo m se poravna na poravnano sporočilo $MP = 0^{r-1} || m$, kjer je $1 \leqslant r \leqslant 8$ oziroma dolžina poravnanega sporočila MP je mnogokratnik števila osem.
2. Razširjeno sporočilo se pridobi iz poravnanega sporočila MP z ponavljanjem strnjevanja sporočila MP s samim seboj, dokler ni dosežena želena dolžina t $ME = ME_t || ME_{(t-1)} || ... || ME_2 || ME_1$ vsak ME predstavlja oktet. Če dolžina t ni mnogokratnik z , potem so zadnji okteti, ki se strnjujejo deli sporočila MP , nahajajo pa se v sporočilu MP kot zaporedni okteti z leve strani. Natančneje $ME_{i+1} = m_{(i \bmod z)+1}$ za $0 \leqslant i \leqslant t-1$
3. Redundanca je dodana v sporočilo ME tako, da dobimo niz MR dobimo z prepletanjem oktetov sporočila ME z redundančnimi okteti, na koncu še prilagodimo MR_{2z} oktet nastalega sporočila. Natančneje $MR_{2i} - 1 = ME_i$ in

$MR_{2i}=S(ME_i)$ za $1\leqslant i\leqslant t$, kjer se $S(u)$ imenuje senčna funkcija okteta u . Če $u=u_2\|u_1$ kjer u_1 in u_2 sta dolgi štiri bite, potem $S(u)=\pi(u_2)\|\pi(u_1)$, kjer je π permutacija

$$\pi=\begin{Bmatrix}0123456778ABCDEF,\\E358942F0DB67AC1\end{Bmatrix}$$

Za boljšo predstavo so π vrednosti predstavljene z heksadecimalnimi znaki.

4. Iz sporočila MR se konstruira k bitno celo število IR po postopku:
   - k najmanj pomembnemu bitu $k-1$ sporočila MR se z leve pripne dodaten bit z vrednostjo ena,
   - zamenja se najmanj pomemben oktet $u_1\|u_2$ z $u_1\|0110$
5. Za podpisovanje se uporabi podpisni algoritem, ki preslika sporočilo z enakim številom bitov in omogoča iz luščenje sporočila. IR je podpisan z izbranim algoritmom, ki poda podpis s .

ISO/IEC 9796 algoritem je bil napisan za uporabo z RSA in Rabinovim algoritmom. Za ta dva primera je generiranje podpisa tudi bolj detajlno opisano.

3.3.2 Postopek postopek preverjanja podpis po ISO/IEC 9796 standardu

Verifikacijo podpisa po standardu ISO/IEC9796 se deli na tri stopnje in sicer:

1. Pri odpiranju podpisa s se:
   - uporabi javna transformacija na podpisu s , da dobimo IR' ,
   - se zavrne podpis, če je napačne dolžine oziroma, če prvi najbolj pomemben bit ni 1 in najmanj pomemben zlog ni 0110.

2. Sporočilo MR' se izlušči iz niza IR' v naslednjih korakih:
   - X je niz $k-1$ najmanj pomembnih bitov IR' ,
   - če so $u_4\|u_3\|u_2\|0110$ najmanj pomembni zlogi od X , potem se zamenja najmanj pomemben oktet X z $\pi^{-1}(u_4)\|u_2$,
   - MR' se izlušči iz X s pomočjo združevanj ničel, tako da rezultira dolžino $2t$ oktetov.

   Vrednosti z in r se izračunajo po naslednjem postopku:
   - Iz $2t$ oktetov MR' se izračuna $MR'_{2i}\oplus S(MR'_{2i-1}),\ 1\leqslant i\leqslant t$, v primeru, da so vse vsote enake 0 se podpis zvrne.
   - spremenljivki z priredimo najmanjšo vrednost i za katerega velja $MR'\oplus S(MR'_{2i-1})\neq 0$
   - r je najmanj pomemben zlog vsote izračunane v prejšnjem koraku. Podpis zavrnemo, če r nima vrednosti med 1 in 8.

