Algoritmi za kreiranje digitalnog potpisa

Vranješ, Antonio

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU

FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA

I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Sveučilišni studij

ALGORITMI ZA KREIRANJE DIGITALNOG POTPISA

Diplomski rad

Antonio Vranješ

Osijek, 2017.

Sadržaj

1. UVOD

U današnjem svijetu uznapredovale tehnologije i visoko razvijenih komunikacijskih sustava, unutar kojih se svakodnevno šalje velika količina informacija, uz dovoljno napredno kodiranje i zaštitu informacija na visokom nivou, vrlo je važno imati i jamstvo koje će posvjedočiti kako je poruka koja putuje od pošiljatelja do primatelja zaista poslana od strane osobe koja tvrdi da je pošiljatelj i kako poruka nije neovlašteno izmijenjena. Ta se dva uvjeta mogu svesti pod jedan zajednički pojam – autentičnost. Upravo je autentičnost poruke jamstvo koje digitalni potpisi (eng. *digital signatures)* u svijetu komunikacija pružaju i to je njihova glavna zadaća. Prema [1], osim autentičnosti, digitalni potpisi osiguravaju i integritet poruke što podrazumijeva utvrđivanje je li se poruka putem do primatelja mijenjala provjerom sažetka poruke te neporecivost koja isključuje mogućnost da pošiljatelj poruke porekne sudjelovanje u transakciji. Zadaća ovog rada jest analizirati najčešće algoritme koji se koriste u kriptosustavu digitalnog potpisivanja sa strogim naglaskom na matematičku podlogu, način rada i sigurnost te međusobno usporediti odabrane algoritme. Osim analize, rad je obuhvatio kratak pregled povijesnog razvoja digitalnog potpisivanja gdje se opisuje postupni prijelaz s tradicionalnih, papirnatih potpisa na digitalne potpise te se spominju najbitniji znanstvenici i inženjeri koji su doprinijeli razvoju digitalnog potpisivanja. Rad opisuje i principe na kojima je digitalno potpisivanje temeljeno unutar čega su opisane najčešće metode pomoću kojih se utvrđuje vjerodostojnost pošiljatelja. Nadalje, rad donosi analizu sigurnosti te moguće napade na odabrane algoritme. U radu je navedeno i opisano nekoliko područja primjene digitalnih potpisa, a jedno poglavlje je posvećeno i budućnosti digitalnog potpisivanja obzirom da se radi o tehnologiji za koju se procjenjuje da će u budućnosti biti od velikog značaja.

2. POVIJESNI RAZVOJ DIGITALNOG POTPISIVANJA

U današnje vrijeme svatko ima vlastiti potpis, ljudi potpisuju pravne dokumente, račune, formulare. Prošlo je gotovo cijelo tisućljeće od 1069. godine u kojoj se prvi put pojavio potpisani dokument od strane poznate povijesne ličnosti. Prema [2], potpisnik tog dokumenta je bio srednjevjekovni španjolski plemić i vojskovođa, El Cid. Do polovice 17. stoljeća, ručni potpisi su se proširili Europom, a Engleski parlament je u to vrijeme već bio dizajnirao ugovore koji su zahtijevali ručni potpis. Jedan od poznatijih povijesnih dokumenata potpisanih rukom je i američka Deklaracija nezavisnosti koju je vlastoručno potpisao tadašnji predsjednik Kontinentalnog kongresa, John Hancock. Prijenos poruka telegrafom je tehnologija koja je započela oko 1860. godine, a već 9 godina kasnije je američki sud *New Hampshire Supreme Court* potpise prenesene telegrafom proglasio važećima i pravomoćnima. U osamdesetim godinama 20. stoljeća, telefaks je bio raširena tehnologija putem koje su fizičke i pravne osobe hitno prenosile dokumente. Mnogi su sudovi u to vrijeme potpise prenesene telefaksom proglasili važećima. Kako stoji u [1], unatoč tome što se takvi potpisi fizički nalaze na papiru, njihov prijenos se vrši elektronički. Takva vrsta potpisa se naziva elektroničkim potpisima. Digitalni potpisi su podskupina elektroničkih potpisa koji koriste različite metode u kriptografiji. Zbog toga je razvoj digitalnog potpisivanja usko povezan s razvojem same kriptografije. Prvi autor koji je obradio područje kriptografije s javnim ključem jest William Stanley Jevons u svojoj knjizi *„The Principles of Science: A Treatise on Logic and Scientific Method"*. U ranim 1970 – im godinama su trojica znanstvenika, James Ellis, Clifford Cocks i Malcolm Williamson osmislili prve algoritme koji se temelje na asimetričnom ključu. Ipak, svoje radove nisu objavili. Veliki napredak u razvoju kriptografije predstavlja algoritam koji su 1976. godine, pod utjecajem radova Ralpha Merkela, razvili Whitfeld Diffie i Martin Hellman. Algoritam je po svojim autorima kasnije prozvan *Diffie – Helman* algoritmom i predstavlja prvu praktično primjenjivu metodu za razmjenu ključeva. Još jedan krucijalni algoritam u kriptografiji, RSA algoritam, su 1977. razvili Ron Rivest, Adi Shamir i Leonard Adleman. Algoritam je svoj naziv dobio akronimskim spojem prezimenâ trojice autora. To je prvi algoritam za enkripciju i potpisivanje koji se i danas smatra sigurnim i pouzdanim ukoliko se koriste dovoljno dugi ključevi i rade pravovremena ažuriranja. Ubrzo nakon RSA – a, razvijen je DSA (eng. *Digital Signature Algorithm).* Tijekom 1985. godine, Neal Koblitz i Victor Miller predlažu doradu DSA algoritmu na način da se nad konačnim poljima u kriptografskim algoritmima s javnim ključem uvede primjena eliptičnih krivulja. Prijedlog je prihvaćen i time je nastao ECDSA (eng. *Elliptic Curve Digital Signature Algorithm)*. Tijekom devedesetih godina 20. stoljeća, Sjedinjene Američke Države su započele proces standardizacije algoritama za digitalni

potpis. 1994. godine je *National Institute of Standards and Technology* izdao standard na koji je stavio oznaku FIBS PUBS 186 (eng. *Federal Information Processing Standards Publications).* Ubrzo nakon toga, i *American National Standards Institute* izdaje ANSI X9.30 standard. Nedugo zatim, standardizacija se proširila i na zemlje članice Europske unije, ali i na neke zemlje Europe koje nisu članice.

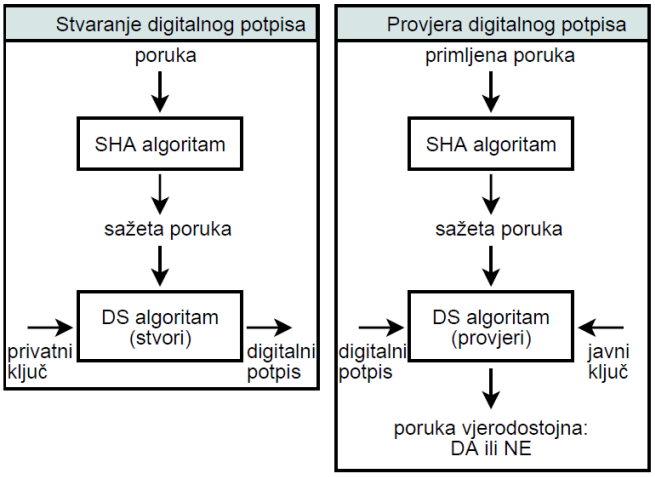
3. TEMELJNI PRINCIPI DIGITALNOG POTPISIVANJA

Kao što je spomenuto u uvodu, digitalno potpisivanje je način kojim se osigurava autentičnost elektroničkog dokumenta. Autentičnost podrazumijeva poznat identitet autora dokumenta i jamstvo da se dokument nije mijenjao na putu do svog primatelja. Digitalno potpisivanje se oslanja na različite tipove enkripcije kako bi se osigurala autentičnost. Enkripcija je proces kodiranja svih podataka koje pošiljatelj šalje primatelju na način da je jedino primatelj u mogućnosti dekodirati podatke. Autentikacija je proces provjere dolazi li informacija s izvora kojem se može vjerovati. Ta dva procesa su temelj na kojima počiva digitalno potpisivanje. Prema [3], postoji nekoliko načina za autentikaciju osobe ili informacije:

- **Lozinka** – upotreba korisničkog imena i lozinke je najčešći način za autentikaciju osobe. Osoba unosi svoje korisničko ime i odgovarajuću lozinku kada računalo to zatraži. Ukoliko se unese pogrešno korisničko ime ili pogrešna lozinka, daljnji pristup se odbija.
- **Ispitni zbroj (eng. *checksum)*** – iako je jedna od najstarijih metoda i koristi se prvenstveno za provjeru je li pristigla informacija ispravna, metoda ispitnog zbrajanja može poslužiti i kao način za autenticiranje obzirom da neispravan zbroj sugerira da su podaci na jedan od načina izmijenjeni.
- **CRC (eng. *Cyclic Redundancy Check)*** – CRC koncept je vrlo sličan metodi ispitnog zbrajanja. Razlika je u tome što CRC metoda koristi dijeljenje polinoma kako bi utvrdila vrijednost CRC – a koja je najčešće duga 16 bita ili 32 bita. Prednost CRC – a je vrlo velika točnost. U slučaju da se jedan bit poruke putem do primatelja izmijenio, CRC vrijednosti se neće podudarati. Iako su i metoda ispitnog zbrajanja i CRC metoda efikasne u prevenciji nasumičnih pogrešaka u prijenosu kroz kanal, pružaju slabu zaštitu od zlonamjernih napada na podatke u prijenosu. Od takvih napada su puno efikasnije metode koje u radu slijede.
- **Enkripcija s privatnim ključem (eng. *Private key encription)*** – pojam privatnog ključa podrazumijeva sustav u kojem svako računalo ima tajni ključ kojim šifrira informacije koje preko mreže šalje ostalim računalima. Također, potrebno je znati koja će računala međusobno komunicirati i instalirati ključ na svako od njih. Enkripcija s privatnim ključem se može shvatiti kao komunikacija putem tajnog kôda koji računala moraju znati ako žele dekodirati informaciju. Jedan od najstarijih primjera je Cezarova šifra unutar koje svako slovo u šifriranom tekstu (šifratu) stoji u poruci umjesto točno određenog drugog slova iz abecede, ovisno o dogovorenom, fiksnom abecednom pomaku. Vrijednost tog pomaka se može smatrati privatnim ključem čiju vrijednost primatelj zna. U tom slučaju, ako

pošiljatelj pošalje šifriranu poruku, primatelj će ju primiti i on će ju, obzirom da zna za koliko je mjesta abeceda pomaknuta pri procesu šifriranja, jedini biti u mogućnosti uspješno dešifrirati. Ostali potencijalni zlonamjerni napadači na kanal će, kako ne znaju vrijednost pomaka, umjesto smislene poruke vidjeti samo besmisleni niz znakova.

- **Enkripcija s javnim ključem (eng. *Public key encription*)** – ova vrsta enkripcije koristi kombinaciju javnog i privatnog ključa. Privatni ključ je poznat samo svome vlasniku kako bi se izbjeglo krivotvorenje potpisa, dok je javni ključ pojedinog računala poznat svim sudionicima koji žele međusobno sigurno komunicirati. Kako bi uspješno dekodirao poruku, primatelj se koristi javnim ključem pošiljatelja i vlastitim privatnim ključem. Kako je navedeno u [1], za potrebe analize, informacije koje se digitalno potpisuju se skraćeno nazivaju porukom. Kako je prikazano na slici 3.1. iz [1], proces digitalnog potpisivanja započinje sažimanjem originalne poruke u saženu inačicu (eng. *message digest*). Za dobivanje sažete inačice poruke koriste se tzv. *hash* algoritmi. To je posebna vrsta algoritama koja predstavlja sigurnu jednosmjernu funkciju, najčešće SHA (eng. *Secure Hash Algorithm*), njezin izlaz se vrlo jednostavno izračuna, a vrlo teško joj je naći inverznu funkciju. Tako dobivena sažeta inačica poruke se potpisuje. Poruka se, zajedno s pripadajućim potpisom, šalje primatelju koji na zahtjev dobiva pošiljateljev javni ključ. Pošiljateljev javni ključ koji je sada u posjedu primatelja nije isti, ali odgovara tajnom ključu pošiljatelja i na taj način primatelj utvrđuje vjerodostojnost poruke i samog postupka digitalnog potpisivanja. Važno je da se u procesu provjere koristi SHA algoritam istovjetan onome koji se koristio i prilikom potpisivanja.



Slika 3.1. *Shematski prikaz stvaranja i provjere digitalnog potpisa*

Prema [1], u postupku koji je opisan, potpisuje se sažeta inačica poruke, a ne cijela poruka iz sljedećih razloga:

- **Efikasnost**: stvaranje sažetka poruke je puno brže od stvaranja potpisa pa će stoga potpis biti puno kraći, a cjelokupni postupak brži,
- **Javnost dokumenta**: dokumenti kao što su diplome, ugovori, dozvole i potvrde trebaju biti javno dostupni pa se prenose bez enkripcijske zaštite. Potpis koji je priložen jamči vjerodostojnost pojedinog dokumenta,
- **Integritet**: potpisani tekst treba biti kraći od duljine privatnog ključa. Obzirom da poruka koju se potpisuje najčešće nije, u slučaju bez sažimanja, bilo bi ju potrebno razlomiti na dijelove i svaki dio zasebno potpisati i poslati. U takvom scenariju primatelj ne bi mogao sa sigurnošću znati je li koji dio poruke izbrisan ili izgubljen tijekom prijenosa.

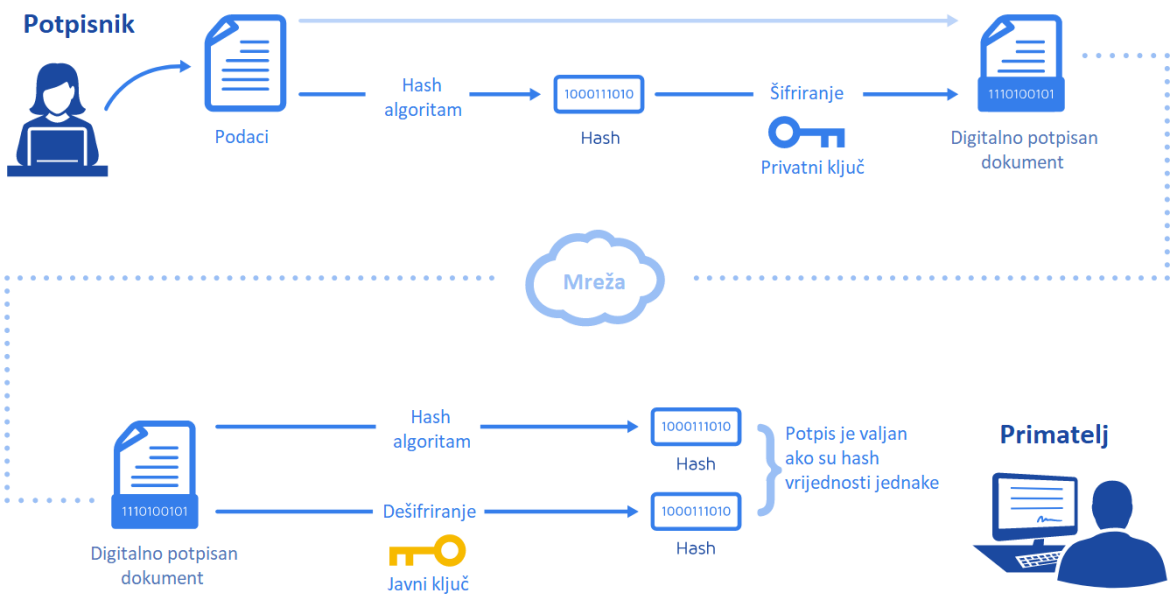
- **Digitalni certifikati –** implementacija enkripcije s javnim ključem u sustavima velikim kao što su mrežni poslužitelji zahtijeva drugačiji pristup. Zbog toga su razvijeni digitalni certifikati. Digitalni certifikat u svojoj suštini predstavlja informaciju koja potvrđuje da se mrežnom poslužitelju može vjerovati. Kako stoji u [1], odluku o povjerenju ili nepovjerenju donosi i, ovisno o tome. certifikate izdaje neovisno tijelo osnovano u tu svrhu poznatije kao *Certificate Authority* koje je dio PKI (eng. P*ublic Key Infrastructure*) sustava. Prema [3], neovisno tijelo predstavlja posrednika između dva računala kojemu oba računala vjeruju. Ono potvrđuje da je svako računalo zaista ono za koje se predstavlja i svakom od njih tada šalje javne ključeve ostalih računala. Kako je navedeno u [1], certifikati povezuju javne ključeve s podacima o identitetu vlasnika čime sprječavaju neovlašten pristup podacima objavljivanjem lažnog javnog ključa. Alternativu centraliziranim PKI sustavima predstavlja mreža povjerenja (eng. *web of trust*). Ona radi na način da korisnici, koristeći vlastite privatne ključeve, potpisuju certifikate drugih korisnika. Navedeni certifikati mogu sadržavati informacije o svojim vlasnicima kao što su javni ključevi i ostali podaci o vlasniku. U mreži povjerenja korisnik može ocijeniti certifikat vjerodostojnim ako ga je potpisalo tri ili više korisnika kojima navedeni korisnik djelomično vjeruje ili jedan dokazano vjerodostojan korisnik.

