Portál pre odborné publikovanie ISSN 1338-0087

Chybová analýza prúdových šifier

Hromada Viliam · Informačné technológie, Študentské práce

11.01.2012

Článok sa zaoberá problematikou chybovej analýzy prúdových šifier, resp. chybových útokov. Sú predstavené základné typy útokov na dva typy konštrukcie prúdových šifier založených na lineárnych spätnoväzobných registroch a dva konkrétne chybové útoky na prúdovú šifru LILI-128, kandidáta v súťaži NESSIE.

1. Úvod

Dnešné kryptosystémy, resp. šifrovacie algoritmy, delíme podľa toho, ako šifrujú ten istý blok textu, na blokové a prúdové šifry. Zatiaľ, čo blokové šifry zašifrujú vždy ten istý otvorený text, resp. jeho časť, na ten istý zašifrovaný text (v základnom režime ECB), pri prúdových šifrách to neplatí, t.j. rovnaké bloky textu zašifruje bloková šifra na iné bloky zašifrovaného textu.

Útoky na tieto kryptosystémy možno deliť na dve triedy: na priame útoky a na nepriame útoky. Priame útoky sú zamerané na algoritmickú podstatu kryptosystému, bez ohľadu na jeho implementáciu. Nepriame útoky využívajú (zneužívajú) fyzickú implementáciu kryptosystému a zahŕňajú širokú paletu techník, ktoré alebo poskytujú útočníkovi nejakú „vnútornú“ informáciu o procese šifrovania (napríklad časová alebo napäťová analýza), alebo mu dovoľujú tento proces ovplyvniť (preklápanie bitov v pamäti zariadenia pomocou žiarenia, atď.). Chybová analýza študuje, aký efekt majú jednotlivé indukované chyby na zašifrovaný text, s cieľom získať aspoň čiastočnú informáciu alebo o kľúči, alebo o vnútornom stave šifrovacieho zariadenia.

Chybová analýza bola prvýkrát použitá v roku 1996 kryptoanalytikmi Bonehom, Demillom a Liptonom na útok voči kryptosystémom s verejným kľúčom založených na problémoch vyplývajúcich z teórie čísel (konkrétne išlo o útok na RSA s chybne implementovaným algoritmom počítania Čínskej zvyškovej vety) a neskôr bola použitá Bihamom a Shamirom ako základ útoku na súčinové blokové šifry (napríklad DES). Zatiaľ, čo tieto techniky boli zovšeobecnené a aplikované na útoky voči iným blokovým šifrám a šifrovacím systémom s verejným kľúčom, donedávna existovalo málo výsledkov zameraných na podobné útoky na prúdové šifry.

Tento článok popisuje základné techniky chybových útokov na dva typy konštrukcií prúdových šifier založených na lineárnych spätnoväzobných registroch, konkrétne na lineárny spätnoväzobný register s nelineárnou filtrovacou funkciu a na lineárny spätnoväzobný register, ktorého časové riadenie sa riadi výstupom iného lineárneho

spätnoväzobného registra. Na záver je uvedená ukážka dvoch konkrétnych útokov na prúdovú šifru LILI-128, ktorá predstavuje kombináciu použitia filtrovacej funkcie a časového riadenia.

2. Chybové útoky na prúdové šifry

Jedným zo základných stavebných kameňov prúdových šifier sú tzv. lineárne spätnoväzobné registre (ďalej len LFSR z anglického Linear Feedback Shift Register). LFSR sa v praxi používajú najmä pre ich jednoduchú hardvérovú implementáciu, dobré štatistické vlastnosti výstupných postupností a veľké periódy výstupných postupností (v prípade správne zvolených charakteristických polynómov). Avšak, ich nevýhodou je, že sú lineárne, t.j. každý výstupný bit je lineárnou kombináciou bitov počiatočného naplnenia (v súlade s príslušnou lineárnou diferenčnou rovnicou). Preto dochádza k ich spájaniu s nelineárnymi komponentmi. V podstate existujú 3 typy konštrukcií prúdových šifier založených na LFSR:

- výstup LFSR je filtrovaný pomocou nelineárnej funkcie
- taktovanie LFSR je riadené výstupom iného LFSR
- výstup LFSR je filtrovaný pomocou konečného stavového automatu

My sa zameriame na prvé dve konštrukcie. Útok považujeme za úspešný, ak sa nám podarí nájsť počiatočné naplnenie registrov.

