Enkratni podpisi

Aljaž Nunčič

Avgust 2020

1 Uvod

V življenju velikokrat želimo, da lahko z gotovostjo trdimo, kdo je lastnik sporočila, ki smo ga prejeli. V ta namen moramo sporočilu dodati podpis, za katerega velja, da je njegova poneverba nemogoča oz. zelo malo verjetna. Tekom predavanj smo spoznali več načinov podpisovanja, sam pa se bom posvetil enkratnim podpisom. To so podpisi, pri katerih lahko en ključ uporabimo zgolj enkrat. Cilj članka je predstaviti celotno področje enkratnih podpisov, saj sem tekom iskanja literature naletel na vire, ki pokrivajo zelo ozka področja, samih člankov pa tudi ni veliko. Članek je tako namenjen vsakomur, ki želi dobiti osnoven pregled nad področjem, lahko pa služi tudi kot motivacija za magistrsko nalogo.

V začetku bom predstavil, zakaj sploh uporabljati enkratne podpise in jih umestil med že spoznane tekom predmeta. Nato se bomo lotili nekaj podpisnih shem v kronološkem vrstnem redu. Na začetku si bomo pogledali Lamportovo shemo, ki velja za prvo na področju enkratnih shem. Največja omejitev enkratnih podpisov je ravno enkratna uporaba zasebnih in javnih ključev. Ta problem bomo delno rešili s pomočjo Merklovega drevesa. Sledili bosta podpisni shemi, ki sta nadaljevanje Lamportove - Winternitzova shema in shema HORS. Za konec pa si bomo pogledali še shemo koši in žogice, ki sem jo izbral zaradi zanimivosti in tudi zelo pogostega omenjanja v novejši literaturi. Zanimivo pri njej je, da izkorišča paradoks rojstnega dne, kar v kriptografiji ni ravno pogosto. Pri vsaki shemi si bomo pogledali tako postopek podpisovanja, kot preverjanja podpisov. Pa začnimo.

2 Zakaj enkratni podpisi

Morda se komu zdijo enkratni podpisi neuporabni ravno zaradi enkratne uporabe. To lahko izboljšamo do neke mere in jih uporabimo n-krat, a je ta n še vedno omejen in dokaj nizek. Kako to storimo, bom predstavil v nadaljevanju, sedaj pa naj najprej odgovorim na vprašanje, zakaj uporabljati enkratne podpise, ko pa imamo na primer RSA, ElGamal ali pa algoritem za podpisovanje s pomočjo eliptičnih krivulj (ECDSA), ki smo jih spoznali pri predmetu [11]. Te sheme bodo s prihodom kvantnih računalnikov postale ranljive, med tem ko bodo enkratni podpisi ostali varni, saj temeljijo na zgoščevalnih funkcijah.

3 Lamportova shema

Vse skupaj se je začelo leta 1979, ko je Leslie Lamport predstavil prvo shemo za enkratne podpise, ki temelji na zgoščevalnih funkcijah [7]. Ideja Lamportove sheme je preprosta. V nadaljevanju bom predstavil postopek podpisovanja in preverjanje podpisa s pomočjo primera. Pri primerih v članku

bom zgolj zaradi enostavnosti ključe (tudi sporočilo) večkrat predstavil kot tabele, saj si na ta način lažje predstavljamo razdelitev in pozicioniranje znotraj ključa. Denimo da Bojan pošlje sporočilo Ani.

Slika 1: Leslie Lamport - avtor Lamportove sheme. [2]

3.1 Podpisovanje

Bojan mora najprej generirati dve tabeli naključnih vrednosti. Ti morata biti vsaj tako dolgi, kot bo dolgo poslano sporočilo in predstavljata zasebna ključa. Naj bo prvi ključ sk_0 in drugi sk_1. Javna ključa dobi tako, da izračuna zgoščeno vrednost vsakega elementa zasebnega ključa. Tako dobimo dve tabeli pk_0 in pk_1, ki sta javna ključa. Sam postopek podpisovanja od tukaj naprej pa je zelo preprost. Če je na n-tem mestu v sporočilu bit 0, potem na n-to mesto v podpisu vstavi n-ti element iz sk_0. Enako je za bit 1 v sporočilu, le da tokrat uporabi ključ sk_1.