4. ALGORITMI ZA DIGITALNO POTPISIVANJE

Sustav digitalnog potpisivanja se sastoji od tri različita algoritma:

- Algoritam za generiranje javnog i privatnog ključa – izlaz iz algoritma su privatni ključ i odgovarajući javni ključ. Pomoću privatnog ključa će poruka biti potpisana, a pomoću javnog verificirana.
- Algoritam za izradu potpisa – generira se digitalni potpis na osnovu sažetka poruke i privatnog ključa
- Algoritam za provjeru potpisa – koristeći poruku, javni ključ i potvrđuje ili opovrgava autentičnost poruke.

Shematski prikaz međudjelovanja ova tri algoritma je prikazan na slici 4.1.



Slika 4.1 *Shematski prikaz algoritama za stvaranje i provjeru digitalno potpisane poruke*

Unazad posljednjih 60 godina, razvijen je poveći broj algoritama koji podržavaju digitalno potpisivanje, a neki od njih su:

- ElGamalova shema potpisivanja
- Schnorrov potpis
- Pointcheval – Sternov algoritam za potpisivanje
- Rabinov algoritam za potpisivanje
- Spojna potpisna shema

Ipak, ovaj rad analizira tri najčešća i najpouzdanija algoritma koji se koriste za digitalno potpisivanje, a to su:

1. **DSA *(Digital Signature Algorithm)***
2. **ECDSA *(Elliptic Curve Digital Signature Algorithm)***
3. **RSA (*Rivest – Shamir – Adleman)***

4.1. DSA (Digital Signature Algorithm)

Prema [1] i [4], *Digital Signature Algorithm* (DSA) je algoritam za digitalno potpisivanje odobren od strane DSS (eng. *Digital Signature Standard*) standarda savezne vlade Sjedinjenih Američkih Država. Koriste ga sve vladine organizacije te sve nevladine tvrtke i organizacije koje surađuju s vladom. Algoritam je 1991. godine razvio David Kravitz, bivši agent u NSA – u (eng. *National Security Agency*) i može se koristiti besplatno. Kako je navedeno na početku četvrtog poglavlja ovog rada, svaki se algoritam za digitalno potpisivanje, pa tako i DSA, sastoji od tri podalgoritma.

4.1.1. Generiranje ključeva

Prema [4], proces generiranja ključeva se može podijeliti u dvije faze:

1) Generiranje parametara

a) Odabrati odobrenu kriptografsku *hash* funkciju *H*. U originalnom DSS – u, preporuka je bila koristiti SHA – 1, ali u trenutnom DSS – u je odobrena upotreba i SHA – 2. Izlaz iz *hash* funkcije se može smanjiti na veličinu para ključeva.

b) Odabrati duljine ključeva *L* i *N* (oznake se odnose na **duljine** ključeva u bitovima, a ne na nazive samih ključeva). Duljine ključeva su prvi pokazatelji kriptografske snage ključeva. Prvi DSS je uvjetovao da *L* bude broj između 512 i 1024 (uključujući i granične vrijednosti) te da bude djeljiv sa 64. Današnje preporuke za parove duljina ključeva (*L, N)* jesu (1024, 160), (2048, 224), (2048, 256) i (3072, 256). Duljina *N* mora biti manja ili jednaka duljini izlaza iz funkcije *H*.

c) Odabrati prosti broj *q* duljine *N* bita.

d) Odabrati prosti broj *p* duljine *L* bita tako da vrijedi $p = qz + 1$ za neki cijeli broj *z,* vrijednost *L* mora biti između 512 i 1024 (uključujući i granice intervala) i djeljiva sa 64.

e) Odabrati proizvoljno broj *h* takav da vrijedi $1 < h < (p - 1)$

f) Izračunati broj g takav da vrijedi $g = h^z \bmod p$. Ako se pokaže da je $g = 1$, treba odabrati drugačiju vrijednost h. Ipak, većina odabira će odvesti do upotrebljivog broja g, a najčešće se za vrijednost h uzima broj 2.

Algoritamski parametri (p, q, g) se sada mogu distribuirati svim korisnicima u sustavu.

2) Izračun ključeva za korisnike

Uz izračunate parametre, druga faza obuhvaća izračun privatnog i javnog ključa za pojedinog korisnika.

a) Odabrati tajni ključ x jednom od nasumičnih metoda takav da je $0 < x < q$.
b) Izračunati javni ključ $y = g^x \bmod p$

4.1.2. Potpisivanje poruke

Poruka m se uz pomoć *hash* funkcije H (SHA – 1 ili SHA – 2) potpisuje na sljedeći način:

a) Odabrati nasumični broj k takav da vrijedi $1 < k < q$.
b) Izračunati $r = (g^k \bmod p) \bmod q$. U malo vjerojatnom slučaju da je $r = 0$, odabrati drugačiju vrijednost broja k.
c) Izračunati $s = k^{-1} [H(m) + xr) \bmod q$.
d) Ponoviti postupak s drugačijim k ako su $s = 0$ ili $r = 0$ (malo vjerojatno).
e) Potpis je uređeni par (r, s).

Dio algoritma koji zahtijeva najviše resursa jest modularno eksponenciranje u koracima 1f i 2b. Vrijednosti navedene u spomenutim koracima mogu se izračunati i prije nego poruka prođe kroz funkciju H. Modularni inverz iz koraka c) u ovom odjeljku (4.1.2.) je drugi najzahtjevniji korak po pitanju korištenja resursa, a računa se pomoću proširenog Euklidovog algoritma ili Malog Fermatovog teorema kao $k^{q-2} \bmod q$.

4.1.3. Provjera vjerodostojnosti poruke

a) Provjeriti vrijede li izrazi $0 < r < q$ i $0 < s < q$. Ukoliko samo jedan od njih nije zadovoljen, potpis se smatra nevažećim, neispravnim.
b) Izračunati $w = s^{-1} \bmod q$.
c) Izračunati $u_1 = H(m) \cdot w \bmod q$.
d) Izračunati $u_2 = (r \cdot w) \bmod p$.

e) Izračunati $v = (g^{u_1} \cdot y^{u_2} \bmod p) \bmod q$.

f) Potpis je valjan ako vrijedi $v = r$, u protivnom je potpis nevaljan.

4.1.4. Osjetljivost algoritma na krivi odabir parametara

Kako je navedeno u [4], za pravilan i siguran rad DSA algoritma, ključna su tri zahtjeva:

- Entropija
- Tajnost
- Jedinstvenost.

Ova se tri zahtjeva odnose na nasumičnu vrijednost *k* koja se proizvoljno odabire u drugom od tri dijela algoritma, pri potpisivanju poruke. Navedeni zahtjevi su od kritične važnosti obzirom da je neispunjenje samo jednog od njih dovoljno da zlonamjerni napadač na kanal otkrije privatni ključ pošiljatelja. Algoritam je toliko osjetljiv na pogrešan izbor parametara da je ponavljanje iste vrijednosti *k* u dva slučaja (iako je vrijednost *k* tajna), korištenje predvidljivog broja za vrijednost *k* ili čak otkrivanje samo nekoliko bita vrijednosti *k* i više nego dovoljno za probijanje DSA zaštite. Ovakva osjetljivost je obilježje i DSA algoritma i algoritma izvedenog iz njega koji ovaj rad također obrađuje, ECDSA algoritma. U prosincu 2010. godine, grupa koja sebe naziva *fail0verflow* je objavila kako je uspjela otkriti tajni ključ ECDSA algoritma koji je tvrtka *Sony* koristila za potpisivanje softvera za igraću konzolu *PlayStation 3*. Napad im je omogućio nemar inženjera koji su radili na algoritmu obzirom da su za svaki potpis koristili istu vrijednost *k*. Ovakav napad se može izbjeći ukoliko se vrijednost *k* izvede deterministički iz tajnog ključa i *hash* poruke kao što je opisano u [5]. Takav pristup osigurava da će broj *k* za svaku hash funkciju biti drugačiji i nepredvidljiv za napadače koji ne znaju tajni ključ *x*.

4.1.5. Dokaz valjanosti algoritma za provjeru potpisa

U ovom potpoglavlju se nalazi dokaz kako algoritam za provjeru potpisa dobar potpis prihvaća, odnosno, ne odbacuje. Kako je navedeno u [7], najprije primijetimo da vrijedi:

$$g^q \equiv h^{zq} \equiv h^{p-1} \equiv 1 \pmod{p}, \qquad prema\ Fermatovom\ teoremu.$$

Obzirom da je $g > 1$ i q je prost broj, može se zaključiti da je q red od g modulo p. Iz relacije

$$s \equiv \left(k^{-1}\,(H(m) + xr)\right) \pmod{q}$$

slijedi relacija

$$k \equiv H(m)s^{-1} + xrs^{-1} \quad (mod\ q)$$

što se koristi u raspisivanju od $r = g^k \ mod\ p$:

$$g^k \equiv g^{k\ mod\ q} \equiv g^{H(m)s^{-1}}\, g^{xrs^{-1}} \equiv g^{u_1} y^{u_2} \quad (mod\ p).$$

Iz navedenoga slijedi $r = v$.

4.2. ECDSA (Elliptic Curve Digital Signature Algorithm)

ECDSA je algoritam nastao iz ranije opisanog DSA algoritma. Prema [6], kao ANSI standard prihvaćen je 1999. godine, a kao IEEE i NIST standard, prihvaćen je godinu kasnije. Za razliku od prethodnika na čijim je temeljima nastao, ECDSA se ne oslanja na problem faktorizacije velikih brojeva, već na eliptičke krivulje što snagu njegovih ključeva čini mnogo većom.

4.2.1. Primjena eliptičkih krivulja u kriptografiji

Svaka se vrsta enkripcije temelji na složenosti rješavanja određenog problema i upravo su te složenosti iskorištavali tvorci dosad konstruiranih enkripcijskih algoritama kako bi razvili što sigurnije i neprobojnije algoritme. Do pojave primjene eliptičkih krivulja u kriptografiji, većina algoritama se temeljila na složenosti problema faktorizacije velikih brojeva. Prema [7], primjenu eliptičkih krivulja u kriptografiji su prvi uveli Koblitz i Miller 1985. godine iskoristivši činjenicu da je u grupi eliptičkih krivulja matematička operacija potenciranja puno lakša od matematičke operacije logaritmiranja. Da bi primjena eliptičkih krivulja bila jasna, u dijelu rada koji slijedi su objašnjenje postavke i matematičke operacije nad grupom eliptičkih krivulja koje su potrebne za primjenu spomenutih krivulja u kriptografiji.

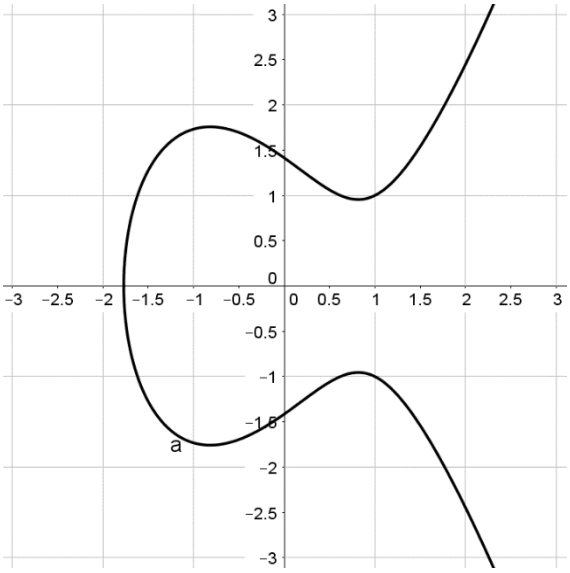
4.2.2. Problem diskretnog logaritma za eliptičke krivulje

Eliptička krivulja *E* je krivulja koja odgovara jednadžbi:

$$E = \{(x, y) |\ y^2 = x^3 + ax + b\}$$

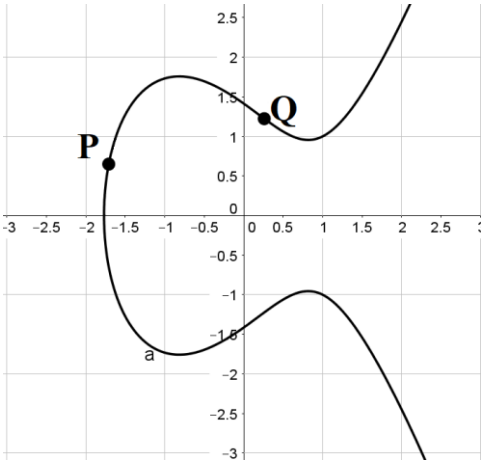
gdje su parametri a i b te točka u beskonačnosti, $\mathcal{O}$, elementi polja *K*, $(a, b, \mathcal{O} \in K)$. Polje *K* može biti polje u $\mathbb{R}$, $\mathbb{Q}$, $\mathbb{C}$ i $\mathbb{Z}/p\mathbb{Z}$ (cjeli brojevi s modulo aritmetikom).

Reprezentativna eliptička krivulja jednadžbe $y^2 = x^3 - 2x + 2$ na kojoj će ovaj rad objasniti operacije je prikazana na slici 4.2.

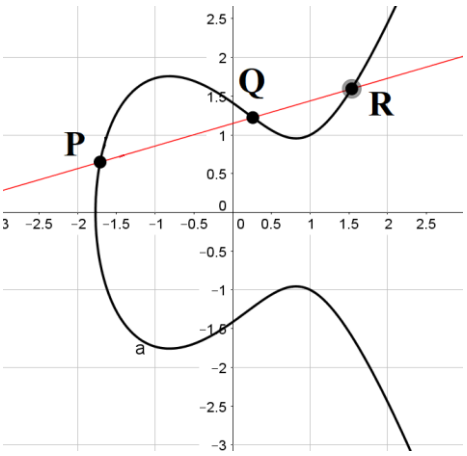


Slika 4.2. *Reprezentativna eliptička krivulja*

Da bi se razumjela upotreba eliptičkih krivulja u kriptografiji, potrebno je najprije definirati i pojasniti operacije koje su potrebne za implementaciju ovakvih krivulja u kriptografiju. Prva operacija je zbrajanje. Potrebno je pronaći zbroj dviju točaka krivulje, stoga je definirana operacija zbrajanja na krivulji *E*. Kako je prikazano na slici 4.3., odabrane su dvije točke, *P* i *Q*. Pravac koji prolazi kroz točke *P* i *Q* siječe krivulju u još jednoj točki koja je nazvana *R* (slika 4.4.).

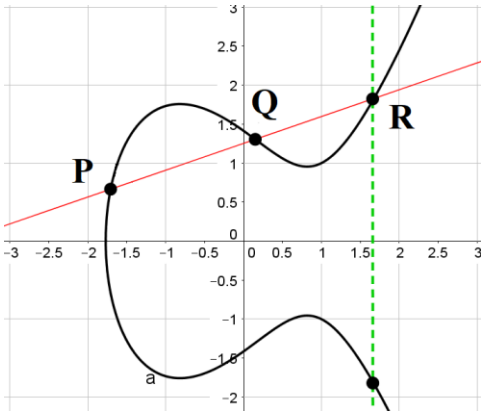


Slika 4.3. *Dvije točke na krivulji*

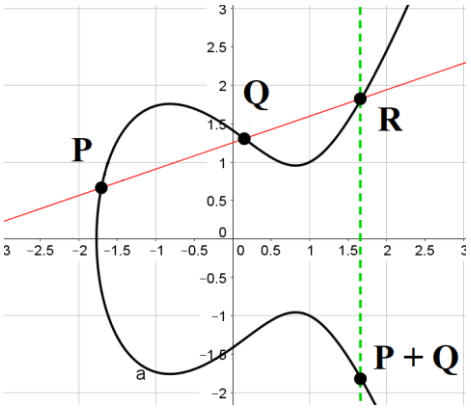


Slika 4.4. *Pravac kroz krivulju i treća točka* R

Točka *R* služi kao referentna točka pomoću koje će se dobiti točka koja predstavlja zbroj točaka *P* i *Q*. Sljedeći korak u zbrajanju jest povlačenje pravca koji siječe točku *R* i apscisu te je paralelan s ordinatom. Taj je korak vidljiv na slici 4.5. Točka koja će predstavljati zbroj točaka *P* i *Q* jest točka koja se nalazi na zadnje navedenom pravcu i siječe krivulju, odnosno, točka koja predstavlja *P* + *Q* je simetrična točki *R* obzirom na apscisu (slika 4.6.)