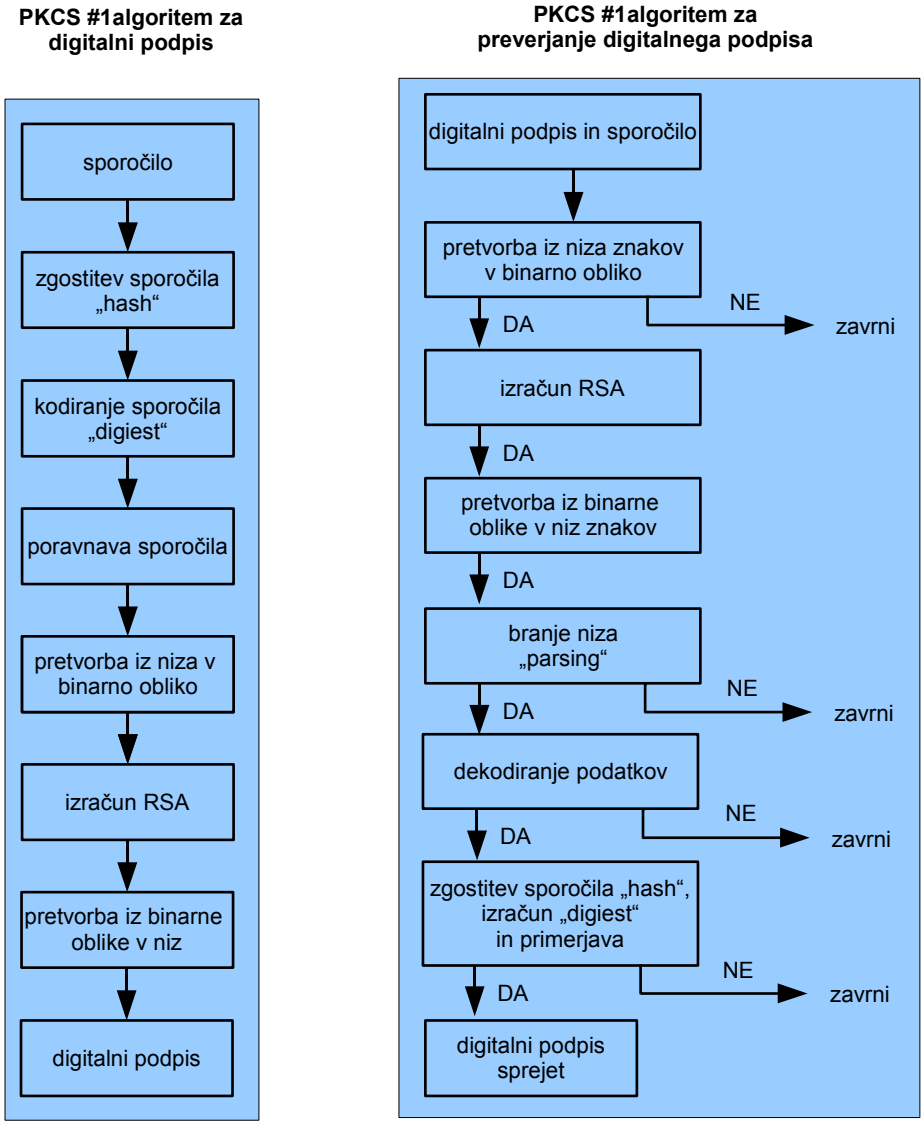
   Iz MR' se sestavi MP' dolžine z zlogov po sledečem postopku:
   - $MP'=MR'_{(2i-1)}$ za $1\leqslant i\leqslant z$,
   - podpis se zavrne, če najbolj pomembni biti $r-1$ sporočila MP' niso vsi enaki 0,
   - Izračunamo M' , ki je $8z-r+1$ najmanj pomembnih bitov MP' .

3. V zadnjem koraku preverimo redundanco podpisa po sledečem postopku:

- Iz M' sestavimo MR'' z uporabo, razširjanja, redundance in ostale postopke, ki so bili uporabljeni pri podpisovanju.
- Podpis se sprejme samo v primeru, da so najmanj pomembni $k-1$ biti MR'' enaki $k-1$ najbolj pomembnim bitom MR'

3.4 PKSC #1 algoritem

Javni šifrirni standardi šifriranja z javnimi ključi oziroma PKCS (Public-key Cryptography Standard) predstavljajo nabor specifikacij, ki vključujejo tudi RSA šifriranje in digitalno podpisovanje.
PKCS algoritem je algoritem digitalnega podpisovanja z pripenjanjem podpisa na sporočilo, zato ta algoritem za svoje delovanje potrebuje zgoščevalno funkcijo, kot so na primer MD2 in MD5.