3.2 Preverjanje

Ko Ana prejme sporočilo in podpis ga preveri tako, da v primeru, ko je v sporočilu na n-tem mestu bit 0, zgosti n-ti element v podpisu in preveri, da se ujema z n-tim elementom v pk_0. Enako je za bit 1 v sporočilu, le da tokrat uporabi javni ključ pk_1.

3.3 Mogoča izboljšava

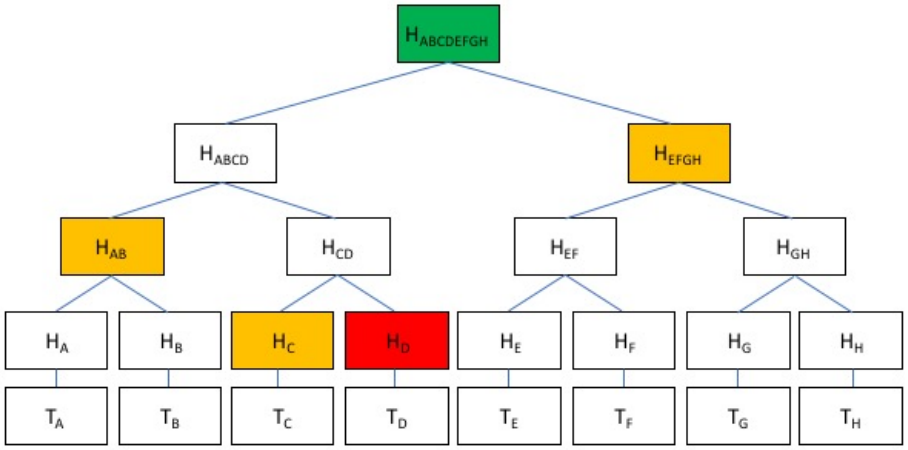
Trenutna shema je zelo potratna, saj za sporočilo dolžine l potrebujemo dva, vsaj l dolga zasebna ključa (in tudi javna). To lahko izboljšamo na način, da podpisujemo le bite 1 (ali 0). Ker pa bi samo s takšno shemo napadalec z lahkoto spremenil bit 1 v sporočilu in odstranil ustrezni element v podpisu, moramo dodati varnostno vsoto. Ta je enaka številu 1 v sporočilu. Na koncu podpišemo tudi samo varnostno vsoto.

4 Merklovo drevo

Enkratni podpisi so v praksi zanimivi, a imajo eno slabost [8]. En javni ključ lahko uporabimo zgolj enkrat, saj bi v nasprotnem primeru bilo mogoče podpis ponarediti. Želimo si, da bi en javni ključ lahko uporabili večkrat, recimo n krat.

Prva ideja, na katero pomislimo je, da iz več javnih ključev sestavimo enega. To storimo tako, da $n-1$ krat uporabimo preprosto operacijo stika a hitro vidimo, da ta način ni najbolj učinkovit. Če je namreč dolžina enega prvotnega ključa k, je po tem postopku dolžina ključa $n \cdot k$. V nadaljevanju si bomo pogledali način, kako lahko to dolžino zmanjšamo.

Definirajmo drevesno strukturo, kot je prikazana na sliki 3. Denimo da imamo 8 ključev T_A do T_H. Vsakega izmed njih zgostimo z zgoščevalno funkcijo v H_A do H_H. Pozor, gradnja Merklovega drevesa ni povezana z ostalim postopkom podpisovanja, temveč je od njega neodvisna. Omogoča nam le gradnjo enega javnega ključe iz večih, pri čemer dolžina ni enaka $n \cdot k$. Prav tako ni pomembno, da uporabimo isto zgoščevalno funkcijo, ki jo uporabimo med samim podpisovanjem. Vozlišča v drevesih nato paroma združujemo tako, da sinova združimo z operacijo stika ter nad njima izvedemo zgoščevalno funkcijo, s čimer dobimo starša. To počnemo tako dolgo, da pridemo do enega samega vozlišča (v našem primeru vozlišča $H_{ABCDEFGH}$), korena, ki predstavlja skupni javni ključ.



