UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

KRYPTOANALÝZA ŠIFIER V MOBILNÝCH SIEŤACH

Diplomová práca

2018 **Bc. Ján Kotrady**

UNIVERZITA PAVLA JOZEFA ŠAFÁRIKA V KOŠICIACH
PRÍRODOVEDECKÁ FAKULTA

KRYPTOANALÝZA ŠIFIER V MOBILNÝCH SIEŤACH

Diplomová práca

Študijný program:	Informatika
Študijný odbor:	9.2.1. Informatika/9.2.9. Aplikovaná informatika
Školiace pracovisko:	Ústav informatiky
Vedúci práce:	RNDr. Rastislav Krivoš-Belluš, PhD.

Košice 2018 **Bc. Ján Kotrady**

Univerzita P. J. Šafárika v Košiciach
Prírodovedecká fakulta

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Ján Kotrady
Študijný program: Informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: Diplomová práca
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Kryptoanalýza šifier v mobilných sieťach

Názov EN: Cryptanalysis in Mobile Networks

Cieľ:
1. Analyzovať a porovnať publikované útoky na šifry používané v mobilných sieťach.
2. Preskúmať praktické implementácie analyzovaných útokov.
3. Navrhnúť nové metódy kryptoanalýzy týchto šifier.

Literatúra:
E. Biham, O, Dunkelman: Techniques for Cryptanalysis of Block Ciphers (Information Security and Cryptography), Springer, 2017, ISBN 978-3642172311
W. Stallings: Cryptography and Network Security: Principles and Practice, 7th edition, Pearson, 2016, ISBN 978-0134444284
A. G. H. Naim: Cryptanalysis of Some Block Ciphers, PhD. thesis, University of London, 2014

Kľúčové slová: kryptoanalýza, mobilné siete, šifry

Vedúci: RNDr. Rastislav Krivoš-Belluš, PhD.
Oponent: doc. RNDr. Jozef Jirásek, PhD.
Ústav : ÚINF - Ústav informatiky
Riaditeľ ústavu: prof. RNDr. Viliam Geffert, DrSc.

Dátum schválenia: 23.04.2018

Poďakovanie

Ďakujem vedúcemu svojej práce, RNDr. Rastislavi Krivoš-Bellušovi, PhD. za cenné pripomienky, podporu a odbornú konzultáciu počas prípravy práce, ktoré mi boli veľkou pomocou.

Abstrakt

V tejto záverečnej práci sa zaoberáme možnosťami kryptoanalýzy šifier, ktoré sa používajú v mobilných sieťach. Cieľom práce je analyzovať súčasné metódy kryptoanalýzy a navrhnúť nové metódy kryptoanalýzy. V práci sme popísali možnosti kryptoanalýzy, ktoré sme následne vylepšili tak, že je možné previesť kryptoanalýzu v praktickom čase aj vzhľadom na ochranu, ktorú jednotliví mobilní operátori implementujú. Taktiež sme popísali prístupy ku generovaniu nových typov dúhových tabuliek, ktoré sa môžu využívať aj v iných prípadoch a sú všeobecného charakteru. Analyzovali sme možnosti využitia týchto dúhových tabuliek a ich časové zložitosti, pričom tieto dúhové tabuľky sú v princípe takého charakteru, že je možné ich použiť na reálnu kryptoanalýzu v praxi. Okrem toho sa venujeme z časti aj diferenčnej kryptoanalýze, kde sme popísali možnosti pracovania s diferenciami v rámci diferenčnej kryptoanalýzy, aplikovanej na šifru KASUMI.

Kľúčové slová: *kryptoanalýza, mobilné siete, kasumi, prúdové šifry.*

Abstract

In this thesis, we deal with cryptanalysis cipher capabilities that are used in mobile networks. The aim of the thesis is to analyze current methods of cryptanalysis and to propose new methods. We have described the possibilities of cryptanalysis, which we have subsequently improved in such a way that cryptanalysis can be transferred in the practical time and in view of the protection that individual mobile operators implement. We have described approaches to generating new types of rainbow tables, which can also be used in other cases and are of a general nature. We have analyzed the possibilities of using these rainbow tables and their time complexity, and these rainbow tables are in principle of such a nature that they can be used for real cryptanalysis in practice. In addition, we also deal with the differential cryptanalysis, where we have described the possibilities of working with differentials within the differential cryptanalysis applied on the KASUMI cipher.

Keywords: *cryptoanalysis, mobile network, kasumi, stream cipher.*

Obsah

Zoznam ilustrácií a tabuliek **8**

Úvod **9**

1 Základné kryptografické prvky používané v mobilných sieťach **11**

1.1 Základné prvky a popis mobilných sietí 11

1.1.1 Protokol v mobilných sieťach 2G 13

1.1.2 Protokol v móde 3G a 4G 15

1.2 Šifrovacie algoritmy používane v sieti 2G 18

1.2.1 A5/1 19

1.2.2 A5/2 20

1.3 Šifrovacie algoritmy používane v sieti 3G a 4G 21

1.3.1 KASUMI 22

1.3.2 AES 22

1.3.3 Snow3G 23

1.4 Frame number 24

1.5 Nezabezpečená komunikačná linka operátora 25

2 Kryptoanalýza šifry A5/1 **26**

2.1 Myšlienka útoku 26

2.2 Dúhové tabuľky 27

2.2.1 Všeobecný popis dúhových tabuliek 27

2.2.2 Zavedenie význačných bodov 29

2.2.3 Využitie dúhových tabuliek 30

2.3 Dúhové tabuľky pre šifru A5/1 31

2.3.1 Formálna definícia a časová zložitosť 31

2.3.2 Generovanie tabuliek pre šifru A5/1 32

2.3.3 Vlastnosť dúhovej tabuľky 33

2.4 Spätné točenie šifry A5/1 34

2.5 Praktická implementácia 36
2.6 Zverejnené dúhové tabuľky 37
2.7 Postup útoku 38
2.7.1 Zachytenie mobilnej komunikácie 39

3 Kryptoanalýza šifry KASUMI 41
3.1 Diferečná kryptoanalýza 41
3.1.1 Diferenčná charakteristika 42
3.1.2 Filtrovanie 43
3.1.3 Použitie správnej diferencie pre nájdenie kľúča 43
3.2 Boomerang attack 44
3.2.1 Amplified Boomerang attack 45
3.3 Rectangle attack 46
3.3.1 Získanie kľúča 47
3.4 Related key Rectangle Boomerang attack 49
3.5 Related key Rectangle attack aplikovaný na šifru KASUMI 50
3.5.1 Diferencia pre prvé 4 kola 51
3.5.2 Diferencia pre 5 až 7 kolo 51
3.5.3 Útok 51
3.5.4 Analýza útoku 53
3.6 Related key Sandwich attack 54

4 Návrh kryptoanalytických metód a vylepšenia kryptografických systémov 56
4.1 Vlastnosti šifry A5/1 a útokov na šifru A5/1 56
4.2 Vnútorné stavy podľa frame number 60
4.2.1 Algoritmus a analýza útoku dúhových tabuliek pre fixný frame number 62
4.3 Rozdelenie výstupu dúhových tabuliek 64
4.3.1 Algoritmus a analýza útoku dúhových tabuliek pre rozdelený výstup 65
4.4 Vylepšenie filtrovania diferenčnej kryptoanalýzy šifry KASUMI 68

Záver 71

Zoznam použitej literatúry 76

Prílohy 77

Zoznam ilustrácií a tabuliek

Obrázok 1: Schéma siete používanej v mobilnej komunikácií. 13
Obrázok 2: Protokol používany v sieťach 2G. 14
Obrázok 3: Protokol používany v sieťach 3G. 17
Obrázok 4: Útok na siete 3G a 4G. 18
Obrázok 5: Schéma šifry A5/1. 20
Obrázok 6: Schéma šifry A5/2. 21
Obrázok 7: Príklad frame number. 24
Obrázok 8: Pomery stavov pri spätnom točení. 36
Obrázok 9: Schéma dúhových tabuliek. 37
Obrázok 10: Posuvné okno. 38
Obrázok 11: Schéma Boomerang attack. 45
Obrázok 12: Schéma Amplified Boomerang attack. 46
Obrázok 13: Schéma Related key Rectangle attack. 50
Obrázok 14: Možné hodnoty kľúčov. 53
Obrázok 15: Porovnanie Related key Boomerang attack a Sandwich attack. 54
Obrázok 16: Príklad uvažovania výstupu zo šifry A5/1. 58
Obrázok 17: Schéma pôvodného útoku. 58
Obrázok 18: Schéma dúhovej tabuľky pre dva koncové stavy. 61
Obrázok 19: Schéma výpočtu vzdialeností. 62
Obrázok 20: Spájanie výstupných bodov. 63
Obrázok 21: Príklad generovania tabuliek. 65
Obrázok 22: Označenie časti funkcie FL. 69

Tabuľka 1: Tabuľka funkcií používaných v štandarde 3G a 4G. 17
Tabuľka 2: Počet predchádzajúcich stavov pre spätné točenie. 34
Tabuľka 3: Podmienená distribúcia počtu riešení. 34
Tabuľka 4: Prehľad značenia. 49

Úvod

S rýchlo rastúcou technológiou prichádza aj rozsiahly vývoj v oblasti mobilných sietí. Či už je to vývoj rôznych protokolov, technológií a komunikačných zariadení, prichádza aj k vývoju rôznych kryptografických primitív. Mobilné siete a mobilné telefóny prešli za posledné desaťročie odlišným vývojom ako na úrovni hardvérovej, tak aj softvérovej a aplikačnej. Pre spätnú kompatibilitu sú dodnes v mobilných sieťach použité zastaralé algoritmy, používané na zabezpečenie dôvernosti, ktoré už v súčasnej dobe predstavujú bezpečnostnú hrozbu. Naopak, s tým ako prichádzal vývoj nových zariadený a hardvér sa stal rýchlejší, môžeme pozorovať aj používanie bezpečnejších kryptografických primitív, ktoré ale pre zachovanie spätnej kompatibility sú častokrát vynechávané.

V našej práci budeme skúmať dve základne kryptografické šifry, a to šifru A5/1 a šifru KASUMI. V rámci práce sa pozrieme na praktickú kryptoanalýzu, ktorú je možné aplikovať v súčasnej mobilnej infraštruktúre a taktiež sa budeme zaoberať aj novými útokmi, ktoré je možné na tieto šifry aplikovať. Okrem kryptoanalýzy šifier sa budeme z časti venovať aj praktickým útokom na mobilné siete a pozrieme sa aj na protokolové zabezpečenie v týchto sieťach.

Práca je rozdelená na štyri kapitoly, kde v prvej kapitole popisujeme princípy a základné technológie použité v mobilných sieťach. Špecifikujeme štandardy a základné vlastnosti týchto sietí, a taktiež sa venujeme aj protokolovému zabezpečeniu. Vzhľadom na to, že problém mobilných sietí je obsiahli, vybrali sme tie špecifiká, ktoré sú pre nás v tejto práci najdôležitejšie.

V druhej kapitole sa venujeme aktuálnym možnostiam kryptoanalýzy šifry A5/1, jej vlastnostiam a možnostiam generovania dúhových tabuliek. Popisujeme, ako fungujú aktuálne útoky a ako je možné ich využívať v praxi.

V tretej kapitole sa zameriame na šifru KASUMI, kde špecifikujeme možnosti kryptoanalýzy tejto šifry, popisujeme aktuálne útoky a prístupy ku kryptoanalýze tejto šifry. Vzhľadom na charakter používania tejto šifry nie je v súčasnosti možný útok na túto šifru tak, ako je používaná v mobilných sieťach. Okrem toho sa venujeme aj diferenčnej kryptoanalýze, ktorú veľmi z voľna popisujeme. Pre bližšie špecifikovanie

diferenčnej kryptoanalýzy nie je priestor v tejto práci.

V poslednej kapitole navrhujeme nové metódy, ktoré sme uviedli v teoretickej rovine, ktoré je možné použiť aj v praxi na kryptoanalýzu šifry A5/1. Nakoniec popisujeme možnosti pracovania s diferenciami pri kryptoanalýze šifry KASUMI, čím sme zlepšili časovú zložitosť.

Súčasťou práce je aj príloha vo forme CD, na ktorom je zdrojový kód, ktorý je možné použiť pri generovaný dúhových tabuliek. V prílohách prikladáme špecifikáciu šifry KASUMI vo forme schémy a špecifikáciu šifry Snow3G, taktiež vo forme schémy.

1 Základné kryptografické prvky používané v mobilných sieťach

Predmetom tejto kapitoly je zhrnutie základných kryptografických primitív, používaných v mobilných sieťach, približný popis útoku na tieto kryptografické primitíva, vlastnosti tých sietí a spôsob autentifikácie užívateľa. V rámci tejto kapitoly popíšeme, ako je možné útočiť na mobilné siete s dôrazom na narušenie dôvernosti. Venovať sa budeme dvom základným typom, druhom mobilných sietí, a to konkrétne sieť GSM, ktorá je prvá používaná mobilná sieť a jej nasledovníka sieť s označením 3G (3GPP). Vlastnostiam a možnostiam útoku na sieť 4G sa v tejto práci venovať nebudeme, a to z dôvodu jej nerelevantnosti [12]. Totižto, mobilné siete 4G a vyššie (5G, LTE) používajú AES kryptografický štandard, na ktorý dodnes neexistuje efektívny útok (pričom táto šifra je skúmaná už niekoľko rokov) a využívajú kryptografické protokoly, ktoré sú doposiaľ považované za bezpečné [35].

1.1 Základné prvky a popis mobilných sietí

Preto, aby sme pochopili, aké mobilné siete existujú a aké útoky sú možné v mobilných sieťach, potrebujeme popísať, ako vlastne funguje mobilná sieť a ako prebieha autentifikácia v tejto sieti a taktiež sa pozrieme na metódy dohody kľúča, čo neskôr využijeme pri popise jednotlivých útokov na tieto siete. V našej práci pre jednoduchosť budeme rozlišovať dva základné druhy mobilných sietí, a to sieť s názvom 2G a sieť s názvom 3G. Podľa rôznych zdrojov sa tieto siete označujú rôzne, a to napríklad 3GPP, LTE, GSM atď. V našom prípade, ale použijeme univerzálne označenie 2G pre siete typu GSM, pričom tento druh je prvým mobilným sieťovým systémom, ktorého vznik sa datuje už na 90. roky. Následný nový štandard 3G, niekde označovaný aj 3GPP (3GPP je označenie pre projekt, z ktorého vznikla sieť typu 3G), alebo označovaný aj ako UMTS. Novými štandardami (druhmi siete) ako 4G a 5G sa zaoberať ďalej nebudeme.

V praxi sa často stretávame s tým, že aj keď siete typu 3G až 5G sú vhodné

ako pre komunikáciu (hlasový hovor, sms), tak aj pre internetovú komunikáciu (IP protokol), nie sú používane pre hlasovú komunikáciu z dôvodu, aby prenosové rýchlosti boli dostatočné. Tento dôležitý poznatok využijeme neskôr pri kryptoanalýze šifier - ak sa stále využíva sieť typu 2G, tak je postačujúce vedieť dostatočne dobre útočiť na tento druh siete [13, 35].

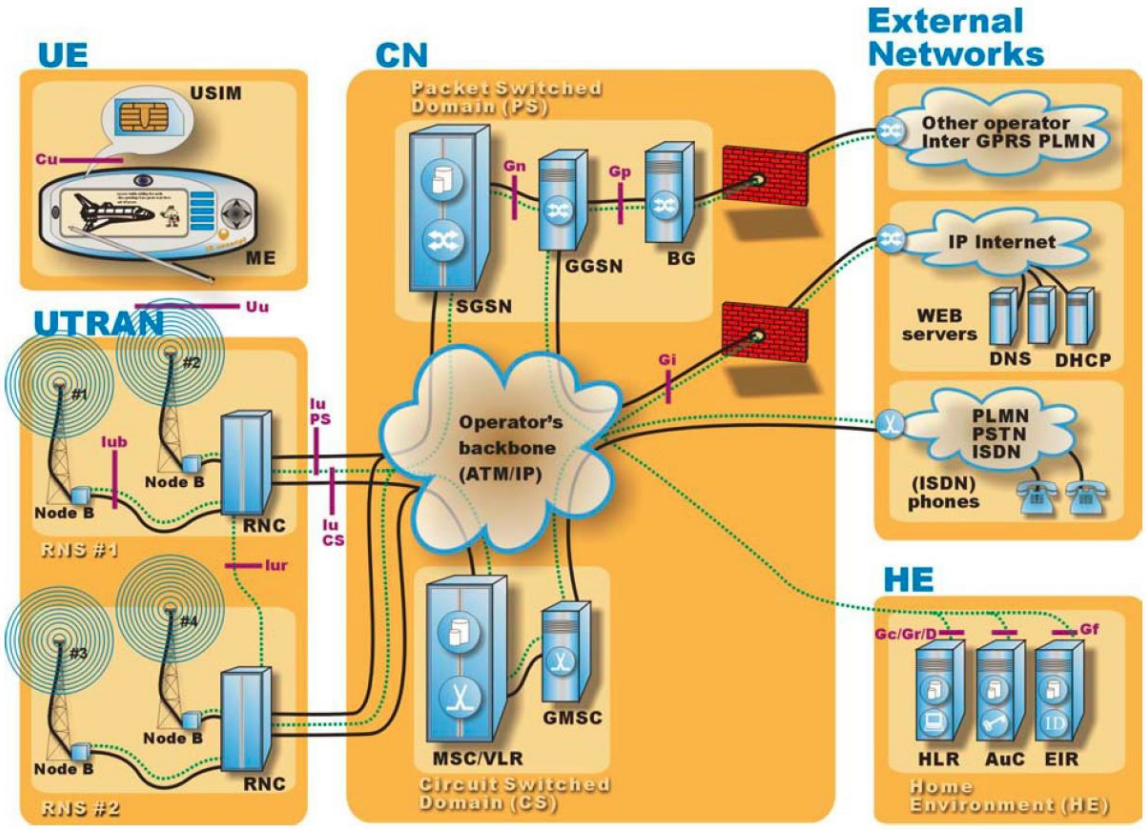
Mobilná sieť pozostáva z niekoľkých častí, no pre nás tie najdôležitejšie sú [13]:

1. BTS – Ide o prvú stanicu, s ktorou je mobilné zariadenie pripojené a s ktorou priamo komunikuje. Na obrázku 1 v našom prípade ide o komplex zariadení Node B, niekde do daného pojmu spadajú aj zariadenia RNC. Pre účely tejto práce budeme postačovať s tým, že ide o zariadenie, ktoré stojí za prijatím a odosielaním správ priamo do mobilného zariadenia cez nezabezpečený kanál.

