

UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Računalništvo in matematika – 2. stopnja

Tim Kalan

VEČSTRANSKI PODPISI

Magistrsko delo

Mentor: doc. dr. Tilen Marc

Ljubljana, 2024

Zahvala

Neobvezno. Zahvaljujem se ...

Kazalo

1	Uvod	1
2	Kriptografske osnove	1
2.1	Aritmetika v \mathbb{Z}_p^*	1
2.2	Pridobivanje velikih naključnih praštevil	2
2.3	Zgoščevalne funkcije	2
2.4	Kriptografija javnega ključa	3
2.5	Digitalni podpisi	5
2.6	Varnost	6
3	Schnorrov podpis	7
3.1	Varnost Schnorrovega podpisa	9
4	Pregled skupinskih podpisov	10
4.1	Skupinski podpisi	10
4.2	Pragovni podpisi	11
4.3	Večstranski podpisi	11
4.4	Agregirani podpisi	11
5	Večstranski Schnorrov podpis	11
5.1	Večstranski podpis podskupine z odgovornostjo	12
5.1.1	Robustnost, varnost in napadalec	13
5.1.2	Model slučajnega oraklja	14
5.2	Konstrukcija večstranskega Schnorrovega podpisa	14
5.2.1	Naivna verzija	14
5.2.2	Generiranje parametrov: Predpostavka DL	15
5.2.3	Napad na generiranje ključev: Dokazi brez razkritja znanja	16
5.3	Definicija večstranskega Schnorrovega podpisa	17
5.4	Varnost večstranskega Schnorrovega podpisa	17
6	Večstranski podpisi v splošnem	17
	Literatura	19

Program dela

Mentor naj napiše program dela skupaj z osnovno literaturo.

Osnovna literatura

1. S. Micali, K. Ohta in L. Reyzin, *Accountable-subgroup multisignatures*, v: Proceedings of the 8th ACM conference on Computer and Communications Security (ur. P. Samarati), ACM, Philadelphia, PA, USA, 2001, str. 245–254, DOI: 10.1145/501983.502017, dostopno na <https://doi.org/10.1145/501983.502017>.

Podpis mentorja:

Večstranski podpisi

POVZETEK

Tukaj napišemo povzetek vsebine. Sem sodi razlaga vsebine in ne opis tega, kako je delo organizirano.

Multisignatures

ABSTRACT

An abstract of the work is written here. This includes a short description of the content and not the structure of your work.

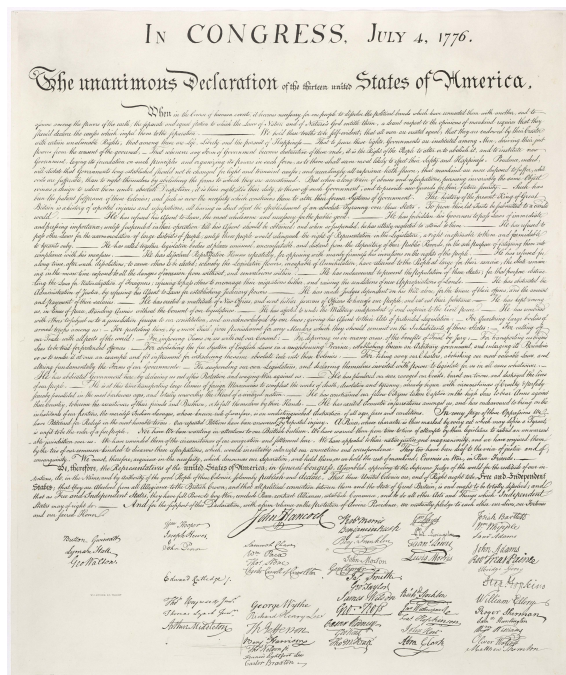
Math. Subj. Class. (2020): 94A60, 11T71

Ključne besede: digitalni podpis, kriptografija

Keywords: digital signature, cryptography

1 Uvod

Odkar se je na svetu pojavil koncept (ročnega) podpisa, je večina primerov uporabe temeljila na pridobivanju podpisov več deležnikov. Odličen primer je npr. Deklaracija neodvisnosti Združenih držav Amerike 1.



Slika 1: Deklaracija neodvisnosti Združenih držav Amerike s podpisi podpornikov spodaj.

V prejšnjem stoletju je vzpon računalnika in napredek v kriptografiji privedel do *digitalnih podpisov*. Ti odlično nadomeščajo ročni podpis, prav tako omogočajo, da se skupina podpiše tako, da vsak član poda svoj podpis. Vendar tu lahko z malo matematike poskrbimo, da se skupina lahko podpiše na način, da vsi člani skupaj oddajo en sam podpis, ki priča o podpisu celotne skupine. Tako razbremenimo preverjalca podpisov, kar je ključno v sistemih, kjer je računska moč omejena ali pa draga (npr. pri tehnologiji veriženja blokov).

2 Kriptografske osnove

Preden si lahko pogledamo točno kako lahko skupina generira en sam podpis besedila, si moramo pogledati nekaj kriptografskih osnov. Bolj komplicirane stvari bodo opisane sproti, ideja tega poglavja je predstaviti stvari, ki so predpogoj za branje kakršnegakoli kriptografskega besedila.

2.1 Aritmetika v \mathbb{Z}_p^*

V kriptografiji imamo pogosto opravka z multiplikativnimi grupami, najenostavnejša med njimi (in tudi tradicionalno največ uporabljena) je *multiplikativna grupa naravnih števil modulo p* . Njeni elementi so števila v $\{0, 1, \dots, p-1\}$, ki so tuja

številu p . V posebnem primeru, ko je p praštevilo, so to torej števila $\{1, 2, \dots, p-1\}$ in je red grupe $\text{ord}(\mathbb{Z}_p^*) = |\mathbb{Z}_p^*| = p-1$. Operacija v tej grupi je, kot ime že nakazuje, množenje modulo p .

Spomnimo se, da je red elementa g najmanjše naravno število q , da velja $g^q \equiv 1 \pmod{p}$, kjer je 1 enota za množenje. V primeru, da je p praštevilo, je grupa \mathbb{Z}_p^* ciklična, kar pomeni, da v njej obstaja element g , katerega red je enak redu grupe, torej $\text{ord}(g) = p-1$. V tem primeru se g imenuje *generator*.