Slika 2: Merklovo drevo [6]

4.1 Merklov dokaz

Kadar uporabljamo Merklovo drevo, kot javni ključ posredujemo le koren. Pri tem je pomembno, da ga posredujemo varno in nespremenjenega, saj lahko le tako prejemnik kasneje uspešno izvede preverjanje podpisa. Kasneje prejemniku poleg samega sporočila pošljemo tudi "prvotni" javni ključ, ki je del Merklovega drevesa in naj bo to v našem primeru vozlišče T_D. Hkrati mora seveda prejemnik vedeti tudi, katera zgoščevalna funkcija se je uporabljala med gradnjo drevesa. Prejemnik nato gradi pot po drevesu navzgor do korena, pri čemer mu pošiljatelj zagotovi še ostale potrebne zgoščene vrednosti, ki jih pri tem potrebuje (pri nas: H_C, H_{AB} in H_{EFGH}). Če se prejemnikov koren nato ujema s predhodno prejetim javnim ključem, potem je tudi "prvotni" javni ključ del

drevesa ter s tem pripada isti osebi. Pri tem opozorimo, da za pošiljanje vseh ostalih elementov v drevesu razen korena, ki je javni ključ, ne potrebujemo skrbeti za varen prenos do uporabnika, saj jih bo prejemnik v primeru spremembe zavrnil v postopku Merklovega dokaza. Časovna zahtevnost dokaza je $O(\log n)$.

5 Winternitzova shema - WOTS

Ena izmed izboljšav Lamportove sheme je tudi Witernetzova shema, kjer namesto, da bi podpisovali vsak bit posamezno, podpisujemo po več bitov hkrati [7, 5]. Denimo, da se odločimo podpisovati po 8 bitov na enkrat. Te skupine bitov si lahko predstavljamo kot števila in sicer od 0 do 255. Da se izognemo velikemu številu ključev pa naredimo preprost trik. Za podpisovanje vrednosti 0 uporabimo sk_0, ki je enak neki naključni vrednosti. Za podpisovanje vrednosti 1 uporabimo sk_1, ki je enak zgoščeni vrednosti sk_0. Če torej uporabljamo zgoščevalno funkcijo H, velja $sk_n = H(sk_{n-1})$ za vse $1 \leq n \leq 255$. Za javni ključ pa uporabimo kar sk_{255}. In ne, ni napaka, da sta javni ključ in sk_{255} enaka. Zasebni ključ sk_{255}, ki ga uporabljamo v primeru podpisa vrednosti 255 (8 zaporednih bitov 1), je hkrati tudi javni ključ. Ta poenostavitev namreč ne zniža varnosti.

5.1 Podpisovanje

Ker so ključi med seboj odvisni, potrebujemo imeti shranjen le sk_0. Sporočilo nato podpisujemo tako, da izračunamo vrednost 8 zaporednih bitov (lahko bi izbrali tudi katerokoli drugo število in s tem le spremenili število zasebnih ključev), ki naj bo enaka n. V podpis tako dodamo ustrezen element sk_0, nad katerim izvedemo zgoščevalno funkcijo n-krat. Zaradi medsebojne odvisnosti zasebnih ključev, moremo tudi tukaj dodati varnostno vsoto, ki jo prav tako podpišemo.

5.2 Preverjanje

Preverjanje je enostavno. Tudi tukaj najprej izračunamo vrednost zaporednih bitov v sporočilu, ki naj bo n. Nato nad ustreznim delom elementa v podpisu izvedemo zgoščevalno funkcijo $(255 - n)$-krat. Na koncu le še preverimo, da se dobljena vrednost ujema z vrednostjo v javnem ključu.