Slika 8: Shematski prikaz podpisovanj in verifikacije podpisa PKCS algoritma

3.4.1 PKCS algoritem za podpisovanje

Shematski prikaz PKCS algoritma za podpis lahko vidimo na sliki 8 natančnejši opis pa lahko razdelimo na šest korakov, ki predstavljajo:

1. Z *„digest“* zgoščevalno funkcijo se iz sporočila M dobi MD .
2. Dobljeni MD in identifikator zgoščevalnega algoritma sta združena v ASN.1 (abstract syntax notation) in nato še kodirana z BER (basic encoding rules). Rezultat vseh operacij je oktetni niz D
3. Podatke v nizu D uredimo, tako da dobimo oktetni niz EB
4. Oktetni niz pretvorimo v niz celih števil. Če oktete EB predstavimo z $EB_1 \| EB_2 \| ... \| EB_k$, definiramo $\widetilde{EB_i}$ kot celo številčno obliko EB_i . Celo številčna oblika EB pa je $\sum_{i=1}^{k} 2^{8(k-i)} \widetilde{EB_i}$
5. Izračun RSA $s = m^d \bmod n$.
6. Niz oktetov s se pretvori v niz znakov $ED = ED_1 \| ED_2 \| ... \| EB_k$, kjer okteti ED_i izpolnjujejo pogoj $s = \sum_{i=1}^{k} 2^{8(k-i)} \widetilde{ED_i}$. Podpis pa je $S = ED$

PKCS shemo preverjanje podpisa lahko vidimo na sliki 8 poteka v sledečih korakih:

1. Pretvorba S iz niza znakov v s niz celih števil, zavrnitev podpisa v primeru, da dolžina S ni mnogokratnik števila 8 in v primeru, da je $s > n$,
2. izračun RSA $m = s^e \bmod n$,
3. pretvorba m v niz znakov EB dolžine k oktetov.
4. Pretvorba EB v BT oziroma v niz PS in niz podatkov D . Podpis se zavrne v primerih: ko pretvorba na EB ni možna, BT ni ne 00 in ne 01, ter ko je $PS < 8$ oziroma ni konsistenten z BT
5. z BER dešifrirno funkcijo se dekodira D , da se dobi MD in identifikator zgoščevalnega algoritma. V primeru napačnega identifikatorja zgoščevalnega algoritma se podpis zvrne.
6. Izračuna se „digest“ sporočila MD' , ter primerja z MD pridobljenim iz podpisa. Podpis S se sprejme za sporočilo M samo v primeru, ko velja $MD' = MD$

3.5 Fiat-Shamirjev algoritem

Fiat-Shamirjev algoritem je prav tako kot PKCS algoritem z pripenjanjem digitalnega podpisa na sporočila. Algoritem uporablja enosmerno zgoščevalno funkcijo za preverjanje avtentičnosti sporočila. Za razliko od RSA podpisnega algoritma, je za Fiat-shamirrjev algoritem potrebna tretja oseba, ki generira praštevili p in q ter javni in privatni ključ za vsako osebo. Prednost tega algoritma je v tem, da je varen tudi proti adaptivnimi napadi na sporočila.
Podpisni algoritem je zelo hiter, saj se izvede le 6% računskih operacij, ki bi jih potrebovali za RSA algoritem. Iz tega stališča je Fiat-Shamirjev algoritem primeren za aplikacije, ker je potrebno generirati veliko število podpisov z predpostavko, da poraba večjega pomnilniškega prostora ni problematična.

3.6 GQ algoritem

Guillou-Quisquater protokol za identifikacijo se da prilagoditi za digitalno podpisovanje z zamenjavo zgoščevalne funkcije z enosmerno zgoščevalno funkcijo. Zahtevnostno je GQ algoritem podoben Fiat-Shamirevem algoritmu, saj je precej hitrejši od RSA in malo počasnejši od Fiat-Shamirjevega algoritma. Prednost pa je je v tem, da zasede manj pomnilniškega prostora kot Fiat-Shamirjev algoritem.

4 DSA in podobni algoritmi

Vse metode digitalnega podpisovanja v tem poglavju so metode digitalnega podpisovanja z uporabo naključnih števil. Varnost DSA algoritmov, podobno kot pri RSA algoritmih temelji na računski zahtevnosti računanja logaritmov in Z_p^* . Računska zahtevnost je sicer nujen pogoj za varnost, vendar do sedaj ni dokazov, da sta RSA in DSA algoritma varna.

4.1 DSA algoritem za digitalni podpis

DSA algoritem je bil napisan leta 1991 in je bil v Ameriki sprejet kot standard za digitalno podpisovanje oziroma DSS (Digital Signature Standard). Algoritem je izpeljanka ElGamallovaga algoritma za digitalni podpis z pripenjanjem digitalnega podpisa na sporočilo. Algoritem za svoje delovanje potrebuje zgoščevalno funkcijo h in neko naključno celo število q .

