3. Domača naloga

Matija Kerkoč

May 24, 2019

Opisi funkcij so podani v Python datoteki, kjer so funkcije definirane, v dokumentu bomo opisali ideje, ki se skrivajo za njimi. Pri programiranju rešitve "verno" sledimo navodilom asistenta ter prosojnicam s predavanj, saj so navodila jasno podana, ter nas korak za korakom "vodijo" do rešitve.

Program **kvazi_trk.py** je namenjen generiranju trkov, praštevil p in q ter DSA ključa.

Iskanje kvazi-trka za SHA-1

Uporabimo lahko *paradoks rojstnih dni*. Le ta nam zagotavlja dovolj veliko verjetnost, da bomo že med nekaj naključnimi števili, izbranimi iz večje množice, našli kvazi-trk. Tako nam ni treba preverjati ogromno možnosti. Predpostavimo lahko tudi, da bo naš trk različen od vseh, ki so jih našli sošolci, saj je verjetnost, da bi našli isti trk dovolj majhna. Zaradi majhnega števila rešitev (cca. 20) lahko ta pogoj preverimo tudi sami brez pomoči računalnika.

Iskanje praštevil p in q

Za pomoč pri iskanju definiramo nekaj pomožnih funkcij. Generiramo praštevilo q in nato še praštevilo p, ki mora biti oblike $q|(p-1)$. Ker je praštevil, ki ustrezajo pogojem veliko, načeloma ni bojazni, da bi par $(p_0, q0)$ morebiti že izbral sošolec, seveda pa raje preverimo, saj je to lahko zelo velik problem. Če sošolec pozna naši praštevili p in q, lahko potem prebira vsa sporočila, ki so namenjena nam, prav tako pa tudi sporočila, ki jih mi pošiljamo drugim.

Praštevilskosti generiranih števil ne moremo preveriti na običajen način, zato uporabimo Miller-Rabin (ali Solovay–Strassen) probabilističen algoritem, ki nam z *dovolj veliko* verjetnostjo pove ali je število res praštevilo.

Generiranje DSA ključa

Vse funkcije v tem sklopu so napisane po prosojnicah s predavanj. Najpomembnejša je $inverz(a, q)$, saj nam pametno izračuna inverz. Idejo za tak

izračun sem dobil na strani StackOverflow. V resnici je to ideja iz algebre, ki nam pove, če gledamo inverz po modulu q, kjer je q praštevilo lahko inverz izračunamo pametneje, kot v kateri drugi grupi $\mathbb{Z}_m$, kjer m ni praštevilo.

Program **koncna_resitev.py** je namenjen podpisovanju rešitve s vnaprej generiranimi ključi ter iskanju veljavne rešitve po navodilih iz naloge.

Podpisovanje rešitve z DSA ključem

Našo rešitev po navodilih podpišemo s prej generiranim ključem. Funkcija prejme besedilo in ključ ter vrne podpisano besedilo oz. par (γ, δ).

Iskanje veljavne rešitve

Združimo vse funkcije skupaj po pravilih ter speminjamo, dokler končni hash ni veljavne oblike (hash se začne z "0000000").

Končna rešitev za Blokchain

Izbrani trk: 70347229 in 42888415,

Hash trka: 177703517928781483524900561654343230014068721055,

p: 232284838602164655545664240919671552933467791475214624003557
18388515063730806939828755776692858977242878982853152994612578
13087465461666677742624333914670682013093752526962948632609293
44623920728719865744397259889196060113621536156550796645998589
18556639680109034730628685328183936924924994532147505385571640563,

q: 1180278857667110501955963277249236557928848108173,

alfa: 11485982442857234985017737432108753116127071469251436069
03453253341394767908999710292214093272033047757683779250584679
86835796600466213699131682555861047075808967081328836953258587
78411792203294641741720365560383869220352738308137295010871322
00437401125887221319175401115015215173257881971718885854516534942388,

beta: 28503972274823058105014213675450876546254749254879800376553632
40982896009574133179429234162900358405990065423344542256665381609132
94550162008901221895889524125271833878377354069966251735550145796027
830268828652836428573291517651766349380467473798396237709967241659074
982144610158779464687432933502566875024301,

a: 31608283315653480989630074607762127350140874 2876.

gama: 827834273328481633893147227894769047440956337699,

delta: 2023909268092324576922376647820454456826018905 59,

Četrta vrstica: 83820753,

Blok sošolca pred mano: "000000076330d2c1cb7f1a511c22716ee78$bacc$3",

Moj veljaven blok: "00000000a77e76a42f52$ffbca$1910dc1882c1b16".

Zaključek

Največji izziv domače naloge je bilo generiranje velikih praštevil ter računanje z njimi. Treba je spisati učinkovite funkcije, saj drugače računanje porabi preveč časa. Seveda pa je časovno najpotratnejša zadnja funkcija, ki poišče veljaven blok, saj mora le ta pregledati veliko število možnih četrtih vrstic ter za vsako izračunati hash-e.

Izjemno previdni moramo biti pri generiranju praštevil p in q! Naši praštevili morata biti različni od praštevil sošolca, saj lahko v primeru, ko izberemo isti praštevili, ta sošolec podpisuje naša sporočila ter bere sporočila namenjena nam. Pri iskanju trkov načeloma ni potrebno biti previden, unikaten trk je le pogoj domače naloge.

Naloga je bila zelo zanimiva, saj smo lahko poskusili konstruiratu delček programov, ki oblikujejo eno izmed bolj "vročih" področij ta hip: kripto valute. Te namreč v veliki meri bazirajo na metodi blockchain-a. Pohvalil pa bi tudi sošolce, ki so bili zelo iznajdljivi pri iskanju trkov.