UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Lan Žagar

STROJNI GENERATORJI PRAVIH NAKLJUČNIH ŠTEVIL

Seminarska naloga pri predmetu
Kriptografija in teorija kodiranja 2

Mentor: izr. prof. dr. Aleksandar Jurišić

Ljubljana, 2008

Kazalo

Povzetek

Uporaba naključnih števil je pomemben sestavni del mnogih aplikacij, predvsem v kriptografiji, pri modeliranju in znanstvenih simulacijah pa tudi pri igrah na srečo. V ta namen želimo imeti dobre in hitre generatorje od katerih pričakujemo zaporedja števil, ki se po lastnostih čim bolj približajo teoretičnim rezultatom idealnega generatorja naključnih števil.

V grobem lahko ločimo generatorje psevdonaključnih števil ter generatorje pravih naključnih števil. Prvi so bili v literaturi že podrobno obdelani in so splošno bolje poznani. Drugi pa sicer postajajo vedno bolj razširjeni, a jih še ne poznamo dovolj dobro.

V tej seminarski nalogi sem zato skušal natančneje predstaviti tudi to vrsto generatorjev, jih primerjati s psevdonaključnimi generatorji in pokazati prednosti in pomakljivosti obeh vrst. Opravil sem tudi praktična testiranja treh generatorjev (računalnik Epia in pametni kartici IBM ter Schlumberger). Izkazalo se je, da vsi trije generatorji prestanejo zbirko statističnih testov.

Poglavje 1

Uvod

Naključna števila so potrebna v velikem številu aplikacij iz raznih področij, čeprav se mogoče kljub dnevni uporabi vsi tega ne zavedamo. Ne uporabljajo se samo v zahtevne, znanstvene namene, ampak tudi za bolj preproste in zabavne, kot na primer pri igrah in hazardiranju. Tudi povsem osnovni in stari problemi s katerimi se soočamo odkar živimo v skupnostih, so z njimi rešljivi. Ko hočemo izbrati nekoga v množici ljudi z enakimi kvalifikacijami lahko to rešimo z žrebom – naključno.

Kaj je naključnost? Točnega oz. enotnega odgovora na to ni, saj imamo v različnih kontekstih različne predstave o tem pojmu. Nekakšen preprost in splošen odgovor bi lahko bil, da je naključnost pomanjkanje urejenosti, vzorca, predvidljivosti, namena oz. vzročnosti. Naključnost je objektivna – neko zaporedje števil je naključno ali ne, čeprav se lahko nenaključno zaporedje zdi naključno oz. nepredvidljivo nekomu, ki ne pozna postopka generiranja ali uporabljenega ključa. Pri tem se pojavi vprašanje ali v našem vesolju prava naključnost sploh obstaja ali bi bilo mogoče (vsaj v teoriji) deterministično določiti vsak prihodnji dogodek na podlagi popolnega znanja o trenutnem stanju in poznavanja naravnih zakonov. Toda taki problemi so že domena filozofije.

Naključno zaporedje bitov si torej lahko predstavljamo kot rezultat neodvisnih metov “poštenega” kovanca z vrednostima ”0”in ”1”. Neodvisnost metov pomeni, da prihodnji meti niso odvisni od rezultatov prejšnjih metov, in “poštenost” kovanca, da je pri vsakem metu verjetnost vrednosti ”0”in ”1”enaka $\frac{1}{2}$. Tak (teoretični) kovanec predstavlja idealni generator naključnih števil oz. v tem primeru bitov, čeprav

navadno govorimo kar o generatorjih števil. Ker so večja števila v računalnikih tako ali tako predstavljena z več biti, sta namreč generator bitov in števil ekvivalentna – iz prvega se da dobiti drugega in obratno.