Slika 4.5. *Pomoćni pravac kroz točku* R



Slika 4.6. *Način dobivanja točke* P + Q

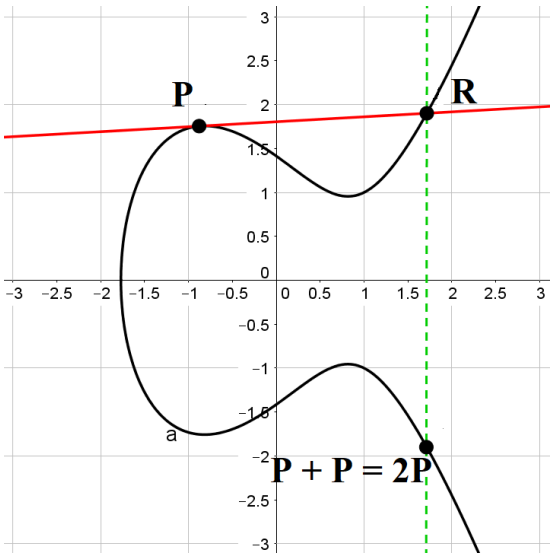
Točka $P + Q$ će u daljnjem radu biti navođena kao točka S (eng. *sum*). Postupak izračuna koordinatâ točke S počinje izračunom nagiba pravca koji prolazi kroz točke P i Q. Nagib n se računa pomoću formule

$$n = \frac{y_P - y_Q}{x_P - x_Q}$$

Koordinate točke S se tada uz pomoć nagiba računaju prema sljedećim formulama:

$$x_S = n^2 - (x_P + x_Q) \qquad y_S = n(x_P - x_R) - y_P.$$

Sljedeća operacija koja je definirana nad eliptičkim krivuljama jest umnožavanje točaka. To je slučaj u kojem je točka P istovjetna točki Q. Pravac koji prolazi kroz točke P i R je tangenta na točku P. Kako su P i Q točke s istim koordinatama, postupak je identičan prethodno opisanom. Slučaj je prikazan na slici 4.7.



Slika 4.7. *Zbrajanje kada su* P *i* Q *istovjetne točke*

Nagib tangente se računa prema formuli

$$n = \frac{3x_P^2 + a}{2y_P},$$

a koordinate točke *2P* (ili *S*) se računaju na sljedeći način:

$$x_S = n^2 - 2x_P \qquad y_S = n(x_P - x_R) - y_P.$$

Sljedeća operacija definirana nad eliptičkim krivuljama jest skalarno množenje. Neka je točka P element krivulje E, $P \in E$, a koeficijent k cijeli broj, $k \in \mathbb{Z}$. U tom slučaju će se neka točka Q moći zapisati kao umnožak koeficijenta k i točke P,

$$Q = kP.$$

U jednostavnijem zapisu, točku Q će se moći zapisati kao

$$Q = P + P + \ldots + P, \qquad k \text{ puta zaredom.}$$

Kako bi se na temelju eliptičkih krivulja efikasno kriptirali podaci, potrebno je razviti jednosmjernu funkciju, funkciju koju će biti relativno lagano provesti u jednom smjeru, a vrlo složeno u drugom smjeru. Ono što je u DSA algoritmu bila *hash* funkcija, u ECDSA algoritmu se naziva problemom diskretnog logaritma. Prema [8], operacija skalarnog množenja predstavlja tu jednosmjernu funkciju koja se traži. Problem se očituje u sljedećem: ako je krivulja definirana nad skupom cijelih brojeva modulo prost broj p, $\mathbb{Z}/p\mathbb{Z}$, ako postoje točke P i Q koje se nalaze na krivulji E ($P, Q \in E$ ($\mathbb{Z}/p\mathbb{Z}$)) i ako je točka Q višekratnik točke P, $Q = kP$, na način da je k puta veća od točke P, pronaći brojčanu vrijednost koeficijenta k je vrlo složen zadatak. Ovaj problem je temeljni princip primjene eliptičkih krivulja u kriptografiji. Sljedeći parametar koji je potreban za implementaciju eliptičkih krivulja u kriptografske procese je osnovna točka (eng. *base point*) ili generator *G*. Generator je točka na krivulji koja generira cikličku podgrupu točaka i to na način da se svaka točka podgrupe može dobiti ponavljanjem generatora određeni broj puta. Definiran je i red generatora, *ord(G) = m*. Red generatora je broj koji ukazuje na veličinu nastale podgrupe, odnosno na broj točaka koji je generator izgenerirao. Također, definiran je i kofaktor *h* kao omjer broja svih točaka na krivulji i reda generatora,

$$h = \frac{|E\ (\mathbb{Z}/p\mathbb{Z}\ )|}{m}.$$

Poželjno je da kofaktor *h* bude veličine 1, $h = 1$ u idealnom slučaju. Eliptičke krivulje i generatori koji u svom međudjelovanju postižu kofaktore preko 4 su nepoželjni obzirom da su puno

osjetljiviji na napade. Ranije je objašnjeno na koji se način točke na krivulji mogu međusobno zbrajati te kako ih se može množiti s cjelobrojnom vrijednošću. Zakonitosti i svojstva koja vrijede nad krivuljama, a koja su od koristi u kriptografiji su prikazani u sljedeća 3 primjera:

- $P + Q = Q + P$
- $3P + P = 2P + 2P = 4P$
- $5 \cdot (7P) = 7 \cdot (5P)$.

Prednost ovakvog kriptiranja digitalnog potpisa jest brzina računanja. U navedenom primjeru se u samo 8 koraka došlo od P do $100P$:

1. $P \rightarrow 2P$
2. $2P \rightarrow 3P$
3. $3P \rightarrow 6P$
4. $6P \rightarrow 12P$
5. $12P \rightarrow 24P$
6. $24P \rightarrow 25P$
7. $25P \rightarrow 50P$
8. $50P \rightarrow 100P$

Kao što je ranije u radu navedeno, a ovdje potkrijepljeno i primjerom, operacija u suprotnom smjeru je vrlo spora. U izrazu $Q = kP$ najbrži način za pronaći koeficijent k jest krenuti i pokušavati izjednačiti jednadžbu uvrštavajući redom sve cijele brojeve. U ozbiljnim kriptografskim rješenjima, koeficijent k može biti broj i od 50 znamenaka što znatno podiže sigurnost ECDSA algoritma u usporedbi s algoritmima temeljenim na problematici faktoriziranja velikih brojeva.

4.2.3. Osjetljivost algoritma na krivi odabir parametara

Iako je ECDSA racionalniji izbor od DSA algoritma, postoji mogućnost pogrešnog izbora parametara koji znatno smanjuje sigurnost algoritma. Prema [8], prvi previd koji je moguće napraviti jest odabrati takve parametre da se formira petlja bilo kojeg reda veličine, primjerice, odabrati parametre tako da vrijedi $101P = P$. U takvoj situaciji bi interval mogućih vrijednosti koeficijenta k s gotovo beskonačnog broja mogućnosti pao na samo 100 mogućnosti, bez obzira koliko je koeficijent k velik. Algoritam će i dalje savršeno raditi, ali će potencijalnom napadaču uvelike biti olakšan posao obzirom da će morati isprobati najviše 100 brojeva. Uzimajući u obzir snagu današnjih računala, problem je trivijalan. Također, potrebno je pripaziti i na odabir eliptičke krivulje. Prema [7], važno je da problem diskretnog algoritma kod odabrane krivulje bude težak. Generalno navedeni problem i jest vrlo težak, ali postoje krivulje kod kojih je taj problem nešto lakši, a ponekad i puno lakši. Takve krivulje treba izbjegavati. Analogan problem se pojavljuje i kod algoritama koji svoju neprobojnost temelje na teškoći faktorizacije velikih prirodnih brojeva,

i kod njih je broj oblika *pq*, čiji su faktori veliki prosti brojevi, teško rastaviti na faktore samo ako se *p* i *q* odaberu s pažnjom.

4.2.4. Generiranje ključeva

Kako je navedeno u [7], proces generiranja ključeva se odvija na sljedeći način: definirana je eliptička krivulja E nad poljem K, a točka P je točka prostog reda q na krivulji E. Upute za svakog korisnika koji sudjeluje u transakciji su sljedeće:

a) odabrati slučajan broj d iz intervala $<1, q - 1>$,
b) izračunati $Q = dP$,
c) Q predstavlja javni, a d tajni ključ.

4.2.5. Potpisivanje poruke

U modelu kojim kriptografija pojašnjava svoje algoritme, najčešće se nalaze dva sugovornika, A i B sugovornik te potencijalni napadač. Zbog lakšeg snalaženja u shemama modela, najčešće se za korisnika A koristi ime Alice, a za korisnika B ime Bob. Pri generiranju potpisa za poruku m, Alice će napraviti sljedeće:

a) odabrati slučajan broj k iz intervala $<1, q - 1>$,
b) izračunati $kP = (x_1, y_1)$ i $r = x_1 \bmod q$,
c) ako je $r = 0$, odabrati drugi broj iz intervala navedenog u koraku a),
d) izračunati $k^{-1} \bmod n$,
e) izračunati $s = k^{-1} (H(m) + dr) \bmod q$, gdje je H *hash* funkcija
f) ako je $s = 0$, vratiti se na korak a),
g) potpis poruke m je uređeni par (r, s).

4.2.6. Provjera vjerodostojnosti poruke

Kako bi verificirao Alicein potpis (r, s) poruke m, Bob mora učiniti sljedeće:

a) priskrbiti si Alicein javni ključ Q,
b) provjeriti jesu li r i s cijeli brojevi iz intervala $<1, q - 1>$,
c) izračunati $w = s^{-1} \bmod q$ i $H(m)$,
d) izračunati $u_1 = H(m)w \bmod q$ i $u_2 = rw \bmod q$,
e) izračunati $u_1P + u_2Q = (x_0, y_0)$ i $v = x_0 \bmod q$,
f) ocijeniti potpis vjerodostojnim ako i samo ako vrijedi $v = r$.

Uvjet $r \neq 0$ potvrđuje da se u potpisivanju koristio stvarno Alicein tajni ključ d, a uvjet $s \neq 0$ se pojavljuje zbog 3. koraka u algoritmu provjere potpisa.

4.3. RSA (Rivest – Shamir – Adleman)

Prema [10] i [11], RSA algoritam je jedan od prvih praktičnih kriptografskih algoritama koji koristi ideju javnog ključa. Asimetrija algoritma se očituje u praktičnoj nemogućnosti faktorizacije produkta dva velika prosta broja. Algoritam je nazvan po prvim slovima prezimenâ njegovih tvoraca, Rona Rivesta, Adija Shamira i Leonarda Adlemana koji su algoritam prvi puta predstavili 1978. godine. Korisnik RSA algoritma kreira i objavljuje javni ključ koji se bazira na dva velika prosta broja, zajedno s pomoćnom vrijednošću. Pritom je važno da odabrani prosti brojevi ostanu tajni. Svatko može šifrirati poruku pomoću javnog ključa, ali dešifrirati ju može samo osoba koja zna o kojim se prostim brojevima radi. Do danas se algoritam pokazao prilično sigurnim iako mnoge stvari, koje se u algoritmu podrazumijevaju, nisu dokazane. Primjerice, nije dokazano da je modulo aritmetika najbolji izbor za izračun velikih prostih brojeva. Također, nije niti dokazano da je temelj ovog algoritma, složenost izračuna prostih brojeva, toliko složen problem kao što se čini. U teoriji, postoji mogućnost da će napredak u teoriji brojeva donijeti nova otkrića. Prema [1] i [11], algoritam koji koristi RSA za digitalno potpisivanje se sastoji od 5 koraka koji slijede.

4.3.1. Generiranje ključeva

Prvi korak u algoritmu je generiranje privatnog i javnog ključa. Taj se proces odvija kroz sljedeće korake:

a) nasumično odabrati dva velika prosta broja p i q,
b) izračunati $n_1 = pq$,
c) izračun produkta $\Phi(n1) = (p-1)(q-1)$,
d) odabrati cijeli broj e_1 tako da vrijedi $1 < e_1 < \Phi(n1)$ te da e_1 i $\Phi(n1)$ nemaju zajedničkih djelitelja osim broja 1,
e) odabrati vrijednost d_1 tako da vrijedi $(d_1 \cdot e_1) mod\ \Phi(n_1) = 1$,
f) Javni ključ je $\boldsymbol{K_{javni} = (e_1, n_1)}$, a tajni $\boldsymbol{K_{tajni} = (d_1, n_1)}$.

4.3.2. Potpisivanje poruke

Potpisivanje poruke pomoću RSA algoritma se vrši kroz sljedeće korake:

a) Izračunati *hash* vrijednost *h* poruke *m*, $h = H(m)$, gdje je *H* kriptografska *hash* funkcija, može biti SHA – 1 ili SHA – 2,
b) Izračunati potpis $s = h^{d_1} mod\ n_1$,
c) Dodati potpis *s* poruci *m*.

4.3.3. Šifriranje poruke

Potpisanu poruku je potrebno šifrirati na sljedeći način:

a) Dohvatiti primateljev javni ključ, $K_{javni} = (e_2, n_2)$,
b) Primjenom međusobno dogovorenog povratnog protokola, iz poruke *m* dobiti broj *M* takav da vrijedi $M < n$,
c) Izračunati šifrat poruke $c = M^{e_2}\ mod\ n_2$

4.3.4. Dešifriranje poruke

Šifriranu poruku koja je primatelju prispjela kanalom se dešifrira prateći sljedeće korake:

a) Iz primljene poruke *c* dobiti *M* prema izrazu $M = c^{d_2}\ mod\ n_2$
b) Primjenom inverza dogovorenog protokola iz koraka 4.3.3. b) iz *M* izračunati izvornu poruku *m*.

4.3.5. Provjera vjerodostojnosti poruke

Da bi primatelj bio siguran da je poruku poslala osoba koja to i tvrdi, treba slijediti navedene korake:

a) Izračunati *hash* vrijednost *h* poruke *m*, $h = H(m)$, gdje je važno da *hash* funkcija bude identična onoj koja je korištena za generiranje potpisa,
b) Izračunati $h_1 = s^{(e_1 mod\ n_1)}$
c) Potpis će biti valjan ukoliko vrijedi $h_1 = h$.

4.3.6. Primjer potpisivanja RSA algoritmom

U ovom odjeljku je prikazan praktičan primjer kako Alice šalje Bobu potpisanu poruku $h = MORE$ te kako ju Bob dešifrira i verificira. Brojčani prikaz navedene riječi će biti $h = 13151805$, sukladno rednom broju slova u engleskoj abecedi gdje *00* vrijedi za razmak, a *01* za slovo *A*. Zbog jednostavnosti izračuna i prikaza će se koristiti mali brojevi, u praksi se radi o brojevima s više desetaka znamenaka. Kako je navedeno u [12], Alice najprije bira dva prosta broja p i q:

$$p = 89, \quad q = 113$$

te potom računa

$$n_1 = pq = 10057 \quad \text{i} \quad \Phi(n1) = (p-1)(q-1) = 9856$$

Zatim bira slučajan broj $e_1 = 17$ takav da vrijedi $1 < e_1 < \Phi(n1)$ te da e_1 i $\Phi(n1)$ nemaju zajedničkih djelitelja osim broja 1, a pomoću proširenog Euklidovog algoritma izračuna broj $d_1 = 7537$ tako da vrijedi $(d_1 \cdot e_1) mod\ \Phi(n_1) = 1$. Sada je javni ključ

$$K_{javni} = (e_1, n_1) = (17, 10057),$$ a tajni $$K_{tajni} = (d_1, n_1) = (7537, 10057).$$

Nakon generiranja ključeva, Alice generira potpis što je prikazano funkcijom *sig*. Da bi to učinila, računa sljedeće:

$$s = sig_1(h) = sig_1(13151805) = h^{d_1} \ mod\ n_1 = 13151805^{7537} \ mod\ 10057 = 23869521.$$

Bobovi parametri su sljedeći:

$$p = 97,\ q = 101,\ n_2 = 9797,\ \Phi(n_2) = 9600,\ e_2 = 19,\ d_2 = 7579.$$

Alice poruku i potpis nakon toga šifrira koristeći Bobov javni ključ $e_2 = 19$ računajući

$$c_1 = h^{e_2} mod\ n_2 = 13151805^{19} \ mod\ 9797 = 19167288 \text{ i}$$

$$c_2 = s^{e_2} \ mod\ n_2 = 23869521^{19} \ mod\ 9797 = 39544580,$$

a zatim Bobu šalje par $(c_1, c_2) = (19167288, 39544580)$ koji predstavlja šifrirane te *hashirane* poruku h i potpis s. Kako bi Bob provjerio Alicein potpis, pomoću svog tajnog ključa d_2 računa

$$c_1^{d_2} \ mod\ n_2 = 19167288^{7579} \ mod\ 9797 = 13151805$$

$$c_2^{d_2} \ mod\ n_2 = 39544580^{7579} \ mod\ 9797 = 23869521.$$

Uporabom Aliceinog javnog ključa e_1, verificira njen potpis računajući

$$s^{e_1} mod\ n_1 = 23869521^{17}\ mod\ 10057 = 13151805.$$

Bob je otvoreni tekst dobio već u vrijednosti c_1, ali je provjerom Aliceinog potpisa dobio također originalni tekst tako da sa sigurnošću može zaključiti da mu je poruku zaista poslala Alice.