2.1 Chybový útok na prúdovú šifru s nelineárne filtrovaným LFSR

Nech $(x_1, x_2, \ldots, x_n)$ je vnútorný stav LFSR, kde $x_i \in \{0, 1\}$. LFSR nazývame nelineárne filtrovaným, ak sa na bity registra aplikuje nelineárna booleovská funkcia $f(x_{i1}, x_{i2}, \ldots, x_{it})$, ktorej vstup tvorí podmnožina vnútorných stavových bitov registra s mohutnosťou t. Vo všeobecnosti môže vstup do filtrovacej funkcie pochádzať z viacerých lineárnych spätnoväzobných registrov. Každý výstupný bit sa teda určí alebo na základe výpočtu funkcie f, alebo sa použije tzv. „look-up“ tabuľka s vopred vypočítanými hodnotami funkcie f pre všetky možné hodnoty premenných $x_{i1}, x_{i2}, \ldots, x_{it}$. Predpokladajme teraz, že útočník dokáže vyvolať chyby s nízkou Hammingovou váhou na vnútorných stavoch registra (t.j. dokáže preklopiť niektoré bity registra). Toto sa dá následne využiť na útok nasledovným spôsobom:

Algoritmus 1 Útok na nelineárne filtrovaný LFSR

1. Spôsob chybu a vygeneruj príslušný prúdový kľúč
2. Odhadni miesto indukovania chyby
3. Skontroluj odhad chyby použitím algoritmu 2, ak si neuhádol, hádaj znova
4. Opakuj kroky 1. – 3. kým nenazbieraš O(t) identifikovaných chýb
5. Vytvor a vyrieš sústavu rovníc nad poľom GF(2) nad pôvodným naplnením registra

Algoritmus 2 Kontrola odhadu chyby

1. Predikuj vývoj budúcich diferencií vo vstupe funkcie f v závislosti na počiatočnej chybe
2. Nájdi vstupné bity f s predikovanou nulovou diferenciou
3. Ak sa na niektorom mieste vyskytuje nenulová výstupná diferencia, zamietni odhad

Jedinou podmienkou fungovania tohto útoku je schopnosť predikovať vývoj chyby (vývoj diferencie). Preto sa útok dá zovšeobecniť na viacero lineárnych spätnoväzobných registrov pripojených na tú istú nelineárnu filtrovaciu funkciu. Žiaľ, tento útok funguje len v prípade, že Hammigova váha indukovanej chyby je malá.

2.1.1 Odhad chyby

V algoritme 1 hrá dôležitú úlohu odhad chyby, t.j. určenie pozície registri, kde došlo k preklopeniu pôvodného bitu. Tento odhad je náhodný, t.j. tipneme si, na ktorých miestach došlo k chybe a snažíme sa overiť náš odhad. Na to potrebujeme vedieť, ako sa indukovaná chyba ďalej šíri v registri (registroch). Vďaka linearite LFSR, resp. vďaka lineárnej operácií taktovania (označme ju L), nie je problém predikovať šírenie chyby v ďalších taktoch, ak poznáme počiatočnú diferenciu (označme ju Δ).

Diferencia v i-tom takte je teda $L^i(\Delta)$. Na overenie nášho odhadu potrebujeme predikovať diferencie t bitov, ktoré vstupujú do funkcie f. Ak bol náš počiatočný odhad správny, očakávame, že ak vstupná diferencia týchto bitov je nulová, aj výstupná diferencia z funkcie f bude nulová. Ak bol náš odhad nesprávny, očakávame, že v približne polovici prípadov bude nulovej vstupnej diferencií odpovedať nenulová výstupná diferencia. Čiže v priemere po 2^{t+1} výstupných bitoch sme schopní vylúčiť zlý odhad.