S prihodom računalnikov se je pojavila tudi zahteva po hitrih in kvalitetnih generatorjih naključnih števil. Sam po sebi lahko računalnik namreč samo sledi nekemu v naprej definiranemu postopku in zato da pri istem vhodu vedno isti izhod. Vendar je uporaba naključnih števil nepogrešljiva pri moderni kriptografiji, statističnem modeliranju in simuliranju iger v kazinih ter žrebanjih na loterijah. Vse te aplikacije seveda potrebujejo nek vir naključnosti, ki ga v računalnik vnesemo z uporabo generatorjev naključnih števil. Nekatere zahteve lahko zadovoljimo s programskimi generatorji psevdonaključnih števil, ponekod pa so kriteriji višji in moramo poseči po strojnih generatorjih pravih naključnih števil.

V naslednjem poglavju bomo spoznali in pregledali področje generiranja naključnih števil, kjer bo opisana delitev generatorjev in primerjava njihovih lastnosti in značilnosti. V 3. poglavju bodo bolj natančno opisani strojni generatorji, principi po katerih delujejo ter njihove značilnosti. Sledi razdelek o beljenju, pomembnem sestavnem členu vseh strojnih generatorjev. Na koncu pa bodo predstavljeni načini testiranja ter ocenjevanja različnih generatorjev in povzetek praktičnih testov, ki sem jih izvedel na generatorjih računalnika Epia (s procesorjem iz družine VIA C3 s strojnim generatorjem naključnih števil) ter dveh različnih modelih pametnih kartic (IBM in Schlumberger).

Poglavje 2

Generiranje naključnih števil

2.1 Vrste generatorjev

Generator naključnih števil je računska ali fizična naprava namenjena generiranju naključnih števil. Kot je bilo omenjeno že v uvodu, se to bolj splošno ime uporablja tudi za generatorje naključnih bitov. Še posebej v primeru strojnega generiranja v resnici skoraj vedno operiramo z biti, ki jih naknadno lahko pretvorimo v večja števila. V nadaljevanju bom zato uporabljal kar enoten izraz "Generator naključnih števil". Ker se različni tipi generatorjev razlikujejo že v samih osnovah, si poglejmo najprej delitev v tri večje razrede in opišimo značilnosti vsakega posebej.

Generatorji psevdonaključnih števil (ang. pseudo random number generators – PRNG) oz. deterministični generatorji naključnih števil (ang. deterministic random number generators - DRNG) so programi oziroma postopki, ki na podlagi računanja generirajo psevdonaključna števila. O pravi naključnosti ne moremo govoriti, ker generiranje poteka po točno določenem, determinističnem postopku. Da je izhod vsakič drugačen, poskrbijo različna začetna notranja stanja (ang. seed). Če pa začnemo z istim stanjem, bo tudi izhod vsakič isti. Velikost notranjega stanja je torej odločilnega pomena za to vrsto generatorjev. Zaradi njihove večje razširjenosti, bolj jasnih matematičnih lastnosti in povezave s tokovnimi šiframi, je bil ta tip generatorjev zelo dobro obdelan in raziskan.

Znani predstavniki so linearni kongruenčni generator, LFSR in njegove nelinearne različice, generator Blum Blum Shub, lahko pa v ta namen uporabimo

tudi večino tokovnih šifer in zgoščevalnih funkcij.

Generatorji pravih naključnih števil so fizične naprave, ki izkoriščajo različne fizikalne pojave za generiranje naključnih števil. Dve enaki napravi naj ne bi dali istega izhoda, čeprav je kakršnokoli začetno stanje enako (tipično take naprave niti nimajo notranjega stanja). Te naprave se pojavljajo pod dvema imenoma : strojni generatorji naključnih števil (ang. hardware random number generators – HRNG) ali generatorji pravih naključnih števil (ang. true random number generators – TRNG). "Strojni" saj je za take generatorje potrebna posebna strojna oprema (v nasprotju s psevdogeneratorji, ki so lahko implementirani programsko) in "pravi", ker ne generirajo psevdonaključnih ampak prava, nepredvidljiva naključna števila. Pri prvem izrazu moramo biti pazljivi, saj so generatorji psevdonaključnih števil lahko implementirani tudi s strojno opremo (namensko vezje), tako da bi lahko imeli strojni generator psevdonaključnih števil. Vendar pa se za take naprave navadno ne uporablja ta izraz, saj je iz konteksta očitno, za katero vrsto generatorja gre.