2. USIM/SIM – Každé mobilné zariadenie obsahuje takzvanú SIM kartu. V prípade, že rozprávame o USIM, ide o novší typ SIM karty, ktorý sa používa už v 3G sieťach a zabezpečuje viac funkcií ako SIM karta.

3. AuC – Ide o autentifikačné centrum operátora. Každý operátor ma svoje vlastné autentifikačné centrum, ktoré je zodpovedné za autentifikáciu, autorizáciu a integritu dát (a mnoho iných vlastností). V prípade, že je používateľ pripojený v cudzej sieti (napríklad pri ceste do zahraničia) a nastáva autentifikácia, tak zahraničná mobilná sieť je povinná autentifikovať užívateľa s domovským AuC. Takže cudzia sieť musí posielať požiadavku domovskej siete. AuC taktiež obsahuje takzvané záznamy IMSI, autentifikačné a autorizačné dáta, taktiež aj zvolené šifrovacie kľúče.

4. UE, ME – User Equipment, Mobile Equipment označuje v prípade UE komplex mobilného zariadenia so SIM, resp. USIM kartou a ME označuje komplex mobilného zariadenia (mimo SIM kariet).

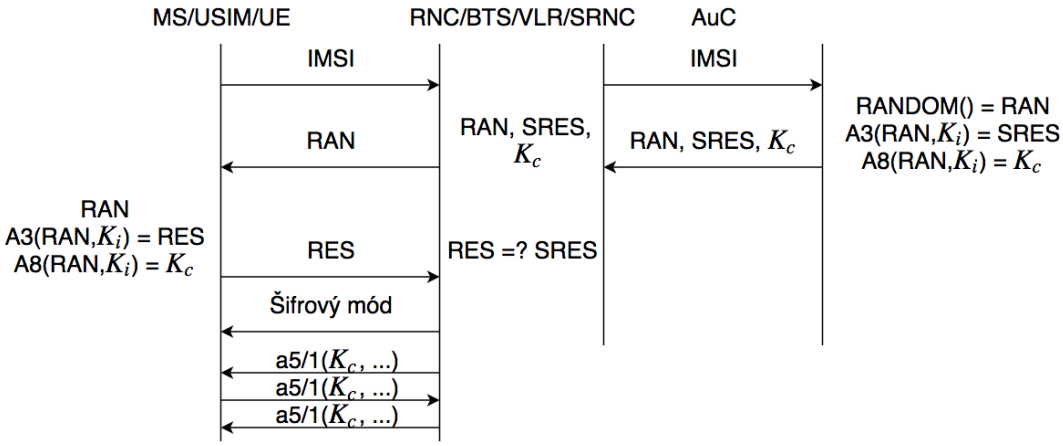


Obr. 1: Schéma siete používanej v mobilnej komunikácií [13].

1.1.1 Protokol v mobilných sieťach 2G

Mobilné zariadenie (UE) je pripojené do mobilnej siete, či už domáceho alebo cudzieho operátora. Následne sa ohlási svojím IMSI. Operátor na základe údajov, ktoré dostal od UE pošle prostredníctvom svojej vlastnej siete správu s obsahom IMSI autentifikačnému centru (AuC). Autentifikačné centrum vyberie zo zoznamu známych IMSI (tie sú uložené v autentifikačnom centre pri vyrábaní/vydávaní SIM karty, spolu s kľúčmi) zdieľaný kľúč K_i, vygeneruje náhodné číslo RAN a následne použije funkciu A3 so vstupom RAN a K_i, výstup označíme SRES. Funkciu A3 bližšie špecifikovať taktiež nebudeme, keďže nie je predmetom tejto práce. Taktiež použije funkciu A8 so vstupom K_i a náhodným číslom RAN, výstup označíme K_c. Tieto dáta (RAN, SRES a K_c) prepošle AuC späť, k BTS stanici, od ktorej predtým požiadavka prišla. Následne stanica BTS vyzve UE tým, že jej prepošle náhodnú hodnotu RAN, vygenerovanú stanicou AuC. V prípade, že ide o správneho používateľa, tak tento používateľ by mal mať znalosť K_i a ak predpokladáme, že bolo doručené správne RAN, UE (SIM/USIM) je schopné zo znalosti K_i a RAN vygenerovať hodnoty SRES a K_c použitím rovnakých funkcií A3 a A8 ako AuC. Ďalším krokom je preposlanie vygenerovaného SRES BTS

stanici. Ak bude BTS stanici doručený rovnaký SRES od UE ako bol doručený od AuC, stanica autentifikuje UE a následne prebieha šifrovaná komunikácia pomocou už dohodnutého kľúča K_c. Treba si uvedomiť, že dôležitá vlastnosť mobilných sieti je možnosť BTS stanice zvoliť si šifrovací mód. Buď bude nasledujúca komunikácia nešifrovaná, alebo sa použije šifrovací algoritmus A5/1 respektíve A5/2, v oboch prípadoch bude šifrovanie prebiehať kľúčom K_c [13, 35].



Obr. 2: Protokol používaný v sieťach 2G [13].

Ako je možné z vyššie popísaného vyvodiť, tak v sieťach typu 2G sa autentifikuje len používateľ vystupujúci ako UE voči BTS stanici, no nie naopak, čo tvorí priestor pre možné útoky. Autentifikácia vyplýva zo znalosti vygenerovať správnu odpoveď RES (respektíve SRES) a tú poslať BTS stanici, pričom BTS stanica tieto informácie dostáva od autentifikačného centra domovskej siete. Nikde sa v celom protokole nezverejňuje kľúč K_i, ktorý v mobilných sieťach vystupuje len v rámci SIM karty, kde je uložený a AuC centra. Je možné tento kľúč z danej karty exportovať iným útokom, ktorý je skôr zameraný na zraniteľnosť čipov na SIM kartách. Celý výpočet funkcií A3 a A8 je prístupný len ako funkcia, ktorú mobilné zariadene volá z danej SIM karty, preto nie je možné tento kľúč K_i získať jednoduchým útokom [1]. Pre bližšie informácie o tom, aké sú možné útoky na tajný zdieľaný kľúč K_i, odporúčam literatúru [1], ktorá sa práve zaoberá možnosťami klonovania SIM kariet. Taktiež skúmajú, ako je možné získať kľúč K_i. Tieto útoky sú skôr útoky, ktoré sa venujú elektrotechnike, a preto sa im bližšie venovať nebudeme.

Tým, že klient vygeneruje správnu odpoveď SRES, respektíve RES, BTS stanica môže považovať, že používateľ je autentifikovaný v rámci domovskej siete a sprístupňuje takzvaný roaming klientovi. Ako sme už ale spomenuli, BTS stanica sa neautentifikuje klientovi, a preto je možné túto zraniteľnosť využiť. Totižto, ak útočník vytvorí falošnú

BTS stanicu, je schopný odchytiť všetky správy, ktoré sa cieľ posiela BTS stanici, no na druhej strane bude vystupovať pre BTS stanicu ako cieľ útoku. Takto dokáže útočník odchytiť a preposielať správy, ktoré dostane ako od cieľa útoku tak aj od BTS stanice pre cieľ. Všetky správy preposiela, pokiaľ nedostane výzvu na šifrovací mód. Vtedy prepošle obom stranám správu, že sa nepoužíva žiadny šifrovací mód [31].

1.1.2 Protokol v móde 3G a 4G

Známe útoky popísané v kapitole vyššie sa snažili autori módov 3G a 4G odstrániť. Navrhli autentifikáciu aj BTS stanice voči zariadeniu UE. Celý protokol na autentifikáciu a dohodu o kľúči prebieha podobne, ako v prípade módu 2G, no AuC generuje v tomto prípade celý autentifikačný vektor, ktorý pozostáva z nasledujúcich častí [29]:

1. MAC - hash, ktorý slúži na overenie integrity správy.
2. XRES - očakávaná odpoveď.
3. C_k - kľúč, ktorý slúži na ďalšie šifrovanie dát.
4. I_k - kľúč integrity, ktorý zabezpečuje integritu posielaných správ.
5. A_k - takzvaný anonymný kľúč - kľúč, ktorý sa použije na overenie integrity autentifikačného vektora.
6. SQN - sekvenčné číslo autentifikačného vektora - USIM aj AuC počíta sekvenčné čísla autentifikačného vektora, ktoré pri každom generovaní inkrementuje. Toto sekvenčné číslo sa funkciou XOR spojí s A_k.

V podstate ide o to, že AuC vygeneruje náhodné číslo a vypočíta na základe náhodného čísla a IMSI niekoľko ďalších hodnôt. V tomto prípade vygeneruje znova C_k ako kľúč, ktorý sa použije na šifrovanie dát, očakávanú XRES (očakávanú odpoveď od UE), kľúč integrity I_k a veľmi dôležitú časť, tzv. autentifikačný token, ktorý slúži na autentifikáciu UE stanice voči BTS. V rámci protokolu v móde 3G je implementovaná aj takzvaná MAC funkcia, ktorá slúži ako ochrana integrity správy. Totižto, ak UE prijme správu s nesprávnym AUTN, nebude považovať sieť za bezpečnú. Ak by sme aj v tomto prípade použili falošnú BTS stanicu, autentifikáciu by sme vyžiadali od správneho AuC, nedostali by sme sa k dátam, ktoré sa používajú na autentifikáciu správ (MAC kód), takže v našom prípade by sme neboli schopný poslať korektnú správu UE s tým, že volíme mód šifrovania taký, že sa správa nebude šifrovať. AUTN musí byť poslané UE, pretože ak by AUTN nebolo poslané UE, mohlo by dôjsť k situácií, žeby UE bolo pripojené na falošnú BTS stanicu a boli by podvrhnuté dáta ako je RANDOM číslo,

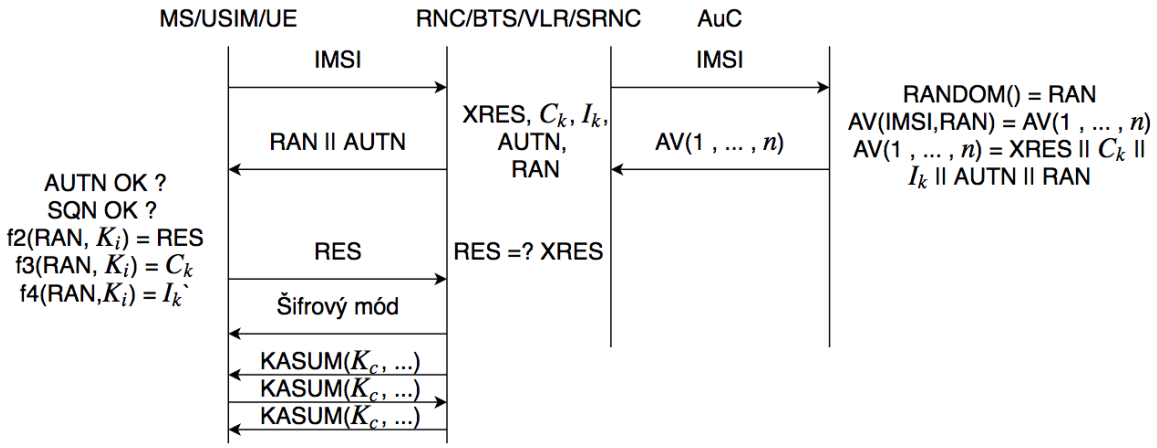
respektíve sekvenčné číslo, ktoré sa používa v autentifikačnom vektore [13]. Aj v tomto prípade existujú možnosti útokov, ktoré nie sú z implementačného hľadiska v našom prípade reálne. Ide o napríklad vytvorenie falošného mobilného operátora, ktorý by bol schopný dostávať správy od AuC. Tieto scenáre sú možné len v prípade, ak by sme povolili vytvorenie a zaregistrovanie osoby, subjektu ako mobilného operátora a potom by sme mohli využívať falošné - respektíve v tomto prípade už skutočné BTS stanice.

V rámci autentifikácie AuC vygeneruje autentifikačný vektor a podpis výzvy RAND, na čo použije funkciu f1. Tým je zaručená integrita náhodnej výzvy RAND. Totižto, UE pri prijatí autentifikačného vektora overí integritu RAND rovnakým výpočtom ako AuC. BTS stanica negeneruje žiadne dáta, nevytvára zo začiatku žiadny podpis a slúži len na preposielanie správ. Ak UE zistí, že RAND je podpísaný správne, bude sa spoliehať na spoľahlivosť operátora a bude predpokladať, že BTS stanica nebude zdieľať kľúč na podpisovanie správ, takzvaný I_k. Tým je zaručená ochrana voči podvrhnutým správam zo strany falošnej BTS stanice [13]. Okrem týchto algoritmov existujú aj algoritmy resynchronizačné, ktoré sa použijú na nové generovanie SQN čísla v prípade, že dôjde zo strany operátora k chybe. Je dôležité poznamenať, že ak dôjde k prekročeniu nejakej vopred stanovenej hranici v sekvenčných číslach SQN je nutné manuálne resynchronizovať SIM kartu.

Ako sme už spomenuli, každá správa, ktorá je posielaná medzi BTS stanicou a UE zariadením musí byť podpísaná funkciou f9, okrem úvodných správ, ktoré sú individuálne podpísané funkciami f0, respektíve f1 pri resynchronizácií. Z tabuľky 1 je možné vidieť, aký účel majú jednotlivé funkcie a kde sa nachádzajú, respektíve kto dané funkcie implementuje. Funkcie f8 a f9 sú implementované v zariadení a BTS stanici, pričom zvyšné funkcie je možné implementovať u operátora individuálne. Ak operátor zvolí chybnú implementáciu funkcií f1 až f5, je potom možné zistiť zdieľaný kľúč K. Tieto funkcie sa nenachádzajú v BTS stanici, BTS stanica potrebuje v danom implementovať iba overenie výstupu z AuC a výstupu z UE.

	Popis	Výstup	Lokácia	#Bit
f0	Náhodné čísla	RAN	AuC	128
f1	Sieťová autentifikácia	(X)MAC-A	USIM, AuC	64
f1*	Sieťová autentifikácia resyn.	(X)MAC-S	USIM, AuC	64
f2	Užívateľská autentifikácia	(X)RES	USIM, AuC	32-128
f3	Derivácia kľúča pre šifru	C_k	USIM, AuC	128
f4	Derivácia kľúča pre integritu	I_k	USIM, AuC	128
f5	Derivácia kľúča pre anonymitu	A_k	USIM, AuC	48
f5*	Derivácia kľúča pre anonymitu resyn.	A_k	USIM, AuC	48
f8	Šifrovacia funkcia		MS, RNC (BTS)	48
f9	Generovanie integrity	A_k	MS, RNC (BTS)	32

Tabuľka 1: Tabuľka funkcií používaných v štandarde 3G a 4G



Obr. 3: Protokol v používaný v sieťach 3G [13].

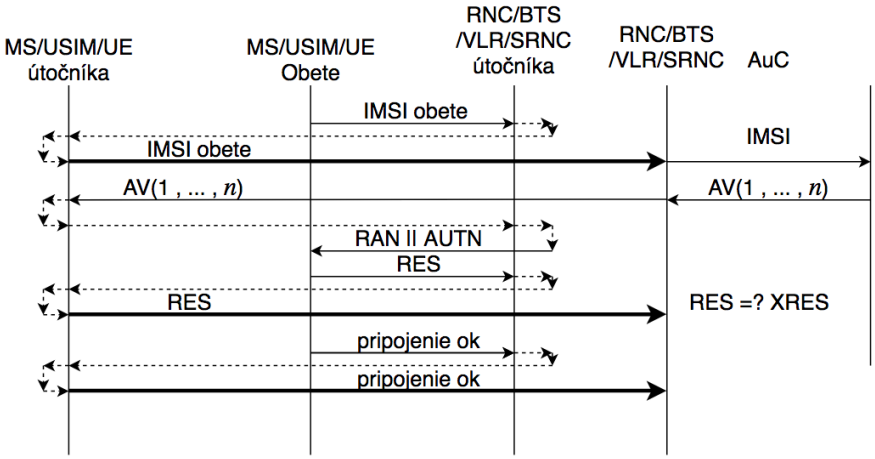
Ak by sme ale možnosť vytvorenia falošného operátora zavrhli, vytvorenie falošnej BTS stanice by nám v tomto prípade nepomohlo, skôr naopak, fungovali by sme len ako repeater, tj. nejaké zariadenie, ktoré by šírilo signál bez možnosti reálneho útoku.