4.1.1 DSA algoritem za generiranje ključa

Vsaka oseba A generira par privatnega in javnega ključa. Za generiranje ključa mora oseba A narediti sledeče:

1. izbrati praštevilo q , ki ustreza pogoju $2^{159} < q < 2^{160}$,
2. izbrati t tako, da bo izpolnjen pogoj $0 \leqslant t \leqslant 8$ ter izbrati praštevilo p , ki izpolnjuje pogoj $2^{511+64t} < p < 2^{512+64t}$ in je delitelj $(p-1)$,
3. izbere element $g \in \mathbb{Z}_p^*$ in izračuna $\alpha = g^{(p-1)/q} \bmod p$, če je $\alpha = 1$ postopek ponovi,
4. izbere naključno celo število a , ki zadostuje pogoju $1 \leqslant a \leqslant q-1$,
5. izračuna $y = \alpha^a \bmod p$,
6. iz tega dobi javni ključ (p, q, α, y) in privatni ključ a .

Pri izbiri števil p in q je pomembno, da je najprej izbrano praštevilo q in šele nato praštevilo p , ki je delitelj $(p-1)$

4.1.2 DSA algoritem za generiranje in verifikacijo podpisa

Oseba A podpiše binarno sporočilo m poljubne dolžine, katera koli oseba B lahko podpis preveri z uporabo javnega ključa osebe A.

1. Za generiranje podpisa mora oseba A narediti sledeče:
   - izbrati naključno celo število k , ki je $0<k<q$,
   - izračunati $r=(\alpha^k \bmod p) \bmod q$,
   - izračunati $k^{-1} \bmod q$,
   - izračunati $s=k^{(-1)}[h(m)+ar] \bmod q$,
   - iz tega dobi digitalni podpis (r,s) za sporočilo m
2. za preverjanje podpisa (r,s) na sporočilu m mora oseba B narediti sledeče:
   - pridobiti A-jev javni ključ (p,q,α,y) ,
   - preveriti, da $0<r<q$ in $0<s<q$, če ni zavrne podpis,
   - izračuna $w=s^{-1} \bmod q$ in $h(m)$,
   - izračuna $u_1=w \cdot h(m) \bmod q$ in $u_2=rw \bmod q$,
   - izračuna $v=(\alpha^{u_1} y^{u_2} \bmod p) \bmod q$,
   - sprejme podpis, če je izpolnjen pogoj $v=r$

DSA algoritem temelji na znani problematiki računske zahtevnosti diskretnih logaritmov. Prvi logaritemski problem je v prostoru $\mathbb{Z}_p^*$, kjer nastopa zahtevno računanje krožnih indeksov, drugi logaritemski problem ciklično pod grupiranje reda q .

4.2 ECDSA algoritem za digitalni podpis

ECDSA algoritem je izpeljanka DSA algoritma, ki temelji na uporabi eliptičnih krivulj. Prednost tega algoritma je, da so ECDSA digitalni podpisi po velikosti manjši od DSA pri isti stopnji varnosti. Druga prednost ECDSA algoritma je da porabi za enako število operacij kot DSA za isto dolžino ključa, pri čemer moramo upoštevati, da ista dolžina ključa ECDSA v primerjavi za DSA zagotavlja precej veečjo stopnjo varnosti.

4.2.1 ECDSA algoritem za generiranje ključa

Vsaka oseba mora generirati par ključev javnega in privatnega. Postopek generiraja ključa je nasledeni:

1. Izbrani morajo biti prametri krivulj (q, FR, a, b, G, n, h)
2. izbrati privatni ključ d_A , ki je naključno izbrano celo število na intervalu $1<d_A<(n-1)$ in
3. javni ključ Q_A , ki je $Q_A=d_{AG}$

4.2.2 ECDSA algoritem za generiranje podpisa

Oseba A, ki hoče podpisati sporočilo m mora narediti sledeče:

1. izračunati e , ki je rezultat zgoščevalnega algoritma SHA-1,
2. izbrati naključno celo število k iz območja $1<k<(n-1)$,
3. izračunati $r=x_1(mod\, n)$, kjer $(x_{1,}\, y_1)=kG$, če je $r=0$ se vrne na drugi korak,
4. izračuna $s=k^{-1}(e+d_A r)(mod\, n)$, če je $s=0$ se vrne na drugi korak,
5. dobljeni podpis je par $(r\,,s)$.