Hibridni generatorji naključnih števil so naprave sestavljene iz komponent obeh zgoraj omenjenih vrst. V tipični zasnovi se strojni generatorji uporabljajo kot vir entropije, ki se potem vnaša v notranje stanje generatorja psevdonaključnih števil, ki to entropijo razporedi med daljše zaporedje števil. Ponavadi se to vrsto generatorjev obravnava kar v okviru zgornjih dveh tipov. Če namreč nadziramo količino vnesene entropije in proces ustavimo, ko je ni zadosti, se tak generator obnaša podobno kot strojni generator. Če pa tega ne počnemo in nadaljujemo s postopkom tudi ko ni dovolj nove entropije, se v tem času generator obnaša kot psevdonaključni generatorji in ga pri večini analiz, ki opazujejo najslabši možen (ang. worst-case) scenarij, tudi obravnavamo kot slednjega. Najbolj znana predstavnika tega razreda sta verjetno `/dev/random` in `/dev/urandom` generatorja operacijskega sistema Linux. Vir entropije so uporabnikove interakcije in gonilniki naprav. Razlikujeta se v tem, da prvi blokira v primeru pomanjkanja entropije in je zato podoben generatorjem pravih naključnih števil, drugi pa nadaljuje z delovanjem in je zato podoben psevdonaključnim generatorjem.

2.2 Razlike in primerjava

Zaradi bistvenih razlik v delovanju in konceptu obeh glavnih vrst generatorjev, psevdonaključnih in strojnih (pravih), jih ne moremo neposredno primerjati in reči, kateri tip je boljši. Vsaka zasnova ima svoje prednosti in slabosti, ki jih moramo poznati, da lahko izberemo pravi tip za določeno opravilo. Sledeča primerjava je zato namenjena informiranju in spoznavanju obeh vrst.

Pa začnimo z razlogi, zakaj generatorji pravih naključnih števil še niso izpodrinili psevdonaključnih. Na prvi pogled bi se namreč marsikomu zdelo, da je prava naključnost očitno boljša od psevdonaključnosti. Glavni razlogi so gotovo **dostopnost**, **praktičnost** ter **cena**. Za pravi vir naključnosti potrebujemo napravo, ki meri oz. izkorišča neke fizikalne pojave. To pomeni dodaten kos strojne opreme, ki pri večini osebnih računalnikov (še) ni standard (čeprav se zadnje čase spreminja tudi to, predvsem zaradi potreb varnosti). Psevdonaključni generatorji so po drugi strani navadni programi, ki jih lahko izvajamo na vseh računalnikih. Zelo veliko jih je prosto dostopnih z možnostjo preverjanja in spreminjanja kode.

Psevdonaključni generatorji imajo tudi dejanske prednosti, ki presegajo zgolj večjo prikladnost in razširjenost. Kjer zahteve po kvaliteti niso glavna ovira, je pa zelo pomemben in pogosto najbolj omejujoč faktor **hitrost**, je ponavadi prva izbira programski generator psevdonaključnih števil. Le-ti so lahko zelo hitri in še vedno ohranjajo dobre statistične lastnosti, strojni generatorji pa morajo ponavadi za večjo hitrost žrtvovati kvaliteto. Razlika je še bolj izrazita ko potrebujemo večja števila, saj se vse operacije v večini današnjih računalnikov vršijo nad 32 ali 64 bitnimi števili, medtem ko strojni generatorji operirajo s posameznimi biti oz. bajti. Ta problem je sicer rešljiv z malo več denarja, saj imamo lahko v eni napravi več strojnih generatorjev, ki delujejo vzporedno. Prav tako ne smemo pozabiti, da je delovanje strojnih generatorjev **asinhrono**, kar pomeni, da naključna števila lahko računajo ločeno in jih imajo že pripravljena, ko jih potrebujemo. Prednost psevdonaključnih generatorjev je tudi v tem, da so bolj **preizkušeni**, zato zbujajo večje zaupanje. Lahko se zanesemo, da bodo delovali v vseh pogojih (dokler pravilno deluje računalnik) in se obnašali tako, kot so se med njihovim ocenjevanjem in preučevanjem.