Primer 2.1 (Grupa \mathbb{Z}_{11}^*). Ker je 11 praštevilo, v grupi \mathbb{Z}_{11}^* obstaja generator, oz. je grupa ciklična z redom $10 = 11-1$. Z zaporednim računanjem potenc lahko vidimo, da je $\text{ord}(2) = 10$, torej je 2 generator.

$$\begin{array}{ll} 2^1 \equiv 2 \pmod{11} & 2^6 \equiv 9 \pmod{11} \\ 2^2 \equiv 4 \pmod{11} & 2^7 \equiv 7 \pmod{11} \\ 2^3 \equiv 8 \pmod{11} & 2^8 \equiv 3 \pmod{11} \\ 2^4 \equiv 5 \pmod{11} & 2^9 \equiv 6 \pmod{11} \\ 2^5 \equiv 10 \pmod{11} & 2^{10} \equiv 1 \pmod{11} \end{array}$$

◇

Opomba 2.2. Spomnimo se *kongruence*: $a \equiv b \pmod{m} \iff m \mid a-b$.

2.2 Pridobivanje velikih naključnih praštevil

2.3 Zgoščevalne funkcije

V grobem so (kriptografske) *zgoščevalne funkcije* funkcije, ki prejmejo poljubno dolg binarni niz (ki lahko predstavlja besede, številke, celotne dokumente, ...), vrnejo pa binarni niz, ki ima vnaprej določeno dolžino. Tem rezultatom pravimo *zgostitve*. Namen zgoščevalnih funkcij je za dokument ustvariti unikaten niz, ki zelo verjetno identifikira dokument. V grobem si od zgoščevalnih funkcij želimo naslednje lastnosti:

- **Določenost** pomeni, da bo zgoščevanje enakih nizov vedno privedlo do enake zgostitve.
- **Učinkovitost** pomeni, da lahko računalnik izračuna poljubno zgostitev v doglednem času. Izračun zgostitve mora biti računsko učinkovit.
- **Enosmernost** pomeni, da iz predložene zgostitve zelo težko ugotovimo, kateri niz je funkcija prejela kot vhod. Tej lastnosti pravimo tudi *odpornost na praslisko*.
- **Odpornost na drugo praslisko** pomeni, da če poznamo niz in njegovo zgostitev, je zelo težko najdemo drug niz z enako zgostitvijo.
- **Skoraj brez trčenj** pomeni, da je verjetnost, da imata dva izraza enako zgostitev, majhna. Želimo tudi, da je zelo težko najti dva niza z enako zgostitvijo.
- **Učinek plaz** pomeni, da majhna sprememba v vhodnem nizu povzroči veliko spremembo v zgostitvi.

Definicija 2.3. Kriptografska zgoščevalna funkcija $H : \{0, 1\}^* \rightarrow \{0, 1\}^n$ je funkcija, ki slika binarne nize m poljubne dolžine v njihove **zgostitve** $H(m)$, tj. binarne nize vnaprej določene dolžine n . Zadoščati mora naslednjim lastnostim:

- (določenost) $\forall m : ((h_1 = H(m) \wedge h_2 = H(m)) \implies h_1 = h_2)$.
- (učinkovitost) Izračun funkcije H mora biti računsko učinkovit.
- (odpornost na prasluko) Če poznamo zgostitev h je računsko neizvedljivo najti niz m , da velja $h = H(m)$.
- (odpornost na drugo prasluko) Če poznamo niz m_1 je računsko neizvedljivo najti zgostitev m_2 , da velja $H(m_1) = H(m_2)$.
- (odpornost na trčenja) Računsko neizvedljivo je najti dva niza m_1 in m_2 , da velja $H(m_1) = H(m_2)$.
- (učinek plaz) Vsaka sprememba vhoda povzroči, veliko spremembo v zgostitvi. Vsak bit zgostitve se spremeni z verjetnostjo vsaj $1/2$.

Primer 2.4. Ena izmed najbolj znanih zgoščevalnih funkcij je **SHA-256**. Njeno ime pomeni *Secure Hash Algorithm* (slov. varen zgoščevalni algoritem), 256 pa predstavlja dolžino zgostitve. Pogostokrat to ime zasledimo pri nameščanju programske opreme, služi kot avtentikator, da smo res naložili pravo stvar.

Za primer si lahko ogledamo zgostitvi dveh podobnih nizov, *Ljubljana* in *Ljubljena*. Kljub podobnosti bomo videli, da sta rezultata popolnoma drugačna, kar si tudi želimo pri zgoščevalnih funkcijah.

SHA-256(Ljubljana) =
b7f147d8b4a6703a951336654355071f9752385f85d0860379e99b484aee7a82

SHA-256(Ljubljena) =
995d2d8ffb40e1838219e65dd2c665701ba34a90e11f7195a4b791838b6787fe

Za preglednost nismo prevajali besed v binarne nize, to bi storili npr. z ASCII ali UTF-8 tabelo. Prav tako smo rezultat napisali v šestnajstiškem sistemu, saj je tako krajši. \diamond

2.4 Kriptografija javnega ključa

Prve šifre, ki smo jih uporabljali ljudje, so bile *simetrične*, kar pomeni, da sta osebi za komunikacijo obe morali poznati skriven *ključ*, ki je definiral, kako je bila šifra ustvarjena.

Primer 2.5 (Cezarjeva šifra). Ena najbolj znanih šifer, ki izvira iz Antičnega Rima, je *Cezarjeva šifra*. Njen ključ je število, ki je krajše od dolžine naše abecede, v Cezarjevem primeru je bilo to število 3. Šifra potem deluje tako, da vsako črko zamaknemo za toliko mest v abecedi, kolikor definira ključ. Npr. za slovensko abecedo, bi šifra zamaknila črke:

A B C Č D E F G H I J K L M N O P R S Š T U V Z Ž
 Č D E F G H I J K L M N O P R S Š T U V Z Ž A B C

To bi izraz JAVNI KLJUČ preslikalo v MČARL NOMŽF. Cezarjeva šifra se imenuje tudi *zamična šifra*. ◇

V prejšnjem stoletju pa se je pojavila alternativa, imenovana *asimetrična kriptografija*, oz. *kriptografija javnega ključa*. Glavna prednost te je, da osebi za komunikacijo ne rabita poznati enakega skrivnega ključa, vendar ima vsak od njiju par ključev, ki ju imenujemo *javni ključ* (angl. *public key*) in *zasebni ključ* (angl. *secret/private key*) in označimo kot par (pk, sk) . Vsaka oseba objavi svoj javni ključ in poskrbi, da nihče ne izve, kaj je njen zasebni ključ.

Šifriranje potem poteka tako, da pridobimo javni ključ od osebe, s katero želi komunicirati, ga uporabi za šifriranje in objavi šifrirano sporočilo. Lastnik ustreznega zasebnega ključa (vsakemu javnemu pripada natanko en zasebni) potem pridobi šifrirano sporočilo in ga z zasebnim ključem odšifrira. Kriptosistemi delujejo na način, da lahko sporočilo, šifrirano z javnim ključem odšifrira samo ustrezen zasebni ključ. Tako zagotovimo varno komunikacijo.