4.2.3 ECDSA algoritem za verifikacijo podpisa

Oseba B mora pridobiti javni ključ Q_A od osebe, digitalni podpis pa verificira po sledečem postopku:

1. preveri, če sta števili r in s v območju $1<(r\,,s)<(n-1)$,
2. izračuna e , ki je rezultat zgoščevalnega algoritma SHA-1,
3. izračuna $w=s^{-1}(mod\, n)$,
4. izračuna $u_1=ew(mod\, n)$ in $u_2=rw(mod\, n)$,
5. izračuna $(x_{1,}\, y_1)=u_{1G}+u_2 Q_A$,
6. podpis je verificiran samo takrat, ko velja $x_1=r(mod\, n)$.

4.2.4 Performančne lastnosti ECDSA algoritma

Performančne lastnosti ECDSA algoritma, kot že omenjeno, so precej boljše od DSA algoritma, druga prednost pa je manjša dolžina ključa. ECDSA je trenutno en perspektivnejših algoritmov na področju digitalnih podpisov.

4.3 ElGamal algoritem za digitalni podpis

ElGamalov algoritem za digitalno podpisovanje je algoritem z uporabo naključnih števil ter pripenjanjem digitalnega podpisa na sporočilo Za ugotavljanje avtentičnosti sporočilo uporablja enosmerne zgoščevalne funkcije, rezultat katerih se vključi v pripeti digitalni podpis. Z AlGamalovim algoritmom lahko podpisujemo binarna sporočila poljubnih dolžin. Kakor je že bilo omenjeno RSA je samo ena vrsta ElGamalovega podpisa.

4.3.1 AlGamalov algoritem za generiranje ključev

Vsaka oseba A generira svoj par ključev, javnega in privatnega, do česar pride v naslednjih korakih:

1. generira veliko naključno praštevilo p in konstanto α za algoritem $\mathbb{Z}_p^*$,
2. izbere naključno število a , ki ustreza pogoju $1\leqslant a\leqslant p\leqslant p-2$,
3. izračuna $y=\alpha^a\, mod\ p$

4. iz tega sledi, da je A-jev javni ključ (p, α, y) privatni pa a .

4.3.2 AlGamalov algoritem za podpisovanje in preverjanje avtentičnosti podpisa

Oseba A je podpisala binarno sporočilo m poljubne dolžine, katera koli oseba B lahko preveri avtentičnost tega podpisa z uporabo A-jevega javnega ključa.

1. Za generiranje podpisa mora oseba A narediti sledeče:
   - izbrati naključno celo število k , ki ustreza pogoju $1 \leqslant k \leqslant p \leqslant p-2$, katerega $gcd(k, p-1)=1$,
   - izračunati $r=\alpha^k \bmod p$,
   - izračunati $k^{-1} \bmod (p-1)$,
   - izračunati $s=k^{-1}\{h(m)-ar\} \bmod (p-1)$
   - iz tega sledi A-jev podpis za sporočilo m je par (r, s) .
2. Preverjanje A-jevega podpisa (r, s) pa poteka po sledečih korakih:
   - pridobiti je potrebno A-jev javni ključ (p, α, y) ,
   - izračunati $v_1 = {}^r r^s \bmod p$,
   - izračunati $h(m)$ in $v_2=\alpha^{h(m)} \bmod p$ in
   - nazadnje sprejeti podpis samo v primeru, ko je $v_1=v_2$.

4.3.3 Varnost AlGamalovega algoritma

Za razbitje podpisa na sporočilu m mora napadalec izbrati naključno število k in izračunati $r=\alpha^k \bmod p$. Če je diskretni logaritem nemogoč za izračun, potem lahko napadalec izbira le naključne parametre s , verjetnost da bo naključno izbran parameter s pravi je p^{-1} , kar je zanemarljivo za velike vrednosti p .
Za povečanje varnosti algoritma, mora oseba A, ki podpisuje sporočilo, za vsak podpis izbrati nov parameter k , sicer se verjetnost, da bi napadalec generial pravi ključ poveča.
Uporaba enosmerne zgoščevalne funkcije h za digitalni podpis $s=k^{-1}\{m-ar\} \bmod (p-1)$ je nujna, sicer lahko napadalec z izbiro katerega koli para (u, v) , ki zadostuje pogoju $gcd(v, p-1)=1$, izračuna $r=\alpha^u y^v \bmod p=\alpha u+av \bmod p$ in $s=-rv^{-1} \bmod (p-1)$. Iz tega sledi, da je par (r, s) veljaven podpis za sporočilo $m=su \bmod (p-1)$, ker velja $(\alpha^m \alpha^{-ar})^{s^{-1}}=\alpha^u y^v=r$.
Pomembno pri verifikaciji podpisa je, da oseba B vedno preveri pogoj $0<r<p$, če tega ne preverja, lahko napadalec izbere svoje sporočilo in ga avtentično podpiše.