2.1.2 Zostrojenie sústavy lineárnych rovníc

Po identifikovaní O(t) chýb potrebujeme zostrojiť sústavu lineárnych rovníc nad GF(2). Zadefinujme si lineárne štruktúry nultého a prvého rádu.

Definícia 1

Lineárna štruktúra nultého rádu n-bitovej funkcie j je n-bitový vektor γ taký, že pre všetky X platí: $f(X) = f(X \oplus \gamma)$.

Definícia 2

Lineárna štruktúra prvého rádu n-bitovej funkcie j je n-bitový vektor γ taký, že pre všetky X platí: $f(X) = f(X \oplus \gamma) \oplus 1$.

Pre každú funkciu f platí, že nulový vektor je triviálna lineárna štruktúra nultého rádu. Sústreďme sa na jednotlivé výstupné bity. Pre každý chybový prúdový kľúč je útočník schopný sledovať výstupnú diferenciu. Taktiež je na základe známej indukovanej chyby schopný určiť vstupnú diferenciu do f. Ak máme viacero chybových výstupných prúdov bitov, sme schopní pozorovať pre každý výstupný bit jednotlivé páry vstupno-výstupných diferencií.

Za predpokladu, že f neobsahuje netriviálne lineárne štruktúry platí, že pre každú vstupnú diferenciu v priemere polovica možných vstupov do funkcie f zodpovedá príslušnej výstupnej diferencií (t.j. funkcia f vracia týmto vstupom rovnakú hodnotu, ako vracia v prípade nám neznámeho pôvodného vstupu so známou vstupnou diferenciou). Čiže každá chyba zredukuje počet možných vstupov do funkcie f v i-tom takte o polovicu. Preto, ak máme daných t párov (a viac) vstupno-výstupnej diferencie

pre i-ty výstupný bit, sme schopní exhaustívnym prehľadávaním možností jednoznačne určiť konkrétnu t-ticu bitov, ktorá vstupovala do funkcie f v takte i.

Teraz môžu nastať dva prípady: alebo sme rovno určili bit(y) počiatočného naplnenia, alebo sme určili bit(y), ktorý(é) nebol(i) súčasťou počiatočného naplnenia. V prvom prípade sme teda priamo získali hľadané bity. V druhom prípade vieme zostrojiť lineárnu rovnicu nad GF(2) nad pôvodným naplnením registra pomocou jeho príslušnej lineárnej diferenčnej rovnice. Tento postup opakujeme, kým nenazbierame $\Theta(n)$ rovníc. Lineárne štruktúry nultého a prvého rádu si vieme predvypočítať pomocou autokorelačnej funkcie.

Definícia 3

Autokorelačná funkcia funkcie f je definovaná:

$$K_f(\gamma) = \frac{1}{2^t} \sum_{x \in \{0,1\}^t} (-1)^{f(x)+f(x+\gamma)} \quad (1)$$

Lemma 1

Ak $g = f(x \oplus c) \oplus d$ pre fixné $c \in \{0, 1\}^t$ a $d \in \{0, 1\}$, potom $Kf(\gamma) = Kg(\gamma)$.

Všimnime si, že $Kf(\gamma) = 1$ práve vtedy, keď γ je lineárna štruktúra nultého rádu funkcie f. Obdobne, $Kf(\gamma) = -1$ práve vtedy, ak γ je lineárna štruktúra nultého rádu funkcie f.

2.1.3 Neznáma filtrovacia funkcia

Doteraz sme predpokladali, že poznáme predpis nelineárnej filtrovacej funkcie f. Avšak, aplikovať chybový útok na LFSR dokážeme aj v prípade, že tento predpis nepoznáme. Je dobré si uvedomiť, že pri určovaní správnosti nášho odhadu indukovanej chyby v algoritme 2 nepotrebujeme poznať predpis f. Takisto v algoritme 1 vieme vykonať kroky 1 – 4 bez znalosti f.

Definícia 4

Nech D(i) je množina vstupno-výstupných diferenčných chybových párov prislúchajúcich pozícií i vo výstupnom chybovom prúdovom kľúči. $D_x(i)$ je výstupná diferencia na i-tej pozícií zodpovedajúca vstupnej diferencii x.