Okrem spomínaných útokov existujú špecifické útoky, ktorými je možné realizovať rôzne benefity pre útočníkov, ako napríklad posielanie SMS správ alebo zisťovanie (podvrhnutie) polohy obete útoku [21]. Jeden z najvýznamnejších útokov na 3G a 4G siete je práve útok s názvom authentication relay attack, ktorý dovoľuje útočníkovi pripojiť sa na jadro siete bez toho, aby poskytol akékoľvek legitímne údaje, pričom sa bude vydávať za obeť. Pomocou tohto útoku je možné zmeniť lokáciu obete v jadre siete, a tým pripraviť falošné alibi alebo dôkazy, napríklad počas kriminálneho vyšetrovania. Ďalším zaujímavým útokom je takzvaný DoS útok, ktorý je možné na tieto siete vytvo-

riť [21]. Všetky tieto útoky sú útoky na implementačné chyby a nedostatočné overovanie zo strany ako operátora domovskej siete alebo tak aj operátora BTS stanice.

1.1.2.1 Podvrhnutie lokality

Jedným z možných útokov je podvrhnutie lokality obete na základne BTS stanice. Takýto útok je možné realizovať napríklad pre možnosť falšovania pozície pri kriminálnom vyšetrovaní. Útok prebieha tak, že útočník si zvolí obeť, pripraví si falošnú BTS' stanicu a pripraví si falošné UE' zariadenie. Útočník vynúti pripojiť obeť na BTS' stanicu, pričom táto stanica je priamo pripojená s útočníkovým UE' zariadením. Pri výzve na pripojenie zariadenia UE obete prepošle UE obete IMSI číslo falošnej BTS' stanici, pričom táto stanica slúži len na preposielanie výziev falošnému UE'. Ten bude preposielať všetky dáta BTS stanici, ktorá slúži na podvrhnutie lokácie [21]. Schému útoku je možné vidieť z obrázka 4, kde falošná BTS stanica slúži len na preposielanie dát. V tomto prípade útočník nie je schopný vytvoriť žiadne hovory, len SMS správy, vie len vytvoriť podvrhnutú lokáciu zariadenia.



Obr. 4: Útok na siete 3G a 4G [21].

1.2 Šifrovacie algoritmy používane v sieti 2G

V rámci štandardu 2G sa používajú dva rozdiele šifry. Jedna je označovaná ako A5/1 a druhá ako A5/2. Tieto šifry boli rozdielne z dôvodu zákazu vývozu šifier z USA a šifra A5/2 bola explicitne určená šifra pre krajiny východu, respektíve pre krajiny ZSSR. Navrhnutá bola tak, aby jej kryptoanalýza bola jednoduchšia ako v prípade šifry A5/1 [2].

1.2.1 A5/1

Je to prúdová šifra, ktorá je dodnes používaná v štandarde 2G, a tým pádom je aj používaná vo väčšine mobilnej hlasovej komunikácie. Jej algoritmus bol z počiatku utajovaný a reverzným inžinierstvom bol neskôr extrahovaný priamo zo SIM kariet [9]. Šifra pozostáva z troch posuvných registrov a niekoľko XOR hradiel [34]. Výstup zo šifry A5/1 je jednoducho modulo 2 súčet výstupov z každého LFSR registra, ktoré sú definované nasledovne:

$$\text{register } R1\text{: } x^{19} + x^5 + x^2 + x + 1$$

$$\text{register } R2\text{: } x^{22} + x + 1$$

$$\text{register } R3\text{: } x^{23} + x^{15} + x^2 + x + 1$$

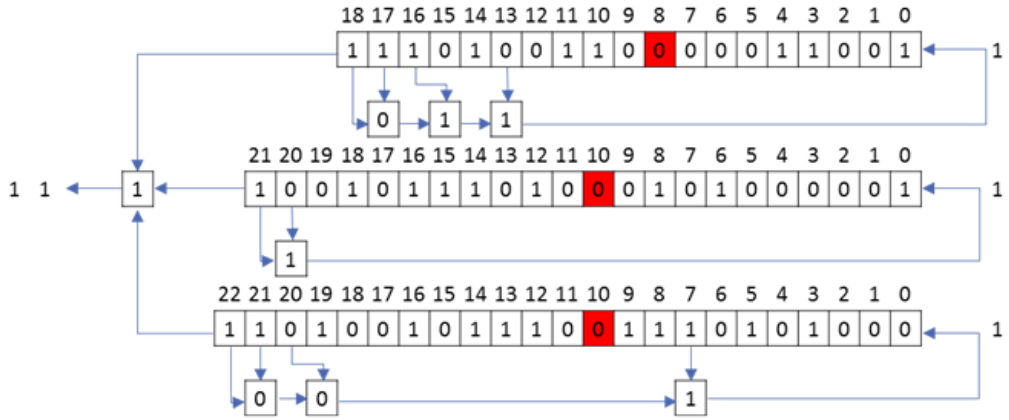
Definujme $R1(i)$ i-tý bit registra R1, $i \in \{0, \ldots, 18\}$, $R2(i)$ i-tý bit registra R2 , $i \in \{0, \ldots, 22\}$, $R3(i)$ i-tý bit registra R3, $i \in \{0, \ldots, 23\}$, a nech S je suma $S = R1(8) + R2(10) + R3(10)$. Položme $M = 1$ ak $S >= 2$ a $M = 0$ ak $S <= 1$. Nech CLK_R je posuvný vstup do registra R (ak CLK_R je rovné 1, tak register R sa bude posúvať) [34]. Potom definujeme

$$CLK_{R1} = (R1(8) \text{ XOR } M)^{`}$$

$$CLK_{R2} = (R2(10) \text{ XOR } M)^{`}$$

$$CLK_{R3} = (R3(10) \text{ XOR } M)^{`}$$

Ak to celé zhrnieme, šifra A5/1 je posuvná šifra s majoritným posunom. To znamená, že len tie register, ktoré zdieľajú majoritný bit z bitov A_8, B_{10}, C_{10} sa budú posúvať. Schematický popis šifry A5/1 uvádzame v obrázku 5, kde sú červenou farbou zobrazené bity, na základe ktorých je daný register posúvaný (v prípade obrázka 5 sa budú posúvať všetky 3 registre).



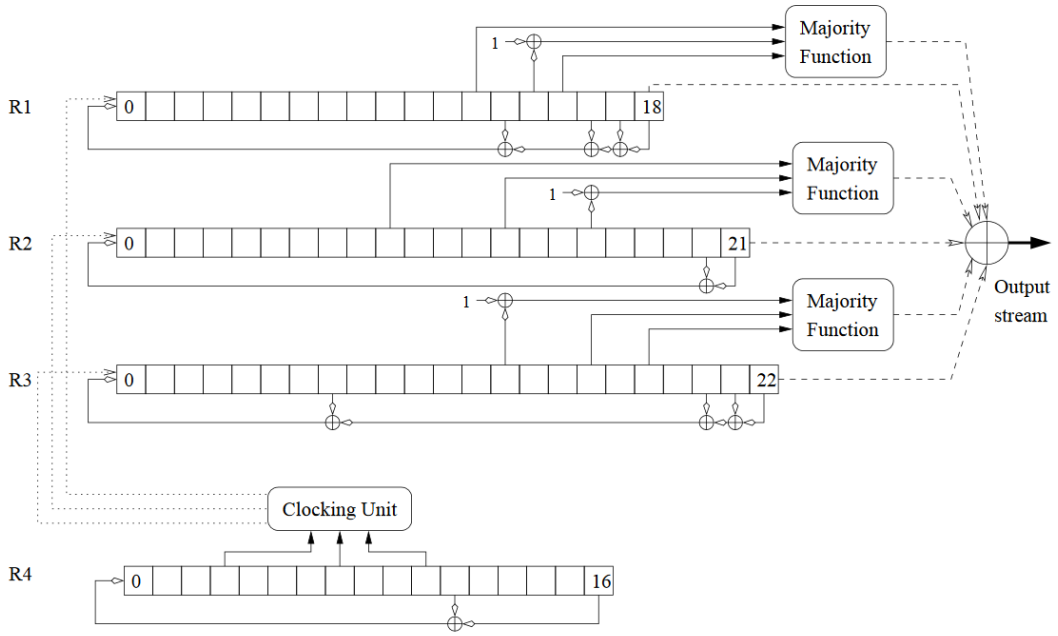
Obr. 5: Schéma šifry A5/1.

Inicializácia šifry prebieha tak, že najprv sa všetky tri registre nastavia na hodnotu 0. Následné sa bude zľava vkladať kľúč K_c a takzvaný frame number, ktorý popíšeme bližšie ďalej, označovaný ako FN. K_c je odvodený z protokolu popísanom v kapitole 1.1.1 funkciou označovanou ako A8 - pre nás bližšie nešpecifikovanou, keďže nie je predmetom tejto práce. Frame number je verejne známy. Na začiatku pri inicializácií je kľúč pridaný k poslednému bitu každého z registrov pomocou funkcie XOR s vypnutým majoritným posúvaním. Taktiež je rovnako pridaný aj frame number po inicializácií kľúčom. Kľúč má veľkosť 64 bitov a frame number má veľkosť 22 bitov. Následne je šifra točená 100 krát bez výstupu a už s aktivovaným majoritným posúvaním. Potom je vygenerovaný výstup o veľkosti 114 bitov pre dáta prichádzajúce k UE a ďalších 114 bitov pre dáta odchádzajúce od UE. Následne je len zmenený frame number a inicializácia prebieha odznova [16]. Ďalšími vlastnosťami šifry A5/1 sa budeme zaoberať v nasledujúcich kapitolách.

1.2.2 A5/2

Ako už bolo spomenuté, šifra A5/2 vznikla ako exportná verzia šifry A5/1 pre krajiny východnej Európy. Táto šifra je pre našu prácu nie príliš zaujímavá, keďže už pred niekoľkými rokmi bola poskytnutá metóda kryptoanalýzy tejto šifry v reálnom čase, do niekoľkých sekúnd [2]. Napriek tomu pre úplnosť popíšeme jej princíp. V podstate ide o verziu šifry A5/1 s troma posuvnými registrami o veľkosti 19,22 a 23 bitov, s mierne pozmeneným systémom majoritného posúvania. Šifra používa jeden register naviac, ktorý je inicializovaný pri inicializovaní kľúča a taktiež aj frame number. Tento register je zodpovedný za to, že jednotlivé registre $R1, R2, R3$ budú posúvané, a to tak, že ak sa 10. bit registra $R4$ zhoduje s majoritou, tak register $R1$ sa posúva. Ak

sa 3. bit registra $R4$ zhoduje s majoritou, tak sa posúva register $R2$ a ak sa zhoduje 7. bit registra $R4$ tak sa posúva register $R3$ [2]. Výstup šifry je počítaný ako funkcia XOR z 19.,22.,23., bitu registrov $R1, R2, R3$ a taktiež ako XOR funkcia z niektorých bitov pomocou majoritnej funkcie. Pre bližšie informácie a implementáciu šifry A5/2 odporúčame článok, v ktorom je aj popísaný útok na danú šifru v reálnom čase [2].



Obr. 6: Schéma šifry a5/2 [2].

1.3 Šifrovacie algoritmy používane v sieti 3G a 4G

Ďalšími algoritmami, ktorým sa budeme venovať sú šifrovacie algoritmy v mobilných sieťach typu 3G a 4G. Medzi základné algoritmy patrí šifra Kasumi, AES a Snow3G. AES a Snow3G patria medzi nové šifry v mobilných sieťach, pričom stále je možné zvoliť mód šifrovania, respektíve druh šifry použitej pri šifrovaní. Šifra Snow3G nie je ešte tak dôkladne preskúmaná ako šifra AES, no ak by sa v budúcnosti našli chyby v tejto šifre, existuje veľmi jednoduché riešenie problému, a to je prepnutie len na mód šifrovania AES. Totižto BTS stanica má stále možnosť vybrať, aké módy šifrovania bude podporovať a následne pošle správu mobilnému zariadeniu, ktoré šifrové algoritmy podporuje a UE si vyberie jemu vyhovujúce algoritmy. Týmto je vlastne zabezpečené to, že ak by prišlo niekedy k prelomeniu jednej zo šifier AES alebo Snow3G, tak sa jednoduchým krokom jedna z týchto šifier vypne.

1.3.1 KASUMI

KASUMI je na rozdiel od predchádzajúcich dvoch šifier, šifra bloková, s permutačnou sieťou a Feistelovou sieťou, pričom ide o rekurzívnu asymetrickú Feistelovú sieť. Pôvodne bola táto šifra odvodená od šifry MYSTI1 tým, že bola zmenená schéma odvodzovania kľúča pre jednotlivé kolá z nelineárneho modelu na lineárny model, a to tak, že sa k jednotlivým kľúčom pomocou funkcie XOR pridá konštanta a následne sa kľúč podľa toho, v ktorom kole sa nachádzame bitovo rotuje o fixný počet kôl. Generovanie kľúča a schému šifry uvádzame v prílohe A a v prílohe B. Nasledujúci popis sme čerpali zo zdroja [15].

Šifra KASUMI je 64 bitová šifra a má veľkosť kľúča 128 bitov. KASUMI bola navrhnutá na optimalizáciu hardvéru, a tak bola jej bezpečnosť zoslabená ako sa ukáže v ďalších kapitolách. KASUMI má rekurzívnu štruktúru. Každý z ôsmich Feistelových kôl je zložený z funkcie FO, ktorá je troj-kolová 32 bitová Feistelová sieť a z FL funkcií, ktoré pridávajú 32 bitový podkľúč k dátam. Poradie funkcií sa v jednotlivých kolách mení (v nepárnych kolách je funkcia FL prvá, v parných kolách je funkcia FO prvá).

Funkcia FO ma taktiež rekurzívnu štruktúru. Každé z troch kôl funkcie FO pozostáva z pridávania kľúča a z aplikácie funkcie FI, ďalšia trojkolová Feistelová sieť. Funkcia FI používa dve nelineárne S-boxy $S7$ a $S9$, kde $S7$ je permutácia zo 7bitov na 7bitov a $S9$ je permutácia z 9 bitov na 9 bitov, a funkcia FI aplikuje taktiež 16 bitový podkľúč, ktorý je pridaný k dátam. Celkovo 96 bitov podkľúča vstupuje do funkcie FO v každom kole.

Vstup do funkcie FL je 32 bitový vstupný šifrovaný text a 16 bitov podkľúča. Jeden podkľúč ovplyvňuje dáta použitím OR funkcie a druhý podkľúč ovplyvňuje dáta použitím AND funkcie.

Podkľúče sú odvodené z kľúča lineárnym spôsobom: 128 bitový kľúč je najprv rozdelený na osem 16-bitových časti: $K_1, K_2, K_3, \ldots, K_8$. Každý K_i je použitý na výpočet $K_i' = K_i \oplus C_i$, kde K_i sú známe fixované konštanty. V každom kole, osem slov je použitých ako podkľúč (až na nejakú rotáciu). Celkovo teda 128 bitov podkľúča každého kola je lineárne závislých od tajného kľúča. Presné odvodenie kľúča sa nachádza v prílohe 2 a v prílohe 3 sú popísané konštanty.

1.3.2 AES

AES je bloková šifra, [11] založená na substitučno permutačnej siete, ktorá nepoužíva Feistelovú sieť. Používa veľkosť bloku na šifrovanie 128 bitov a veľkosti kľúča môžu byť 128, 192 alebo až 256 bitov. Šifra pracuje s maticou o veľkosti 16 bajtov, ktorá vstupuje v jednotlivých kolách do 4 rôzných funkcií, a to funkcia SubBytes, Shift-

Rows, MixColumns, AddRoundKey. Počet kôl je rôzny v závislosti od veľkosti kľúča, a to 10 kôl pre 128 bitový kľúč, 12 kôl pre 192 bitový kľúč a 14 kôl pre 256 bitový kľúč.

Funkcia SubBytes využíva nelineárny S-box, ktorý je konštantný pre každé kolo. Nasledujúca funkcia ShiftRows vykonáva posun bajtov v šifrovacej matici. Predposledná funkcia MixColumns pre každý stĺpec matice aplikuje nad $GF(2^8)$ násobenie modulo $01_{16} \cdot z^4 + 01_{16}$ fixným polynómom $c(z) = 03_{16} \cdot z^3 + 01_{16} \cdot z^2 + 01_{16} \cdot z\ + 02_{16}$. Posledná funkcia AddRoundKey je funkcia, ktorá ku každému bitu použitím funkcie XOR pridá kľúč. Pre bližší popis odporúčame knihu autora šifry AES [11].

Podľa dosial známych skutočnosti, efektívny útok na šifru AES doposiaľ neexistuje, a z nášho pohľadu ju budeme považovať za bezpečnú, a preto sa jej nebudeme ďalej venovať. Existujú ale útoky, ktoré vedia časovú zložitosť znížiť o nejaký pomerne malý počet bitov. Jeden z najlepších útokov, ktoré na šifru AES existujú je takzvaný bicyklický útok [8], ktorý je schopný znížiť časovú zložitosť oproti útoku hrubou silou na 2^{126} bitov pre 128 bitovú šifru, na čo je potrebné mať vopred pripravených 2^{88} dát, čo je celkovo viac dát ako je na celom svete [8].