4.3.7. Usporedba RSA s ostalim kriptosustavima javnog ključa

RSA algoritam ima puno prednosti. Neke od njih su:

- Velika brzina odvijanja,
- Jednostavnost šifriranja i verifikacije,
- Jednostavnost razumijevanja,
- Široka rasprostranjenost,
- Velika primjenjivost u industriji.

Ipak, RSA algoritam posjeduje i određene mane:

- Vrlo sporo generiranje ključeva
- Sporo dešifriranje
- Podložnost napadima u slučaju loše implementacije.

U usporedbi s algoritmima zasnovanim na eliptičkim krivuljama, RSA je jednostavniji za implementaciju, ali ono zbog čega izbor danas sve češće pada na ECDSA jest činjenica da će se, prema [7], kod kriptosustava zasnovanih na eliptičkim krivuljama postići zadovoljavajuća sigurnost s puno kraćim ključevima nego kod kriptosustava zasnovanih na faktorizaciji velikih brojeva. Primjerice, za istu razinu sigurnosti kao kod RSA kriptosustava s duljinom ključa od 1024 bita, što je danas standardna vrijednost, kod algoritama koji koriste eliptičke krivulje je dovoljno uzeti ključ duljine 160 bita što je danas standardna vrijednost za takve algoritme. U članku [13], objavljenom 1991. godine, nizozemski inženjeri Lenstra i Verheul su konstruirali tablicu s preporukama za duljine ključeva ako se traži određena sigurnost. Osim tadašnjih preporuka, u tablici su dali i svoja predviđanja koliko će dugi ključevi morati biti u budućnosti do 2050. godine. Prema [7], u obzir su uzeli nekoliko parametara. Jedan od njih uzima u obzir valjanu pretpostavku da najbolji javno objavljeni rezultati u razbijanju pojedinih kriptosustava neće ujedno biti i najbolji mogući. U tablici 4.1. su predstavljene preporučene duljine ključeva u bitovima za kriptosustave zasnovane na faktorizaciji ili diskretnom logaritmu u konačnom polju (čiji je predstavnik u ovom radu RSA) te za kriptosustave zasnovane na diskretnom logaritmu u polju eliptičkih krivulja (ECDSA). Također, priložena je i procjena kompjuterskog vremena potrebnog za probijanje šifre u *MIPS* godinama. Jedna MIPS (eng. *million instructions per second*) godina je definirana kao

količina računanja koje se provede u godini dana na računalu koje je sposobno provesti milijun naredbi u sekundi.

Tablica 4.1. *Predviđene duljine ključeva u budućnosti*

Godina	RSA duljina ključa	ECDSA duljina ključa	Omjer ECDSA/RSA	MIPS godina
1990.	622	117	1:5	$3.51 \cdot 10^{7}$
2000.	952	132	1:7	$7.13 \cdot 10^{9}$
2010.	1369	146	1:9	$1.45 \cdot 10^{12}$
2020.	1881	161	1:12	$2.94 \cdot 10^{14}$
2030.	2493	176	1:14	$5.98 \cdot 10^{16}$
2040.	3214	191	1:17	$1.22 \cdot 10^{19}$
2050.	4047	206	1:20	$2.47 \cdot 10^{21}$

Iz tablice je vidljivo kako danas kriptosustavi zasnovani na eliptičkim krivuljama daju istu sigurnost kao i RSA algoritam uz desetak puta kraću duljinu ključeva i da će u budućnosti taj omjer dodatno rasti u korist algoritama s eliptičkim krivuljama. Ovakva svojstva tih algoritama su važna u onim primjenama i medijima koji imaju ograničen prostor za pohranu ključeva. Kako predviđaju autori u [7], intenzivnijim proučavanjem eliptičkih krivulja se očekuje da će nestati i jedan od rijetkih argumenata protiv njihove upotrebe, nedovoljna istraženost problema na kojima se zasniva ovakva vrsta kriptografije. U usporedbi s DSA algoritmom, RSA je brži u šifriranju i verifikaciji potpisa, dok je DSA brži u generiranju ključeva i dešifriranju. Algoritmi su podjednako jaki, ali u slučaju digitalnog potpisivanja, ipak se mala prednost pri odabiru daje DSA algoritmu.

5. NAPADI NA ALGORITME DIGITALNOG POTPISA

Kako je navedeno u [1], napadi na algoritme digitalnog potpisa se dijele u dvije skupine:

- Napadi uz poznavanje ključa – napadaču je dostupan javni ključ potpisnika
- Napadi uz pristup porukama – napadač ima pristup potpisanim porukama

Postoji nekoliko načina na koje si napadač može priskrbiti pristup porukama:

1. Napad na poznate poruke – napadač je ostvario pristup skupu poruka koje nije on osobno odabrao.
2. Napad na generički odabrane poruke – napadač prije krivotvorenja potpisa odabrani skup poruka daje korisniku na potpis. pri odabiru poruka koje će podmetnuti potpisniku, napadač nema uvid niti u jedan vjerodostojan potpis pa je ovakav napad neprilagodljiv. Izbor poruka ne ovisi o korisnikovom javnom ključu te se stoga napad naziva generičkim – isti skup poruka se koristi za napade na potpise svih korisnika.
3. Usmjereni napad na odabrane poruke – sličan napadu na generički odabrane poruke, napadač odabire poruke na temelju korisnikovog javnog ključa, ali bez uvida u vjerodostojan potpis. Kao i prethodni, i ovaj napad je neprilagodljiv, ali nije generički jer se napada pojedinog korisnika.
4. Prilagodljivi napad na odabrane poruke – napadač korisniku daje na potpis poruke odabrane na temelju korisnikova javnog ključa i prethodno pribavljenih potpisa.

Navedeni načini na koji su poruke dostupne napadaču su poredani prema težini, rastući prema najtežem. Najopasnija vrsta napada jesu prilagodljivi napadi na odabrane poruke, ujedno i predstavljaju najčešću vrstu napada na sustave digitalnog potpisivanja. Ukoliko se napad bilo koje vrste provede na sustav digitalnog potpisivanja, slijedi jedno od stanja:

1. Potpuno kompromitiranje sustava – napadač je u posjedu korisnikova privatnog ključa.
2. Univerzalno krivotvorenje – napadač je uspostavio algoritam funkcionalno jednak korisnikovom algoritmu za digitalno potpisivanje. Može se temeljiti na različitom, ali ekvivalentnom privatnom ključu.
3. Selektivno krivotvorenje – napadač može krivotvoriti potpise samo na odabranom skupu poruka.
4. Egzistencijalno krivotvorenje – napadač može krivotvoriti potpis najmanje jedne proizvoljne poruke.

Prema gore navedenom, sustav digitalnog potpisivanja je moguće kompromitirati potpuno, univerzalno, selektivno i egzistencijalno. Sustav za koji je dokazano da ga je nemoguće egzistencijalno kompromitirati važi za sigurniji sustav od onoga kojemu je dokazana samo nemogućnost potpunog kompromitiranja.

5.1. Napad mjerenjem vremena

Kako stoji u [14], sustav digitalnog potpisivanja je moguće napasti zahvaljujući činjenici da stvaranje digitalnog potpisa ima različita trajanja za različite poruke. Neki od razloga tomu su različite optimizacije performansi u ovisnosti o dostupnim resursima, različita stanja RAM memorije i procesorske operacije s različitim vremenom izvođenja. Preciznim mjerenjem vremena koje je potrebno za stvaranje digitalnog potpisa, vješt napadač može doći u posjed tajnog ključa korisnika. Ovaj napad je moguće primijeniti na RSA, DSA i druge algoritme digitalnog potpisa. Uz javno dostupne informacije, zatim one do kojih je moguće doći prisluškivanjem kanala te uz mjerenje vremena potrebnog za potpisivanje većeg broja poruka pomoću istog tajnog ključa, napadač može doći do navedenog tajnog ključa, bit po bit. Zbog toga je napad mjerenjem vremena neprimjenjiv ako se za svaki potpis koristi različiti tajni ključ, primjer je naveden u pododjeljku 4.1.4. Vremena koja napadač mjeri u pokušaju izvođenja ovakvog napada jesu vrijeme primitka poruke u napadnutom sustavu i vrijeme odgovora na spomenutu poruku. Općenito, napad se može promatrati kao problem uočavanja signala. Signal se sastoji od vremenskih varijacija koje su uzrokovane bitom tajnog ključa, bitom kojeg napadač pokušava otkriti i šuma koji je sastavljen od vremenskih varijacija koje uzrokuju ostali, još nepoznati bitovi, tajnog ključa. Neka postoji j poruka ($y_0, y_1, \ldots, y_{j-1}$) sa svojim vremenom trajanja potpisivanja $T_0, T_1, \ldots, T_{j-1}$). U takvom slučaju, vjerojatnost pogađanja (x_b) prvih b bitova tajnog ključa proporcionalna je izrazu:

$$P(x_b) \sim \prod_{i=0}^{j-1} F\ (T_i - t(y_i, x_b))$$

gdje su:

- $t(y_i, x_b)$ – vrijeme potrebno za prvih b iteracija digitalnog potpisa pomoću ključa x_b
- F – pretpostavljena funkcija raspodjele vjerojatnosti $T_i - t(y_i, x_b)$ za sve vrijednosti y i za ispravan x_b.

Ako se ispravno odredi x_{b-1}, vjerojatnost da je x_b točan, a x_b' netočan jest

$$\frac{\prod_{i=0}^{j-1} F\ (T_i - t(y_i, x_b))}{\prod_{i=0}^{j-1} F\ (T_i - t(y_i, x_b)) + \prod_{i=0}^{j-1} F\ (T_i - t(y_i, x_b'))}.$$

U praktičnim situacijama, ova formula nije od prevelike koristi obzirom da je funkciju *F* vrlo zahtjevno pronaći.

5.2. Napad na algoritam maskiranja poruke RSA digitalnog potpisa

Prema [15], neke implementacije RSA algoritma koriste algoritam za maskiranje poruke (eng. *padding*) prema RSASSA – PKCS1 – v1_5 specifikaciji. Unutar takvih implementacija, otkriven je propust koji omogućuje krivotvorenje potpisa. Vrijedi napomenuti kako ovo nije napad na sami RSA algoritam, već na nepravilne implementacije u procesu verifikacije potpisa. Na simpoziju kriptografije *Crypto 2006*, Daniel Bleichenebacher je izveo napad na digitalni potpis izveden RSA algoritmom. Napad je uspio uz javni parametar *e = 3*, a izveden je prema dolje navedenom.

Hash vrijednost poruke *M* se prije potpisivanja maskira kako bi se otežalo krivotvorenje potpisa pa jedan fragment poruke izgleda ovako:

`00 01 FF FF … FF 00 || ASN.1 || H(M)`

gdje je `00 01 FF FF … FF 00` vrijednost korištena za maskiranje, `ASN.1` predstavlja informacije o *hash* funkciji kao što je duljina *hash* vrijednosti, a `H(M)` je sama *hash* vrijednost poruke. Valja primijetiti kako je *hash* vrijednost poravnata udesno, odnosno, ona dolazi nakon vrijednosti maskiranja i ASN.1 informacija. U ovom slučaju, nakon što je digitalni potpis dešifriran putem javnog parametra *e = 3*, korisniku je vidljiv gore naveden znakovni niz. Iz tako vidljive poruke, izdvaja se *H(M)* i uspoređuje s *hash* vrijednošću poruke na koju se potpis odnosi. Proces izdvajanja navedene *hash* vrijednosti se izvodi na način da se označi, izdvoji sve iza maskirnih vrijednosti i ASN.1 informacija. Verifikacija potpisa se izvodi usporedbom izdvojene *hash* vrijednosti s *hash* vrijednošću izračunatom iz primljene poruke. Kako je već navedeno u opisu RSA algoritma, ako se dvije *hash* vrijednosti podudaraju, potpis se smatra valjanim. Problem se javlja u situaciji gdje neke implementacije RSA algoritma odvajaju određeni broj bitova *hash* vrijednosti na temelju njihove relativne pozicije u odnosu na maskirne vrijednosti i ASN.1, bez dodatne provjere je li netko nešto nadodao na poziciju desno od *hash* vrijednosti u znakovnom nizu. U slučaju specifikacije navedene na početku ovog potpoglavlja, kao *hash* vrijednost se

izdvaja sve što stoji iza ASN.1 informacija u znakovnom nizu. Za bilo koju poruku *M'* s *hash* vrijednošću *H(M'),* lako je pronaći treći korijen znakovnog niza koji je nakon napada u obliku

`00 01 FF FF … FF 00 || ASN.1 || H(M)|| smeće` *`ubačeno napadom`*`.`

Napadač smanjuje broj ponavljanja znakova *FF*, a veličinu smeća koje je ubacio određuje vješto na način da cijela poruka predstavlja veličinu koja je treća potencija nekog broja. Primjer je priložen uz parametar *e = 3*, ali napad je moguće izvesti i u drugim slučajevima u kojima se koristi mala vrijednosti *e* za koju se *e* – ti korijen znakovnog niza može pronaći vrlo lako. Napad se može spriječiti na način da se broj 3 ne odabire kao parametar *e* te dodatnom provjerom znakovnog niza koja provjerava ima li u znakovnom nizu dodatnih podataka osim *H(M)* vrijednosti.

6. PRIMJENA DIGITALNOG POTPISA

Digitalni potpis je koncept koji je s vremenom postao neizostavan dio svakodnevne komunikacije koja zahtijeva usluge koje koncept digitalnog potpisa može ustupiti pa je stoga ovakav sustav potpisivanja primjenjiv na širok spektar usluga i djelatnosti. Ipak, područja u kojima se digitalni potpisi najviše koriste jesu potpisivanje dokumenata, slijepi potpisi, potpisi u internetskim aplikacijama i zaštita multimedijalnih sadržaja.

6.1. Digitalno potpisivanje dokumenata

Kako je već navedeno u početku ovog rada, digitalno potpisivanje u mnogim zemljama ima status jednake pravne važnosti kao i ručno potpisivanje što potpisnika pravno obvezuje prema uvjetima dogovorenim u potpisanom dokumentu. Prema [1], to je razlog iz kojeg se preporučuje koristiti različiti par ključeva za šifriranje i za potpisivanje poruka. Prednost ovakvog izbora ključeva je mogućnost sudjelovanja u komunikaciji koja je kriptirana parom ključeva za enkripciju, što ne znači da se svaka poslana poruka potpisuje. Poruka se potpisuje tek na kraju, kada je postignut dogovor između dviju strana, parom ključeva za potpis i tek tada je potpisnik pravno vezan uz dokument koji je digitalno potpisao. Zloporaba digitalno potpisanih dokumenata je moguća obzirom da algoritmi i protokoli za digitalni potpis ne daju informaciju o vremenu u kojem je dokument potpisan. Jedan od način rješavanja ovog problema jest mogućnost da potpisnik uključi vremensku oznaku unutar digitalnog potpisa ili da u samom dokumentu navede vrijeme potpisivanja. Ipak, takav pristup nije uputan obzirom da ostavlja mogućnost namjernog navođenja krivog datuma i vremena potpisivanja. Problem vremenskog označavanja digitalnog potpisa se efikasno rješava uvođenjem sigurnih vremenskih oznaka koje dodjeljuje neovisna i pouzdana treća strana koja se naziva TSA (eng. *Time Stamping Authority*). Zbog dodatne zaštite, u potpis se dodaju višestruke sigurne oznake izdane od različitih TSA – ova. Možda i najveća prednost korištenja digitalnog potpisa pri potpisivanju dokumenata, osim osiguravanja autentičnosti i integriteta dokumenta, jest onemogućavanje nepriznavanja dokumenta od strane osobe koja potpisuje. Potpisnik ne može zanijekati poruku koju je potpisao tajnim ključem koji je poznat samo njemu osim ako se otkrije potpisnikov tajni ključ. Zbog toga je od najveće važnost tajni ključ doista i držati u strogoj tajnosti. Tajni ključ je uputno čuvati na pametnoj kartici, dodatna mogućnost je čuvanje ključa na računalu. Ipak, u tom slučaju, korisnik je u mogućnosti potpisivati dokumente samo na tom računalu, a razina sigurnosti tajnog ključa pada na onu razinu sigurnosti na kojoj se nalazi samo računalo.