Primer 2.6 (RSA). En prvih algoritmov javnega ključa, ki se uporablja še danes, je *RSA*. Njegova varnost izhaja iz (domnevne) težavnosti problema iskanja prafaktorjev. Svoj ključ definiramo tako, da si izberemo dve (zelo veliki) praštevili p in q , ter ju zmnožimo v $n = pq$. Za primer vzemimo $p = 23$ in $q = 17$. n je potem enak 391. Izbrati si moramo še eksponent e , vzemimo npr. $e = 3$. Naš javni ključ je potem par

$$(n, e) = (391, 3).$$

Postopek šifriranja poteka tako, da oseba, s katero komuniciramo, izbere sporočilo m , npr. $m = 10$, pridobi naš javni ključ, in izračuna šifro c kot

$$c = m^e \bmod n = 10^3 \bmod 391 = 218.$$

Dogovoriti se moramo še o zasebnem ključu. Za to bomo potrebovali eksponent za odšifriranje d , tako da bo veljalo

$$(m^e)^d \equiv 1 \pmod{\varphi(n)},$$

kjer φ označuje Eulerjevo funkcijo ϕ . Iščemo torej multiplikativni inverz eksponenta e , modulo $\varphi(n)$. V našem primeru je to $d = 235$. Zasebni ključ je potem

$$(p, q, d) = (23, 17, 235).$$

Iz zasebnega ključa torej lahko kadarkoli izračunamo javnega, saj enostavno zmnožimo p in q ter izračunamo inverz, v splošnem pa iz n učinkovito ne moremo pridobiti faktorjev p in q , kar nam daje varnost.

Ko prejmemo šifrirano sporočilo c , ga odšifriramo tako, da izračunamo

$$m = c^d \bmod n = 218^{235} \bmod 391 = 10.$$

◇

Poleg šifriranja, brez da bi si delili ključ, pa je kriptografija javnega ključa omogočila tudi *digitalne podpise*. Ti so uporabljeni vsakič, ko pošljemo e-pošto ali dostopamo do katerekoli spletne strani. Delujejo na podoben način, kot šifriranje z javnim ključem, le da najprej uporabimo zasebni ključ na sporočilu, prek javnega ključa pa preverjamo veljavnost podpisa. Ponavadi sta šifriranje in podisovanje uporabljena hkrati, saj tako pošljemo šifrirano sporočilo, za katerega lahko oseba, s katero komuniciramo preveri, da je res prišlo od nas.

2.5 Digitalni podpisi

Ideja *kriptografskih* ali *digitalnih* podpisov je, da služijo kot izboljšava človeškega ročnega podpisa. Za razliko od ročnega podpisa, lahko z digitalnim dosežemo pravo identifikacijo posameznika, ki temelji na njegovem zasebnem ključu. Tako smo lahko za digitalno podpisan dokument prepričani, da ga je res podpisal lastnik točno določenega zasebnega ključa.

Podpis dokumenta poteka nekoliko drugače, kot pri ročnih podpisih. Pri ročnem podpisu ta postane del dokumenta, digitalni podpis pa je od njega ločen, vseeno pa nastane s pomočjo zgostitve podpisanega dokumenta, zato bo podpis za dva različna dokumenta vedno drugačen.

Ostane še vprašanje preverjanja avtentičnosti podpisa. Pri ročnem podpisu to lahko storimo prek primerjave z znanim, preverjeno avtentičnim podpisom. Ta postopek je zamuden in nenatančen, veliko večino ročnih podpisov je moč ponarediti z nekaj prakse. Preverjanje digitalnega podpisa pa temelji na kriptografiji javnega ključa. Ker je podpis nastal s pomočjo podpisnikovega zasebnega ključa, lahko s pomočjo ujema jučega javnega ključa preverimo avtentičnost.

Definicija 2.7. Digitalni ali kriptografski podpis $\mathcal{S} = (\mathcal{G}, \mathcal{S}, \mathcal{V})$ je trojica učinkovitih algoritmov \mathcal{G} za ustvarjanje ključa, \mathcal{S} za podpisovanje in \mathcal{V} za preverjanje podpisa. Definirana je nad končno množico možnih sporočil \mathcal{M} , vrnjeni podpis pa leži v končni množici podpisov Σ .

- \mathcal{G} je naključnostni algoritem za ustvarjanje para ključev (pk, sk) , ki ne prejme nobenega argumenta. pk je javni ključ za preverjanje avtentičnosti podpisa, sk pa je zasebni ključ za podpisovanje.
- \mathcal{S} je naključnostni algoritem, ki za svoja argumenta prejme zasebni ključ sk in sporočilo m , vrne pa podpis σ spročila m z zasebnim ključem sk oz.

$$\sigma = \mathcal{S}(sk, m).$$

- \mathcal{V} je determinističen algoritem, ki preverja veljavnost podpisov. Za svoje argumente prejme javni ključ pk , sporočilo m in podpis σ , vrne *veljaven*, če je podpis veljaven in *neveljaven*, sicer. Velja torej

$$\mathcal{V}(pk, m, \sigma) = \begin{cases} \text{veljaven}, & \sigma = \mathcal{S}(sk, m), \\ \text{neveljaven}, & \sigma \neq \mathcal{S}(sk, m). \end{cases}$$

2.6 Varnost

Glavna stvar, ki nas zanima pri obravnavi kateregakoli kriptosistema, je njegova *varnost*. Ker je cilj digitalnih podpisov sogovorniku zagotoviti, da sporočilo res pošljamo mi, nas glede varnosti najbolj skrbi, da bi *napadalec* lahko ponaredil naš podpis in nam s tem ukradel identiteto. Pri tem je lahko uspešen na več nivojih, ki so od najmanj do najbolj škodljivega:

- **Eksistencialno ponarejanje** (angl. *existential forgery*) pomeni, da napadalec lahko ponaredi vsaj en podpis. To pomeni, da lahko najde vsaj en par (m, σ) , da velja $\mathcal{V}(\text{pk}, m, \sigma) = \text{veljaven}$.
- **Selektivno ponarejanje** (angl. *selective forgery*) pomeni, da lahko napadalec z nezanemarljivo verjetnostjo podpiše sporočilo, ki mu ga da nekdo drug in ga mi še nismo podpisali. Torej, če napadalcu nekdo predloži sporočilo m , lahko z nezanemarljivo verjetnostjo najde podpis σ , da velja $\mathcal{V}(\text{pk}, m, \sigma) = \text{veljaven}$.
- **Popoln zlom** (angl. *total break*) pomeni, da je napadalec ugotovil naš zasebni ključ in s tem podpisovalni algoritem. V našem imenu lahko podpiše karkoli.

Poleg zgoraj definiranih *ciljev napadalca*, lahko za vsak kriptosistem definiramo tudi *model napada*, in pa *tip varnosti*, ki ga zagotavlja shema. Varnost večine shem za digitalne podpise temelji na (domnevni) težavnosti določenih matematičnih problemov.