1.3.3 Snow3G

Poslednou šifrou, ktorá sa využíva v mobilných sieťach je šifra Snow3G. Ide o šifru, ktorá je orientovaná na generovanie slov o veľkosti 32 bitov s použitím 128 bitového kľúča a 128 bitovej inicializácie [29]. Pri inicializácií je šifra posúvaná bez výstupu a následne sa prepne do šifrovacieho módu. Šifra používa základné funkcie [23]:

1. $MUL_\alpha(c) = (MULxPOW(c, 23, 0xA9)||MULxPOW(c, 245, 0xA9)||$ $MULxPOW(c, 48, 0xA9)||MULxPOW(c, 239, 0xA9))$, kde c je 8 bitov, funkcia MUL_α mapuje 8 bitový vstup na 32 bitov.

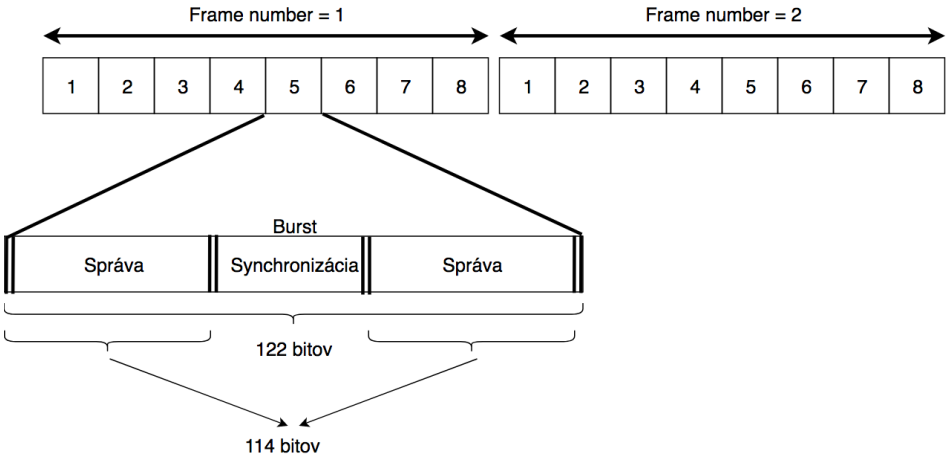
2. $DIV_\alpha(c) = (MULxPOW(c, 16, 0xA9)||MULxPOW(c, 39, 0xA9)||$ $MULxPOW(c, 6, 0xA9)||MULxPOW(c, 64, 0xA9))$, kde c je 8 bitov, funkcia DIV_α mapuje 8 bitový vstup na 32 bitov.

3. $MULxPOW(V, i, c)$ je definovaná rekurzivne: Ak i je rovné 0, potom $MULxPOW(V, i, c) = V$, inač $MULxPOW(V, i, c) = MULx(MULxPOW(V, i-1, c), c)$.

Vo výpočte výstupu zo šifry Snow3G sa počíta medzihodnota $v = (s_{0,1}||s_{0,2}||s_{0,3}||0x00) \oplus MUL_\alpha(s_{0,0}) \oplus s_2 \oplus (0x00||s_{11,0}||s_{11,1}||s_{11,2}) \oplus DIV_\alpha(s_{11,3}) \oplus F$ a nakoniec sa nastaví $s_{15} = v$, kde $s_{i,j}$ označuje i-ty bajt a j-ty bit v i-tom bajte. Okrem daných operácií Snow3G obsahuje aj S-boxy, S1 a S2, založené na S-boxov šifry AES. Schéma šifry je v prílohe D. Pre bližšie informácie odporúčame článok o implementácií danej šifry

a vylepšeniach jej implementácie [23, 24]. Pre danú šifru taktiež v súčasnosti neexistujú efektívne metódy kryptoanalýzy, ktoré by danú šifru prelomili v reálnom čase.

1.4 Frame number



Obr. 7: Príklad frame number.

Komunikácia v GSM sieti používa tzv. Frame number, ktoré sú verejné známe. Každá BTS stanica vysiela správu v takzvaných frame a burst. Dĺžka burstu je 122 bitov, pričom prenášaná informácia je dĺžky 114 bitov. Práve týchto 114 bitov je generovaných ako výstup zo šifry A5/1. Správa, ktorú prenášame GSM (napríklad SMS správa) je rozdelená medzi niekoľko burstov. Napríklad SMS správa, ktorá je kódovaná 7 bitovým kódovaním (štandard z pred pár rokov, dnes sa používa UTF-8) a obsahuje 120 znakov je rozdelená do 7 rôznych burstov. Okrem danej správy burst ešte obsahuje informácie od operátora, kontrolné súčty a niekedy aj rôzne nepredvídateľné dáta. Preto je potrebné danej problematike venovať pozornosť. Každých osem burstov je odosielaná naraz, počas jedného frame. Takto je obslúžených naraz až 8 mobilných zariadení. Frame number je verejné známe číslo, je nastavený na začiatku komunikácie (posiela sa na začiatku komunikácie ako ohlásenie, že takýto je frame number, správa typu: Next frame number is NNNN). Frame number je číslo veľkosti 22 bitov a inicializácia šifry A5/1, popísanej vyššie nezávisí od frame number. Ak sa pošle 8 burstov, počítač zvýši frame number o 1 (inkrementuje danú hodnotu) a šifra A5/1 sa reinicializuje s novým frame number no s pôvodným inicializačným kľúčom [35]. Dĺžka intervalu trvania jedného frame number je 4.615 ms [27].

1.5 Nezabezpečená komunikačná linka operátora

Mobilný operátor nemá povinnosť šifrovať dáta, ktoré prechádzajú jeho internou mobilnou sieťou, a ani to v mnohých prípadoch nerobí. Dokonca zákon Slovenskej republiky (Zákon č. 351/2011 Z. z., §64, ods. 14) hovorí o tom, že je povinné pre operátora zabezpečiť také podmienky, aby bolo možné na žiadosť orgánov činných v trestnom konaní zachytiť nešifrovanú mobilnú komunikáciu, a tým pádom ju aj odpočúvať. Práve z týchto dôvodov je akákoľvek mobilná komunikácia nešifrovaná na úrovni vnútornej siete operátora. Ak by bol niekto schopný dáta, ktoré sa šíria po vnútornej sieti mobilného operátora zachytiť, získal by priamy prístup k všetkým mobilným hovorom, ktoré sa na danej časti sieti uskutočňujú. Na druhej strane, ten istý zákon hovorí o tom, že je operátor povinný zabezpečiť mobilnú sieť tak, aby nebolo možné odpočúvanie komunikácie, ak na to nebol daný súhlas alebo tento zákon nehovorí inač (v prípade vyžiadania odpočúvania mobilnej komunikácie zo strany orgánov činných v trestnom konaní).

Stretol som sa s tvrdeniami na rôznych sociálnych sieťach, že užívateľ bol schopný sa napojiť priamo na linku operátora (napríklad býva v obytnom dome, kde na streche budovy je namontovaná BTS stanica), a z danej linky vie čítať dáta, ktoré sa k danej BTS stanici posielajú (a odosielajú). Zdroje k týmto príspevkom uvádzať nebudem, keďže zo strany dotyčnej osoby ide o trestný čin.

Z daného vyplýva, že hlasový hovor a SMS správy nie sú na strane operátora šifrované a táto linka sa považuje za dôvernú, aj keď to takto nemusí byť. Šifrovanie medzi jednotlivými BTS stanicami, ktoré sú schopné obslúžiť pomerne veľké množstvo ľudí, by bolo nákladné riešenie pre mobilných operátorov.

2 Kryptoanalýza šifry A5/1

V nasledujúcej kapitole sa pozrieme na možnosti útoku na šifru A5/1, jej nedostatky, a taktiež sa zameriame na aktuálne možnosti a praktickú implementáciu na danú šifru. Šifra A5/1 je od roku 2010 považovaná za prelomenú [30], kedy bol reálny útok predvedený na danú šifru na konferencií Black Hat v USA. Predtým boli popísané aj iné útoky, ktoré ale neboli z pamäťového hľadiska optimálne - boli potrebné obrovské dátové úložiská na uloženie vopred predvypočítanej tabuľky a kryptoanalýza bola možná v reálnom čase (rádovo v milisekundách) [17]. Súčasné metódy využívajú takzvané dúhové tabuľky, ktoré sú vlastne kompromisom medzi uloženým miestom a časom potrebným na kryptoanalýzu šifry. Myšlienka dúhových tabuliek je v podstate mať prepočítané stavy šifry (ak je to možné), a potom celú množinu (ktorá je obvykle obrovská) zmenšiť na množinu takú, že existujú jednotlivé prechody medzi stavmi šifry.

2.1 Myšlienka útoku

Šifra A5/1 je prúdová šifra so 64 bitami ako vnútorné stavy a ktorá používa 64 bitový kľúč. Aby sme boli schopní z vnútorného stavu dostať späť kľúč, a teoreticky by sme vedeli, poznali vnútorný stav šifry, vieme získať kľúč, ktorý bol použitý na šifrovanie [30, 27]. Lenže výstup z tejto šifry tvorí postupnosť bitov, konkrétne prvých 114 bitov pre šifrovanú komunikáciu jedným smerom a 114 bitov pre šifrovanú komunikáciu druhým smerom. Následne sa tento výstup funkciou XOR pridá ku správe, ktorú chceme šifrovať a táto správa je následne poslaná mobilnou sieťou do BTS stanice. Ak by sme poznali správu, ktorú chce mobilné zariadenie šifrovať, vedeli by sme spätne vygenerovať to, čo sa nachádza vo vnútornom stave šifry A5/1 predtým, ako šifra vygenerovala výstup? Do tohto bodu vstupuje takzvaná dúhová tabuľka. Dúhovú tabuľku môžeme vnímať ako databázu vnútorných stavov spolu s ich výstupom po zatočení šifrou A5/1. V takom prípade vieme zobrať to, čo bolo vnútorným stavom šifry A5/1 predtým, ako šifra vygenerovala a využiť backclocking, ktorým sa dostaneme späť k pôvodnému kľúču [18, 27, 30].

Takýmto spôsobom by sme ale potrebovali mať k dispozícií disk o kapacite $2^{64} \cdot 128$

bitov - pretože ak uvážime, že kompletne možný vnútorný stav šifry A5/1 je 64 bitov, potom celkový počet všetkých možností je 2^{64}, a pre každý vnútorný stav potrebujeme uložiť samotný vnútorný stav a k nemu aj to, čo tento vnútorný stav obsahuje. Preto sa vymýšľali rôzne metódy, kedy a ako je možné danú veľkosť priestoru na disku zmenšiť.

Približne 20-30 možných stavov dostaneme pri točení šifry A5/1 spätne pri generovaní kľúča použitého na šifrovanie pri točení šifry z komunikácie, ktorá prebieha smerom k zariadeniu a približne 50-60 možných kľúčov, ak to budeme robiť z komunikácie, ktorá prebieha od UE k BTS stanici [30]. Vyskúšanie možných kľúčov a následným odstránením frame number získavame kľúč, ktorý bol použitý pri generovaní vnútorného stavu šifry A5/1. Keďže šifrovanie v mobilných sieťach 2G funguje tak, že sa použije následne len iný frame number, tak získavame kľúč, ktorý bude platný až do doby zmeny kľúča pri autentifikácií operátora. Táto doba závisí od toho, v akej sieti sme - niektoré telekomunikačné spoločnosti používajú ten istý kľúč do doby, kým mobilné zariadenie nezmení operátora. Jednotlivý kľúč sa presúva spolu so zariadením medzi BTS stanice [35]. Iný operátor má časovú pečiatku platnosti alebo implementuje novú dohodu na kľúči pri každej zmene BTS stanice, čo je ale časovo náročné hlavne napríklad pri ceste automobilom.

Dôležitým prvkom je ešte zachytenie mobilnej komunikácie, no tomu sa venovať veľmi nebudeme, len v individuálnej kapitole popíšeme možné prístupy.

2.2 Dúhové tabuľky

2.2.1 Všeobecný popis dúhových tabuliek

Dúhové tabuľky sú predvýpočítané tabuľky, ktoré obsahujú reťaz dát. To znamená, že nepotrebujeme na kryptoanalýzu mať na disku uložené všetky dáta, ale uložíme si len nejakú časť a zvyšok dát dopočítame nejakým algoritmom. Nie všetky šifry sú vhodné na použitie kryptoanalýzy pomocou dúhových tabuliek ale základným kryptografickým primitívom je HASH funkcia, ktorá keď vygeneruje výstup, tento výstup sa použije ako vzor pre ďalší vstup do tej istej funkcie. Takýmto reťazením vznikajú dúhové (rainbow) tabuľky. Myšlienka je nasledujúca: Z počiatočného otvoreného textu P_0 a korešpondujúceho šifrovaného textu C_0, sa metóda snaží nájsť možný kľúč $k \in \mathbb{N}$, ktorý bol použitý na šifrovanie tak, že

$$C_0 = S_k(P_0),$$

ak vygenerujeme všetky možnosti šifrovaného textu s otvoreným textom so všetkými možnosťami kľúča $k \in \mathbb{N}$. Šifrovaný text je organizovaný do takzvaných reťazcov, kde

iba prvý a posledný element týchto reťazcov je uložený v pamäti [32]. Uloženie iba prvého a posledného reťazca sa nazýva takzvaný trade-off, vyváženie pomeru tým, že ušetríme pamäť za cenu kryptoanalytického času. Tieto reťazce sú tvorené redukčnou funkciou R, ktorá vytvára kľúč zo šifrovaného textu a rekurzívnym aplikovaním šifry S a redukčnej funkcie vytvoríme dané reťazce striedavých kľúčov a šifrovaných textov [32].

$$k_i \xrightarrow{S_{k_i}(P_0)} C_i \xrightarrow{R(C_i)} k_{i+1}$$

Následovník $R(S_k(P_0))$ je zvyčajne aj označovaný ako $f(k)$ a generovanie kľúča z kľúča, respektíve šifrovaného textu tvorí reťaz [32]:

$$k_i \xrightarrow{f} k_{i+1} \xrightarrow{f} k_{i+2} \xrightarrow{f} \dots$$

Nakoniec je vytvorených m takýchto reťazcov o dĺžke t. Ak predpokladáme, že máme dokonalú šifru, to znamená, že pri každom rozdielnom kľúči sa nám nikdy nestane, že šifra skončí v rovnakom vnútornom stave týchto tabuliek, pravdepodobnosť úspechu je nasledovná [19]:

$$P_{table} \geq \frac{1}{N} \sum_{i=1}^{m} \sum_{j=0}^{t-1} (1 - \frac{ti}{N})^{j+1} \tag{1}$$

Úspešnosť jednej tabuľky rapídne klesá s jej veľkosťou, a aby sme získali lepšiu úspešnosť je dobré vygenerovať viac tabuliek s rozdielnou redukčnou funkciou. Pravdepodobnosť úspechu l tabuliek je vyjadrená (samozrejme, pri dokonalej šifre) ako:

$$P_{success} \geq 1 - (1 - \frac{1}{N} \sum_{i=1}^{m} \sum_{j=0}^{t-1} (1 - \frac{ti}{N})^{j+1})^l \tag{2}$$

Ak použijeme rozdielnu redukčnú funkciu R, tak ak sa dostaneme ku kolízii v dúhovej tabuľke, to znamená, že v nejakom reťazci dôjde k výpočtu pomocou redukčnej funkcie k rovnakým hodnotám, tak v prípade rozdielnych redukčných funkcií sa dostaneme k tomu, že síce v určitom bode sa nám reťazce dostanú do rovnakého bodu, ale keďže používame rozdielne redukčné funkcie, tak musí platiť, že ak sú tie redukčné funkcie dobré (zaručene produkujú rozdielny výstup pri rozdielnych vstupoch, sú bijektívne), reťazce sa dostanú následne mimo kolíziu [19].

Lenže implementácia a nájdenie takýchto redukčných funkcií a použitie viacerých tabuliek nie je efektívne. Práve preto boli zavedené takzvané dúhové tabuľky [32]. Používajú redukčnú funkciu v každom bode reťazca a začínajú s redukčnou funkciou 1 a končia s redukčnou funkciou $t - 1$ [32]. Takže ak nastane kolízia, potom sa jednotlivé reťazce spoja len ak nastala kolízia v rovnakom bode výpočtu. Ak nenastala

kolízia v rovnakom bode výpočtu, obe reťazce taktiež budú pokračovať s rozdielnym výsledkom nasledujúcej redukcie a tým pádom sa dva rozdielne reťazce nespoja. Pre reťazec dĺžka t, ak nastane kolízia, šanca že sa daný reťazec spojí je iba $\frac{1}{t}$ [32]. Pravdepodobnosť úspechu, že sa daný reťazec spojí je [32]:

$$P_{table} \geq 1 - \prod_{i=1}^{t}(1 - \frac{m_i}{N}) \tag{3}$$

kde $m_1 = m$ a $m_{n+1} = N(1 - e^{-\frac{m_n}{N}})$

V praxi sa reálne používa namiesto rôznych redukčných funkcií ako redukcia, pripočítanie určitej konštanty. Ak si napríklad zvolíme, že ak chceme mať reťazec dĺžky t tak prvých t operácií budeme stále pripočítavať konštantu takú, aby zmenila celý výstup z redukčnej funkcie. Následne sa konštanta zmení a celý postup môžeme opakovať znova.