Po drugi strani je jasno, kaj je glavna odlika strojnih generatorjev. Postopek pridobivanja naključnih števil ni **deterministično**, kar pomeni pravo nepredvidljivost in ne samo pogojne, računske. Entropija se namreč veča sorazmerno z dolžino zaporedja in ni, kot pri psevdonaključnih generatorjih, omejena z začetno entropijo. To je torej edini pravi in varen način pridobivanja nove entropije, ki jo potrebujemo tudi za inicializacijo notranjega stanja pri generatorjih psevdonaključnih števil. Še ena prednost je **aperiodičnost** generiranega zaporedja (nikoli se ne začne ponavljati), čeprav tudi periodični generatorji zaradi izjemno dolgih period s tem navadno nimajo težav.

Kratek povzetek zgoraj opisanih prednosti enih in drugih generatorjev je v tabeli 2.1.

HRNG	PRNG
nedeterminističnost	hitrost
aperiodičnost	cena, dostopnost
asinhronost	raziskanost
	zanesljivost

Tabela 2.1: Prednosti pravih ter psevdonaključnih generatorjev

Zaradi zgoraj omenjenih prednosti in slabosti je priporočljiva vrsta generatorja odvisna od ciljne aplikacije in njenih zahtev ter pogojev. Nekaj tipičnih področij uporabe je naštetih v tabeli 2.2.

HRNG	PRNG
kriptografija	simulacije
varnost	vzorčenje
igralni avtomati	programiranje
inicializacija PRNG	

Tabela 2.2: Uporaba pravih ter psevdonaključnih generatorjev

Poglavje 3

Strojni generatorji pravih naključnih števil

V prejšnjem poglavju so bili predstavljeni strojni generatorji pravih naključnih števil kot eden glavnih vrst generatorjev. Izpostavljene so bile glavne razlike v lastnostih ter uporabi v primerjavi z generatorji psevdonaključnih števil. V tem poglavju se bomo pobližje spoznali z zgradbo in principi delovanja strojnih generatorjev, ter z njihovimi specifičnimi lastnostmi in možnimi problemi.

Vsi strojni generatorji v principu so zgrajeni podobno. Vir naključnosti je nek fizikalni pojav z ustreznimi lastnostmi. Naprava ima senzor, ki meri analogni signal, pri čemer ga moramo velikokrat za to najprej ojačati (če se ukvarjamo s težko zaznavnimi šumi). Digitalizator nato analogni signal pretvori v digitalnega – binarno (0/1) spremenljivko. Ti rezultati se še dodatno obdelajo takoj pri/po zajemu ali malo kasneje, da se poskrbi za boljše statistične lastnosti. Naprava nato shranjuje pridobljena naključna števila v za to namenjenem medpomnilniku. Tu se lahko na malo večjem vzorcu preverja, da števila res izgledajo naključno in naprava pravilno deluje. Ko se generator uporabi, se iz medpomnilnika prebirajo števila, v primeru da jih imamo na zalogi vsaj neko v naprej določeno minimalno količino.

3.1 Viri naključnosti

Omenili smo, da pravo naključnost strojni generatorji črpajo iz fizikalnih pojavov. Kateri pojavi izkazujejo zadostno naključnost? Čeprav smo navajeni, da se

makroskopski pojavi obnašajo po zakonih newtonske fizike, so lahko tudi ti dober vir naključnosti. Utemeljitev za to lahko najdemo v teoriji kaosa in dinamičnih sistemov. To omogoča, da so se preprosti sistemi, kot sta na primer ruleta in naprava z žogicami za žrebanje številk pri loteriji, obdržali do danes. Naključnost, ki ju z njima dosežemo, je namreč dovolj dobra tudi za današnje zelo visoke standarde. Tudi primitivni napravi kot sta kocka in kovanec sta dovolj dobri, a seveda neuporabni, če želimo, da se samodejno pridobi večja količina naključnih števil. Tudi za avtomatsko in predvsem dovolj hitro izvajanje, potrebno za uporabo z računalniki, se da razviti naprave, ki temeljijo na kaotičnih sistemih. To so pokazali v [1], kjer so predstavili prednosti svojega načrta za kaotični generator naključnih števil. Tako so dokazali, da ne temeljijo vsi moderni strojni generatorji na kvantnih pojavih. Vseeno pa lahko rečemo, da jih večina temelji na kvantnih procesih, ki se odvijajo na mikroskopski oz. že kar (sub)atomski ravni.