6.2. Slijepi potpisi

Slijepi potpisi su vrsta digitalnih potpisa kod kojih je sadržaj poruke koja se potpisuje skriven od potpisnika prije samog potpisivanja. Najčešća primjena ovakve vrste digitalnih potpisa jesu situacije u kojima autor i potpisnik poruke nisu ista osoba. Primjer takvih situacija su digitalizirani kriptografski sustavi za glasovanje i sigurni elektronički platežni sustavi. Prema [1], još jedno područje primjene slijepih potpisa dolazi iz potrebe sprječavanja potpisnika da poveže potpisanu skrivenu poruku s kasnije otkrivenom porukom. Takve se situacije najčešće javljaju kada se zahtijeva anonimnost sudionika u komunikaciji. Slijepi se potpisi implementiraju uz pomoć algoritama digitalnog potpisivanja s javnim ključem kao što su RSA i DSA. Poruku se prije potpisivanja skriva, a zatim ju se i potpisuje nekim od navedenih algoritama. Kao i u standardnom digitalnom potpisivanju, vjerodostojnost potpisane skrivene poruke je moguće utvrditi pomoću potpisnikova javnog ključa.

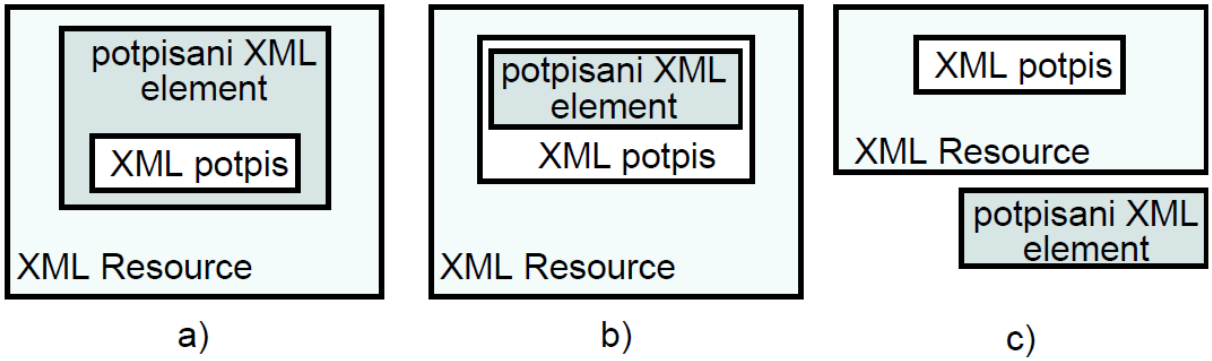
6.3. Digitalni potpisi u internetskim aplikacijama

Kako je navedeno u [1], različite sadržaje internetskih stranica je moguće potpisati takozvanim XML digitalnim potpisom. Takva vrsta potpisa je regulirana *W3C XML Signature* standardom koju izdaje krovna međunarodna standardizacijska organizacija na području internetskih tehnologija pod nazivom *World Wide Web Consortium*. XML potpis može potpisati tipove podataka kao što su:

- XML elementi, XML čvorovi,
- URI oznake
- Binarne datoteke
- Binarni podaci ugrađeni u XML dokument.

Na željenoj Internet stranici je moguće potpisati bilo koji njen element. Elementi koji se najčešće digitalno potpisuju su dijelovi HTML ili XML koda, različite vrste formulara i njihovi sadržaji na stranicama. XML potpis se može izvesti na tri načina, kako je navedeno na listi ispod i slici 6.1.

- Omotani potpis – potpis se ugrađuje u podatke koje potpisuje,
- Omotavajući potpis – potpisani podaci se ugrađuju u potpis,
- Odvojeni potpis – potpis i njime potpisani podaci su odvojeni.

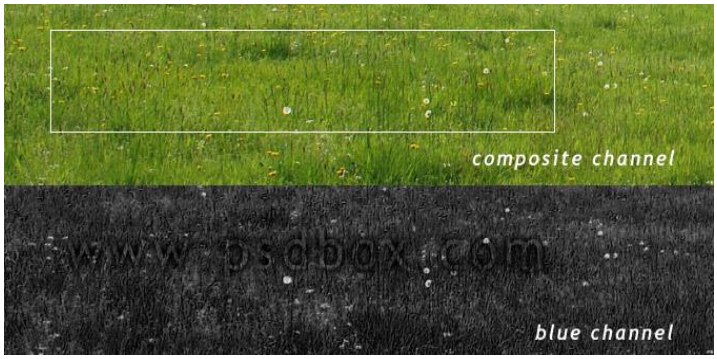


Slika 6.1. *XML potpisi: a) omotani, b)omotavajući, c) odvojeni*

6.4. Digitalno potpisivanje multimedijalnih sadržaja

Prema [1] i [17], multimedijalne sadržaje je moguće zaštititi na dva načina; vodenim žigom ili digitalnim potpisivanjem. Vodeni žig predstavlja skup informacija već ugrađenih u multimedijski sadržaj na način da ugrađeni vodeni žig može i ne mora biti vidljiv krajnjem korisniku. Vidljivi žig se primjenjuje kako bi se ograničila upotreba multimedijskog sadržaja na koji je vidljivi žig stavljen (slika 6.2.), a skriveni se žig koristi za utvrđivanje porijekla sadržaja na koji je žig stavljen (slika 6.3.).

Slika 6.2. *Vidljivi vodeni žig*



Slika 6.3. *Nevidljivi vodeni žig*

Vodeni se žigovi najčešće koriste za potpis sadržaja koji se ne obrađuje i ne sažima jer je vodeni žig izravno povezan s podacima na koje se odnosi pa ga je moguće jednostavno provjeriti. Osim toga, unutar takve vrste podataka postoji dovoljno prostora za ugradnju žiga bez narušavanja vidljivosti i razumljivosti sadržaja. Vodeni žigovi mogu biti krhki što ih čini ranjivima na obradu i sažimanje, ali ih se može izvesti i vrlo robusnima kako bi se povećala njihova otpornost na obradu multimedijskog sadržaja, a time i zaštita samog sadržaja. Ipak, obzirom da se multimedijski sadržaji gotovo nikada ne prenose bez sažimanja, puno ih je prikladnije zaštititi digitalnim potpisom. Većina standarda za sažimanje koji se danas koriste svojim djelovanjem u zaglavlju

ostavljaju posebne odjeljke za proizvoljne korisničke informacije u koje se može pohraniti digitalni potpis. Ukoliko se koristi neki standard koji ne ostavlja prostor u zaglavlju za korisničke informacije, moguće je sadržaj potpisati u odvojenoj datoteci te ju poslati zajedno sa sadržajem kojeg potpisuje. Nema prevelike razlike u potpisivanju multimedijskih sadržaja i ostalih vrsta podataka, tek u informacijama koje se koriste za stvaranje potpisa. Ukoliko se djeluje na tekstualne dokumente ili programski kôd, potpisuju se nizovi bitova koji predstavljaju tekst ili kôd što osigurava da će promjena već jednog znaka biti zamijećena tijekom provjere vjerodostojnosti potpisa. Kod multimedijalnih sadržaja se potpis vrši na način da se zaštiti njihov sadržaj, odnosno vizualne i zvučne informacije koje korisnik prima tijekom reprodukcije. Prednost je što se takva vrsta potpisa ne gubi sažimanjem jer je stopljena s podacima koje štiti. U slučaju da napadač pokuša promijeniti potpisani sadržaj, doći će do izmjene potpisanih informacija koje je moguće otkriti tijekom provjere vjerodostojnosti.

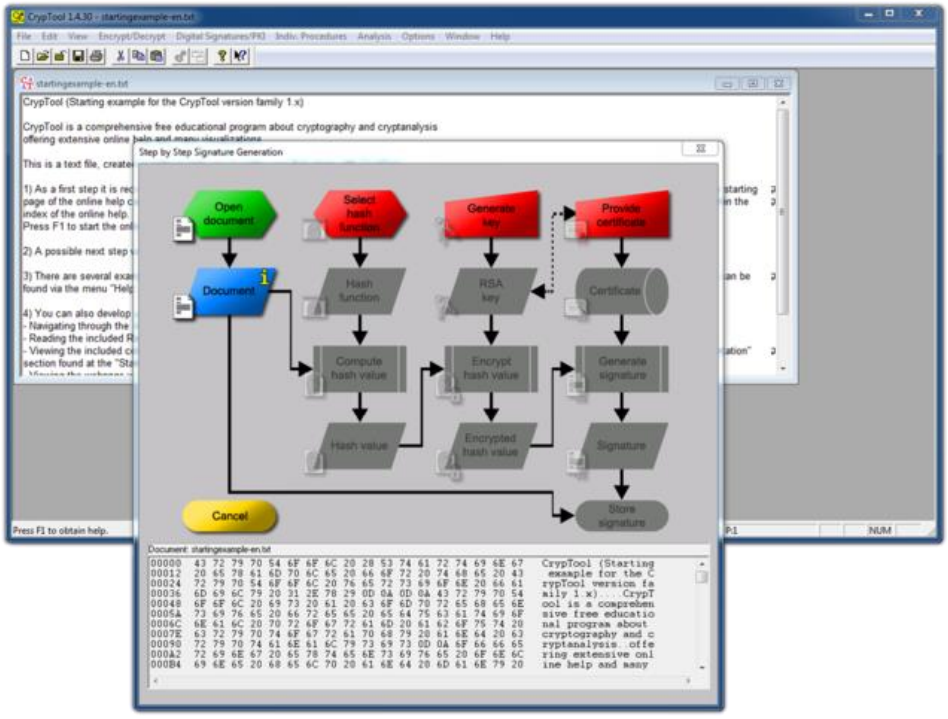
7. PROGRAMSKA IMPLEMENTACIJA DS ALGORITAMA

Jedan od zadataka ovog rada bio je obrađene algoritme primijeniti za digitalno potpisivanje proizvoljno odabrane poruke te ih međusobno usporediti. Programsko rješenje koje objedinjuje sva tri obrađena algoritma i unutar kojeg je proveden proces digitalnog potpisivanja trima različitim algoritmima jest *CrypTool 1.4.31 Beta 6b* (slike 7.1. i 7.3.).

7.1. *CrypTool* okruženje

Kako je navedeno u [18], *CrypTool* je besplatna, *open – source* aplikacija za *Windows* sustave specijalizirana za kriptografiju i kriptoanalizu. Dostupan je u 5 jezika i najrašireniji je softver svoje vrste obzirom da se istovremeno koristi i kao metoda učenja u obrazovnim sustavima zemalja svijeta i kao platforma za obuku zaposlenika u privredi. Iako je originalno napisan za interne potrebe industrije za obukom zaposlenika, do danas se razvio u važni *open – source* projekt u području kriptologije s ciljem osvještavanja o sigurnosti IT sustava. Neke od mogućnosti kojima *CrypTool* raspolaže su:

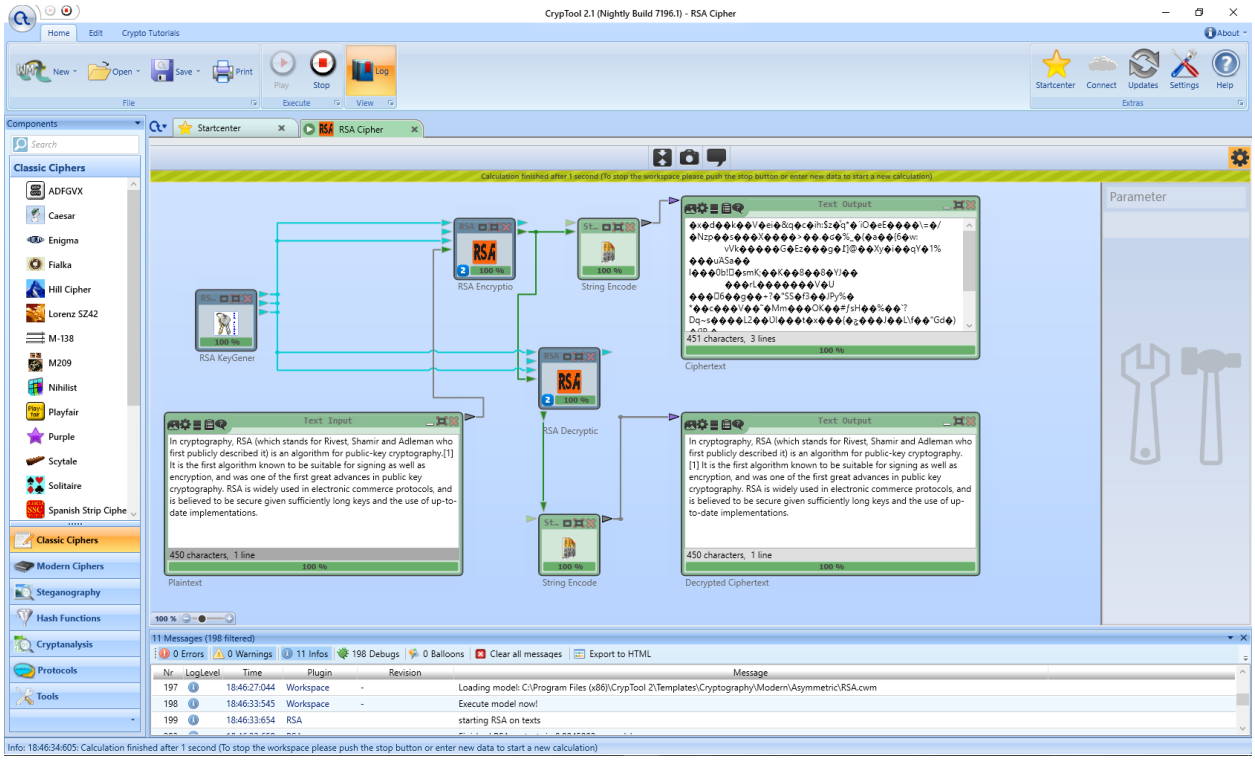
- Obrada podataka brojnim tradicionalnim i modernim kriptografskim algoritmima (enkripcija, dekripcija, generiranje ključeva, autentifikacija itd.)
- Vizualizacija različitih algoritama (Cezarom, RSA, Diffie – Helmanov, AES itd.)
- Kriptoanaliza različitih algoritama (Vigenerov, RSA, AES itd.)



Slika 7.1. *Sučelje za digitalno potpisivanje CrypTool 1 aplikacije*

Do danas su razvijene 2 glavne verzije *CrypToola*, a *CrypTool 2* (na slici 7.2.) je moderni nasljednik prethodne verzije. Poboljšanja u odnosu na prethodnu verziju su:

- **Moderno *Plug'n'Play* sučelje** – ova verzija omogućuje korisničko sučelje za vizualno programiranje kako bi protok podataka mogao biti vizualno i intuitivno popraćen,
- **Vizualizacija algoritama** – sve komponente unutar grafičkog sučelja mogu vizualno prikazati i sve svoje unutarnje operacije što olakšava korisniku shvaćanje svih kriptografskih detalja algoritama,
- **Razumljive kriptoanalitičke funkcije** – nova verzija nudi korisniku mnoštvo kriptografskih alata za analizu i probijanje tradicionalnih i modernih algoritama za šifriranje podataka.