Ďalej budeme uvažovať o takzvaných prechodových funkciách a označovať ich ako f. Tieto prechodové funkcie sú definované ako zloženie redukčnej funkcie a funkcie prechodu od jedného stavu k druhému, na základe ktorých je možné z týchto f funkcií získať hľadanú hodnotu (kľúč, vnútorný stav). Tieto f funkcie napríklad mapujú výstup z HASH funkcie ako vstup do ďalšej časti HASH funkcie v dúhových tabuľkách, pričom využívajú napríklad skracovanie výstupu, pridanie farby a ľubovoľný ďalší výpočet.

2.2.2 Zavedenie význačných bodov

Ako je teda možné zistiť, kedy jednotlivú farbu môžeme zmeniť a kedy končí daný reťazec? Existujú možnosti, ako si počítať počet už vygenerovaných reťazcov, no tento spôsob je veľmi neefektívny, a preto sa zavádzajú takzvané význačné body [28]. Ako význačný bod môže byť v podstate čokoľvek, zvolene rozumne. Napríklad po použití redukčnej funkcie R uvidíme, že časť prvých bitov sú len samé jednotky, prehlásime to za význačný bod a môžeme zmeniť farbu, tj. konštantu. Celkovo použitie dúhových tabuliek na konci vyzerá takto:

$$k_i \xrightarrow{f} k_{i+1} \xrightarrow{f} k_{i+2} \xrightarrow{f} \ldots \xrightarrow{f} \text{DP} \xrightarrow{f} \ldots \xrightarrow{f} \text{DP}$$

Význačné body sú vhodné na to, aby sme ušetrili miesto. Totižto, ak použijeme takéto význačne body, vieme simulovať možnosť použitia viacerých farieb pri každom význačnom bode tak, ako keby sme prechádzali dúhovú tabuľku odznova. Podľa toho, aký použijeme význačný bod, dostávame rôzne priestorové zložitosti. Totižto ak napríklad za význačný bod použijeme konštantný koniec vygenerovaného ďalšieho vstupu o veľkosti napríklad 10 bitov, celková veľkosť tabuľky (bez použitia farby) sa nám zmenši $2^2 2$ krát.

2.2.3 Využitie dúhových tabuliek

Predstavme si, že dostaneme za úlohu nájsť kľúč, ktorý bol použitý na šifrovanie pre nás známeho otvoreného text a šifrovaného textu a predpokladajme, že máme k dispozícii takúto tabuľku, vopred predpočítanú. Každý riadok má dĺžku t, pričom používame 4 rozdielne farby. Predpokladajme, že veľkosť kľúča je 12 bitov. Potom na to, aby sme pokryli celú množinu generovaných kľúčov potrebujeme $m = \frac{2^{12}}{\cdot t \cdot 4}$ riadkov. Na to, aby sme uložili kompletnú tabuľku, bez použitia farieb a dĺžky reťazcov by sme potrebovali $2^{12} riadkov$, čím sme ušetrili pomerne dosť veľkú kapacitu miesta na disku. Útok bude prebiehať tak, že ak dostaneme text, ktorý potrebujeme prelomiť (šifrovaný postačuje), použijeme algoritmus redukcie a následného generovania ďalšieho prvku v riadku kým neprídeme k poslednému, vopred zvolenému význačnému bodu. Nakoniec sa pozrieme do tabuľky, či daný bod v nej existuje a ak áno, tak vyberieme zvolený riadok, čím získame začiatočný bod a postupne generujeme ďalší prvok v reťazci, až kým neprídeme k reťazcu, ktorý potrebujeme prelomiť. Tým, že generujeme celý riadok odznova, vieme aj predchádzajúcu hodnotu, ktorá sa rovná nášmu hľadanému kľúču. Samozrejme, treba ešte vyriešiť problém, s akou farbou chceme začať, aby sme zväčšili pokrytie - no tento algoritmus vieme spustiť 4 krát po sebe s rozdielnou počiatočnou farbou.

Algoritmus 1 Prechod dúhovými tabuľkami.

```
procedure NÁJDI K_c(C, t)   ▷ Nájdi v dúhovej tabuľke t šifrovací kľúč reťazca C
    x ← C
    for i ∈ {1, ..., n} do
        while x ∉ EP do                          ▷ EP je množina endpoint-ov
            x ← f(x) ⊕ c[j], j ∈ {i, ..., n}     ▷ c[i] je i-ta farba
    y ← t[x]   ▷ t[x] je počiatočná hodnota riadka tabuľky odpovedajúca koncu x
    for i ∈ {1, ..., n} do
        while x ∉ EP do                          ▷ EP je množina endpoint-ov
            x ← f(y) ⊕ c[i]                      ▷ c[i] je i-ta farba
            if x = C then return R(y)
            y ← x
    return nenájdene
```

Algoritmus 1 popisuje približne postup získania hodnoty kľúča pomocou dúhových tabuliek. EP označuje množinu koncov, ktorá je jednoznačne určená napríklad význačnými bodmi. Tie môžu byt charakteristické tak, že sa nejaká časť bitov, napríklad na konci reťazca, rovná 0. Pole $c[i]$ označuje pole farieb, ktoré sa v danom algoritme používajú.

Tento algoritmus je len ilustračný, pre lepšie pochopenie dúhových tabuliek.

2.3 Dúhové tabuľky pre šifru A5/1

2.3.1 Formálna definícia a časová zložitosť

V tejto kapitole formálnejšie definujeme dúhové tabuľky, pretože to budeme nasledovne potrebovať pre vetu, ktorá hovorí o pokrytí dúhových tabuliek pri vhodne zvolenom algoritme. Najprv si definujeme parametre podľa [28]:

1. m: definuje počet začiatočných bodov dúhovej tabuľky.
2. s: označuje počet funkcií f, označovaných $f_1, \ldots, f_s$, zvyčajne konštruovaných tak, ako funkcia f popísaná vyššie s tým, že sa používa rozdielna konštanta (farba).
3. v: počet funkcií f_i v rovnakom stĺpci, tj. počet aplikácií rovnakej funkcie f_i.
4. r: počet sekcií $(f_s)^v \circ \ldots \circ (f_2)^v \circ (f_1)^v$.
5. $t = rsv$.

Útočník potom vytvára dúhové tabuľky nasledovne [28]:

1. Vyber m začiatočný bod uniforme náhodne z množiny $\{1, 2, \ldots, N\}$ a ďalej označíme tieto body ako $SP_1, SP_2, \ldots, SP_m$.
2. Pre $1 \leq i \leq m$:
   (a) Položme $x_{i,0} = SP_i$.
   (b) Pre $q = 1, 2, \ldots, r$:
   Pre $j = 1, 2, \ldots, s$:
   Pre $l = 1, 2, \ldots, v$:
   - Vypočítaj $x_{i,(q-1)sv+(j-1)v+l} = f_j(x_{i,(q-1)sv+(j-1)v+l-1})$
   - Ak $(q = r)$ a $(j = s)$ a $(l = v)$, hodnota $x_{i,rvs}$ je označená ako konečný bod (endpoint) korešpondujúci k SP_i a označíme ho ako EP_i.
3. Ulož m párov (SP_i, EP_i) do tabuľky usporiadanej podľa hodnôt koncových bodov.

Pre účely tvrdenia popíšeme aj presný postup generovania spätnej hodnoty v dúhovej tabuľke [28]:

1. Pre $j = s, s\ -1, \ldots, 1$:

   (a) Použijeme jednoduchú operáciu konkatenovanú s f na konštrukciu f_j na daný výstup y, ktorý označíme y'_0.

   (b) Pre každé $l = 1, 2, \ldots, v$:

      i. Polož $y_0 = y'_0$ ak $j = s$, inač vypočítaj $y_0 = (f_s)^v \circ \ldots \circ (f_{j+1})^v(y'_0)$

      ii. Pre $q = 1, 2, \ldots, r$:

         A. Skontroluj, či y_{q-1} je endpoint v danej tabuľke. Ak y_{q-1} sa nezhoduje so žiadnym endpointom, pokračujeme ďalším podkrokom. Ak y_{q-1} sa zhoduje s nejakým endpointom, napríklad EP_i, potom vygenerujeme $x_{i,(r-q)sv+jv-l}$ z korešpondujúceho štartovacieho bodu SP_i ako $(f_1)^{v-l} \circ [(f_s)^v \circ \ldots \circ (f_2)^v \circ (f_1)^v]^{r-q}(SP_i)$ ak $j = 1$, inač $(f_j)^{v-l} \circ (f_{j-1})^v \circ \ldots \circ (f_2)^v \circ (f_1)^v \circ [(f_s)^v \circ \ldots \circ (f_2)^v \circ (f_1)^v]^{r-q}(SP_i)$. Nakoniec $x_{i,(r-q)sv+jv-l}$ je pravdepodobne správne riešenie (ak je, ukonč výpočet)

         B. Ak $q \neq r$, vypočítaj $y_q = (f_s)^v \circ \ldots \circ (f_2)^v \circ (f_1)^v(y_{q-1})$

      iii. Ak $l \neq v$, vypočítaj $y'_0 = f_j(y'_0)$.

   Ak sa žiadny z možných riešení v EP_i nenájde, útok zlyhal.

Časová zložitosť tejto kryptoanalýzy je približne $\sum_{j=1}^{s-1}\sum_{l=1}^{v}[(s-j)v] + \sum_{j=1}^{s}\sum_{l=1}^{v}\sum_{q=1}^{r-1} sv + \sum_{j=1}^{s}\sum_{l=1}^{v-1} 1 = \frac{t^2}{r} - \frac{t^2}{2r^2} - \frac{t^2}{2sr^2} + \frac{t}{r} - s$ výpočtu a časová zložitosť $\sum_{j=1}^{s}\sum_{l=1}^{v}\sum_{q=1}^{r} 1 = vrs = t$ hľadaní v tabuľke [28]. Ak do daného útoku pridáme význačné body, tak sa časová zložitosť výpočtu zmení nasledovne [28]:

$$\sum_{j=1}^{s}[q + (j-1)qv + (r-1)qsv]v = tsv + \frac{t}{r} - \frac{tsv}{2r} - \frac{tv}{2r}$$

výpočtov funkcie f, plus rvs hľadaní v tabuľke. Vyššie spomínaná definícia nie je pre dúhové tabuľky, ktoré boli použité v roku 2010 pre prvý dôkaz správnosti daného útoku. No v našom prípade ju budeme potrebovať, pretože ide o definíciu, s ktorou budeme neskôr pracovať. Dôležitým rozdielom je to, ako bude zvolená funkcia f a jej následný výpočet.

2.3.2 Generovanie tabuliek pre šifru A5/1

Veľmi dôležitou vlastnosťou šifry A5/1 je nasledujúce tvrdenie [28]:

Stav šifry (všetky vnútorné stavy troch posuvných registrov) okamžite po inicializácii kľúča šifry A5/1 je systém 64 lineárnych funkcií 64 bitov tajného kľúča. Týchto 64 bitov tajného kľúča môže byť ľahko získaných vyriešením 64 lineárnych rovníc pre

64 neznámych. Ak je daný 64 bitový stav ihneď po inicializácií frame number šifry, môžeme okamžite vygenerovať stav pred frame number s časovou zložitosťou 22 točení.

Vzhľadom na šifry A5/1, funkcia f začína po pridaní frame number a končí generovaním prvých 64 bitov výstupu. Ako vstup dostane inicializovaný stav a ako výstup vygeneruje 64 bitovú sekvenciu. Týchto vygenerovaných 64 bitov je vstupom do ďalšej f funkcie ako inicializačný stav.

K daným tvrdeniam treba poznamenať, že ak sme sa nachádzali v ľubovoľnom stave, tak f funkcia ignoruje to, aký frame number bol nastavený - ten sa jednoducho vzhľadom na kryptoanalýzu dopočíta a vznikne len iný kľúč. Totiž pre každý frame number z každého možného vnútorného stavu šifry A5/1 vieme prejsť späť k nejakému inému kľúču.

Funkcia f_i je v prípade šifry A5/1 definovaná ako $f_i = f \oplus i$, kde i je farba - podľa toho, ako je generovaná tabuľka.

Výpočet, pri zafixovanom význačnom bode ako posledných 12 bitov jednotlivého výstupu z redukčnej funkcie, pri veľkosti dúhovej tabuľky 1,6 TB s použitím 5 výkonných grafických kariet trval približne 60 dní [28]. Výpočet bol ukladaný na SSD disky o celkovej veľkosti 2 TB, čo činilo celkové náklady 15000 Amerických dolárov len na hardvér [28]. Okrem vyššie spomínanej funkcie f sa z počiatku používala taká funkcia, ktorá je „až za točením prvých 100 krát bez výstupu“, čo znamená, že funkcia f zobrala to, čo šifra A5/1 dáva na výstup po inicializácií kľúčom, frame number, a taktiež aj prvotným zatočením šifry A5/1 o 100 krát bez výstupu, takzvaného mixovania kľúča.

2.3.3 Vlastnosť dúhovej tabuľky

Pravdepodobnosť, že význačný bod bude dosiahnutý na najviac t operácií je v prípade dúhových tabuliek pre šifru A5/1 $p_{DP}(t) = 1 - (1 - \frac{1}{D})^t$, kde $D = 2^d$ a d je dĺžka nulovej bitovej masky význačného bodu [32]. Na druhej strane, pravdepodobnosť, že štartovací bod úspešne vygeneruje nejaký koncový bod je $p_{DP}(r) = (p_{DP}(r \cdot D))^S = (1-(1-\frac{1}{D})^{r \cdot D})^S$, kde M_0 a M sú celková hodnota počtu SP a platná hodnota počtu SP [32]. Ak zvolíme $M_0 = 2^{36,678}$, čo odpovedá približne 1,6 TB, máme $p_{DP} = 0,9999721$, potom $M_0 - M = 2^{21,55}$ štartovacích bodov, ktoré nemôžu úspešne generovať EP budú zahodených. Priemerný počet A5/1 funkcií operácií T potrebných na M validných reťazcov môže byť približne vyjadrený ako $T \approx \frac{1}{M} \sum rt = 1M_0(p_{DP}(t) - p_{DP}(t-1))t \approx D \cdot S$ [32].

2.4 Spätné točenie šifry A5/1

Vzhľadom k definícií šifry A5/1 označíme bity $c_1 = R1(8)$, $c_2 = R2(10)$, $c_3 = R3(10)$, to znamená, že c_i označuje bit i-tého registra, ktorý je kontrolný a slúži pre vstup do majoritnej funkcie - je to bit, ktorý rozhoduje, či sa daný register bude točiť, alebo nie. Označme c'_1, c'_2, c'_3 im predchádzajúce bity. Ak budeme šifrou A5/1 točiť spätne o jeden bit, môže nastať nasledujúcich šesť možností [32]:

1. Pre ľubovoľné k, ak $c'_i = c'_j \neq c'_k = c_k$, potom i-tý a i-tý LFSR je točený.
2. Pre ľubovoľné k, ak $c'_i = c'_j \neq c'_k \neq c_k$, potom tento stav nemá predchádzajúci stav, to znamená, že ho nie je možné točiť späť.
3. Ak $c'_i = c'_j = c'_k = c_i = c_j = c_k$, potom všetky LFSR sú točené spätne.
4. Pre ľubovoľné k, ak $c'_i = c'_j = c'_k = c_i = c_j \neq c_k$, potom existujú dve možnosti:
   (a) i-tý a j-tý LFSR je točený alebo
   (b) všetky LFSR sú točené.
5. Pre ľubovoľné i, ak $c'_i = c'_j = c'_k = c_i \neq c_j = c_k$, potom existujú tri možnosti:
   (a) i-tý a j-tý LFSR je točený alebo
   (b) i-tý a k-tý LFSR je točený alebo
   (c) všetky LFSR sú točené.
6. Ak $c'_i = c'_j = c'_k \neq c_i = c_j = c_k$, potom existujú 4 možnosti a to každý pár z troch LFSR je točený alebo všetky LFSR sú točené.