Nekaj pogostejših virov je :

- zrnati šum
- termični šum
- atmosferski radijski šum
- čas med radioaktivnim razpadom delcev
- odstopanja pri frekvenci ur in oscilatorjev
- ...

3.2 Posebnosti

Ko se ukvarjamo s strojnimi generatorji naključnih števil se moramo zavedati nekaj pomembnih posebnosti, da jih ne obravnavamo enako kot ostale generatorje (bolj smo namreč navajeni dela z psevdonaključnimi generatorji). Posebej moramo paziti na dejstvo, da delovanje strojnih generatorjev ni odvisno samo od zgradbe in principov delovanja, ampak tudi od pogojov in okolja v trenutku uporabe. To odpira vrata za nove vrste napadov, če ima napadalec fizični dostop do uporabljene naprave. Analogni signali, ki jih strojni generatorji merijo, so namreč pogosto odvisni od

dejavnikov, kot so temperatura in zunanje motnje (npr. močnejši radijski signali, elektromagnetna valovanja ...).

Omeniti velja tudi, da kot pri psevdonaključnih generatorjih tudi tu pogosto ne moremo pravilnega delovanja dokazati. V takih primerih je pomembna daljša prisotnost med stroko in razkriti principi delovanja, da lahko v uporabljane rešitve zaupamo in verjamemo, da je odkritje (ali celo že prisotnost) napadov malo verjetna. V tem smislu so strojni generatorji preučevani malo premalo časa. Preveč je tudi skrivanja konceptov in premalo odprte dokumentacije.

Poglavje 4

Beljenje

Strojni generatorji imajo bolj kot s predvidljivostjo probleme s slabimi statističnimi lastnostmi, kar je posledica načina pridobivanja naključnih številk. Največji težavi sta povezani predvsem s pristranskostjo (verjetnost "0" je več ali manj kot 0.5) in korelacijo (sosednji biti v zaporedju niso neodvisni). Toda v primeru, da dobivamo zadostno količino entropije, se da take preproste statistične probleme tudi naknadno popraviti. Procesiranju, ki je temu namenjeno, pravimo beljenje (ang. whitening), saj prvotni signal (vhodno zaporedje) približa belemu šumu (naključnemu zaporedju). V literaturi se uporabljajo tudi drugi izrazi kot na primer "de-skewing" (v [2]) ali kar ekstrakcija entropije [3].

Obstajata dva pristopa beljenja.

Pri prvem za odpravljanje pristranskosti poskrbimo že na nivoju strojne opreme, takoj za izvedbo samih meritev fizikalnega pojava.

Popravljanje praga

Vzemimo primer, kjer pri digitalizaciji signala pretvorimo analogno meritev v binarno vrednost glede na to ali je večja ali manjša od neke srednje vrednosti (praga). Zaradi zunanjih vplivov ali staranja se lahko dejanska srednja vrednost s časom spreminja, kar povzroči pristranskost generatorja. Možna rešitev je sprotno spremljanje zajetih analognih vrednosti in popravljanje srednje vrednosti, ki se uporabi pri digitalizaciji [4]. Vendar pa tak postopek lahko na račun nepristranskosti uvede dodatno korelacijo [5] – popravljena srednja vrednost, ki vpliva na naslednje bite, je namreč odvisna od prejšnjih bitov.

Novejše rešitve raje popravljajo pristranskost v korektorju, takoj za procesom digitalizacije. V [6] so za to uporabili binarni števec, uspelo pa jim je tudi izpeljati formulo za zgornjo mejo korelacije med biti.