Ak platí pre nejakú pozíciu i vo výstupnom prúde bitov, že $|D(i)| = 2^t$, tak dokážeme vypočítať lineárne štruktúry nultého a prvého rádu f. Majme funkciu g takú, že $g(x) = D_x(i)$ a nech c je bezchybový vstup do funkcie f v čase i (v i-tom takte). Potom platí: $g(x) = f(x \oplus c) \oplus f(c)$. Podľa lemmy 1 platí, že autokorelačná funkcia funkcie g nadobúda rovnaké hodnoty ako autokorelačná funkcia funkcie f. Preto vypočítaním autokorelačnej funkcie g dokážeme zistiť lineárne štruktúry nultého a prvého rádu f.

Platí, že ak pre dve pozície i, j $D(i) = D(j)$ a $|D(i)| = 2^t$, tak môžu nastať 3 situácie: alebo sú pôvodné vstupy funkcie X, Y rovnaké, alebo platí, že $X \oplus Y$ je lineárna štruktúra nultého rádu f, alebo platí, že $X \oplus Y$ je lineárna štruktúra prvého rádu. V prvých dvoch prípadoch dokážeme zostrojiť sústavu lineárnych rovníc nad GF(2) ako v prípade, že poznáme filtrovaciu funkciu. V treťom prípade dokážeme situáciu posúdiť na základe

bezchybového výstupu funkcie f. Aby platilo, že X = Y, musí platiť, že aj príslušné bity i, j v bezchybovom výstupe f sa rovnajú. V prípade, že sa nerovnajú (t.j. pre príslušné výstupné bity f_i, f_j platí $f_i \oplus f_j = 1$) platí, že X ⊕ Y je lineárna štruktúra prvého rádu funkcie f.

Na zabezpečenie toho, aby $|D(i)| = 2^t$, je potreba indukovať $O(t2^t)$ chýb. Na posúdenie vzťahu vstupov X, Y nám stačí, aby prienik množín D(i), D(j) bol dostatočne veľký (aspoň t).

2.2 Chybový útok na prúdovú šifru s časovým riadením

Základná konštrukcia prúdovej šifry s časovým riadením pozostáva z dvoch komponentov: časového LFSR a dátového LFSR. Výstup šifry je podpostupnosť výstupu dátového LFSR, ktorá je určená výstupom časového LFSR. Napríklad, tzv. „one-step/two-step“ generátor generuje výstup tak, že v prípade, že je výstup časového LFSR nulový bit (0), dátový LFSR sa taktne jeden krát a v prípade, že je výstup časového LFSR jednotkový bit (1), dátový LFSR sa taktne dvakrát.

Ďalšou variantou je tzv. stop-and-go generátor, kde v prípade, že výstup časového LFSR je jednotkový bit, dôjde k taktovaniu dátového LFSR a jeho výstup tvorí ďalší bit prúdového kľúča a v prípade, že výstup časového LFSR je nulový bit, zopakuje sa posledný výstup dátového LFSR a ten tvorí ďalší bit prúdového kľúča (samotný dátový LFSR sa netaktuje). Inou variantou je možnosť, že taktovanie dátového LFSR ovplyvňuje viac ako jeden bit časového LFSR, napríklad v prípade prúdovej šifry LILI-128 ovplyvňujú 2 bity časového LFSR taktovanie dátového LFSR a spôsobujú jeho posun o 1 až 4 takty.

2.2.1 Chybový útok na one-step/two-step generátor

Jedným z možných typov chybových útokov na časovo riadené generátory je tzv. „phase-shift attack“, čo je útok, pri ktorom dochádza k posunu jedného komponentu o jeden takt (prípadne viac taktov), zatiaľ čo druhý komponent sa neposunie. Jedná sa napríklad o posun dátového registra o jeden takt pred samotným šifrovaním, čo nám umožní získať informácie o bitoch časového LFSR. V prípade one-step/two-step generátora môžeme popísať útok nasledovným spôsobom:

Algoritmus 3 Fázový útok na one-step/two-step generátor

1. Vygeneruj bezchybový prúdový kľúč
2. Spôsob fázový posun dátového LFSR o jeden takt a vygeneruj príslušný chybový prúdový kľúč
3. Nájdi bit bezchybového prúdového kľúča s pozíciou i, pre ktorý platí, že sa nerovná bitu na pozícií i-1 v chybovom prúdovom kľúči, z čoho vyplýva, že v i-tom takte generátora došlo k posunu dátového registra o 2 takty (t.j. i-ty bit v časovom LFSR mal hodnotu 1).
4. Opakuj 3. krok kým nenazbieraš dostatočný počet lineárnych rovníc nad GF(2) nad pôvodným naplnením časového LFSR
5. Zo známeho naplnenia časového LFSR a prúdového kľúča urči počiatočné naplnenie dátového LFSR.

2.2.2 Chybový útok na stop-and-go generátor

V prípade útoku na stop-and-go generátor postupujeme podobne, ako v prípade one-step/two-step generátora:

Algoritmus 4 Fázový útok na stop-and-go generátor

1. Vygeneruj bezchybový prúdový kľúč
2. Spôsob fázový posun dátového LFSR o jeden takt a vygeneruj príslušný chybový prúdový kľúč
3. Nájdi bit bezchybového prúdového kľúča s pozíciou i, pre ktorý platí, že sa nerovná bitu na pozícií i-1 v chybovom prúdovom kľúči, z čoho vyplýva, že v i-tom takte generátora došlo k zopakovaniu predchádzajúceho výstupu dátového registra (t.j. i- ty bit v časovom LFSR mal hodnotu 0.
4. V prípade, že dôjde k situácií, že i-ty bit bezchybového kľúča a (i–1)-ty bit chybového kľúča sú rôzne, avšak (i+1)-ty bit bezchybového kľúča a i-ty bit chybového kľúča sú rovnaké, v časovom registri musí byť na pozícií i+1 jednotkový bit.
5. Opakuj kroky 3., 4. kým nenazbieraš dostatočný počet lineárnych rovníc nad GF(2) nad pôvodným naplnením časového LFSR
6. Zo známeho naplnenia časového LFSR a prúdového kľúča urči počiatočné naplnenie dátového LFSR.

3. Chybové útoky na prúdovú šifru LILI-128

3.1 Prúdová šifra LILI-128 [1]

Prúdová šifra LILI-128 [1] bola jedným z kandidátov v projekte NESSIE (neúspešným). Jedná sa o synchrónnu, časovo riadenú prúdovú šifru s nelineárnou filtrovacou funkciou, ktorej kľúč má 128 bitov. Skladá sa z dvoch komponentov, 39-bitového časového lineárneho spätnoväzobného registra $LFSR_C$ a 89-bitového dátového lineárneho spätnoväzobného registra $LFSR_D$. Postup generovania prúdového kľúča je nasledovný:

1. Na množinu 10 bitov dátového registra $LFSR_D$ sa aplikuje nelineárna filtrovacia funkcia, jej výstup tvorí bit prúdového kľúča.
2. Časový register $LFSR_C$ sa taktne jeden krát. Podľa jeho dvoch bitov sa určí číslo c z množiny {1, 2, 3, 4}.
3. Dátový register $LFSR_D$ sa taktne c krát.

Inicializácia registrov sa robí rozdelením bitov kľúča. Prvých 39 bitov sa použije ako počiatočné naplnenie časového registra, zvyšných 89 registrov sa použije ako počiatočné naplnenie dátového registra. Nulové naplnenia sa neberú do úvahy.

3.2 Útok na LILI-128 (Hoch, Shamir) [2]

Prvou fázou útoku je indukcia jednobitových chýb na náhodných miestach a vyprodukovanie príslušného chybového prúdového kľúča. Následne sa zariadenie „zresetuje“ a postup sa zopakuje, kým nezískame 89 rôznych prúdových kľúčov, čo zodpovedá indukcií jednobitovej chyby v každom bite registra. Toto zopakujeme, avšak pred indukciou chyby posunieme dátový register o jeden takt. Pozorujeme, že množiny prúdových kľúčov obsahujú niekoľko rovnakých prúdových kľúčov.