Predpokladajme, že c_1, c_2, c_3 sú náhodne vybrané hodnoty pri uniformnej distribúcií, potom počet riešení pre c'_1, c'_2, c'_3 spĺňajú pravdepodobnostnú distribúciu: $P_1(0) = \frac{3}{8}$,$P_1(1) = \frac{13}{3}2$,$P_1(2) = \frac{3}{32}$,$P_1(3) = \frac{3}{32}$,$P_1(4) = \frac{1}{32}$, kde $P_1(k)$ označuje pravdepodobnosť, že pri točení o 1 bit nastane k riešení [32]. Ak napríklad $c_1 = 0, c_2 = 1, c_3 = 1$ a $c'_1 = 1, c'_2 = 0, c'_3 = 1$, je ľahko možné pozorovať z tabuľky 2, že toto je stav, ktorý nie je možné riešiť, pretože je to v spore s majoritným princípom šifry A5/1. Výpočty týchto pravdepodobností sú založené na tabuľke 2. Ak je riešenie uhádnuté, súčasné reverzné točenie (c_i, c_j, c_k) je nahradené c'_i, c'_j, c'_k a c'_i, c'_j, c'_k sú nahradené predchádzajúcimi bitmi (c''_i, c''_j, c''_k), kde $c''_1 = R1(10), c''_2 = R2(12), c''_3 = R3(12)$. Nasledujúce každé spätné točenie zvýši počet bitov maximálne o 3. Ak vezmeme v úvahu funkciu XOR, ktorá sa

$(c'_i c'_j c'_k) \backslash (c_i c_j c_k)$	(000)	(001)	(010)	(011)	(100)	(101)	(110)	(111)
(000)	1	2	2	3	2	3	3	4
(001)	0	1	0	1	0	1	0	1
(010)	0	0	1	1	0	0	1	1
(011)	1	1	1	1	0	0	0	0
(100)	0	0	0	0	1	1	1	1
(101)	1	1	0	0	1	1	0	0
(110)	1	0	1	0	1	0	1	0
(111)	4	3	3	2	3	2	2	1

Tabuľka 2: Počet predchádzajúcich stavov (riešení) pre spätné točenie [32]

#točení \ #riešení	0	1	2	3	4	5	6	7	8	9	10
1	$\frac{12}{32}$	$\frac{13}{32}$	$\frac{3}{32}$	$\frac{3}{32}$	0	0	0	0	0	0	0
2	$\frac{12}{256}$	$\frac{97}{256}$	$\frac{12}{256}$	$\frac{27}{256}$	$\frac{8}{256}$	$\frac{3}{256}$	0	$\frac{1}{256}$	0	0	0
3	$\frac{36}{2048}$	$\frac{733}{2048}$	$\frac{108}{2048}$	$\frac{204}{2048}$	$\frac{60}{2048}$	$\frac{27}{2048}$	$\frac{3}{2048}$	$\frac{12}{2048}$	0	0	$\frac{1}{2048}$

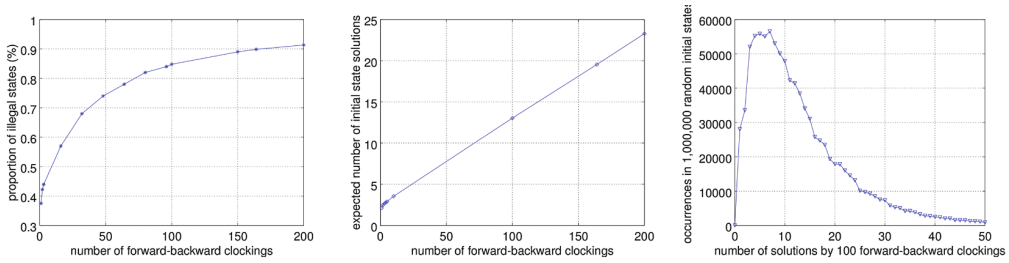
Tabuľka 3: Podmienená distribúcia počtu riešení [32]

v šifre A5/1 vyskytuje, tak ak poznáme aktuálny stav, tak jej určenie je jednoznačné vzhľadom na všetky prechádzajúce možnosti spätného točenia [32].

Zoberme si napríklad mapovanie medzi vnútorným počiatočným stavom a vnútorným počiatočným stavom s jedným spätným točením. V prieme $\frac{12}{32}$ stavov z interného priestoru je nemožných (nedosiahnuteľných) a nemajú predchádzajúci inicializačný stav, $\frac{13}{32}$ vnútorný stavov majú mapovanie jedna k jednej, $\frac{3}{32}$ interných stavov sa nám premietne do 2 stavov každý a tak ďalej [32]. Ak by sme mali uniformne distribuovanú šifru, tak v priemere počet riešení počiatočného stavu je $\frac{12}{32} \cdot 0 + \frac{13}{32} \cdot 1 + \frac{3}{32} \cdot 2 + \frac{3}{32} \cdot 3 + \frac{1}{32} \cdot 4 = 1$, čo by znamenalo, že spätné točenie nevyprodukuje viac stavov v priemer. No ako sa ukazuje, vnútorné stavy nie su uniformne distribuované v reálnom útoku a ako výsledok, $\frac{13}{32}$ stavov má mapovanie jedna k jednej, $\frac{3 \cdot 2}{32}$ stavov reprodukuje dva stavy každý, $\frac{3 \cdot 3}{32}$ stavov reprodukuje tri stavy každý a $\frac{1 \cdot 4}{32}$ reprodukuje 4 stavy každý. Očakávaná hodnota jedného spätného točenia je potom $\frac{13}{32} \cdot 1 + \frac{6}{32} \cdot 2 + \frac{9}{32} \cdot 3 + \frac{4}{32} \cdot 4 = 2,15$ [32]. Ak točenie šifry A5/1 o jeden bit naviac zvýši maximálny počet riešení o 3, očakávaný počet riešení pri r točeniach späť môže byť reprezentovaný ako $\sum_{i=1}^{3r+1} P_r(i) \cdot i^2$, kde $P_r(i)$ je pravdepodobnosť počtu riešení a druhá mocnina vyplýva z neuniformnej distribúcie zobrazenia medzi stavmi pred a po točení. Pre trojkolové spätné točenie, je očakávaná hodnota

vypočítaná ako $\sum_{i=1}^{4} P_1(i) * i^2 = 2.12$, $\sum_{i=1}^{7} P_2(i) * i^2 = 2.5$, $\sum_{i=1}^{10} P_3(i) * i^2 = 2.65$, čo aj splňuje experimentálne výsledky [32].

Ďalšie aproximovanie funkcie P_i je len experimentálne, pretože je veľmi zdĺhavé, preto tieto výsledky boli vypočítané experimentálne. Pre nás je zaujímavou hodnotou spätné točenie o 100 cyklov, kde dosahuje funkcia $P_100(0)$ hodnotu približne $84,7\%$, z čoho vyplýva, že približný počet riešení pri 100 spätných točení je približne 14 riešení [32]. Z daného aj vyplýva, že po zamiešaní kľúča, čo je fáza v ktorej sa 100 krát šifra točí bez výstupu je redukovaný o približne $1 - P_100(0) = 15.3\%$, z daného vyplýva, že celkový priestor kľúča je v šifre A5/1 $2^{61,29}$. To je aj ideálna veľkosť kľúčového stavu šifry A5/1 [32].



(a) (b) (c)

Obr. 8: Pomer korektných a nekorektných stavov (a), očakávaný počet počiatočných stavov (b) výskyt riešení (c) [32] .

Pre bližší popis toho, ako implementovať spätné točenie odporúčame článok [6, **?**], ktorý sa venuje problematike zostrojenia rovníc. V našom prípade použijeme projekt KRAKEN, ktorý má spätné točenie šifry A5/1 implementované. Vlastným experimentom sme zistili, že spätné točenie šifry A5/1 o 100 bitov trvá približne 10 ms na bežnom osobnom počítači, s nijak neoptimalizovaným kódom. Spustili sme 10 000 výpočtov a v prieme dĺžka jedného výpočtu bola okolo 12 ms, pričom priemerný počet riešení pre ľubovoľný stav po fáze mixovania kľúčom bol 13,5.

2.5 Praktická implementácia

V tejto sekcii popíšeme praktickú implementáciu útoku a dúhové tabuľky, ktoré boli generované v roku 2010 [30, 20]. V podstate v tejto kapitole zhrnieme to, ako boli prvé a posledné verejné dúhové tabuľky vygenerované a ako je možné z praktického hľadiska implementovať útok na túto šifru.

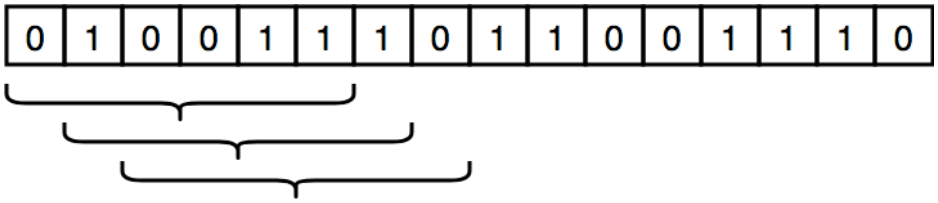
2.6 Zverejnené dúhové tabuľky

V roku 2010 boli zverejnené prvé a posledné dúhové tabuľky [30]. Okrem týchto tabuliek existujú aj rôzne články, ktoré sa venujú generovaniu týchto tabuliek [28, 32], ale ich dúhové tabuľky zverejnené neboli. Ako význačný bod bola zvolená postupnosť núl na konci výstupu zo šifry A5/1 a to tak, že z výstupu funkcie f musí tvoriť na konci reťazca 16 núl. Ako f funkcia bola zvolená funkcia po mixovaní kľúča, to znamená, že v bode kedy je kľúč aj frame number vložený do šifry, následne tá je zatočená o 100 krát bez výstupu s aktivovanou minoritnou funkciou. Celková veľkosť takýchto tabuliek, ak by sme nepoužili žiadny význačný bod, by bola $2^64 \cdot 8 \cdot 2 = 128$ exabajtov. Tým, že veľkosť kľúča vieme zmenšiť, ako je aj popísané v predchádzajúcej kapitole o 3 bity, dostávame veľkosť 18 exabajtov. Následne bola aplikovaná funkcia f pričom jej dĺžka je 2^16, čo vie zredukovať celkovú veľkosť 2^16 násobne menšiu [20]. Teraz celková kapacita tvorí len 4.5 petabajtov, pričom ak zavedieme 8 rozdielnych farieb, dostávame sa na 563 terabajtov. Takúto kapacitu vedia zabezpečiť výkonné superpočítače. Ak ale zredukujeme pokrytie len na 0.5% celkového priestorového stavu, výsledná kapacita je okolo 2.8 terabajtov. Redukciou ukladaných dát, to znamená, že koncové nuly nepotrebujeme ukladať (koncové nuly v rámci koncových bodov) sa dostávame na 1.6 terabajtov. Tieto dúhové tabuľky boli zverejnené projektom autora [30] a sú voľne k dispozícií pre širokú verejnosť. Zdieľajú sa pomocou služby torrent. Tabuľky sú uložené v binárnej forme a my ich máme aj k dispozícií. Ďalej útok s využitím týchto tabuliek implementuje projekt KRAKEN, v ktorom je možné využiť metódy spätného točenia, taktiež implementuje aj prehľadávanie tabuliek a z postupnosti bajtov vie na základe toho, ako boli dúhové tabuľky generované prejsť ku koncovému bodu.

Obr. 9: Schéma dúhových tabuliek z roku 2010.

Ako sme si mohli všimnúť, tabuľky obsahujú len 0.5% pokrytie, čo sa môže na prvý pohľad zdáť málo. Na druhej strane, my sme popísali v prvej kapitole to, že šifra A5/1 sa používa tak, že generuje výstup o veľkosti 114 bitov, pričom v našom prípade je

potrebných len 64 bitov (64 bitov je veľkosť vnútorného stavu). Tým pádom je možné využiť takzvané posuvné okno, ktoré zvyšuje šancu nájdenia kľúča pri týchto tabuľkách na 50% [20]. Totižto ak zvolíme len prvých 64 bitov textu, ktorý chceme prelomiť, už len táto časť nám generuje dostatočný výstup, ktorý je jednoznačne (aspoň ak je šifra uniformná) určený nejakým kľúčom. Ak sa nám nepodarí v tých 64 bitov nájsť daný kľúč pomocou dúhovej tabuľky, zvolíme bity od 2 po 65. Správa, ktorá je šifrovaná šifrou A5/1 je o veľkosti 114 bitov. Z daného vyplýva, že pri každom útoku máme $114 - 64 + 1 = 51$ možností. Tým pádom je pravdepodobnosť takého útoku podľa [30, 20] zvýšená až na 50% len pri 0.5% pokrytí. Obrázok 10 ukazuje návrh takého systému. Preto budeme potrebovať s pribúdajúcim posunom točiť so šifrou A5/1 o posun naviac [30].



Obr. 10: Posuvné okno. Kryptoanalýza šifry A5/1 bude skúmať len časť šifrovaného textu.

Ďalej projekty implementujú aj efektívny spôsob ukladania dúhových tabuliek na disk a ich efektívne prehľadávanie [20]. Taktiež samotný algoritmus šifry A5/1 je vylepšený a nejde len o jednoduché točenie šifry A5/1 vopred, pričom sa implementujú rôzne vylepšenia, mechanizmy na to, aby prehľadávanie bolo čo najrýchlejšie. Týmto veciam sa ďalej venovať nebudeme, keďže sú skôr len implementačného charakteru.

2.7 Postup útoku

Keďže sieť GSM je známa svojím správami, ktoré často musia BTS stanice dookola opakovať, ako napríklad správy typu: Som nemenovaný operátor! [35, 20]. Tieto správy slúžia na to, aby mobilné zariadenia, ktoré sú v dosahu signálu, vedeli o tom, že sú pripojené, respektíve že je možnosť sa pripojiť do danej siete. Tieto správy sú vysielané veľmi často, experimentálne overenia hovoria o tom, že ich frekvencia dosahuje viac ako 50%. Okrem toho, pre každé zariadenie, ktoré je pripojené k sieti a BTS stanica mu nemá čo povedať, tak dané zariadenie vysiela správu NOP - správa, ktorá neobsahuje žiadny text, väčšinou s fixným dorovnaním do veľkosti správy (napríklad: NOPx35a8c6d3f1, kde koncovka za NOP zostáva rovnaká) a tieto správy posiela BTS stanica mobilným zariadeniam taktiež na to, aby mobilné zariadenia vedeli, že majú

signál a posiela ich zašifrovane. Pričom šifra A5/1 funguje tak, že vygeneruje postupnosť bitov, ktoré sa funkciou XOR pridajú k správe, ktorú chceme zachytiť. Tým pádom, ak budeme predpokladať, že daná správa je správa typu NOP, využitím XOR funkcie a jej vlastnosti: ak platí: $x \oplus y = z$, potom $y = z \oplus x$ vieme predpokladať, aký výstup bol vygenerovaný zo šifry A5/1. Nakoniec na celý problém aplikujeme dúhové tabuľky a vypočítame, ako sme to popísali vyššie, tajný kľúč správy, ktorý vieme použiť na dešifrovanie celej nasledovnej komunikácie.

2.7.1 Zachytenie mobilnej komunikácie

Posledný problém, ktorý potrebujeme vyriešiť je, ako je možné zachytiť mobilnú komunikáciu. Na to existuje niekoľko rôznych možností [20]:

1. Využitím RTL-SDR: RTL-SDR je softvérovo definované rádio, pričom bežná cena sa pohybuje okolo 20 Amerických dolárov. Ide o zariadenie, ktoré vieme nastaviť na ľubovoľnú frekvenciu a vie spracovávať digitálny výstup. RTL-SDR sú pasívne zariadenia, ktoré vedia signál len prijímať. Niekoľko projektov, ako získavať dáta z mobilných sietí je popísaných v týchto odkazoch [10, 3].

2. Využitím projektu hackRT: hackRT je aktívne zariadenie a je možné ho využiť aj na simulovanie falošnej BTS stanice. Cena zariadenia sa pohybuje okolo 300 Amerických dolárov

3. Využitím mobilných telefónov s OsmocomBB (napríklad Motorola C115). Projekt OsmocomBB je práve určený na slobodný softvér, ktorý ma otvorený zdrojový kód. V minulosti boli popísané simulácie BTS stanice práve s použitím 5 mobilných telefónov Motorola s operačným systémom OsmocomBB [33].

Na simuláciu alebo aj na odpočúvanie mobilnej komunikácie potrebujeme pokryť určité frekvenčné pásmo [26]. Pretože mobilné siete fungujú na princípe skákajúcej frekvencie. Tento prvok nie je bezpečnostný, ale je to skôr vlastnosťou zabezpečujúcou lepšie pokrytie, respektíve signál [20]. GSM frekvencia je definovaná niekoľkými kanálmi, a z tohto dôvodu nie je postačujúci napríklad iba jeden mobilný telefón alebo len jedno zariadenie typu RTL-SDR na pokrytie celého pásma. Skoky vo frekvenciách sa vyskytujú v pomerne veľkej miere, a preto je potrebné skenovanie celého pásma. Ak jeden hovor, ktorý chceme odpočúvať je potrebné prelomiť, potrebujeme mať skenované celé frekvenčné pásmo, a v prípade úspechu prelomenia budeme následne vedieť aj ďalšie, respektíve nasledujúce frekvencie. Identifikovať mobilné zariadenie, ktoré chceme odpočúvať je pomerne veľký problém. Keďže mobilné zariadenie sa identifikuje len pri prvom prihlásení k BTS stanici a aj to iba pomocou takzvaného IMSI

čísla, potrebujeme vedieť IMSI číslo obete. Okrem daného problému je potrebné pri domácej implementácií vyriešiť problém toho, aby sme nezískali, respektíve neprelomili dáta iného účastníka GSM komunikácia, pretože v tomto prípade sa jedna o trestný čin. Či je samotné odpočúvanie samého seba trestným činom sme sa doposiaľ nedozvedeli zo žiadnych dostupných zdrojov (žiadosť o informáciu, štúdium zákona). Práve z týchto dôvodov sme sa rozhodli, že ďalej sa nebudeme venovať praktickej implementácií a možnosti skenovania mobilných sieti.