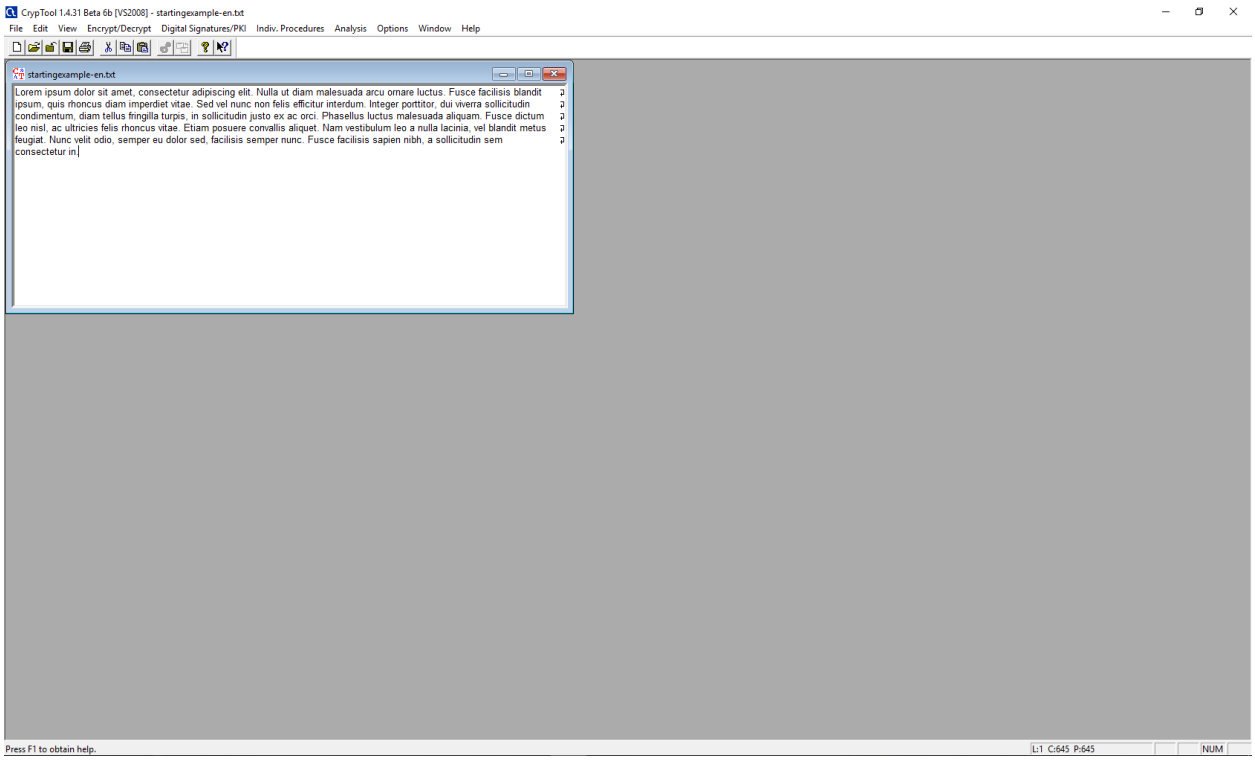
Slika 7.2. *Prikaz RSA algoritma u CrypTool 2 okruženju*

7.2. Implementacija DSA algoritma

Za polaznu poruku je odabran prvi odlomak standardnog generičkog *Lorem Ipsum* teksta:

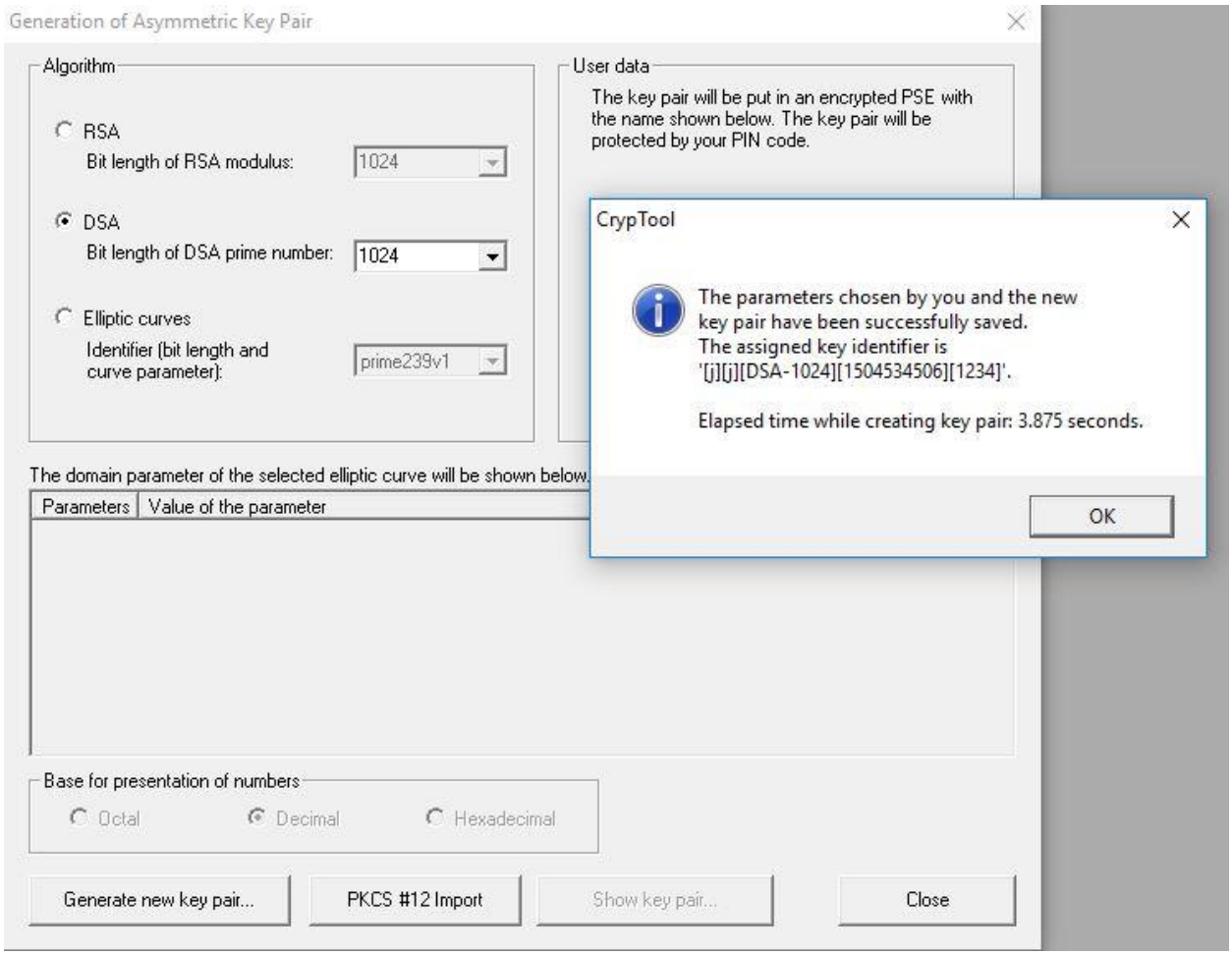
> *Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut diam malesuada arcu ornare luctus. Fusce facilisis blandit ipsum, quis rhoncus diam imperdiet vitae. Sed vel nunc non felis*

efficitur interdum. Integer porttitor, dui viverra sollicitudin condimentum, diam tellus fringilla turpis, in sollicitudin justo ex ac orci. Phasellus luctus malesuada aliquam. Fusce dictum leo nisl, ac ultricies felis rhoncus vitae. Etiam posuere convallis aliquet. Nam vestibulum leo a nulla lacinia, vel blandit metus feugiat. Nunc velit odio, semper eu dolor sed, facilisis semper nunc. Fusce facilisis sapien nibh, a sollicitudin sem consectetur in.



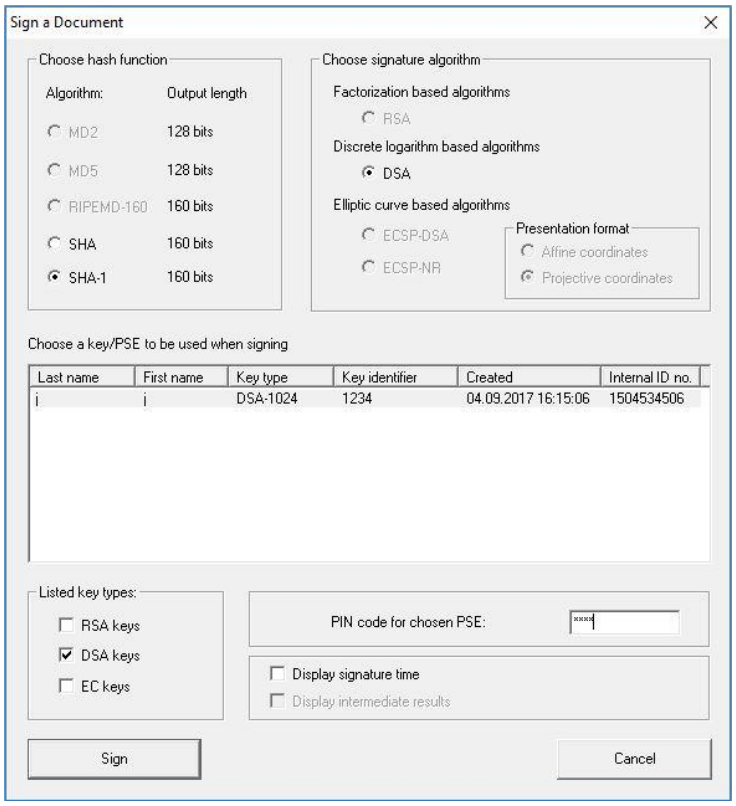
Slika 7.3. *Početni zaslon CrypToola i testna poruka*

Kako bi se mogao primijeniti DSA algoritam na polazni tekst, potrebno je generirati odgovarajući par ključeva. Duljina ključa je postavljena na 1024 bita i *CrypTool* putem generatora slučajnih brojeva uspješno konstruira par ključeva za što mu je bilo potrebno 3.875 sekundi (slika 7.4.).



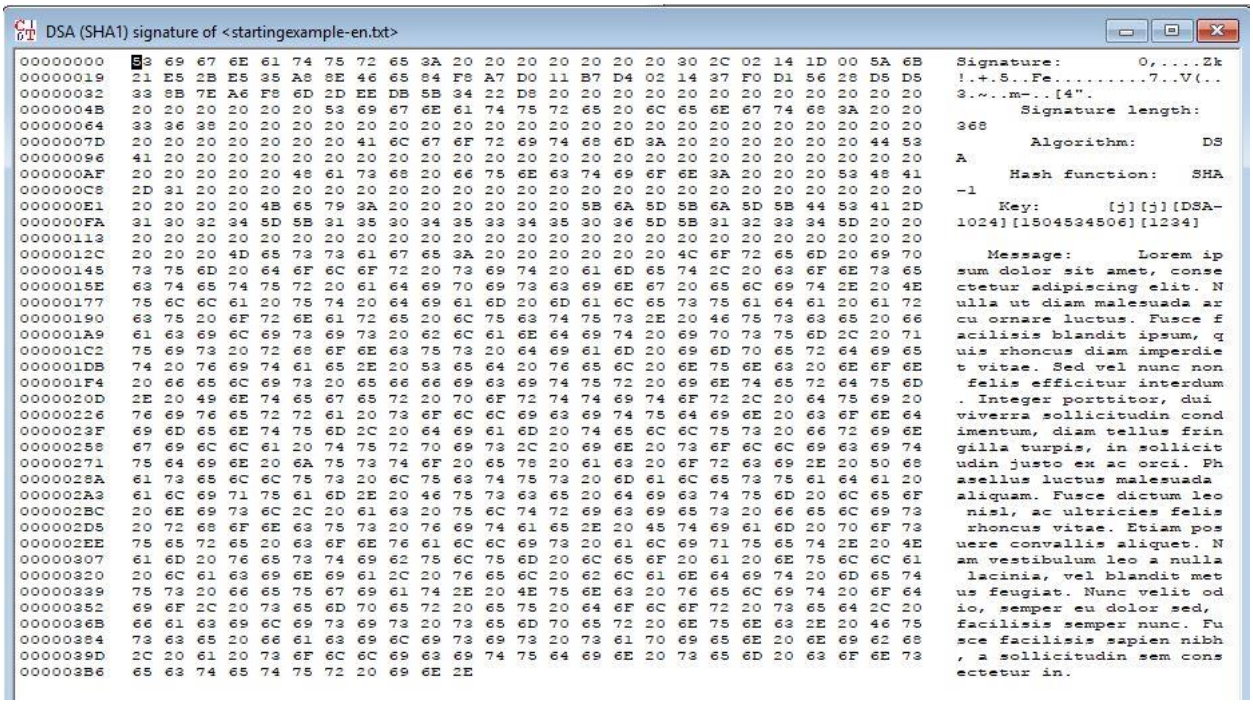
Slika 7.4. *Uspješno generiranje ključa za DSA*

U sljedećem koraku se otvara sučelje za potpisivanje dokumenta. U sučelju je potrebno odabrati jednu od *hash* funkcija koja će sudjelovati u algoritmu, željeni algoritam i par ključeva. Kako se vidi na slici 7.5., odabrani su SHA – 1 *hash* funkcija, DSA algoritam i upravo generirani par ključeva koji odgovara DSA algoritmu.



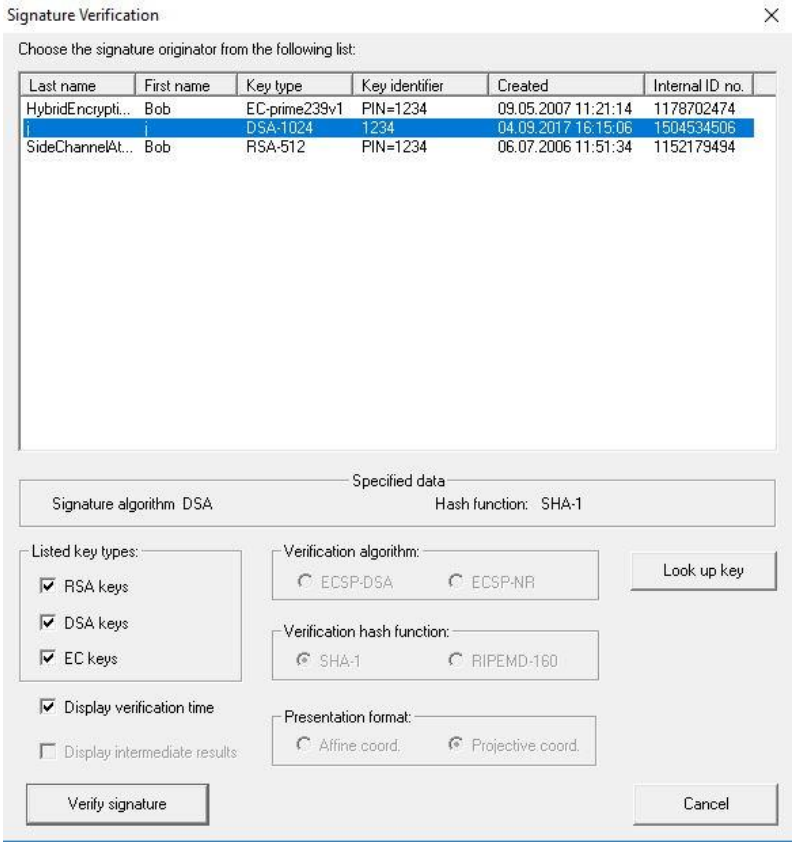
Slika 7.5. *Sučelje za potpis s odabirom parametara*

Nakon klika na gumb za potpisivanje, otvara se prozor u kojem se nalaze polazni tekst zapisan u heksadekadskom brojevnom sustavu i pridruženi mu potpis (slika 7.6.). Za generiranje potpisa je bilo potrebno manje od jedne tisućinke sekunde.



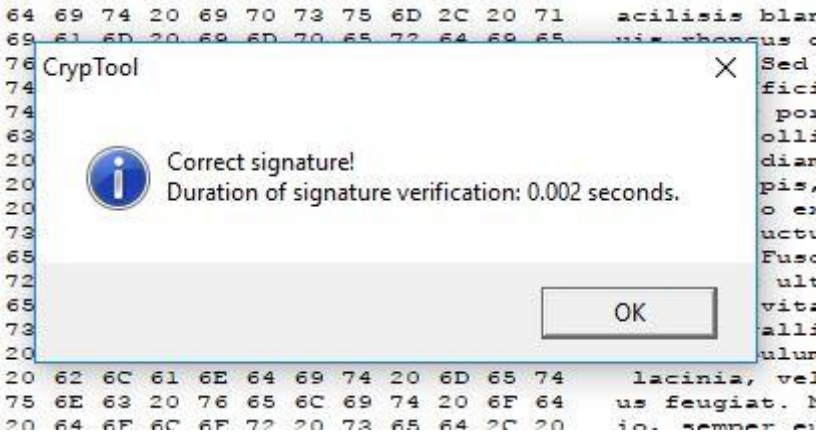
Slika 7.6. *Potpisana polazna poruka*

U sučelju za proces verifikacije je potrebno odabrati par ključeva koji je korišten i za potpisivanje kako bi se verifikacija mogla uspješno odviti (slika 7.7.).



Slika 7.7. *Sučelje za verifikaciju potpisa*

Nakon što su svi parametri pravilno odabrani, klikom na gumb za verifikaciju pojavljuje se obavijest kako je potpis uspješno verificiran. Proces verifikacije je trajao dvije tisućinke sekunde (slika 7.8.).