Drug pristop je, da pridobljene bite naknadno računsko obdelamo. Za tako obdelavo je veliko možnih postopkov, najbolj razširjeni pa so :

Von Neumannov postopek

preprost postopek, ki ga je že leta 1951 predlagal John von Neumann [7]. Z njim lahko pristranskost generatorja popolnoma odpravimo, toda ob močni predpostavki, da so dobljeni biti med sabo neodvisni. Takega zagotovila v praksi žal nimamo in lahko se zgodi, da se pristranskost le zmanjša.

Opis postopka:

vhodno zaporedje $x_1, x_2, \ldots$

izpiši x_{2i}, če in samo če $x_{2i} \neq x_{2i+1}$

Pri generatorju z verjetnostjo $P(X_i = 0) \approx \frac{1}{2}$ (majhno pristranskostjo) je izhodno zaporedje dolgo približno četrtino vhodnega. V primeru večje pristranskosti, pa se izkoristek slabša.

XOR

Zelo pogost način je tudi, da dva ali več bitov seštejemo (XOR). To za razliko od prejšnega postopka tudi v idealnih pogojih (neodvisnost bitov) ne izniči pristranskosti ampak jo samo zmanjša. Vendar ker tako ali tako biti večinoma niso neodvisni, je ta postopek lahko prav tako dober. Matematično ozadje in motivacija za ta pristop slonita na lemi o kopičenju (ang. piling-up lemma), ki jo je v [8] predstavil in dokazal Mitsuru Matsui.

Če za nekorelirano, pristransko zaporedje bitov velja: $P(X_i = 0) = \frac{1}{2} + b_i$

Potem sledi iz leme, da: $P(X_1 \oplus X_2 \oplus \ldots \oplus X_n = 0) = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} b_i$

Prednost tega postopka je, da lahko seštejemo bite, ki so zadosti narazen, da je korelacija med njimi zanemarljivo majhna in se obnašajo (skoraj) kot neodvisni biti. Dobro je tudi da imamo zagotovljeno prepustnost – izhodno zaporedje je veliko točno polovico vodnega (oz. tretjino če seštevamo po 3 bite itd.)

Beljenje s PRNG

Pri tem postopku izhod strojnega generatorja prebelimo z izhodom generatorja psevdonaključnih števil tako, da bite obeh zaporedja seštevamo (XOR). Dober psevdonaključni generator je nepristranski in ima dobre statistične lastnosti, zato je tako tudi izhodno zaporedje. Zaporedje strojnega generatorja pa poskrbi za doprinos entropije. Kombinacija se zdi idealna, toda paziti moramo na varnost. V primeru, da napadalec pozna uporabljeno psevdonaključno zaporedje, lahko dobi tudi neobdelano zaporedje strojnega generatorja. To ima slabše statistične lastnosti in je zato bolj občutljivo na nadaljnje napade. Psevdonaključni generator mora biti zato dovolj dober, kar pa navadno pomeni počasnejše delovanje (npr. Blum Blum Shub, primer kriptografsko varnega generatorja, je zelo počasen).

Zgoščevalne funkcije (ang. Hash functions)

Kljub statističnim pomanjkljivostim neobdelanega zaporedja lahko to vseeno vsebuje dosti entropije (npr. 0.7 na bit ali več). V tem primeru lahko vzamemo večje število bitov in jih z uporabo zgoščevalnih funkcij stisnemo. Rezultat je krajše zaporedje, za katerega upamo, da je zadržalo čim več začetne entropije in ima statistične lastnosti podobne idealnemu naključnemu zaporedju. Dejanski rezultati postopka so močno odvisni od lastnosti zgoščevalne funkcije. V praksi kvaliteta rezultatov zato večinoma ni dokazljiva. Možen pa je tudi bolj formalen pristop, s katerim je mogoče statistične lastnosti dokazati. V [3] to dosežejo z uporabo paroma neodvisnih zgoščevalnih funkcij.