Je to spôsobené tým, že v prípade, že indukovaná chyba nebola indukovaná na mieste, ktoré ovplyvňuje nový bit (podľa diferenčnej rovnice), je jedno, či chybu indukujeme na mieste i a potom zariadenie posunieme o jeden takt, alebo ho najprv posunieme o jeden takt a následne indukujeme chybu na i–1 mieste. Spočítaním, koľko prúdových kľúčov sa v daných množinách zhoduje, vieme zistiť, či bol register $LFSR_D$ taktovaný o 1, 2, 3 alebo 4 takty, čím získame 2 bity pôvodného naplnenia registra $LFSR_C$, resp. získame 2 lineárne rovnice nad GF(2) nad pôvodným naplnením registra $LFSR_C$.

Čiže, po zhruba 20 opakovaniach (indukcií chýb po zhruba 20 rôznych fázových posunoch dátového registra) sme schopní vypočítať pôvodné počiatočné naplnenie časového registra. Po určení počiatočného naplnenia použijeme algoritmus 1 na nájdenie počiatočného naplnenia dátového registra, pričom použijeme už vygenerované chybové prúdové kľúče. Algoritmicky zapísaný útok:

Algoritmus 5 Chybový útok na LILI-128 (Hoch, Shamir) [2]

1. Vygeneruj bezchybový prúdový kľúč
2. Vygeneruj 89 rôznych chybových prúdových kľúčov prislúchajúcich jednobitovým indukovaným chybám
3. Vyhodnoť prúdové kľúče na zistenie bitov $LFSR_C$
4. Opakuj kroky 2., 3. pri rôznych fázových posunoch dátového registra, kým nezískaš 39 lineárne nezávislých rovníc nad GF(2) nad pôvodným naplnením $LFSR_C$.
5. Pomocou známeho naplnenia časového registra použitím algoritmu 1 zisti počiatočné naplnenie dátového registra $LFSR_D$.

3.3 Útok na LILI-128 (Hromada)

Náš útok sa líši od útoku popísaného v časti 3.2 tým, že na zistenie naplnenia časového registra nepožaduje indukovanie chýb v dátovom registri. Miesto toho sa pri hľadaní naplnenia časového registra využije útok fázovým posunom. Vygenerujeme bezchybový prúdový kľúč, „zresetujeme“ zariadenie, posunieme dátový register o jeden takt, opäť vygenerujeme prúdový kľúč. Toto zopakujeme s posunom o dva, tri a štyri takty.

Porovnávaním bezchybového prúdového kľúča a chybových prúdových kľúčov dokážeme zistiť, o akú hodnotu bol taktovaný dátový register. Ak totiž platí, že sa nezhoduje i-ty bit bezchybového prúdového kľúča a (i–1)-ty bit prúdového kľúča zodpovedajúceho dátovému registru posunutého o 1 takt, musí platiť, že v danom mieste muselo prísť ku posunu o 2, 3 alebo 4 takty. Preto porovnáme, či sa zhoduje i-ty bit bezchybového prúdového kľúča a (i–1)-ty bit prúdového kľúča zodpovedajúceho dátovému registru posunutého o 2 takty, a ak nie, znamená to, že muselo dôjsť k posunu o 3 alebo 4 takty.

Preto porovnáme, či sa zhoduje i¬-ty bit bezchybového prúdového kľúča a (i–1)-ty bit prúdového kľúča zodpovedajúceho dátovému registru posunutého o 3 takty a ak nie, vieme, že muselo dôjsť ku posunu o 4 takty. Analogicky dokážeme určiť, či došlo k posunu o 1, 2 alebo 3 takty. Tak zistíme príslušné bity v časovom registri.

Po zistení počiatočného naplnenia časového registra postupujeme pri hľadaní počiatočného naplnenia dátového registra pomocou indukcie 89 jednobitových chýb v

dátovom registri (bez počiatočného fázového posunu). Pomocou algoritmu 1 potom zistíme počiatočné naplnenie dátového registra. Algoritmicky zapísaný útok:

Algoritmus 6 Chybový útok na LILI-128 (Hromada)

1. Vygeneruj bezchybový prúdový kľúč
2. Vygeneruj 4 chybové prúdové kľúče, ktoré zodpovedajú fázovým posunom dátového registra o 1, 2, 3, 4 takty.
3. Analyzuj chybové prúdové kľúče na zistenie bitov časového registra a získanie 39 lineárne nezávislých rovníc nad GF(2) nad počiatočným naplnením časového registra.
4. Pomocou známeho naplnenia časového registra indukovaním 89 chýb v dátovom registri a použitím algoritmu 1 zisti počiatočné naplnenie dátového registra $LFSR_D$.