4.3.4 Performančne lastnosti AlGamalovega algoritma

AlGamalov algoritem za generiranje podpisov je v principu zalo hiter algoritem, kar pomeni, da rabi le par modularnih operacij za podpis, s predpostavko, da ima že v

naprej izračunan eksponentni in Euclideanov del modularnega izračuna. Na drugi strani je AlGamalov algoritem za verifikacijo podpisov malo počasnejši, saj vključuje več eksponentnih modularnih operacij, možno ga je efektivno optimizirati.

4.4 Schnorrov algoritem za digitalno podpisovanje

Schnoorov algoritem je še ena izpeljanka ElGamalovega algoritma bolj podobna DSA algoritmu. Schoorov algoritem za svoje delovanje potrebuje enosmerno zgoščevalno funkcijo, generiranje ključa pa poteka po istem postopku, kot pri RSA algoritmu.
Prednost tega algoritma pred ElGamalovim je v hitrosti, pri pravi izbiri parametrov. Vendar se z hitrostjo tudi zmanjša dolžina podpisa, kar pa tudi zmanjša varnost digitalnega podpisa.

4.5 AlGamalov algoritem digitalnega podpisa z vsebujočim sporočilom

Do sedaj smo navedli že precej izpeljank AlGamalovega algoritma, v bistvu če pomislimo, vidimo da AlGamalovi algoritmi za pripenjanjem digitalnega podpisa na sporočilo v bistvu ni nič drugega, kot digitalni podpis z vsebujočim sporočilom, le da je v tem primeru sporočilo vrednost, ki nam jo poda enosmerna zgoščevalna funkcija. Skratka AlGamalov algoritem za v v sami osnovi omogoča digitalni podpis z vsebujočim sporočilom. Za AlGamalov algoritem digitalnega podpisa z vključenim sporočilom uporabimo redundančno funkcijo (namesto zgoščevalne funkcije), ključ pa se generira po istem principu, kot pri RSA algoritmu, le da ni omejitev glede velikosti p in q koeficientov.

5 Enkratni digitalni podpisi

Enkratni digitalni podpisi služijo enkratnemu podpisovanju dokumentov, z večkratno uporabo enkratnih podpisov obstaja možnost ponarejevanja digitalnega podpisa. Za vsak enkraten podpis je potreben nov javni ključ, ki ga lahko v tem primeru imenujemo validacijski parameter. Z združitvijo preverjanja validacijskega parametra in enkratnega podpisa, lahko dobimo shemo oziroma autentikacijska drevesa, kjer je možnih več digitalnih podpisov.
Večina algoritmov, za enkratne digitalne podpise, je zelo hitrih oziroma učinkovitih, zato se ti algoritmi večinoma uporabljajo v čip karticah, kjer smo omejeni z računsko močjo vgrajenih procesnih enot.

5.1 Rabinov algoritem za enkraten podpis

Rabinov algoritem za enkratno podpisovanje je bil prvi predlog algoritma za digitalni podpis nasploh. Dovoljuje podpis enega samega sporočila, preverjanje podpisa zahteva interakcijo med osebo, ki je sporočilo podpisala in osebo, ki hoče

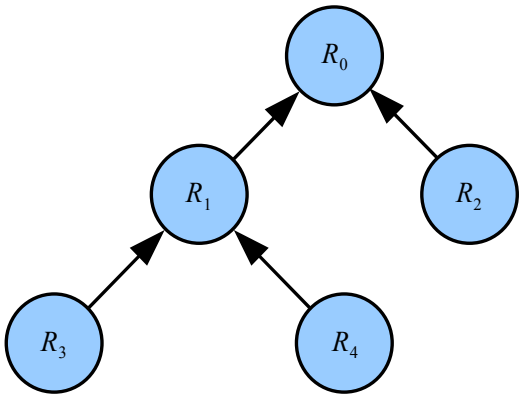
podpis preveriti. V nasprotju z ostalimi algoritmi, je Rabinov podpis verificirati samo enkrat, kar pa je v praksi zelo nepraktično.

5.2 Merkleov enkraten digitalen podpis

Merkleov algoritem se zelo razlikuje od Rabinovega, saj za verifikacijo podpisa ni potrebna prisotnost podpisovalca, ampak za to lahko uporabimo overitelja oziroma zaupanja vredno tretjo osebo. Prednost tega algoritma je kot že omenjeno v hitrosti generiranja podpisa, saj je za podpis potrebno minimalno trivialnih matematičnih operacij, sam podpis pa zagotavlja zadostno stopnjo varnosti s predpostavko, da je privatni ključ uporabljen samo enkrat.