3 Kryptoanalýza šifry KASUMI

V tejto kapitole sa pozrieme na to, ako je možné útočiť na šifru KASUMI, ktorá sa taktiež používa v mobilných sieťach tretej generácie. Šifra KASUMI sa taktiež ukázala ako prelomená ale len na teoretickej báze. Všetky útoky vyžadujú zašifrovanie obrovského množstva textu rovnakým, respektíve veľmi podobným kľúčom [15]. Okrem toho, pri kryptoanalýze sa využíva taktiež diferencia medzi otvorenými textami, ktorú v reálnych podmienkach nevieme dosiahnuť. No na druhej strane, ani v prípade kryptoanalýzi šifry A5/1 sme spočiatku nevedeli, ako je možné túto šifru prelomiť, útoky existovali len v teoretickej rovine. Je možno aj dobre spomenúť, že práve kryptoanalýza šifry KASUMI nie je pri možnostiach odpočúvania veľmi dôležitá. Existujú rušičky signálu, pomocou ktorých vieme v cieľovej oblasti dosiahnuť rušenie signálu 3G a vzhľadom na spätnú kompatibilitu mobilných zariadení a BTS staníc, vieme prejsť na kryptoanalýzu v reálnom čase šifry A5/1. Okrem toho, perličkou je, že šifra KASUMI a šifra A5/1 používajú rovnaký kľúč. V prípade šifry KASUMI ide len o to, že tento kľúč z pôvodných 64 bitov, aký používa šifra A5/1 je dopočítaný expanziou na 128 bitov. Pritom nám bude stačiť prelomiť šifru A5/1 na to, aby sme vedeli a získali tajný kľúč, následne môžeme odblokovať rušenú frekvenciu a pokračovať v odpočúvaní [20].

Na začiatku tejto kapitoly sa pozrieme na to, čo vlastne diferenčná kryptoanalýza je a následne ukážeme možnosti útoku na šifru pomocou takzvaného bumerangového útoku s príbuzným kľúčom. Názvy útokov budem v nasledujúcej kapitole uvádzať pôvodne, v anglickom znení.

3.1 Diferečná kryptoanalýza

V nasledujúcej časti veľmi stručne popíšeme diferenčnú kryptoanalýzu, jej vlastnosti a metódy kryptoanalýzy pomocou diferenčnej kryptoanalýzy. Vzhľadom na to, že budeme pracovať s knihou autora [15], budeme sa držať aj jeho popisu diferenčnej kryptoanalýzy, ktorá nie je úplne jednoduchá pre bežného čítateľa, ktorý sa s diferenčnou kryptoanalýzou ešte nestretol. Týmto preto odporúčame preštudovať literatúru [25],

kde je detailnejšie vysvetlená diferenčná kryptoanalýza na príkladoch.

Diferenčná kryptoanalýza odhaľuje vzťah medzi šifrovaním dvoch otvorených textov P_1 a P_2 pod rovnakým kľúčom K [5]. Ak vezmeme ľubovoľnú šifru, ktorá používa funkciu XOR pri pridávaní kľúča (napríklad šifra DES vo funkcií F) a spustíme šifrovanie dvakrát s rozdielnymi vstupmi P_1 a P_2 pod rovnakým kľúčom K. Ak je diferencia medzi dvoma časťami šifrovania $R \oplus R^*$ známa, potom platí, že diferencia medzi $E(R)$ a $E(R^*)$ je taktiež známa ako $(E(R))' = E(R) \oplus E(R^*) = E(R')$ [15], pretože funkcia XOR s rovnakou konštantou K_r (kľúč kola) nemá význam na diferenciu $(E(R))' = E(R')$.

Ak sa následne aplikuje S-box, ktorý nie je lineárny, tak diferencie, ktoré do S-boxu vstúpia môžu mať rôzne výstupy (ak vezmeme v úvahu nejakú distribúciu). Definujeme nasledovne diferenčnú distribučnú tabuľku S-boxu ako tabuľku, ktorá počíta pre každú vstupnú a výstupnú diferenciu daného S-boxu, koľko párov s danou vstupnou diferenciou má výstupnú diferenciu [15]. Poznamenajme, ak máme vstupnú diferenciu rovnú 0, potom aj výstupná diferencia je rovná 0. Ak je daná vstupná diferencia X' a výstupná diferencia Y', môžeme vypočítať pravdepodobnosť nasledujúceho zobrazenia $X' \rightarrow Y'$. Výpočet je vykonaný pre každý jeden S-box, ktorý sa v danej šifre nachádza (rozdielný).

3.1.1 Diferenčná charakteristika

Jednokolová diferenčná charakteristika $X' \rightarrow Y'$ s pravdepodobnosťou p je predpoveď, že vstupná diferencia X' sa stane výstupnou diferenciou Y' s pravdepodobnosťou p po jednom kole šifrovania. Dve jednokolové charakteristiky môžu byť spojené pomocou podmienky, že výstupná diferencia je nasledujúcou vstupnou diferenciou [15]. Ak sa v šifrovaní používa Feistelová sieť, ktorá prehadzuje vstupy medzi kolami, potom podmienkou na spojenie dvoch diferencií je, že výstupná diferencia z ľavej strany je vstupnou diferenciou do pravej strany Feistelovej siete. Takýmto spôsobom je možné vytvárať charakteristiky aj pre dlhé diferencie, respektíve pre celé šifrovanie [15]. Ak budeme predpokladať, že kola sú nezávislé, potom pravdepodobnosť spájania charakteristík je produktom všetkých pravdepodobností jednokolových charakteristík. Treba ale poznamenať, že vo väčšine šifier jednotlivé kolá nie sú nezávislé [15, 5, 25], vzhľadom na vzťah medzi kľúčmi, v mnoho prípadoch je pravdepodobnosť charakteristiky aproximovateľná produktom jednokolovej charakteristiky.

Taktiež je možné definovať *diferencie* podľa [15]. Diferencia je trojica skladajúca sa zo vstupnej diferencie Ω_P, výstupnej diferencie Ω_T a pravdepodobnosti, že pár so vstupnou diferenciou (v tomto prípade rozlišuj diferenciu ako rozdiel medzi dvoma reťazcami) Ω_P má výstupnú diferenciu Ω_T. Pravdepodobnosť týchto diferencií je suma pravdepodobností pre každú charakteristiku, ktorá zdieľa danú diferenciu Ω_P a Ω_T.

3.1.2 Filtrovanie

Základnou myšlienkou filtrovania je nájsť a identifikovať štatisticky málo bežné distribúcie v diferenciách, ktoré sa vyskytujú. To je práve ten signál, ktorý chceme detekovať a ktorý môže byť maskovaný pármi, ktoré nespĺňajú diferenciu [25]. Je potrebné eliminovať nesprávne páry čim skôr, aby neskôr nezvyšovali časovú zložitosť útoku. Identifikovať nesprávne páry vyžaduje skontrolovať, či daná diferencia je dodržaná alebo nie. Predpokladajme, že máme nasledujúcu charakteristiku $(0, 0, 2, 0) \xrightarrow{S} (0, 0, h, 0)$ pre posledné kolo, kde h môže nadobúdať nasledujúce hodnoty: $\{1, 2, 9, a\}$. Predpokladajme, že používame šifru takú, že posledné kolo neobsahuje žiadnu permutáciu, potom tieto štyri hodnoty musia byť aj výstup z danej šifry. Potom útočník môže preskúmať jednotlivé výstupné páry a okamžite rozhodnúť, či daný pár je zlý pár alebo či je to potencionálny kandidát na vhodnú diferenciu [25]. Tým je dôležité si uvedomiť, že mi budeme len predpokladať, že daný text bude spĺňať diferenciu - on v podstate nemusí spĺňať diferenciu na konci nášho odhadu - stále pracujeme len s nejakou pravdepodobnosťou.

3.1.3 Použitie správnej diferencie pre nájdenie kľúča

Správny kľúč, ktorý spĺňa diferenciu (je ich niekoľko, keďže existuje pravdepodobnosť, s akou sa daná diferencia vyskytne, preto potrebujeme nejakú množinu kľúčov a nie každý pár musí spĺňať danú diferenciu) obsahuje informáciu o kľúči [15]. Informácia môže byť získaná preskúmaním jednotlivých hodnôt, ktoré vstupujú do kôl šifrovania. Hlavná myšlienka analýzy kľúča a jeho spätného získania tkvie v tom, že ak výstupná diferencia funkcie v danom kole je taká, ako je predpokladaná charakteristikou, potom aktuálny vstup do nelineárnej časti je známy alebo je aspoň zmenšení na nejakú menšiu množinu. Preto práve získanie kľúča sa deje v kolách, kde je známy aktuálny vstup [15].

Ak nájdeme správny pár, ktorý spĺňa diferenciu, bude tento pár stále ukazovať na správny bit kľúča, ktorý chceme prelomiť. V podstate vo väčšine útokov je nakoniec časť kľúča v danom kole, na ktoré aplikujeme diferenčnú kryptoanalýzu získaná tak, že budeme skúšať jednotlivé kľúče zaradom (závisí od druhu diferenčnej kryptoanalýzy) hrubou silou a tie kľúče, ktoré budú spĺňať podmienku, budú správne. Ak je kľúč krátky, ale je použitých niekoľko krátkych kľúčov, diferenčná kryptoanalýza takýmto štýlom bude fungovať veľmi efektívne.

3.2 Boomerang attack

Boomerang attack bol prvý krát popísaný Wagnerom [36]. Základná myšlienka je použitie krátkych diferencií (dvoch krátkych diferencií konkrétne) s vysokou pravdepodobnosťou namiesto jednej diferencie viacerých kôl so slabou pravdepodobnosťou.

Predpokladajme, že máme blokovú šifru $E : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$, ktorá môže byť popísaná kaskádovito ako $E = E_1 \circ E_0$, tak, že pre E_0 existuje diferencia $\alpha \to \beta$ s pravdepodobnosťou p a pre E_1 existuje diferencia $\gamma \to \delta$ s pravdepodobnosťou q. Boomgerand attack používa prvú diferenciu $\alpha \to \beta$ pre E_0 vzhľadom na páry (P_1, P_2) a (P_3, P_4) a používa druhú diferenciu $\gamma \to \delta$ pre E_1 na páry C_1, C_3 a (C_2, C_4) [36]. Útok je založený na nasledujúcom procese [15]:

1. Požiadaj o zašifrovanie párov otvoreného textu (P_1, P_2) také, že platí $P_1 \oplus P_2 = \alpha$ a označíme korešpondujúci šifrovaný text (C_1, C_2).

2. Vypočítaj $C_3 = C_1 \oplus \delta$ a $C_4 = C_2 \oplus \delta$ a požiadaj o dešifrovanie párov $(C_3, C4)$. Označíme tento otvorený text (P_3, P_4).

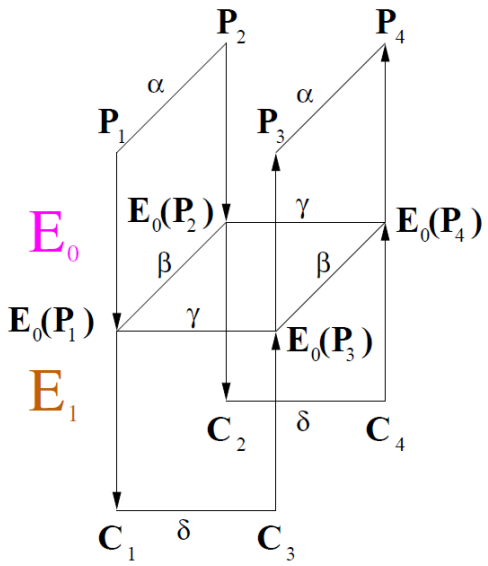
3. Skontroluj, či $P_3 \oplus P_4 = \alpha$.

Takýto postup sa nazýva aj δ-posun, pretože každý pár je zašifrovaný a následne posunutý o δ, potom dešifrovaný. Príklad takého postupu je znázornený na obrázku 11.

Jednotlivé pravdepodobnosti, že pár (P_1, P_2) je správny vzhľadom na diferenciu $\alpha \to \beta$ je p. Pravdepodobnosť, že obidva páry (C_1, C_3) a (C_2, C_4) sú správne páry vzhľadom na druhú diferenciu je q^2 [15] (pretože pravdepodobnosť pre jeden pár je q a predpokladáme uniformnosť - pretože počítame maximálnu najhoršiu pravdepodobnosť). Ak sú všetky páry správne, potom platí $E_1^{-1}(C_3) \oplus E_1^{-1}(C_4) = \beta = E_0(P_3) \oplus E_0(P_4)$ a s pravdepodobnosťou p taktiež platí, že $P_3 \oplus P_4 = \alpha$. Celková pravdepodobnosť, aby splnila Boomerang attack charakteristiku je potom $(pq)^2$. Ak celkovo platí, že $pq > 2^{\frac{-n}{2}}$, Boomerang attack bude fungovať (znamená to, že bude efektívnejší, časovo jednoduchší, pretože n je vlastne veľkosť priestoru vstupu do šifry).

Ďalším dôležitým prvkom je, že diferenčná kryptoanalýza takého charakteru môže byť vedená pre všetky možné β a δ, pokiaľ platí, že $\beta \neq \delta$. Celková pravdepodobnosť je potom $(\hat{p}\hat{q})^2$ [15]:

$$\hat{p} = \sqrt{\sum_{\substack{\beta \\ \alpha \to \beta}} \Pr^2[\alpha \to \beta]}, \quad \text{a} \quad \hat{q} = \sqrt{\sum_{\substack{\gamma \\ \gamma \to \delta}} \Pr^2[\gamma \to \delta]}$$



Obr. 11: Schéma Boomerang attack [15].

Boomerang attack je relatívne efektívny keď $\hat{p}$ a $\hat{q}$ sú relatívne veľké vzhľadom na odmocninu najlepšej diferencie [15].

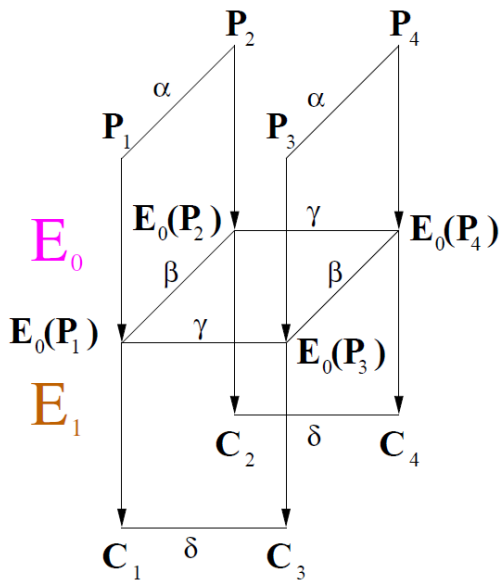
3.2.1 Amplified Boomerang attack

Tento útok je veľmi podobný Boomerang attack. V podstate ide o ten istý útok, len s tým rozdielom, že nebudeme dešifrovať zašifrovaný text ale použijeme šifrovanie niekoľkých textov. Útok vytvára štvorice otvorených textov $((P_1, P_2)(P_3, P_4))$ ktoré spĺňajú niekoľko podmienok. Útok začína s otvorenými textami so vstupnou diferenciou α, t.j. $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$ [15]. Každý z týchto párov má pravdepodobnosť p na splnenie diferencie $\alpha \rightarrow \beta$ v E_0 [15]. Ak obidva páry spĺňajú diferenciu $E_0(P_1) \oplus E_0(P_2) = E_0(P_3) \oplus E_0(P_4) = \beta$ a potom $E_0(P_1) \oplus E_(P_3) = (E_0(P_1) \oplus \beta) \oplus (E_0(P_3) \oplus \beta) = E_0(P_2) \oplus E_0(P_4)$ [15]. Ak diferencia $E_0(P_1) \oplus E_0(P_3)$ je rovná γ, potom aj diferencia $E_0(P_2) \oplus E_0(P_4)$ [15]. Každý z týchto dvoch nových párov má pravdepodobnosť q aby sa stal správnym párom vzhľadom k druhej diferencií $\gamma \rightarrow \delta$, to znamená, že platí $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$. Štvorica splňujúca každú z týchto podmienok je nazývaná správnym párom. Schéma takého útoku je znázornená na obrázku 11. Analýza pravdepodobností je k dispozícii v [15], celkový počet očakávaných párov je:

$$\binom{Np}{2} \cdot 2^{-n} \cdot q^2 = N^2 \cdot 2^{-n-1} \cdot (pq)^2$$

kde N označuje počet štartovacích párov $(P_1, P_2), (P_3, P_4)$ [15]. Pre náhodnú permutáciu (náhodné hodnoty α a δ), očakávaný počet štvorí je $N^2 \cdot 2^{-2n}$, pretože existuje

práve N^2 kandidátov štvoríc a pre každý pár (C_1, C_3) alebo (C_2, C_4) je pravdepodobnosť, že budeme mať špecifickú diferenciu na výstupe 2^{-n} [15]. Teda, ak to celé zhrniem, potom pokiaľ $pq > 2^{\frac{-n}{2}+\frac{1}{2}}$, existuje viac zlých štvoríc, ktoré spĺňajú podmienku ako správnych štvoríc.



Obr. 12: Schéma Amplified Boomerang attack [15].

3.3 Rectangle attack

Rectangle attack je vylepšením takzvaného Amplified Boomerang attack. Správna štvorica je definovaná ako štvorica otvorených textov $((P_1, P_2), (P_3, P_4))$ a korešpondujúcich šifrovaných textov $((C_1, C_2), (C_3, C_4))$, ktorá spĺňa podmienky

$$P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha \quad \text{a} \quad C_1 \oplus C_3 = C_2 \oplus C_4 = \delta,$$

kde α je vstupná diferencia ktorá vstupuje do šifry v časti E_0 a δ je výstupná diferencia E_1 [15]. Tieto podmienky označujeme ako podmienky rectangle útoku. Pre ďalšiu analýzu definujeme čiastočné šifrovanie P_i označením Z_i, to znamená, že $Z_i = E_0(P_i)$. Poznamenajme, že $C_i = E_i(Z_i)$.