Stinson [5] definira naslednje modele napada:

- **Napad samo s ključem** je napad, kjer napadalec pozna samo naš javni ključ pk . Pozna torej algoritem za preverjanje podpisov \mathcal{V} .
- **Napad z znanimi sporočili** je napad, kjer napadalec poseduje seznam parov sporočil in njihovih podpisov $(m_1, \sigma_1), (m_2, \sigma_2), \dots$, kjer za vsak i velja $\sigma_i = \mathcal{S}(\text{sk}, m_i)$.
- **Napad z izbranimi sporočili** je napad, kjer nam napadalec da seznam sporočil m_1, m_2, \dots , mi pa mu vrnemo seznam podpisov, da za vsak i velja $\sigma_i = \mathcal{S}(\text{sk}, m_i)$.

Ostane nam še pregled varnosti, ki jo lahko pričakujemo oz. zahtevamo od sheme za podpisovanje. Takšna shema ne more biti *brezpogojno varna*, kar bi pomenilo, da je tudi z neomejenimi računskimi zmožnostmi nemogoče ponarediti podpis. To je zato, ker lahko napadalec sistematično preveri vse podpise za neko sporočilo s pomočjo algoritma \mathcal{V} , dokler ne najde pravega. Pričakujemo pa lahko *računsko varnost*, kar pomeni, da napadalec ne more najti ponaredka v doglednem času, če ima omejene računske sposobnosti, ali pa *dokazljivo varnost*, kar pomeni, da lahko varnost prevedemo na težavnost nekega matematičnega problema.

3 Schnorrov podpis

Eden izmed najenostavnejših, dokazano varnih podpisov je ravno *Schnorrov podpis* [2]. Kot vsi podpisi, tudi ta potrebuje tri algoritme: za generiranje ključa, podpisovanje in preverjanje podpisa.

Za generiranje para ključev, je potrebno najprej generirati dve praštevili p in q , tako da q deli $p - 1$. Potem je potrebno izbrati element g iz multiplikativne grupe modulo p , torej $g \in \mathbb{Z}_p^*$, ki je reda q . Za konec je potreben še izbor naključnega števila $s \in [0, q - 1]$ in izračun

$$I = g^s \bmod p.$$

Ko vse to opravimo, smo uspešno ustvarili par ključev

$$\begin{aligned} \text{pk} &= (p, q, g, I), \\ \text{sk} &= s. \end{aligned}$$

Ideja v ozadju teh števil je, da p določa multiplikativno grupo števil \mathbb{Z}_p^* . Za podpis je potrebno najti podgrupo, katere red je praštevilo, je pa vseeno dovolj velika, da omogoča varnost. Red te podgrupe je q , določa pa jo generator g . Iz varnostnih razlogov mora p imeti 2048 bitov, q pa 224. Čeprav je grupa \mathbb{Z}_p^* zelo velika, je Schnorrov podpis vseeno učinkovit, saj večinoma deluje znotraj podgrupe, ki jo generira g .

KAKO DOBIMO PARAMETRE??

Poleg parametrov p, q in g , morata podpisnik in preverjalec določiti oz. imeti dostop do varne kriptografske zgoščevalne funkcije $H : \{0, 1\}^* \rightarrow \mathbb{Z}_q$. Za varno funkcijo smatramo vsako, ki zadošča lastnostim iz definicije 2.3. Velikost kodomene te funkcije definira velikost končnega podpisa. Iz zgostitve, dolge $\log_2 q$ bitov, dobimo podpis, dolg $2 \log_2 q$ bitov [5]. Tu se pokaže glavna prednost Schnorrovega podpisa. Vrača relativno kratke podpise, hkrati pa lahko izkoristi velikost grupe \mathbb{Z}_p^* za varnost.

Za podpis enega sporočila mora podpisnik generirati naključno število $r \in [0, q - 1]$ in izračunati *zavezo*

$$X = g^r \bmod p.$$

Ta korak je podoben zadnjemu delu generiranja ključev, le da je skrivni del ključa s uporabljen večkrat, r pa mora biti generiran za vsako sporočilo znova. Potem z uporabo funkcije H izračunamo *izziv*

$$e = H(X || m),$$

kjer $||$ označuje stikanje nizov. Za konec je potrebno izračunati še

$$y = es + r \bmod q,$$

podpis sporočila m pa je potem par (X, y) oz.

$$S(s, m) = (X, y).$$

Za preverjanje veljavnosti podpisa (X', y') sporočila m , je potrebno izračunati

$$e' = H(X' || m)$$

in preveriti, če velja

$$g^{y'} \stackrel{?}{\equiv} X' \cdot I^{e'} \pmod{p}. \quad (3.1)$$

Za to moramo uporabiti nekaj lastnosti cikličnih grup in modularne aritmetike.

Trditev 3.1. *Naj bosta p in q praštevili, $q \mid p-1$. Naj bo g element grupe \mathbb{Z}_p^* reda q , kar pomeni, da je $g^q \equiv 1 \pmod{p}$. Naj bo k naravno število. Potem velja*

$$g^k \bmod p = g^{k \bmod q} \bmod p.$$

Dokaz. Po osnovnem izreku o deljenju naravnih števil, lahko k na en sam način zapišemo kot $k = nq + r$, kjer velja $n \in \mathbb{N}, r < q$.

Leva stran enačbe se potem prepiše

$$\begin{aligned} g^k \bmod p &= g^{nq+r} \bmod p = \\ &= (g^q)^n g^r \bmod p = \\ &= 1^n g^r \bmod p = \\ &= g^r \bmod p. \end{aligned}$$

Desna stran pa se prepiše kot

$$\begin{aligned} g^{k \bmod q} \bmod p &= g^{(nq+r) \bmod q} \bmod p = \\ &= g^r \bmod p. \end{aligned}$$

Ker sta obe strani enaki, je trditev dokazana. \square

Po trditvi 3.1 lahko levo stran enačbe za preverjanje Schnorrovega podpisa (3.1) prepišemo

$$\begin{aligned} g^{y'} \bmod p &= g^{es+r \bmod q} \bmod p = \\ &= g^{es+r} \bmod p. \end{aligned}$$

Za pretvorbo desne strani moramo uporabiti nekaj lastnosti modularne aritmetike.

Trditev 3.2. *Naj bodo a, b in p naravna števila. Potem za modularno množenje in potenciranje velja*

$$a \cdot b \bmod p = (a \bmod p) \cdot (b \bmod p) \bmod p, \quad (3.2)$$

$$a^b \bmod p = (a \bmod p)^b \bmod p. \quad (3.3)$$

Dokaz. a in b lahko po osnovnem izreku o deljenju naravnih števil na en sam način zapišemo kot

$$\begin{aligned} a &= n_a p + r_a, \\ b &= n_b p + r_b, \end{aligned}$$

kjer velja $r_a < p$.