Prednost beljenja s pomočjo zgoščevalnih funkcij je hitrost izvajanja teh funkcij in možnost prilagajanja razmerja med hitrostjo ter kvaliteto. Tak pristop uporablja med drugim Linuxov generator `/dev/random`, pri katerem je za zgoščevalno funkcijo uporabljen algoritem SHA-1.

Poglavje 5

Testiranje in ocenjevanje

Zaradi velikega števila različnih zasnov in implementacij generatorjev naključnih števil je potrebno imeti načine ocenjevanja in preverjanja njihovega delovanja. Področja povezana z varnostjo (kriptografija) in denarjem (loterije, igre na srečo) imajo visoke zahteve po zanesljivosti in kvaliteti. Slabi generatorji v takih primerih preprosto niso sprejemljivi.

5.1 Statistični testi

Statistični testi (pregled in opisi v [9]) so osnoven način preverjanja naključnosti zaporedja. To so testi, ki preverjajo, ali ima dano zaporedje enake statistične lastnosti kot bi jih pričakovali od izhoda idealnega generatorja naključnih števil. Čeprav takega idealnega generatorja v praksi nimamo, lahko veliko lastnosti matematično izračunamo. Vsak posamezen test preverja neko značilnost, pričakujemo pa seveda, da bo naš generator prestal vse teste. Čim pade na enem, imamo namreč način ločevanja izhoda tega generatorja od pravega naključnega zaporedja.

Glavni problem takega testiranja je, da z njim ne moremo potrditi kvalitete generatorja, lahko jo samo z veliko verjetnostjo ovržemo. Da generator opravi vse statistične teste, je namreč potreben pogoj in ne zadosten. Če bi hoteli biti prepričani v pravo naključnost generatorja, bi ta moral prestati vse teoretično možne statistične teste, kar pa je v praksi nemogoče preizkusiti.

Temu se seveda hočemo približati, zato se je pojavilo veliko množic testov (NIST, Diehard, FIPS 140-1,...), ki iščejo možne pomanjkljivosti. Podobno kot pri kriptografiji, kjer je večen boj med razvijalci in razbijalci šifer, se tudi tukaj bojujejo

razvijalci generatorjev naključnih števil ter razvijalci statističnih testov, ki te generatorje razpoznavajo. Zaenkrat gre bolje razvijalcem generatorjev, saj jih je kar nekaj, ki prestanejo vse teste. Če štejemo tudi strojne generatorje, je možno, da se nekaterih generatorjev na podlagi njihovega (končnega) izhoda ne da ločiti od idealnega (teoretičnega) naključnega generatorja.

5.2 Testiranje strojnih generatorjev

Pri psevdonaključnih generatorjih poženemo nabor statističnih testov na zgeneriranem zaporedju in s tem preverimo njegovo kvaliteto. Pri strojnih generatorjih moramo pristopiti malo drugače, saj njihova glavna naloga ni samo, da zaporedje izgleda naključno (ima dobre statistične lastnosti), ampak tudi da vsebuje čim več entropije. To je zelo težko preverjati in morajo navadno avtorji generatorja nekako teoretično utemeljiti količino entropije. Pri tem pogosto uporabljajo predpostavke, ki v resnici niso izpolnjene (oz. jih ne moremo dokazati).

Vseeno želimo tudi sprotno preverjati delovanje generatorja, saj kot smo omenili lahko zaradi zunanjih pogojev ali staranja kvaliteta izrazito pade ali generator popolnoma odpove. V ta namen je bolje testirati surovo (neobdelano) zaporedje pred postopkom beljenja. Slednje lahko namreč tudi ob popolni odpovedi vira naključnosti vseeno tako dobro popravi zaporedje, da prestane statistične teste.

Proizvajalci zato praviloma kontrolirajo delovanje že v sami napravi. To je narejeno z ožjim izborom zelo osnovnih statističnih testov nad števili v medpomnilniku (takoj po digitalizaciji). Preverjajo se samo tiste lastnosti, za katere se ve, da bodo v primeru slabšega delovanja izrazito odstopale od pričakovanega. Teste lahko glede na funkcijo preverjanja razdelimo v 3 skupine : začetno preverjanje (ang. startup test) preveri delovanje generatorja ob zagonu, sprotno preverjanje (ang. online test) preprečuje poslabšanje med delovanjem zaradi neugodnih zunanjih pogojev, tako imenovani “tot test” pa preverja, da ni generator popolnoma odpovedal.