5.2.1 Avtentikacijska drevesa enkratnih podpisov

Avtentikacijska drevesa, so način avtentikacije velikega števila uporabnikov enkratnih podpisov z možnostjo, da uporabniki lahko generirajo več podpisov. Osnovna ideja avtentikacijskih dreves je, vejanje avtentikacijski parametrov. Vejanje se začne pri tretji zaupanja vredni osebi (overovitelj), ki pozna osnoven R_0 nabor verifikacijskih parametrov (slika 9, nato pa uporabnik verifikacijske parametre veja in generira enkratne podpise, ki so sledljivi do „korenine“ oziroma overovitelja.



Slika 9: Avtentikacijska drevesa

5.2.2 GMR algoritem za enkratno podpisovanje

GMR algoritem je algoritem, za enkratni digitalni podpis, ki temelji na „claw-free“ permutacijah. Pri združitvi algoritma z avtentikacijskim drevesom, omogoča verigo enkratnih podpisov. Izvedba algoritma, v praksi ni praktična, čeprav se je povilo kar nekaj izpeljank, ki se tudi niso uporabile v praksi.

6 Drugi algoritmi za digitalno podpisovanje

Pregled drugih algoritmov, ki ne temeljijo na RSA, DSA, Fiat-Shamir oziroma algoritmih za enkratne podpise.

6.1 Arbitražni digitalni podpisi

Arbitražni algoritmi temeljijo na popolnem zaupanju overitelja, tako v procesu generiranja podpisa, kakor tudi v procesu verifikacije podpisa. Varnostni algoritem temelji na simetričnem šifriranju, z možnostjo varne in avtentične distribucije ključev. Prednost algoritma, je prednost simetričnega šifriranja, ki je matematično nezahtevno. Slabost algoritma, je večkratna komunikacija z overoviteljem, kar zelo obremenjuje overovitelje.

6.2 ESIGN

ESIGN je še ena vrsta digitalnega podpisa, katerega varnost temelji na matematični zahtevnosti faktorizacije. Algoritem je spada med podpise z pripenjanjem podpisa na sporočilo, za verifikacijo sporočila pa uporablja enosmerne zgoščevalne funkcije. Prednost algoritma je, da je bistveno hitrejši od RSA algoritma, vendar je potrebno poudariti, da vse možne pomanjkljivosti še niso bile raziskane.

7 Digitalni podpisi z dodatnimi funkcijami

Digitalni podpisi z dodatnimi funkcijami v bistvu sežejo izven področja digitalnih podpisov. V osnovi ponujajo večjo funkcionalnost, kot jo obsega področje digitalnega podpisovanja s tem, da temeljijo na znanih algoritmih za digitalni podpis recimo RSA.

8 Razvoj algoritmov za digitalno podpisovanje

8.1 NESSIE (New European Schemes for Signatures, Integrity and Encryption)

NESSIE je zaključen (2000-2003) Evropski projekt namenjen iskanju ter evaluaciji novejših varnih šifrirnih algoritmov na vseh področjih šifriranja. Projekt je v ta namen razvil metodologijo preverjanja šifrirnih algoritmov, tako z vidika varnosti kakor tudi z vidika hitrosti algoritmov. Poleg metodologije se je razvilo programsko orodje za potrebe vrednotenja in analize hitrosti algoritmov. Končni cilj projekta je bil tudi podpora Evropskemu razvoju in standardizaciji šifriranja ter njene uporabe v industriji.
NESSIE je leta 2000 izbral za dvainštirideset šifrirnih algoritmov iz desetih držav za vrednotenje njihove varnosti in hitrosti. Od teh je v svojem končne poročilu oziroma priporočilu izbral dvanajst algoritmov iz začetnih dvainštirideset ter pet drugih algoritmov, ki so v obdobju projekta nastajali.

S področja digitalnih podpisov NESSIE priporoča izbiro naslednjih algoritmov:

- ECDSA: Certicom Corp., USA and Certicom Corp., Canada;
- RSA-PSS: RSA Laboratories, USA;
- SFLASH: Schlumberger, France.

Izbrani standardi so se predložili standardizacijskim telesom in priporočali drugim uporabnikom.

8.2 CRYPTREC is the Cryptography Research and Evaluation Committee

CRYPTREC je Japonsko združenje evaluacijo novih in nadzor uporabljenih šifrirnih algoritmov, enakovredno NESSIE projektu oziroma njegovemu nasledniku eSTREAM. Ustanovljen je bil z namenom svetovanja Japonski vladi o izbiri novih varnih oziroma opuščanju obstoječih algoritmov na področju e-poslovanja.