(3.2): Levo stran preoblikujemo

$$\begin{aligned} a \cdot b \bmod p &= (n_a p + r_a) \cdot (n_b p + r_b) \bmod p = \\ &= (n_a n_b p^2 + n_a p r_b + n_b p r_a + r_a r_b) \bmod p = \\ &= r_a r_b \bmod p, \end{aligned}$$

desno pa

$$(a \bmod p) \cdot (b \bmod p) \bmod p = (n_a p + r_a \bmod p) \cdot (n_b p + r_b \bmod p) \bmod p = r_a r_b \bmod p.$$

Ker se strani ujemata, je trditev dokazana.

(3.3): Ker je potenciranje samo zaporedna uporaba množenj, lahko trditev pokažemo z indukcijo na b in enačbo (3.2):

- $b = 2$: Primer, ko je $b = 1$ (ali $b = 0$) je trivialen, če pa je $b = 2$, pa se problem reducira v

$$a \cdot a \bmod p \stackrel{?}{=} (a \bmod p) \cdot (a \bmod p) \bmod p,$$

kar drži neposredno po enačbi (3.2).

- $n \rightarrow n + 1$: Predpostavimo, da enačba (3.3) drži za $b = n$ (I.P.). Ko je $b = n + 1$, dobimo

$$\begin{aligned} a^{n+1} \bmod p &= a^n a \bmod p = \\ &\stackrel{(3.2)}{=} (a^n \bmod p)(a \bmod p) \bmod p = \\ &\stackrel{\text{I.P.}}{=} (a \bmod p)^n (a \bmod p) \bmod p = \\ &= (a \bmod p)^{n+1} \bmod p. \end{aligned}$$

S tem je indukcija končana in trditev dokazana. □

Desno stran enačbe (3.1) torej lahko prepisemo

$$\begin{aligned} X' \cdot I^{e'} \bmod p &= g^r \bmod p \cdot (g^s \bmod p)^e \bmod p = \\ &= (g^r \bmod p) \cdot (g^{es} \bmod p) \bmod p = \\ &= g^{es+r} \bmod p, \end{aligned}$$

kjer smo pri prehodu v drugo vrstico uporabili lastnost (3.3), pri prehodu v tretjo pa lastnost (3.2). Ker se obe strani ujemata za veljavne podpisne vrednosti, ta enačba res preveja Schnorrov podpis.

KAKO PA POKAŽEMO, DA SE ZA NAPAČNE VREDNOSTI NE UJEMA??

3.1 Varnost Schnorrovega podpisa

Najbolj očitna nevarnost kateregakoli kriptosistema z javnimi ključi bi bila možnost izračuna zasebnega ključa iz javnega. Slednji je dostopen vsem, zato bi lahko kdorkoli pridobil zasebni ključ, kar popolnoma izniči pomen šifriranja ali podpisovanja.

Pri Schnorrovem podpisu je javni ključ poleg parametrov uporabljene grupe p, q in g , še število I , izračunano kot

$$I = g^s \bmod p.$$

Za izračun je torej neposredno uporabljen zasebni ključ s . Zaradi notacije bi morda kdo hitro pomislil, da lahko zgornjo enačbo obrnemo in s izračunamo kot

$$s = \log_g(I) \bmod p.$$

Taki izračuni v grupah \mathbb{Z}_p^* žal niso tako enostavni, prišli smo do koncepta *diskretnega logaritma*.

Definicija 3.3 (Problem diskretnega logaritma [1]). Naj bo G ciklična grupa reda q , ki jo generira element g . Naj bo h naključni element iz grupe G . Naj velja $g^x = h$. x Potem imenujemo **Diskretni logaritem (DL)**.

Zamislimo si igro, kjer izzivalec in nasprotnik kot vhod prejmeta opis grupe G (torej q in $g \in G$). Izzivalec potem izbere naključen element $\alpha \in G$ in izračuna $h = g^\alpha$. h pošlje nasprotniku, le-ta pa mora odgovoriti nazaj z elementom α . To igro imenujemo **problem diskretnega logaritma (PDL)** (angl. *discrete logarithm problem*).

Pri tej igri nas zanima verjetnost pravilnega odgovora nasprotnika, ki je računsko omejen. S tem mislimo, da ima na voljo polinomsko mnogo časa (glede na velikost grupe). Če je grupa G takšna, da je verjetnost zanemarljiva, pravimo, da za grupo G drži *predpostavka diskretnega logaritma*.

Izkaže se, da za grupe, kot je \mathbb{Z}_p^* , ne poznamo učinkovitega algoritma za izračun diskretnega logaritma, torej v njih drži predpostavka DL. To torej pomeni, da ob pridobljenem javnem ključu, napadalec ne more učinkovito izračunati zasebnega.

4 Pregled skupinskih podpisov

Ko pridemo do podpisovanja skupin, si lahko zamislimo več različnih rešitev. Micali v [4] definira dve lastnosti oz. spektra, ki jim lahko zadošča podpis skupine:

- **Prilagodljivost** (angl. *flexibility*): Popolnoma prilagodljiv podpis skupine je takšen, ki ga lahko proizvede katerakoli podskupina originalne skupine podpisnikov. Ko je podpis preverjen, se mora tisti, ki ga je preveril odločiti, če je ustrezen del skupine podal svoj podpis. Popolnoma neprilagodljiv podpis bi bil takšen, ki ga lahko v imenu skupine ustvari katerkoli član.
- **Odgovornost** (angl. *accountability*): Če lahko iz podpisa ugotovimo, kateri člani so sodelovali pri ustvarjanju, nam podpis omogoča odgovornost. Ta lastnost je lahko zaželeno, če se želimo prepričati, ali je ustrezen del skupine sodeloval pri podpisu (npr. ali je pri podpisovanju sodeloval generalni direktor podjetja). V drugih primerih pa si želimo anonimnost posameznih članov (npr. če bi generiranje podpisa predstavljalo nekakšno glasovanje, bi želeli vedeti samo, koliko članov je sodelovalo).

V nadaljevanju bomo skupino potencialnih podpisnikov (torej podpisnikov, ki imajo možnost sodelovati pri podpisovanju) označili z $G = P_1, \dots, P_L$, kjer ima skupina L članov. Dejanski podpis pa bo generiral samo del skupine $S \subseteq G$.

4.1 Skupinski podpisi

Skupinski podpis (angl. *group signature*) v imenu celotne skupine G ustvari en anonimni član. To torej pomeni, da je podpis popolnoma neprilagodljiv, saj ni možno prisiliti skupine, da bi podpis ustvaril več kot en član. Prav tako v splošnem noben član, niti tisti, ki preverja podpis, ne more ugotoviti, kdo je podpis ustvaril. Da skupinski podpisi omogočijo vsaj delno odgovornost, skupina določi *vodjo skupine*, ki ima možnost razkriti identiteto podpisnika, če pride do težav. V tem primeru seveda vodja predstavlja atraktivno tarčo za napad.