4. Porovnanie útokov na LILI-128

Zamerali sme sa na porovnanie prezentovaných útokov na prúdovú šifru LILI-128, resp. na porovnanie počtu potrebných indukovaných chýb a na porovnanie počtu bitov prúdového kľúča, potrebných na jednoznačné zistenie pôvodného naplnenia oboch registrov (t.j. na zistenie pôvodného kľúča). Očakávame, že v prípade nášho útoku bude nižšia hodnota potrebných indukovaných chýb, avšak bude vyššia hodnota potrebného počtu bitov prúdového kľúča na úspešný útok.

Z tabuľky 1 vidíme, že naše očakávania sa naplnili, nakoľko v prípade nášho útoku (označeného ako (Hr)) narástol počet potrebných bitov troj- až päť-násobne. Počet potrebných indukovaných chýb bol pri chybovom útoku autorov Hocha a Shamira vo všetkých prípadoch 31-krát vyšší ako pri našom útoku (čo znamená, že zatiaľ čo na náš útok je potrebných 89 indukovaných chýb, na útok Hocha a Shamira je potreba 2759 chýb.

Tab. 1 Výsledky chybových útokov na LILI-128

Kľúč	Počet bitov (Ho)	Počet posunov (Ho)	Počet bitov (Hr)
AAAAAAAAAAAAAAAA	46	31	160
aaaaaaaaaaaaaaaa	46	31	187
ABCDEFGHIJKLMNOP	40	31	157
abcdefghijklmnop	43	31	141
123ABC456DEF789G	46	31	146
123abc456def789g	50	31	150
#&@VILKO8789HROM	44	31	174
#&@vilko8789hrom	47	31	111
NBUSR1234567890!	46	31	207
nbusr1234567890!	43	31	129

5. Záver

V článku sme predstavili koncept chybovej analýzy a vybrané techniky chybových útokov na najčastejšie konštrukcie prúdových šifier. Implementovali a popísali sme dva

útoky na prúdovú šifru LILI-128, ktorej konštrukcia je vhodná na demonštráciu týchto útokov, nakoľko v sebe kombinuje ako nelineárne filtrovaný lineárny spätnoväzobný register, tak aj časovo riadený lineárny spätnoväzobný register.

Z vykonaných experimentov vyplýva, že zatiaľ čo útoky, pri ktorých je možné indukovať vyšší počet chýb, vyžadujú menší počet bitov prúdového kľúča, v prípade, ak by bol počet možných indukovaných chýb menší, dá sa tento nedostatok nahradiť pomocou útoku fázovým posunom, ktorému postačuje menší počet indukovaných chýb, avšak je potrebná možnosť vykonať útok fázovým posunom a navyše je aj potreba možnosti generovania dlhšieho prúdového kľúča.

Zoznam použitej literatúry

1. Dawson, E., et. al., „The LILI-128 Keystream Generator“, Dostupné z https://www.cosic.esat.kuleuven.be/nessie/workshop/submissions.html
2. Hoch, J., Shamir, A., „Fault Analysis of Stream Ciphers“, Chryptographic Hardware and Embedded Systems – CHES 2004, Lecture Notes in Computer Science, 2004, p.240 – 253

Spoluautorom článku je Ing. Milan Vojvoda, PhD., Katedra aplikovanej informatiky a výpočtovej techniky, Fakulta elektrotechniky a informatiky Slovenská Technická Univerzita Ilkovičova 3, Bratislava 812 19

Práca bola prezentovaná na Študentskej vedeckej a odbornej činnosti (ŠVOČ 2011) v sekcii Aplikovaná informatika a získala Cenu dekana, ISBN 978-80-227-3508-7