6 Zgoščena vrednost za pridobitev naključne podmnožice - HORS

Naslednja shema za podpise, ki jo bom predstavil je HORS [10, 3]. Denimo, da imamo zgoščevalno 128 bitno funkcijo H. Za začetek ustvarimo tabelo naključnih vrednosti velikosti 256 polj. Ta polja sedaj predstavljajo zasebne ključe sk_0 do sk_{255}. Da dobimo javne ključe, na vsakem elementu tabele uporabimo zgoščevalno funkcijo in dobimo pk_0 do pk_{255}. Denimo, da želi Bojan Ani poslati sporočilo m.

6.1 Podpisovanje

Na začetku Bojan zgosti vrednost sporočila, pri čemer uporabi 128 bitno zgoščevalno funkcijo. Sedaj sporočilo razdeli na 16 delov (vsak vsebuje 8 bitov) in jih predstavi kot vrednosti od 0 do 255. Če je n-ta vrednost sporočila enaka k, potem na n-to mesto v podpisu vstavi sk_k.

Pri algoritmu lahko uporabimo tudi zgoščevalne funkcije z drugačno velikostjo zaloge vrednost, pri čemer se spremenijo tudi ostale številke v algoritmu.

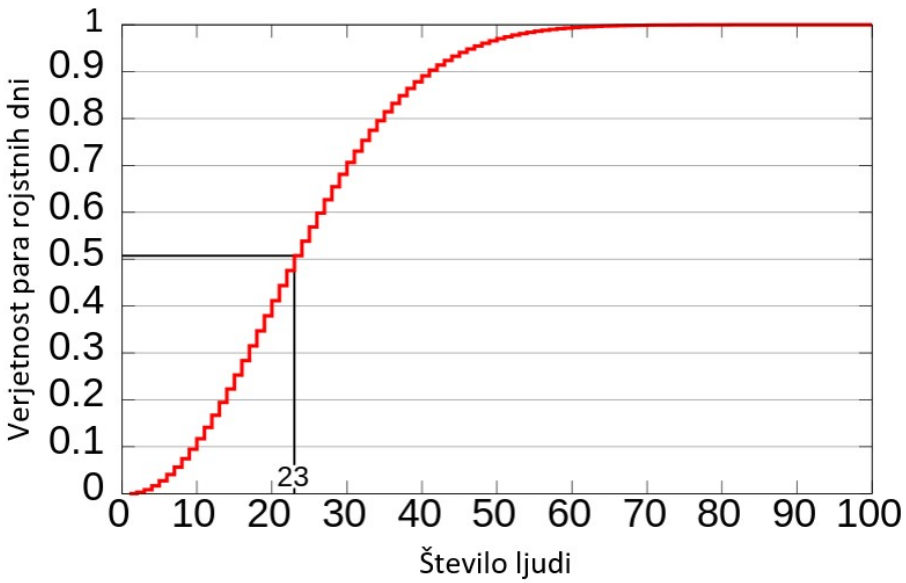
6.2 Preverjanje

Ko Ana prejme sporočilo, tudi ona najprej izračuna zgoščeno vrednost sporočila, ga razdeli na kose po 8 bitov in izračuna njihove vrednosti. Od tu naprej pa se postopek razlikuje. Če je n-ta vrednost sporočila enaka k, potem Ana preveri, da je zgoščena vrednost n-tega mesta v podpisu enaka pk_k.

7 Paradoks rojstnega dne

Paradoks rojstnega dne govori o tem, koliko ljudi naključno moremo izbrati, da bosta imela vsaj dva rojstni dan na isti dan z gotovostjo vsaj 50% [12, 1]. Čeprav bi na začetku pričakovali, da je ta številka dokaj velika (morda 183, kar je polovica od števila dni v letu) se izkaže, da potrebujemo le 23 oseb. Do tega pridemo po naslednjem postopku. Če želimo, da ima druga oseba različen rojstni dan kot prva, imamo $\frac{364}{365}$ možnosti, za tretjo $\frac{363}{365}$ in tako dalje. Imenujmo to verjetnost $P(različni)$. Ker želimo, da imata nasprotni dogodek (vsaj dve osebi imata rojstni dan na isti dan) moremo od 1 odšteti to verjetnost $(1 - P(različni))$.