4.2 Pragovni podpisi

Če želimo zagotoviti, da se s podpisom strinja zadosten delež skupine, lahko uporabimo *pragovni podpis* (angl. *threshold signature*). Ta nam omogoča določeno mero prilagodljivosti, saj lahko katerkoli zadosten delež skupine ustvari podpis. Še vedno je nemogoče upoštevati morebitno hierarhično strukturo skupine. Pragovni podpisi omogočajo tudi popolno anonimnost podpisnikov, in s tem torej nično odgovornost. Intuicija tu je, da večina pragovnih podpisov temelji na interpolaciji polinoma $(l-1)$ -stopnje z l točkami. Podpis je potem ustvarjen s pomočjo vrednosti polinoma v neki točki. Po interpolaciji se informacija o tem, točno katere točke smo uporabili, izgubi. Take podpise imenujemo tudi *l -od- L sheme*. Primer uporabe je odklepanje sefa v banki. Recimo, da ne zaupamo samo osebi z odklepanjem in želimo, da je prisotnih l od L pooblaščenih oseb, ni nam pa važno, katerih. Tu je pragovni podpis odlična rešitev.

4.3 Večstranski podpisi

Za nekatere uporabe podpisov, bi si od njih želeli podobne lastnosti, kot jih ima večstranski ročen podpis. Pri njem lahko hitro preberemo podpisnike, torej imamo popolno prilagodljivost. Vidimo lahko seznam podpisnikov, torej lahko presodimo, če so med njimi tisti, ki smo jih želeli. Prav tako podpisniki nosijo popolno odgovornost, saj na papirju piše njihovo ime.

Podoben učinek bi z digitalnimi podpisi lahko dosegli, če bi namesto enega podpisa skupine, od članov zbrali individualne podpise in jih nanizali v seznam. Dobili bi torej digitalni podpis skupine, ki ponuja popolno prilagodljivost in odgovornost. Težava je samo, da je dolžina podpisa (in s tem čas preverjanja) proporcionalna številu podpisnikov. *Večstranski podpisi* (angl. *multisignatures*) ohranijo lastnosti seznama podpisov, rezultat sheme je pa en sam podpis, ki je enako dolg ne glede na število podpisnikov, prav tako je od števila neodvisen čas preverjanja. Tega s seznamom podpisov ni mogoče doseči, saj tako dolžina podpisa, kot čas preverjanja podpisa raste linearno s številom podpisnikov (vsak doda en podpis seznamu, ki ga je potrebno preveriti). Večstranski podpisi so torej odlična posplošitev ročnih podpisov skupin, ki vseeno ohrani učinkovito preverjanje.

Primer 4.1. Recimo, da imamo nek organ, ki izdaja certifikate avtentičnosti uporabnikov (npr. potrjuje avtentičnost javnih ključev). Za večjo robustnost in varnost, je lahko ta organ razporejen na več strežnikov. Tako preprečimo razpad sistema v primeru izpada enega strežnika. Zato je torej tudi pomembno, da certifikacijo uporabnika potrdi nekaj strežnikov, ne pa nujno vsi. Tu lahko torej neka podskupina vseh strežnikov organa skupaj izda en večstranski podpis, ki potrjuje avtentičnost uporabnika. \diamond

4.4 Agregirani podpisi

5 Večstranski Schnorrov podpis

Micali et al. [4] so prvi definirali formalni model za večstranske podpise in podali formalni dokaz varnosti. Zamislili so si večstranski podpis, ki temelji na Schnorrovem

in ga poimenovali *večstranski podpis podskupine z odgovornostjo* (angl. *Accountable-Subgroup Multisignature (ASM)*). V tem razdelku predstavimo njihov model, podpis in argument varnosti.

5.1 Večstranski podpis podskupine z odgovornostjo

Kljub daljšemu imenu, večstranski podpis podskupine z odgovornostjo le bolj formalno definira idejo večstranskega podpisa, predstavljeno v razdelku 4.3. Ideja oz. cilj večstranskega podpisa je torej, da lahko katerakoli podskupina podpisnikov S , neke skupine G , brez potrebe po *centru zaupanja* (angl. *trusted third party (TTP)*) ustvari podpis. Generiranje ključev je torej popolnoma v domeni skupine G . Podpis, ki ga ustvari S predstavlja splošno preverljiv dokaz strukture S in dejstva, da vsak član skupine stoji za (torej podpisuje) sporočilom M .

Definicija 5.1 (Večstranski podpis (podskupine z odgovornostjo)). Skupina G je sestavljena iz L podpisnikov, torej

$$G = P_1, P_2, \dots, P_L.$$

Podpisnik predstavlja verjetnostni Turingov stroj, omejen s polinomskim časom. Vsak podpisnik pozna svojo identifikacijsko številko (eno od števil $1, \dots, L$) in pa *varnostni parameter* k , ki je enak za vse podpisnike.

Kot vsi digitalni podpisi 2.7, ima tudi ta tri glavne komponente:

- \mathcal{G} je algoritem za ustvarjanje ključev. Za neko skupino podpisnikov G je pognan samo enkrat, na začetku sodelovanja. Vsak podpisnik i dobi kot vhod seznam vseh podpisnikov v G in požene \mathcal{G} , ki vrne par ključev (pk_i, sk_i) . Zapišemo lahko torej

$$\mathcal{G}_i(L) = (pk_i, sk_i),$$

kjer smo brez škode za splošnost predpostavili, da velikost skupine L enolično opiše skupino G .

- \mathcal{S} je algoritem za podpisovanje. Pognan je vsakič, ko neka podskupina S želi ustvariti podpis. Vsak podpisnik kot vhod prejme seznam vseh podpisnikov S , njihove javne ključe pk_j , kjer j teče po identifikacijskih številkah vseh članov S , sporočilo m in lasten zasebni ključ sk_i . Algoritem \mathcal{S} je porazdeljen protokol, pri izvedbi morajo sodelovati vsi člani S . Po uspešni izvedbi lahko en od članov objavi podpis σ .

$$\mathcal{S}_i((id_1, \dots, id_l), (pk_{id_1}, \dots, pk_{id_l}), m, sk_i) = \sigma,$$

kjer je l velikost podskupine S , (id_1, \dots, id_l) pa identifikacijske številke članov.