CRYPTREC s področja digitalnih podpisov priporoča naslednje algoritme:

- DSA
- ECDSA
- RSASSA-PKCS1-v1_5
- RSA-PSS

Izbira šifrirnih algoritmov je podobna kot pri NESSIE projektu, le da je odvzet SFLASH dodana pa sta DSA ter in RSASSA.

8.3 NIST National Institute of Standards and Technology

NIST je Ameriška vladna inštitucija, ki skrbi za napredek na področju meroslovja, znanosti, standardov in tehnologije. Njihovo področje dela obsega preverjanje varnosti uveljavljenih ter vrednotenje novih algoritmov za digitalno podpisovanje. Svoje izsledke in priporočila redno objavljajo v DSS (Digital Signature Standard) standardu. Poleg izbire algoritmov za digitalno podpisovanje, priporočajo tudi izbiro parametrov in priporočila za implementacijo teh algoritmov.
Zadnje preliminarno NIST-ovo priporočilo obsega naslednje algoritme za digitalno podpisovanje:

- DSA
- RSA
- ECDSA

Pri tem je potrebno poudariti, da so pri vseh treh algoritmih podrobno opisani parametri za varno uporabo naštetih algoritmov.

9 Zaključek

Vsi algoritmi za digitalno podpisovanje temeljijo na matematičnih operacijah, ki so zahtevne s stališča računalniškega procesiranja s trenutno tehnologijo, ki je na razpolago. Z razvojem se zmogljivosti računalniških sistemov povečuje, zato je potrebno že uveljavljene algoritme preverjati oziroma prilagajati novim sposobnostim računalnikov, da zagotovili ustrezno raven zaščite. Z druge strani se razvijajo tudi matematična področja, zato je potrebno uveljavljene algoritme preverjati in iskati njihove pomanjkljivosti oziroma izboljšave tudi z analitičnega stališča. Na področju asimetričnega šifriranja je zelo težko napovedati, kateri šifrirni algoritem uporabiti v prihodnosti, do sedaj pa se je izkazalo, da odprti standardi šifrirnih algoritmov so bolj zanesljivi in varnejši.

Pri izbiri trenutno aktualnih algoritmov za digitalno podpisovanje lahko vidimo, da se priporočila uveljavljenih inštitucij prekrivajo. Pri tem je potrebno poudariti, da so nekateri algoritmi pod licenco oziroma plačljivi, kar je pri implementaciji algoritmov lahko pomemben faktor. Licenčni pogoji in trenutne implementacije algoritmov v uporabi so igrali pomembno vlogo v priporočilih uveljavljenih inštitucij, kot so NESSIE, NIST in CRYPTEK.

Na splošno lahko zaključimo, da so trenutno priporočeni algoritmi:

- uveljavljeni DSA,
- RSA in izpeljanke,
- ECDSA, kot trenutno najmočnejši šifrirni algoritem, ki temelji na ekliptičnih krivuljah in
- SFLASH kot najhitrejši algoritem za implementacijo na naprave z omejenimi procesnimi zmogljivostmi na primer SmartCard.

Kljub obsežnim raziskavam, ki so jih znane inštitucije opravile na šifrirnih algoritmih za digitalno podpisovanje in njihovim priporočilom, je potrebno pri uporabi naštetih algoritmov, vedno paziti na trenutno aktualno stanje, saj to področje izredno dinamično.

10 Literatura

[1] Tone Vidmar: Informacijsko-komunikacijski sistem, Pasadena 2002
[2] Sašo Tomažič: Varne komunikacije preko interneta, članek
[3] Sašo Tomažič: Varnost v telekomunikacijah in kako jo zagotoviti, članek
[4] http://rfc-ref.org/RFC-TEXTS/2154/index.html, Digital signature
[5] A. J. Menzes, P. C. van Oorschot, S. A. Vanstone: Handbook of Applied Cryptography
[6] https://www.cosic.esat.kuleuven.be/nessie/
[7] http://www.minrank.org/sflash/
[8] http://csrc.nist.gov/cryptval/dss.htm
[9] http://www.ipa.go.jp/security/enc/CRYPTREC/index-e.html
[10] http://www.cryptrec.jp/english/
[11] http://en.wikipedia.org/wiki/List_of_cryptography_topics
[12] http://en.wikipedia.org/wiki/List_of_prime_numbers
[13] NIST: Draft-FIPS-186-3, March2006