Sam paradoks lahko poenostavimo tudi v splošnejšo obliko, ki je $n \approx 1,177 \cdot \sqrt{N}$, kjer je N število vseh možnosti (v našem primeru 365 dni), n pa število potrebnih izbir (v našem primeru 23 oseb).



Slika 3: Graf prikazuje verjetnost, da imata dve osebi rojstni dan na isti dan v odvisnosti od števila ljudi. [4]

8 Koši in žogice - BiBa

Ko se pri kriptografiji pogovarjamo o kolizijah zgoščevalnih funkcij, imajo le te običajno negativen prizvok, a ni vedno tako [9]. Eden izmed takšnih primerov je shema koši in žogice (ang.: Bins and Balls). Algoritem si lahko predstavljamo kot metanje žogic v koše, pri čemer iščemo tisti dve žogici,

ki pristaneta v istem košu. Da se bo to tekom podpisovanja zgodilo dokaj hitro, nam zagotavlja paradoks rojstnega dne. Hkrati pa tudi želimo, da zlonamerna oseba ne bo morala ponarediti sporočila. Oboje lahko enostavno predstavimo s primerom, ki ga nakazuje že samo ime sheme. Podpisnik ima na voljo večje število žogic, zato bosta hitro dve žogici v istem košu. Težava zlobneža pa je, da ima mnogo manj žogic. Razlog za razliko v številu žogic (žogice so zasebni ključi) je preprost. Podpisnik lahko generira poljubno število zasebnih ključev, za katere mora izračunati zgoščene vrednosti, da dobi javne ključe. Naloga napadalca pa je, da mora najti svoje zasebne ključe, katerih zgoščene vrednosti se ujemajo z enim izmed javnih ključev. Za to nalogo pa vemo, da je veliko težja.

Sedaj si poglejmo postopek podpisovanja in preverjanja. Dogovorimo se, da Bojan Ani pošlje sporočilo m. Denimo, da ima Bojan zasebne ključe sk_1 do sk_n. Zgoščene vrednosti zasebnih ključev pa je že varno posredoval Ani kot množico javnih ključev, ali pa ji je posredoval le koren Merklovega drevesa, ki vsebuje te zgoščene vrednosti zasebnih ključev.

8.1 Podpisovanje

Naj bosta sk_i in sk_j poljubna zasebna ključa. Naloga Bojana pri podpisovanju je, da najde par teh dveh ključev, pri čemer sta zgoščeni vrednosti sporočila, ki ji z operacijo stika dodamo enega izmed ključev, enaki. To lahko zapišemo kot $H(m||sk_i) = H(m||sk_j)$, pri čemer velja, da $sk_i \neq sk_j$. Da lahko Ana nato preveri veljavnost sporočila, ji poleg sporočila pošlje (sk_i, sk_j).

8.2 Preverjanje

Ko Ana dobi sporočilo, mora preveriti, da ga je res podpisal Boris. Najprej mora preveriti veljavnost podpisa. To stori tako, da izračuna $H(m||sk_i)$ in $H(m||sk_j)$ ter preveri, da se vrednosti ujemata. Preveriti mora tudi, da $sk_i \neq sk_j$. Sedaj mora dokazati le še, da podpis pripada Borisu. Če je Boris le objavil množico javnih ključev, mora Ana izračunati zgoščeni vrednost za sk_i in sk_j, ter jih poiskati v množici javnih ključev. V primeru, da pa je Boris uporabil Merklovo drevo, to stori z Merklovim dokazom.

8.3 Mogoče nadgradnje za povečanje varnosti

Možnosti za nadgradnje je več. Naj predstavim dve izmed njih.

Prva možnost je, da za veljaven podpis zahtevamo več parov zasebnih ključev, pri katerih se vrednost zgoščene vrednosti sporočila staknjenega s tem ključem ujemajo. Oziroma drugače, iščemo več parov (sk_i, sk_j), za katere veljajo zgornji pogoji.