- \mathcal{V} je algoritem za preverjanje veljavnosti podpisa. Požene ga tisti, ki želi preveriti veljavnost večstranskega podpisa. Ni nujno, da je to eden izmed podpisnikov iz G . Kot vhod algoritem dobi seznam podpisnikov S , pripadajoče javne ključe, sporočilo m in morebiten podpis σ . Algoritem potem vrne

$$\mathcal{V}((id_1, \dots, id_l), (pk_{id_1}, \dots, pk_{id_l}), m, \sigma) = \begin{cases} \text{veljaven}, & \sigma =, \\ \text{neveljaven}, & \sigma \neq. \end{cases}$$

5.1.1 Robustnost, varnost in napadalec

Definiran večstranski podpis ni robusten. To pomeni, da v primeru izpada enega od podpisnikov P_i , ki je del podskupine S , ta ne more ustvariti večstranskega podpisa. Podpis še vedno lahko ustvari $S \setminus \{P_i\}$. To dejstvo nam dovoli definicijo napadalca, ki ima zelo velik vpliv na celoten sistem, saj ne iščemo robustnosti.

Definicija 5.2 (Napadalec pri ASM). Napadalec v modelu večstranskih podpisov podskupine z odgovornostjo ima naslednje zmožnosti:

- Ima popoln nadzor nad vsemi komunikacijskimi kanali med člani skupine G . Lahko bere, spreminja in preprečuje dostavo vseh sporočil. Prav tako lahko v imenu kateregakoli podpisnika pošlje sporočilo.
- Kadarkoli lahko *pokvari* kateregakoli podpisnika. Ko je igralec pokvarjen, napadalec izve celotno stanje igralca, vključno z vsemi skrivnostmi.
- Nadzira lahko vhod algoritma za ustvarjanje ključev \mathcal{G} za vse podpisnike. Vsakemu lahko poda drugačno skupino G .
- Od kateregakoli nepokvarjenega igralca lahko kadarkoli zahteva podpis nekega sporočila skupaj s podskupino, ki jo določi napadalec. To je *napad z izbranim sporočilom in podskupino*.

Zaradi obširnih zmožnosti napadalca, ta lahko kadarkoli prepreči podpis sporočila. Naš cilj, kar se tiče varnosti, je, da preprečimo eksistencialno ponarejanje podpisa. Želimo torej, da napadalec ni zmožen ponarediti podpisa za katerololi sporočilo v imenu katerekoli podskupine.

Definicija 5.3 (Varnost pri ASM). Naj bo k varnostni parameter (ki si ga delijo vsi podpisniki). Naj bo $c > 0$ poljubna konstanta. Naj bo F napadalec, ki je omejen s polinomskim časom v parametru k . Naj bo p verjetnost, da F vrne trojico (σ, m, S) , za katero velja:

- σ je veljaven večstranski podpis sporočila m s strani skupine S .
- Obstaja nepokvarjen igralec P iz skupine S , od katerega F ni zahteval podpisa sporočila m s strani skupine S .

Shemi za večstranske podpise podskupine z odgovornostjo bomo rekli, da je *varna*, če velja

$$p < k^{-c}.$$

Kot ponavadi pri digitalnih podpisih, tudi tu predpostavimo, da preverjalec podpisov lahko vedno pridobi prave javne ključe podpisnikov iz S . Zaradi močnega napadalca pri ASM, lahko tudi pridemo v situacijo, ko podpisnik P_1 ne ve, kdo je zares podpisnik P_2 . Tukaj predpostavimo, da preverjalec natančno ve, kdo je P_2 v resnici, in lahko pridobi ustrezen javni ključ.

5.1.2 Model slučajnega oraklja

Ko obravnavamo varnost kriptosistemov, ponavadi pogovarjamo o *standardnemu modelu* kriptografije. Ta model ima samo eno predpostavko: napadalec je omejen samo s časom in količino računske moči, ki mu je na voljo. Občasno se znajdemo v primeru, ko moramo za dokaz varnosti sprejeti dodatne predpostavke. V tem primeru se znajdemo v alternativnih modelih kriptografije.

Ko imamo opravka z zgoščevalnimi funkcijami, je pogosto potrebno sprejeti dodatne predpostavke, da lahko pokažemo varnost. Specifično, ko imamo opravka z zgoščevalno funkcijo $H : A \rightarrow B$ predpostavimo, da je bila ta funkcija izbrana naključno med vsemi funkcijami, ki slikajo A v B . To idealizirano verzijo zgoščevalne funkcije imenujemo **slučajni orakelj** (angl. *random oracle*).

Model slučajnega oraklja (angl. *random oracle model*) je model kriptografije, kjer poleg standardnih predpostavk, vsako uporabo zgoščevalne funkcije nadomestimo s slučajnim orakljem [1]. Predpostavimo, da imajo do oraklja dostop vsi vpleteni v kriptosistem, vključno z napadalcem.

Definicija 5.4 (Slučajni orakelj pri ASM). Varnostna obravnava shem za večstranske podpise podskupine z odgovornostjo zahteva model slučajnega oraklja. Zato predpostavimo, da je k_2 še en varnostni parameter. Vsi člani skupine G in napadalec imajo dostop do slučajnega oraklja $H : \{0, 1\}^* \rightarrow \{0, 1\}^{k_2}$, ki je naključno izbrana funkcija med vsemi funkcijami, ki slikajo med $\{0, 1\}^*$ in $\{0, 1\}^{k_2}$.

5.2 Konstrukcija večstranskega Schnorrovega podpisa

Večstranska verzija temelji na Schnorrovem podpisu, predstavljenemu v poglavju 3. Ideja konstrukcije je, da začnemo z naivno verzijo, nato pa rešimo njene probleme, kar nas privede do formalne definicije v razdelku 5.3 in dokaza v razdelku 5.4.

5.2.1 Naivna verzija

Vsi podpisniki v skupini G poznajo skupne parametre p, q in g . Vsak podpisnik P_i si neodvisno in naključno izbere $s_i \in [0, \dots, q-1]$ in izračuna

$$I_i = g^{s_i} \bmod p.$$

Tako vsak podpisnik ustvari svoj par ključev

$$\begin{aligned} \text{pk}_i &= (p, q, g, I_i), \\ \text{sk}_i &= s_i. \end{aligned}$$

Poljubna podskupina $S = \{P_{id_1}, \dots, P_{id_l}\}$ skupine G podpiše sporočilo m s tremi krogi komunikacije:

1. Vsak podpisnik P_i iz podskupine S si izbere naključen element $r_i \in [0, q-1]$ in izračuna zavezo

$$X_i = g^{r_i} \bmod p.$$

Podpisniki potem pošljejo svoje zaveze izbranemu podpisniku D .

2. D izračuna *skupno zavezo*

$$\tilde{X} = (X_{id_1} \cdot X_{id_2} \cdot \dots \cdot X_{id_l}) \bmod p.$$

in jo pošlje vsem podpisnikom.