5.3 Testi v praksi

Na voljo sem imel računalnik Epia z VIA C3 procesorjem, ki ima tudi strojni varnostni modul (PadLock) z generatorjem pravih naključnih števil. Računalnik je imel naložen operacijski sistem Linux, sistemska RNGja (`/dev/random`, `/dev/urandom`) pa sta znala namesto privzetega vira entropije uporabljati strojni generator. Ta

dva generatorja sem podvrgel standardni seriji statističnih testov Dieharder (opisani v [9]. Oba sta prestala vse preizkuse, tako da se ju s tem naborom testov ne bi dalo ločiti od teoretičnega idelanega generatorja. Zaradi hitrega delovanja (68 kB/s za `/dev/random` in kar 200 kB/s za `/dev/urandom`) sem lahko zgeneriral zadostno količino podaktov (1GB) in teste ponovil tolikokrat, da res ni bilo dvoma o rezultatih.

Preizkusil sem tudi oba načina generiranja (varni kot pri `/dev/random` in psevdonaključni kot pri `/dev/urandom`) naključnih števil na pametnih karticah podjetij IBM ter Schlumberger. V sami dokumentaciji algoritmi in način pridobivanja naključnosti niso eksplicitno opisani (namenoma, saj delovanja nočejo popolnoma izdati), zato je lastno testiranje še toliko bolj smiselno.

Pri obeh generatorjih je bil omejujoč faktor hitrost generiranja (več 10-krat počasneje kot pri Epiji), tako da sem imel na voljo le 16MB podatkov vsake vrste. Vseeno je bilo to dovolj, da se je pokazalo, da tudi obe pametni kartici prestaneta ta nabor statističnih testov.

Zaključek

Spoznali smo glavne značilnosti strojnih generatorjev pravih naključnih števil in njihov osnovni princip delovanja ter zgradbe. Primerjali smo jih z bolj poznanimi generatorji psevdonaključnih števil in izpostavili prednosti enih in drugih. Na podlagi te primerjave smo predlagali katero vrsto generatorjev uporabiti na različnih področjih. Predstavil sem nekaj najbolj razširjenih metod beljenja, ki je pomemben postopek za nakdnadno izboljšavo lastnosti zgeneriranega zaporedja. Opisal sem načine preverjanja generatorjev naključnih števil in določene teste (nabor Dieharder) tudi v praksi izvedel na računalniku Epia ter dveh pametnih karticah (IBM, Schlumberger)

Literatura

[1] D. P. Maher, R. J. Rance, *Random Number Generators Founded on Signal and Information Theory*, Workshop on Cryptographic Hardware and Embedded Systems, 1999

[2] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, *Handbook of applied Cryptography*, 1996

[3] B. Barak, R. Shaltiel, E. Tromer, *True Random Number Generators Secure in a Changing Environment*, Workshop on Cryptographic Hardware and Embedded Systems, 2003

[4] A. Yarza, P. Martinez, *A True Random Pulse Train Generator*, Electronic Engineering 50 No. 614, 1978

[5] C. H. Vincent, *The Generation of Truly Random Binary Numbers*, Journal of Physics E 3 No. 8, 1970

[6] V. Bagini, M. Bucci, *A Design of Reliable True Random Number Generator for Cryptographic Applications*, Workshop on Cryptographic Hardware and Embedded Systems, 1999

[7] J. von Neumann, *Various techniques used in connection with random digits*, Applied Math Series 12, 1951.

[8] M. Matsui, *Linear Cryptanalysis Method for DES cipher*, Advances in Cryptology - Eurocrypt'93, 1993

[9] T. Novak, *Preverjanje naključnosti v kriptografiji*, diplomsko delo, 2000