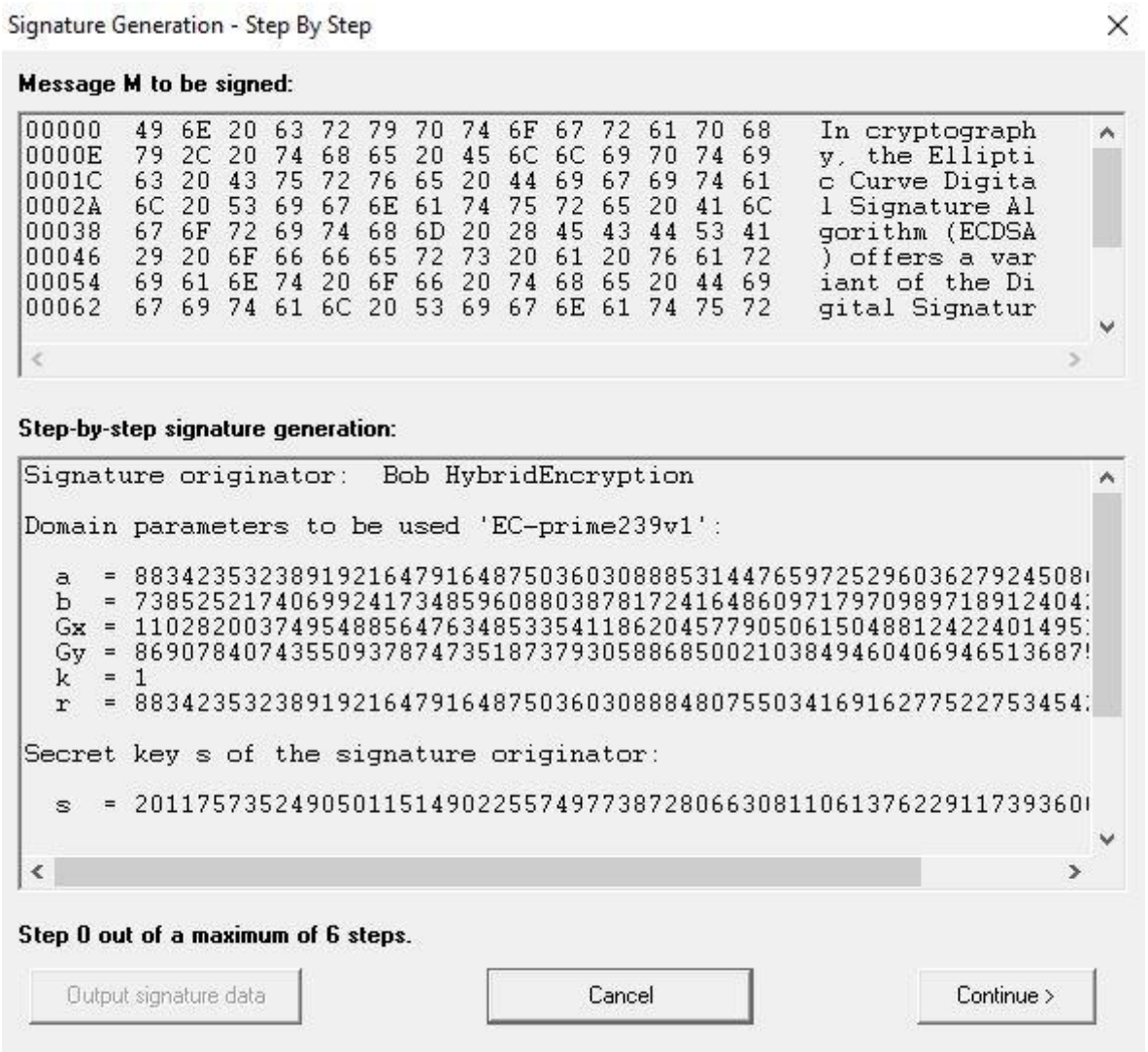
Slika 7.8. *Obavijest o uspješnoj verifikaciji*

7.3. Implementacija ECDSA algoritma

Obzirom da ECDSA zahtijeva više resursa, za polaznu poruku je odabran kraći tekst od prethodnog:

> *In cryptography, the Elliptic Curve Digital Signature Algorithm (ECDSA) offers a variant of the Digital Signature Algorithm (DSA) which uses elliptic curve cryptography.*

Sučelje za odabir parametara ECDSA je identično sučelju sa slike 7.5. Nakon što se krene u proces potpisivanja, pojavljuje se sučelje koje u 6 koraka generira i prikazuje sve parametre (slika 7.9.).



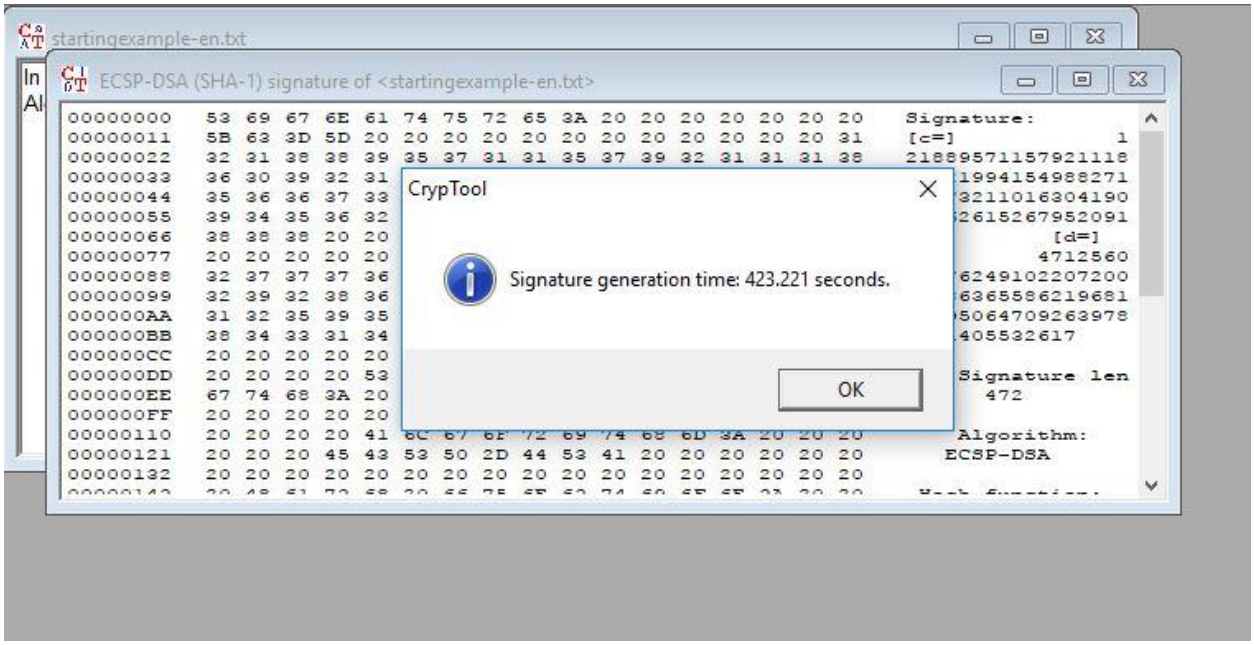
Slika 7.9. *Sučelje za praćenje ECDSA koraka*

Nakon što se izvrši svih 6 koraka, parametri koje je generirao *CrypTool* su sljedeći:

```
Signature originator:  Bob HybridEncryption

Domain parameters to be used 'EC-prime239v1':

  a  = 883423532389192164791648750360308885314476597252960362792450860609699836
  b  = 738525217406992417348596088038781724164860971797098971891240423363193866
  Gx = 110282003749548856476348533541186204577905061504881242240149511594420911
  Gy = 869078407435509378747351873793058868500210384946040694651368759217025454
  k  = 1
  r  = 883423532389192164791648750360308884807550341691627752275345424702807307

Secret key s of the signature originator:

  s  = 201175735249050115149022557497738728066308110613762291173936003932280863

Chosen signature algorithm: ECSP-DSA with hash function SHA-1

Size of message M to be signed: 169 bytes

Continue ...

Calculate a 'hash value' f (message representative) from message M, using the chosen hash function SHA-1.

  f  = 64475410196663561398178316319336199525885365542 2

Continue ...

Create a random one-time key pair (secret key, public key) = (u,V)
with the domain parameters of 'EC-prime239v1' (V=(Vx,Vy) is a point on the elliptic curve):

  u  = 7949303693350289898475002665410569634538012624260961223000871153292886 51
  Vx  = 1218895711579211186092199415498827156673211016304190945626152679520918 88
  Vy  = 8338035467915794683294467671591969657962391740969392445440743219652805 19

Continue ...

Convert the group element Vx (x co-ordinates of point V on elliptic curve) to the number i:

  i  = 121889571157921118609219941549882715667321101630419094562615267952091888

Continue ...

Calculate the number c = i mod r (c not equal to 0):

  c  = 121889571157921118609219941549882715667321101630419094562615267952091888

Continue ...

Calculate the number d = u^(-1)*(f + s*c) mod r (d not equal to 0):

  d  = 471256027776249102207200292863655862196811259506470926397884314055326 17

Continue ...

Signature generation finished.
The signature consists of the two numbers c and d.
```

Slika 7.10. *Koraci u ECDSA potpisivanju*

Nakon što je završio proces potpisivanja, pojavljuje se prozor u kojem se, kao i kod DSA, nalaze poruka u heksadekadskom obliku, odgovarajući potpis i poruka o vremenu potrebnom za izvršavanje algoritma koje iznosi 423 sekunde (slika 7.11.).



Slika 7.11. *Izlazni rezultat ECDSA potpisivanja*

Proces verifikacije se sastoji od 10 koraka unutar kojih se po pravilima ECDSA verifikacije na dva različita načina dolazi do istih vrijednosti što znači da je potpis valjan. Proces verifikacije ECDSA potpisa je trajao nešto manje od 6 sekundi. Proces verifikacije je *CrypTool* izveo na sljedeći način:

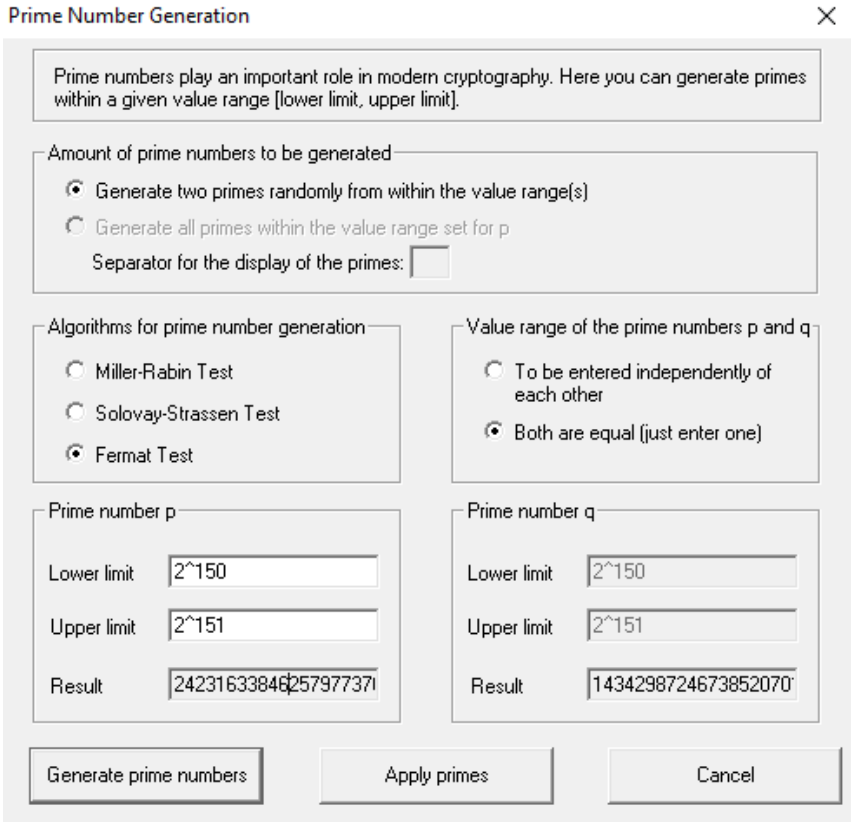
```
Signature originator:  Bob HybridEncryption

Domain parameters to be used 'EC-prime239v1':

  a  = 883423532389192164791648750360308885314476597252960362792450860609699836
  b  = 738525217406992417348596088038781724164860971797098971891240423363193866
  Gx = 110282003749548856476348533541186204577905061504881242240149511594420911
  Gy = 869078407435509378747351873793058868500210384946040694651368759217025454
  k  = 1
  r  = 883423532389192164791648750360308884807550341691627752275345424702807307

Public key W=(Wx,Wy) (W is a point on the elliptic curve) of the signature originator:

  Wx  = 518880653113931651226785874776714756597326097705158858525068487470792596
  Wy  = 733722163816356692518785906411094216416462100885831987856712585576631537

Chosen signature algorithm: ECSP-DSA with hash function SHA-1

Size of message M to be signed: 169 bytes

Bit length of c + bit length of d = 472 bits

Continue ...
```

```
Calculate a 'hash value' f (message representative) from message M, using the chosen hash function SHA-1.

  f  = 64475410196663561398178316319336199525885365542 2

Continue ...

If c or d does not fall within the interval [1, r-1] then the signature is invalid:

  c and d fall within the required interval [1, r-1].

Continue ...

Calculate the number h = d^(-1) mod r:

  h  = 238438320880445822678689368238524152465153977263960188968557857439208108

Continue ...

Calculate the number h1 = f*h mod r:

  h1 = 250215436298127783984312825657615638776789855502948537668061082867149827

Continue ...

Calculate the number h2 = c*h mod r:

  h2 = 764573677317792082263576991717295854662186053608751631087110839427781290

Continue ...

Calculate the elliptic curve point P = h1 G + h2 W
(If P = (Px, Py) = (inf, inf) then the signature is invalid):

  Px = 121889571157921118609219941549882715667321101630419094562615267952091888
  Py = 833803546791579468329446767159196965796239174096939244544074321965280519

Continue ...

Convert the group element Px (x co-ordinates of point P on elliptic curve) to the number i:

  i  = 121889571157921118609219941549882715667321101630419094562615267952091888

Continue ...

Calculate the number c' = i mod r:

  c' = 121889571157921118609219941549882715667321101630419094562615267952091888

Continue ...

If c' = c then the signature is correct; otherwise the signature is invalid:

  Verify results by comparing the two numbers c' and c.
```

Usporedba *c* i *c'* pokazuje da su vrijednosti identične što znači da je potpis valjan.

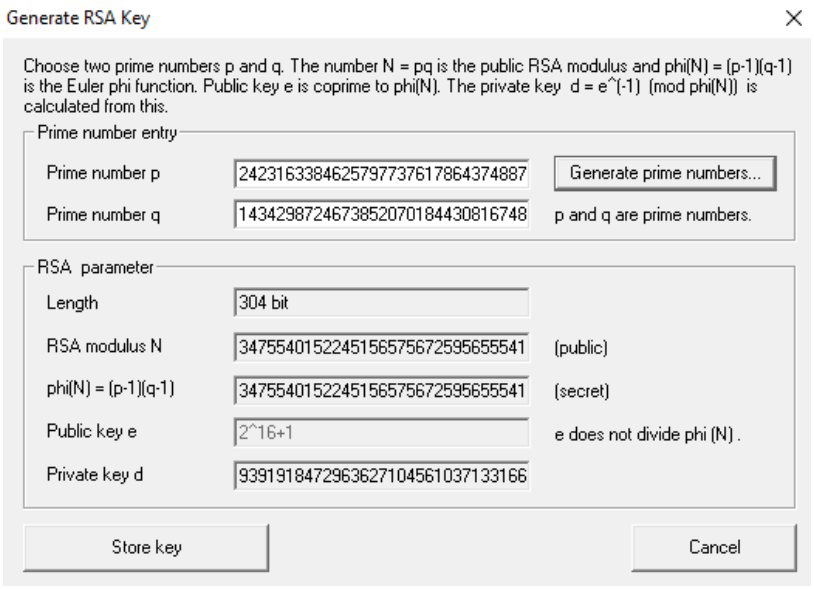
7.4. Implementacija RSA algoritma

Za polaznu poruku je ponovno odabran prvi odlomak *Lorem Ipsum* teksta. Proces potpisivanja počinje generiranjem prostih brojeva za koje treba odabrati parametre (slika 7.12.).