Druga možnost je, da namesto para, za katerega valja $H(m||sk_i) = H(m||sk_j)$ iščemo trojice, četverke oz.: n-terice.

9 Zaključek

Da dokažemo, da smo neko sporočilo res napisali mi, ga podpišemo, pri čemer lahko uporabimo različne skupine podpisov in znotraj njih točno določeno podpisno shemo. V članku smo si pogledali enkratne podpise, za katere velja, da lahko en ključ uporabimo največ enkrat. Največji problem enkratnosti pa je ravno to, da lahko en javni ključ uporabimo zgolj enkrat, kar predstavlja veliko omejitev. Tako moramo namreč za vsako sporočilo najprej varno poslati javni ključ. Do neke mere to lahko izboljšamo s pomočjo Merklovega drevesa, zaradi katerega lahko nek javni ključ uporabimo nekajkrat (n-krat, pri čemer je n omejen). Najenostavnejša je Lamportova shema, kjer vsak bit

podpisujemo posebej, njena nadgradnja je Winternitzova shema, kjer podpisujemo po več (npr.: 8) bitov hkrati. Pogledali smo si tudi shemo zgoščene vrednosti za pridobitev naključne podmnožice. Kot zadnjo shemo pa smo videli, da lahko tudi rojstno dnevni paradoks uporabimo kot idejo in na podlagi njega zasnujemo shemo koši in žogice. Zanimivost pri enkratnih podpisih, ki smo jo lahko opazili v članku je, da so zasebni ključi zasebni le do njihove uporabe, saj so ti kasneje del podpisa.

Literatura

[1] Understanding the birthday paradox. URL `https://betterexplained.com/articles/understanding-the-birthday-paradox/`. Dostopano: 27.7.2020.

[2] Misrafest speaker - leslie lamport: Department of computer science. URL `https://www.cs.utexas.edu/node/70525`. Dostopano: 20.7.2020.

[3] Dec 2015. URL `https://cryptoservices.github.io/quantum/2015/12/07/few-times-signatures.html`. Dostopano: 5.8.2020.

[4] Birthday problem, Aug 2020. URL `https://en.wikipedia.org/wiki/Birthday_problem`. Dostopano: 27.7.2020.

[5] Johannes Buchmann, Erik Dahmen, Sarah Ereth, Andreas Hülsing, and Markus Rückert. On the security of the winternitz one-time signature scheme. In Abderrahmane Nitaj and David Pointcheval, editors, *Progress in Cryptology – AFRICACRYPT 2011*, pages 363–378, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg. ISBN 978-3-642-21969-6.

[6] FlatOutCrypto. Crypto intro: Merkle trees, Dec 2018. URL `https://flatoutcrypto.com/home/cryptointromerkletree`. Dostopano: 24.7.2020.

[7] Matthew Green. Hash-based signatures: An illustrated primer, Apr 2018. URL `https://blog.cryptographyengineering.com/2018/04/07/hash-based-signatures-an-illustrated-primer/`. Dostopano: 20.7.2020.

[8] H. Li, R. Lu, L. Zhou, B. Yang, and X. Shen. An efficient merkle-tree-based authentication scheme for smart grid. *IEEE Systems Journal*, 8(2):655–663, 2014.

[9] Adrian Perrig. The biba one-time signature and broadcast authentication protocol. *Proceedings of the 8th ACM conference on Computer and Communications Security - CCS '01*, 2001. doi: 10.1145/501983.501988.

[10] Leonid Reyzin and Natan Reyzin. Better than biba: Short one-time signatures with fast signing and verifying. In Lynn Batten and Jennifer Seberry, editors, *Information Security and Privacy*, pages 144–153, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. ISBN 978-3-540-45450-2.

[11] F. Shahid, I. Ahmad, M. Imran, and M. Shoaib. Novel one time signatures (nots): A compact post-quantum digital signature scheme. *IEEE Access*, 8:15895–15906, 2020.

[12] Vitalii Tymchyshyn and Andrii Khlevniuk. On the average number of birthdays in birthday paradox setup. 05 2019.