3. Vsak podpisnik s pomočjo slučajnega oraklja izračuna izziv

$$e = H(\tilde{X} || m || S)$$

in svoje *individualne podpise*

$$y_i = es_i + r_i \bmod q.$$

Individualni podpisi so potem spet poslani podpisniku D , ta pa sedaj lahko izračuna

$$\tilde{y} = (y_{id_1} + y_{id_2} + \dots + y_{id_l}) \bmod q$$

in vrne končen večstranski podpis

$$\sigma = (\tilde{X}, \tilde{y}).$$

Preverjanje veljavnosti podpisa je podobno kot pri navadnemu Schnorrovemu podpisu. Da preverimo podpis (\tilde{X}', \tilde{y}') sporočila m najprej izračunamo

$$e' = H(\tilde{X}' || m || S)$$

in preverimo, če velja

$$g^{\tilde{y}'} \stackrel{?}{=} \tilde{X}' \cdot \left(\prod_{P_i \in S} I_i \right)^{e'} \pmod{p}.$$

5.2.2 Generiranje parametrov: Predpostavka DL

Micali et al. [4] so izpostavili in rešili več problemov z zgornjo naivno idejo. Prvi problem je generiranje skupnih parametrov javnega ključa p, q in g . Ker si podpisniki delijo samo varnostni parameter k in pa slučajnega oraklja H , morajo za parametre uporabiti nek dogovorjen način uporabe H . Napadalec bo tako poznal točen postopek ustvarjanja skupnih parametrov, kar mu potencialno da prednost. Izkaže se, da lahko s primerno uporabo H poskrbimo, da to ne predstavlja nevarnosti, in generiranje parametrov dodamo k predpostavki diskretnega logaritma za večstranske podpise.

Definicija 5.5 (Predpostavka diskretnega logaritma pri ASM). Predpostavka o varnosti je poleg standardnega PDL tu razširjena še z algoritmom za pridobivanje deljenih parametrov in jo lahko zapišemo:

- *Parametri*: Sledeč verjetnostni algoritem $Gen(k)$ teče v pričakovanem polinomskega časa:

```

dokler q ni praštevililo in p ni praštevililo:
  q <- naključno izbran k-bitni niz
  p <- 2 * q + 1
vrni p, q

```

- *Težavnost PDL*: Naj bo A algoritem, ki za svoje parametre sprejme
 - praštevili p in q , da velja $p = 2q + 1$ in q je dolg k -bitov,
 - naključni element $g \in \mathbb{Z}_p^*$ reda q ,
 - naključni element I iz podgrupe \mathbb{Z}_p^* , ki jo generira g .

Naj p_k^A označuje verjetnost, da A vrne $s \in [0, q - 1]$, tako da velja

$$I \equiv g^s \pmod{p}.$$

Potem za vsak verjetnostni algoritem A , ki teče v polinomskem času, za vsako konstanto $c > 0$ in za vsak dovolj velik k velja

$$p_k^A < k^{-c}.$$

Če torej predpostavimo 5.5 (kar mora veljati za varnost), nas prvi problem ne skrbi več.

5.2.3 Napad na generiranje ključev: Dokazi brez razkritja znanja

Naslednja težava nastopi v obliki napada na generiranje ključev. Ker napadalec nadzoruje vso komunikacijo, si lahko privošči vstopiti v podpisovanje kot eden izmed igralcev P_L , in lahko zadnji generira svoj par ključev. Če si torej izbere naključen $s \in [0, q - 1]$, lahko izračuna svoj javni ključ kot

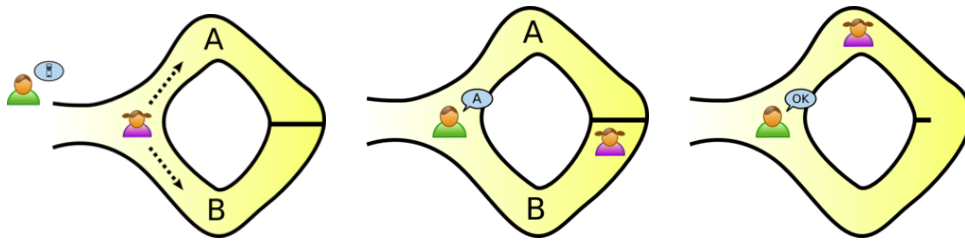
$$I_L = \left(\prod_{i=1}^{L-1} I_i \right)^{-1} \cdot g^s \pmod{p}.$$

To mu omogoča, da se podpisuje v imenu celotne podskupine S .

Za rešitev tega problema potrebujemo nekaj novih kriptografskih orodij. Prvo izmed njih so **dokazi brez razkritja znanja** (angl. *zero-knowledge proofs*). Za intuicijo si najprej pogledjmo primer.

Primer 5.6 (Jama Ali Babe [3]). V zgodbi o Jami Ali Babe nastopata Ana in Bojan. Živita ob Jami Ali Babe, ki je v obliki prstana. Pri vходу sta na levo in desno vidni dve poti, ki se kasneje združita, vendar prehod preprečujejo vrata, ki jih lahko odpre samo skrivno geslo. Ana je ugotovila, kaj to geslo je, vendar ga ne želi povedati Bojanu, vseeno pa ga želi prepričati, da geslo pozna.

Da Ana Bojanu dokaže svoje znanje, si zamisli igro. Najprej bo ona odšla v jamo po eni izmed poti. Potem bo v jamo vstopil Bojan in povedal, če želi da se Ana vrne po levi ali desni poti. Če se bo Ana dovolj velikokrat vrnila po poti, ki jo je povedal Bojan (in nikoli po napačni), bo lahko z veliko verjetnostjo prepričan, da Ana res pozna geslo. Primer enega kroga protokola je prikazan na sliki 2. \diamond



Slika 2: Igra Ane in Bojana v jami Ali Babe.

5.3 Definicija večstranskega Schnorrovega podpisa

5.4 Varnost večstranskega Schnorrovega podpisa

6 Večstranski podpisi v splošnem

Literatura

- [1] D. Boneh in V. Shoup, *A graduate course in applied cryptography*, Stanford University, Stanford, 2023.
- [2] C. P. Schnorr, *Efficient identification and signatures for smart cards*, v: Advances in Cryptology — CRYPTO' 89 Proceedings (ur. G. Brassard), Springer New York, New York, NY, 1990, str. 239–252.
- [3] J.-J. Quisquater in dr. *How to explain zero-knowledge protocols to your children*, v: Advances in Cryptology — CRYPTO' 89 Proceedings (ur. G. Brassard), Springer New York, New York, NY, 1990, str. 628–631.
- [4] S. Micali, K. Ohta in L. Reyzin, *Accountable-subgroup multisignatures*, v: Proceedings of the 8th ACM conference on Computer and Communications Security (ur. P. Samarati), ACM, Philadelphia, PA, USA, 2001, str. 245–254, DOI: 10.1145/501983.502017, dostopno na <https://doi.org/10.1145/501983.502017>.
- [5] D. R. Stinson in M. B. Paterson, *Cryptography: theory and practice*, Textbooks in Mathematics, CRC Press, 2018.