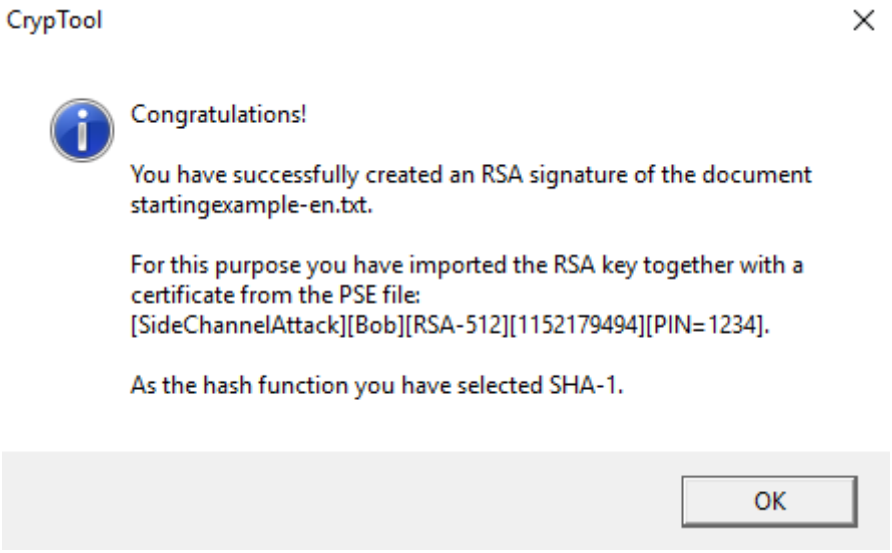
Slika 7.12. *Odabir parametara za generiranje ključeva RSA algoritma*

Nakon primjene generiranih prostih brojeva, *CrypTool* izračunava i ostale parametre te na koncu i ključeve.



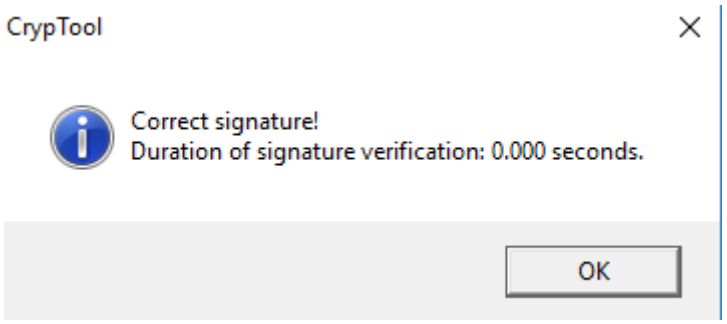
Slika 7.13. *Vrijednosti ostalih parametara i ključeva*

Nakon što se proces odvije, izlazni rezultat je obavijest o uspješnom potpisivanju poruke.



Slika 7.14. *Uspješno potpisivanje RSA algoritmom*

Proces verifikacije se također obavlja vrlo jednostavno i brzo, kao i proces potpisivanja, i verifikacija traje manje od jedne tisućinke sekunde.



Slika 7.15. *Uspješna verifikacija RSA algoritmom*

7.5. Usporedba dobivenih rezultata

Nakon što su obavljena testiranja i prikupljena vremena izvođenja pojedinih dijelova algoritama, dobiveni su sljedeći rezultati. Vrijednosti iz tablice 7.1. se mjere u sekundama.

Tablica 7.1. *Vremena potrebna za izvođenje algoritama*

	DSA	**ECDSA**	**RSA**
Potpisivanje	3.8 (generiranje ključeva) < 0.001 (potpisivanje)	<0.001 (generiranje ključeva) 423.221 (potpisivanje)	< 0.001
Verifikacija	0.002	5.73	< 0.001

Kako je i navedeno u potpoglavlju 4.3.7., RSA algoritam se pokazao kao daleko najbrži algoritam za digitalno potpisivanje. Iako se u teoriji navodi kako je potrebno veliko vrijeme za generiranje RSA ključeva, uz današnju procesorsku moć ta vrijednost pada ispod jedne tisućinke sekunde. Sam proces je vrlo brz, kako potpisivanje, tako i verifikacija za koju je također trebalo manje od jedne tisućinke sekunde vremena. Najsporijim algoritmom se pokazao ECDSA unatoč tome što je polazni tekst bio dosta kraći od polaznog teksta korištenog u potpisivanju DSA i RSA algoritmom. Iako su mu se ključevi vrlo brzo generirali, zbog matematičke složenosti algoritma proces potpisivanja je trajao gotovo 7 minuta. Osim potpisivanja, ECDSA je najsporiji i u verifikaciji za koju mu je trebalo između 5 i 6 sekundi. Kako je vidljivo u tablici 7.1., DSA je vrlo brz u potpisivanju, ali mu je za generiranje ključeva bilo potrebno nešto manje od 4 sekunde. Verifikacija je također bila dosta brza, trajala je 2 tisućinke sekunde.

8. BUDUĆNOST DIGITALNOG POTPISIVANJA

Potpis kao takav je temelj poslovnog djelovanja i transakcija tisućama godina. Ipak, alati i uvjeti na tržištu se mijenjaju pa tako olovke i papire polako zamjenjuju bitovi i bajtovi. Informacije se generiraju i prenose češće i brže nego ikad u povijesti. Moderni alati za komunikaciju su stvorili gotovo neograničene prilike za poboljšanje uvjeta u kojima podaci putuju, ali nisu eliminirali pravnu, kulturološku i praktičnu potrebu za opipljivim i dugotrajnim predstavnikom obvezivanja. Digitalni potpisi su odgovor današnjice na tu staru potrebu. Oni su već sada omogućili svijetu rutinske transakcije u svim djelatnostima putem računalnih mreža. Uspostavljena je politička, pravna i tehnička infrastruktura koja je potrebna da podrži implementaciju na svjetskoj bazi i danas digitalni potpisi za velike iznose smanjuju količinu papirologije te skraćuju vrijeme potrebno za potpisivanje omogućavajući tako da se to vrijeme raspodjeli na druge aktivnosti. Kako je navedeno u [1], ukoliko se nastave današnji trendovi u korištenju digitalnog potpisa, oni će s vremenom u potpunosti zamijeniti pisane potpise na svim mogućim razinama; od osobnih pisama do dugoročnih korporativnih ugovora. Mnogi se današnji računalni stručnjaci slažu da će u budućnosti odgovarajući digitalni potpis biti potreban za pokretanje svih programskih paketa i pristup svim datotekama. Za očekivati je povećanje važnosti i učestalosti primjene digitalnog potpisivanja pa je vrlo važno da organizacije koje imaju ovlasti kao što su IEEE (*Institute of Electrical & Electronics Engineers*) i ANSI (*American National Standards Institute)* konstantno standardiziraju algoritme za digitalno potpisivanje. Također, važno je navesti kako se algoritmi za digitalno potpisivanje s eliptičkim krivuljama sve češće koriste i kako će porast njihove popularnosti s vremenom istisnuti iz uporabe algoritme koji se temelje na drugačijoj vrsti problema. U skladu s time, sve veće prihvaćanje digitalnog potpisa kao ravnopravne zamjene ručnog potpisa će povećati i učestalost napada i pokušaja zloporabe. Zbog toga je i dalje potrebno razvijati moderne sigurnosne mehanizme koji će takve napade spriječiti. Kao najveći prioritet sigurnosti sustava digitalnog potpisivanja postavljen je problem čuvanja tajnosti privatnog ključa.

9. ZAKLJUČAK

Zadatak ovog rada bio je analizirati najčešće algoritme koji se koriste u kriptosustavu digitalnog potpisa. Bilo je potrebno detaljno analizirati matematičku podlogu, načina rada i sigurnost pojedinih algoritama te ih međusobno usporediti. Redom su u poglavljima dani povijesni presjek razvoja sustava potpisivanja, kako ručnog, tako i digitalnog, a zatim su navedeni temeljni principi kriptografije na kojima počiva čitav sustav digitalnog potpisivanja. Nadalje, u glavnom dijelu ovog rada su detaljno opisana tri algoritma za digitalno potpisivanje čija analiza sadrži povijesni presjek, teorijske principe, korake u podalgoritmima generiranja ključeva, potpisivanja i provjere potpisa te primjere za svaki od navedenih algoritama. Nakon toga, opisane su najčešće vrste napada na opisane algoritme te su na nekoliko mjesta u radu priložene upute za optimalan izbor parametara kako ne bi došlo do sigurnosnih propusta. U radu su navedena i najčešća područja primjene digitalnih potpisa, a za kraj je priloženo i predviđanje kako će razvoj koncepta digitalnog potpisa utjecati na svijet komunikacija u budućnosti. U sklopu rada su, osim detaljne teorijske analize odabranih algoritama, izvršene i praktične analize njihove funkcionalnosti. Ostvareni rezultati su većinom opravdali očekivanja postavljena teorijskom analizom algoritama. Odnos između trajanja izvođenja algoritama je bio u skladu s očekivanjima, tek pokoje značajno kvantitativno vremensko odstupanje se može pripisati prilagodbi ulaznih parametara algoritma dostupnim resursima na računalu na kojem su algoritmi izvršavani. Na kraju ovog rada i nakon svega navedenog, može se zaključiti kako su digitalni potpisi bez dileme jedan od najvažnijih principa današnje razmjene informacija. Iako se nerijetko ne osvrćemo na njihovu prisutnost i često ih podrazumijevamo kao nešto što je oduvijek među nama, nemoguće je u dovoljnoj mjeri istaknuti važnost njihove prisutnosti u komunikaciji putem digitalnih medija. Tu tezu potvrđuje i činjenica da se digitalni potpisi koriste na svim mogućim razinama digitalne komunikacije, počevši od, gledano iz perspektive prioriteta važnosti zaštite podataka, najniže, ali i najzastupljenije – civilne razmjene poruka putem mobilnih telefona između fizičkih osoba, preko interkontinentalnih bankovnih transakcija čiji iznosi na dnevnoj bazi dosežu milijarde dolara pa sve do strogo kontroliranih i najtajnijih aktivnosti te razmjene informacija od svjetske važnosti unutar vojnih i obavještajnih djelatnosti. Osvrćući se na ovu temu iz te perspektive, završni zaključak ovog rada ističe kako je cjelokupnu problematiku digitalnog potpisivanja važno držati i održavati na najvišem mogućem nivou i na svim razinama digitalne komunikacije kako bi ona i dalje iz potaje, ali vrlo efikasno i svrsishodno služila čovječanstvu koje danas više ne može bez svakodnevne razmjene neograničene količine informacija.

10.LITERATURA

[1] CARNet CERT, LS&S, *Digitalni potpis*
www.cis.hr/www.edicija/LinkedDocuments/CCERT-PUBDOC-2007-02-182.pdf, lipanj 2017.

[2] *Infographic: the History of the Signature*
https://www.helpsystems.com/resources/articles/infographic-history-signature, lipanj 2017.

[3] *What is a digital signature?*
http://computer.howstuffworks.com/digital-signature.htm, lipanj 2017.

[4] *Digital signature algorithm*
https://en.wikipedia.org/wiki/Digital_Signature_Algorithm, lipanj 2017.

[5] T. Pornin, *Deterministic Usage of the Digital Signatuer Algorithm (DSA) and Elliptic Curve Digital Signature Algorithm (ECDSA)*, RFC 6979
https://tools.ietf.org/pdf/rfc6979.pdf, lipanj 2017.

[6] D. Johnson, A. Menezes, S. Vanstone, *The Elliptic Curve Digital Signature Algorithm (ECDSA)*, 2001.

[7] A. Dujella, M. Maretić, *Kriptografija*, Element, Zagreb, 2007.

[8] *Elliptic Curve Diffie – Hellman*
https://www.youtube.com/watch?v=F3zzNa42-tQ, lipanj 2017.

[9] *Martijn Grooten - Elliptic Curve Cryptography for those who are afraid of maths*
https://www.youtube.com/watch?v=yBr3Q6xiTw4, lipanj 2017.

[10] A. Vranješ, *Sigurnost RSA algoritma – analiza mogućih napada*, 2016.

[11] *RSA (cryptosystem)*
https://en.wikipedia.org/wiki/RSA_(cryptosystem), lipanj 2017.

[12] B. Ibrahimpašić, E. Liđan, *Digitalni potpis*, Osječki matematički list 10(2010), 139 – 148

[13] A.K. Lenstra, E.R. Verheul, *Selecting Cryptographic Key Sizes*, 2001.

[14] P. C. Kocher, *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems*
http://www.cse.msstate.edu/~ramkumar/TimingAttacks.pdf, lipanj 2017.

[15] *An Attack on RSA Digital Signature*

http://csrc.nist.gov/groups/ST/toolkit/documents/dss/RSAstatement_10-12-06.pdf, lipanj 2017.

[16] *RSA Laboratories – 7.3. What is a blind signature scheme?*

https://www.emc.com/emc-plus/rsa-labs/standards-initiatives/what-is-a-blind-signature-scheme.htm, lipanj 2017.

[17] C.Y. Lin, *Watermarking and Digital Signature Techniques for Multimedia Authentication and Copyright Protection*, 2000.

http://www.ee.columbia.edu/ln/dvmm/publications/PhD_theses/cylin-thesis.pdf, lipanj 2017.

[18] *CrypTool – Cryptography for everyone*

https://www.cryptool.org/en/cryptool1, rujan 2017.

11. SAŽETAK

Digitalni potpisi služe kako bi zaštitili poruku od neželjenih izmjena kroz kanal i kako bi posvjedočili da je pošiljatelj doista ona osoba koja to i tvrdi. U svojoj infrastrukturi, sustav digitalnog potpisivanja koristi princip kriptografije s javnim ključem i digitalne certifikate. Najčešći algoritmi za implementaciju digitalnih potpisa jesu RSA, DSA i ECDSA. RSA i DSA se temelje na problemu faktorizacije velikih brojeva, a ECDSA na eliptičkim krivuljama. Općenito, za istu duljinu ključa, ECDSA algoritam je efikasniji od algoritama koji se zasnivaju na problemu faktorizacije velikih brojeva. Postoji nekoliko vrsta napada na algoritme digitalnog potpisivanja od kojih su najčešći napadi mjerenjem vremena i napadi usmjereni na maskirnu vrijednost poruke. Algoritmi za digitalno potpisivanje se primjenjuju za potpisivanje dokumenata, za slijepo potpisivanje, potpisivanje internetskih stranica i sadržaja na njima te za potpisivanje multimedijalnih sadržaja. Očekuje se da će digitalni potpisi u budućnosti potpuno zamijeniti ručno potpisivanje.

ABSTRACT

Digital signatures provide protection to the messages from unwanted changes through the channel and testify that the sender is really the person who claims it. In its infrastructure, the digital signature system uses the principle of public key cryptography and digital certificates. The most common algorithms for implementing digital signatures are RSA, DSA and ECDSA. RSA and DSA are based on the problem of factorization of large numbers, and ECDSA on elliptic curves. Generally, for the same key length, the ECDSA algorithm is more efficient than algorithms based on the problem of factorization of large numbers. There are several types of attacks on digital signature algorithms, most commonly being timing attacks and attack based on the message padding value. Digital signature algorithms apply to signing documents, blind signing, signing up of websites and content on them, and signing up for multimedia content. It is expected that manual signatures in the future will be completely replaced by digital signatures.

12. ŽIVOTOPIS

Antonio Vranješ rođen je 14. listopada 1993. godine u Osijeku. 2000. godine upisuje OŠ „Retfala“ koju završava 2008. godine i u kojoj je u svih osam razreda školovanja postizao odlične uspjehe s prosjekom ocjena 5.0 i uzorno vladanje. 2006. i 2008. je sudjelovao na državnoj smotri LiDraNo u Zadru i Dubrovniku kao glavni urednik najboljeg školskog lista – KBG – a, a na kraju osnovnoškolskog školovanja biva proglašen najboljim učenikom generacije od strane Nastavničkog vijeća. 2008. godine upisuje III. gimnaziju Osijek u kojoj je u četiri godine postizao odlične uspjehe s najvišim prosjecima ocjena te uzorno vladanje. U dva navrata je sudjelovao na županijskim natjecanjima iz informatike u području programiranja. Na kraju srednjoškolskog obrazovanja, kao jedan od najboljih učenika generacije, od strane III. Gimnazije Osijek, nagrađen je izravnim upisom na tadašnji Elektrotehnički fakultet Osijek. 2015. godine završava preddiplomski studij elektrotehnike, smjer Elektroenergetika i stječe zvanje inženjera prvostupnika elektrotehnike. Iste godine na Fakultetu elektrotehnike, računarstva i informacijskih tehnologija u Osijeku upisuje diplomski studij elektrotehnike, smjer Komunikacije i informatika, podsmjer Mrežne tehnologije. Član je reprezentativnog orkestra Tamburaške škole Batorek iz Osijeka s kojim je do sada posjetio Bugarsku i Kinu.

Antonio Vranješ