Ďalej popíšeme vylepšenia daného útoku. Medzi tieto vylepšenia patrí analýza E_1. Namiesto toho, aby sme používali iba hodnoty γ, môžeme použiť ľubovoľné $\gamma' \to \delta$ v E_1 [15]. Pretože špecifická hodnota diferencie σ' je bezvýznamná, pokiaľ obidva páry (Z_1, Z_3) a (Z_2, Z_4) majú ľubovoľnú rovnakú diferenciu σ'. V tomto prípade, potom ale platí, že ak $Z_1 \oplus Z_3 = Z_2 \oplus Z_4 = \sigma'$ a $Z_1 \oplus Z_3 = \beta$, pravdepodobnosť, že podmienky Amplified Boomerang attack sú splnené je [15]

$$\frac{1}{2^n}p^2 \sum_{\substack{E_1 \\ \gamma \to \delta}} \Pr^2[\sigma' \xrightarrow{E_1} \delta] = \frac{p^2\hat{q}^2}{2^n}.$$

Druhé vylepšenie spočíva v tom, že je možné použiť všetky β' hodnoty naraz súčasne. Pokiaľ platí, že $Z_1 \oplus Z_2 = Z_3 \oplus Z_4$ a $Z_1 \oplus Z_3 = \gamma'$, štvorice majú pravdepodobnosť $\Pr^2[\gamma' \to \delta]$, aby sa stali správnymi štvoricami, to znamená takými štvoricami, ktoré splňujú diferenciu. Potom celková pravdepodobnosť spolu s kombináciou z predchádzajúcej rovnice pri uvažovaní uniformného rozdelenia je [15]

$$\frac{1}{2^n} \sum_{\substack{E_0 \\ \alpha \to \beta'}} \Pr^2[\alpha' \xrightarrow{E_0} \beta'] \cdot \sum_{\substack{E_1 \\ \gamma \to \delta}} \Pr^2[\sigma' \xrightarrow{E_1} \delta] = \frac{\hat{p}^2\hat{q}^2}{2^n}.$$

Tretie vylepšenie pri danom útoku spočíva v tom, že v Boomerang attack poznáme ktorý šifrovaný text odpovedá C_1, pretože C_3 je vypočítaný z C_1 [15]. V Rectangle attack je to trochu inak. Pre každý pár párov sú dve možnosti štvoríc, a to: $((P_1, P_2), (P_3, P_4))$ a $((P_1, P_2), (P_4, P_3))$ [15]. Môžeme vlastne testovať každý pár párov $((P_1, P_2), (P_3, P_4))$ či platí, že $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$. Ak tento test neplatí, potom môžeme otestovať druhú dvojicu, a to $((P_1, P_2), (P_4, P_3))$, či platí, že $C_1 \oplus C_4 = C_2 \oplus C_3 = \delta$. Podľa toho bola vytvorená analýza na náhodných dátach, ktorá podľa [22] ukázala, že ak máme daných N otvorených textov, môžeme očakávať práve $\frac{N^2(\hat{p}\hat{q})^2}{2^n}$ správnych štvoríc.

3.3.1 Získanie kľúča

Jedným z hlavných problémov pri tomto útoku je, ako je možné z daných štvoríc získať práve správnu štvoricu. V rámci Boomerang attack sme začali s nejakým otvoreným textom, ktorý sme zašifrovali, vytvorili sme nejaký posun a dešifrovali. Nakoniec sme skontrolovali, že či nový otvorený text po dešifrovaní spĺňa diferenciu, respektíve podmienku Boomerang attack. Pri Rectangle útoku ale máme k dispozícií niekoľko otvorených textov a hľadáme štvoricu, ktorá spĺňa podmienky Rectangle attack. Tieto podmienky môžu byť otestované, iba keď všetky otvorené texty a šifrované texty štvoríc sú dané. Nemôžeme okamžite po tom, ako v prípade Boomerang attack, po zašifrovaní, respektíve dešifrovaní otvoreného textu skontrolovať, či daný text spĺňa diferenciu, keďže v rámci daného útoku vieme prehadzovať texty a práve veľké množstvo dát a náročnosť v identifikovaní správnych štvoríc robí Rectangle attack útok viac zložitým ako jednoduchý Boomerang attack.

Získanie kľúča prebieha nasledovným postupom podľa [15], kde jednotlivé označenia sú vysvetlené v tabuľke číslo 4, vzhľadom na šifru typu $E = E_b \circ E_0 \circ E_1 \circ E_f$:

1. Vygeneruj $S = 2^{\frac{n}{4-r_b}}/\hat{p}\hat{q}$ štruktúry 2^{r_b} otvorených nešifrovaných textov. Štruktúry sú vybudované vybratím náhodného P_0 tak, že sa položí $P_0 \oplus V_b$.

2. Inicializuj pole o veľkosti $2^{m_b+m_f}$. Každý prvok poľa odpovedá rozdielnej predpokladanej, respektíve uhádnutej hodnote podkľúča K_b a K_f.

3. Vlož všetky $N = 2^{n/4}/\hat{p}\hat{q}$ šifrované texty do hešovacej tabuľky indexovanej podľa $n - r_f$ šifrovaných bitov textu, ktorých diferencie sú konštantné vzhľadom na V_r. Ak nastane kolízia, skontroluj diferenciu párov šifrovaného textu v X_f, a ak je to tento prípad, pokračuj s daným párom ďalej.

4. Označme štruktúru, v ktorej je šifrovaný text C_i z množiny štruktúr S ako S_{C_i}. Pre každý pár (C_1, C_3), ktorý spĺňa podmienku $C_1 \oplus C_3 \in X_f$, ak $S_{C_3} > S_{C_1}$ pripoj index S_{C_3} k C_1. Inač pripoj index S_{C_1} k C_3.

5. V každej štruktúre S_i, hľadaj dve šifrované texty C_1 a C_2, pre ktoré existuje iná štruktúra S_j, ktorej index je pripojený k C_1 a k C_2. Pre taký pár skontroluj, či existuje $P_1 \oplus P_2 \in X_b$ a ak áno, preskúmaj šifrovaný text, ktorý sa stretol s C_1 aj s C_2 (samozrejme to platí aj pre C_3 a C_4, ktoré označíme ako C_1 a C_2). Ak otvorené texty P_3, P_4 taktiež spĺňajú podmienku $P_3 \oplus P_4 \in X_b$, pokračuj analýzou štvorice $((P_1, P_2), (P_3, P_4))$

6. Pre každú ostávajúcu štruktúru, ktorá spĺňa predchádzajúcu podmienku, $((P_1, P_2), (P_3, P_4))$ a každú možnú hodnotu $k_b \in K_b$ a $k_f \in K_f$ skontroluj, či platí $E_{b_{k_b}}(P_1) \oplus E_{b_{k_b}}(P_2) = E_{b_{k_b}}(P_3) \oplus E_{b_{k_b}}(P_4) = \alpha$ a $E^{-1}_{f_{k_f}}(C_1) \oplus E^{-1}_{f_{k_f}}(C_3) = E^{-1}_{f_{k_f}}(C_2) \oplus E^{-1}_{f_{k_f}}(C_4) = \delta$. Ak je daná podmienka splnená, zvýš počítadlo v poly na odpovedajúce hodnoty k_b, k_f.

7. Na výstup daj tú hodnotu z poľa, ktorej počítadlo je maximálne.

   Časová zložitosť daného algoritmu je podľa [15]

$$2^{m_b+m_f} + N + N^2 2^{r_f-n-1} + N^2 2^{t_f-n-1} + N^2 2^{2t_f+2r_b-2n-3} +$$
$$+ N^2 2^{-2n-2}(2^{m_b+t_b+2t_f} + 2^{m_f+t_f+2t_b} + 2^{m_b+m_f})$$

   Pri použití $2^{r_b} S = 2^{r_b} 2^{n/2+2-rb}/\hat{p}/\hat{q}$ otvorených textov s celkovou pamäťovou zložitosťou $N + 2^{t_b} + 2^{t_f} + 2^{m_b} + 2^{m_f}$ [15].

Podšifra	Označenie	Význam
E_i	X_i	Množina diferencií, ktoré môžu spôsobiť vstupnú diferenciu vzhľadom na definíciu po kole E_i
	t_i	$\log_2 \mid X_i \mid$
	V_i	Uzáver množiny X_b vzhľadom na funkciu XOR
	r_i	$\log_2 \mid V_i \mid$
	K_i	Množina všetkých možných hodnôt v podkľúči, ktorá vplýva na diferenciu po kole E_i
	m_i	$\log_2 \mid K_i \mid$

Tabuľka 4: Prehľad značenia [15].

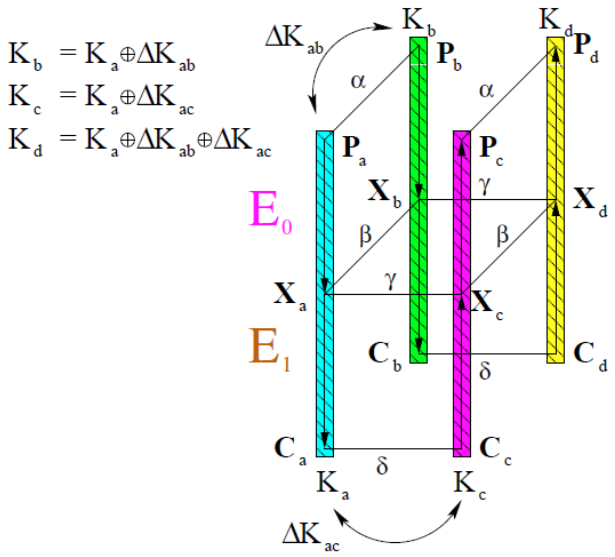
3.4 Related key Rectangle Boomerang attack

Na záver popíšeme posledný útok, ktorý sa využije na kryptoanalýzu šifry KASUMI. Predpokladajme, že šifru vieme rozdeliť na časti E_0 a E_1 a používame diferenciu $\alpha \rightarrow \beta$ pri šifrovaný pomocou časti E_0 s diferenciou v kľúči ΔK_{ab} s pravdepodobnosťou p. V časti E_1 používame druhú diferenciu $\gamma \rightarrow \delta$ pod diferenciou v kľúči ΔK_{ac} s pravdepodobnosťou q [15]. Related key útok používa 4 neznáme kľúče, ktoré sú ale na sebe závislé: $K_a, K_b = K_a \oplus \Delta K_{ab}, K_c = K_a \oplus \Delta K_{ac}$ a $K_d = K_a \oplus \Delta K_{ab} \oplus \Delta K_{ac}$ [15]. Útok prebieha nasledovne:

1. Vyber N otvorených textov $(P_a, P_b = P_a \oplus \alpha)$ náhodné a požiadaj o zašifrovanie P_a pod kľúčom K_a a P_b pod kľúčom K_b.

2. Vyber N otvorených textov $(P_c, P_d = P_c \oplus \alpha)$ náhodné a požiadaj o zašifrovanie P_c pod kľúčom K_c a P_d pod kľúčom K_d.

3. Nájdi štvorice otvorených textov (P_a, P_b, P_c, P_d) a odpovedajúce šifrované texty (C_a, C_b, C_c, C_d) spĺňujúce $C_a \oplus C_c = C_b \oplus C_d = \delta$.



Obr. 13: Schéma Related key Rectangle attack [15].

Ak je generovanie kľúča lineárne, potom daná diferencia kľúča všetkých možných podkľúčov je známa a jednoducho odhadnuteľná. Útok začína s N otvorenými textami so vstupnou diferenciou α, zašifrovanými pod kľúčom K_a a K_b. Očakávame, že $N^2 2^{-n}(\hat{p}\hat{q})^2$ štvoríc bude správnych [15]. Ďalší krok nájdenia správnych štvoríc je možný buď tak, že vyskúšame všetky možné štvorice, čo je ale veľmi neefektívne alebo použijeme získavanie kľúča z prechádzajúcej sekcie.

3.5 Related key Rectangle attack aplikovaný na šifru KASUMI

V šifre KASUMI neexistujú dobré krátke diferencie, no bola popísaná a nájdená dlhá diferencia pre tri kolá šifry KASUMI. Keďže v našom prípade je potrebná diferencia, ktorá by podľa predchádzajúceho popisu spĺňala metódu útoku, v dĺžke sedem kôl, respektíve štvor a trojkolová diferencia. Práve v článku [7] bola popísaná diferencia v šifre KASUMI, ktorá je práve trojkolová. Miernou úpravou, a to tak, že budeme predpokladať v prvej časti uniformnú distribúciu, vieme vybudovať aj diferenciu pre štvrté kolo s tým, že budeme musieť počítať s mierne horšou pravdepodobnosťou. To bude práve prvá časť šifry, označovaná aj ako E_0 a pre druhú časť šifry, označovanú ako E_1 môžeme použiť už len pôvodnú trojkolovú diferenciu.

3.5.1 Diferencia pre prvé 4 kola

Najprv budeme potrebovať zvoliť správne diferencie. Tie už ale boli popísane v článku [7], kde boli nájdené vhodné diferencie pre daný útok. Prvá diferencia pre daný útok vzhľadom na kľúč ΔK_{ab} je $(0,0,1,0,0,0,0,0)$. Iba tretie slovo má nenulovú diferenciu $\Delta K_3 = 0001_x$, kde dolný index x označuje šestnástkovú sústavu. Vstupná diferencia je $\alpha = (0_x, 00200000_x)$ [15]. V [7] bolo ukázané, že daná diferencia má pravdepodobnosť až $\frac{1}{4}$ po troch kolách šifry KASUMI s rovnakou výstupnou diferenciou. Diferencia kľúča štvrtého kola sa zohľadňuje na konci FO funkcii, presnejšie v $FI_{3,4}$. Tým pádom sa nenulová diferencia šíri cez všetky časti funkcie FO, a výstupná diferencia kola je distribuovaná skoro uniformne [15]. Tým pádom je možné použiť diferenciu $\alpha = (0_x, 00200000_x) \rightarrow (y, 00200000_x)$ pre všetky možné hodnoty y. V najhoršom prípade, všetky možné hodnoty y sú rovnako pravdepodobné, potom použitím všetkých 2^{32} hodnôt, každá z nich je očakávaná s pravdepodobnosťou 2^{-32}. Efektívna pravdepodobnosť diferencií, ak sa použijú všetky možné diferencie naraz je $\hat{p} = \sqrt{2^{32} \cdot (2^{-34})^2} = 2^{-18}$ [15]. Ako bolo popísané v [7], útočník môže zvoliť dva bity otvoreného textu tak, že zvýši pravdepodobnosť diferencií: jeden bit otvoreného textu nastaví na jednotku a jeden bit otvoreného textu nastaví na nulu. Tým pádom vie útočník zafixovať jeden bit výstupu OR funkcie, respektíve AND v $FL1$. Presnejšie, ak položíme $P = (P_{LL}, P_{LR}, P_{RL}, P_{RR},)$, kde P_{LL} je 16 bitov otvoreného textu, ktoré vstupuju do AND funkcie v $FL1$ a P_{LR} je zvyšná časť bitov ľavej polovičky otvoreného textu. Útočník nastaví najmenej významný bit P_{LL} a druhý najmenej významný bit P_{LR} na $P_L^0 L = 0$ a $P_L^1 R = 1$, kde mocnina x v označení P^x označuje x-tý bit z ľava [15]. Práve takto zvolené bity spôsobia to, že charakteristika po prvom kole je s pravdepodobnosťou 1 namiesto 1/2 a výsledná diferencia $\alpha = (0_x, 00200000_x) \rightarrow (y, 00200000_x)$ je potom s pravdepodobnosťou 2^{-33} a efektívna pravdepodobnosť je potom 2^{-17} [7].

3.5.2 Diferencia pre 5 až 7 kolo

V tomto prípade zvolíme diferenciu ako $\Delta K_{ac} = (0,0,0,0,0,0,1,0)$ a diferenciu $\gamma = (0_x, 00200000_x) \rightarrow (0_x, 00200000_x) = \delta$ s pravdepodobnosťou $q = \hat{q} = 1/4$ [7].

3.5.3 Útok

Šifru rozdelíme na tri časti. Prvú časť E_0, ktorá pozostáva z prvých štyroch kôl, druhú časť E_1, ktorá pozostáva z kôl päť až sedem a časť E_f určenú na kryptoanalýzu. Nech $K_a, K_b = K_a \oplus \Delta K_{ab}, K_c = K_a \oplus \Delta K_{ab}$ sú neznáme kľúče, ktoré chceme získať.

Pre E_0 použijeme diferenciu popísanú v časti 3.5.1 a pre E_1 použijeme diferenciu popísanú v časti 3.5.2. Pre $N = 2^{51}$ párov otvorených textov zašifrovaných pod kľúčom

K_a a K_b a rovnaký počet párov pod kľúčom K_c a K_d, 2^{102} štvoríc je očakávaných [15]. Zo všetkých štvoríc približne $N^2 \cdot 2^{-64} \cdot 2^{-34} \cdot 2^{-4} = 1$ je správnych štvoríc.

Útok prebieha nasledovne [15]:

1. Zber dát

   (a) Požiadaj o zašifrovanie 2^{51} párov otvorených textov (P_a, P_b), kde $P_b = P_a \oplus \alpha, P^0_{a_{LL}} = 0$ a $P^1_{a_{LR}} = 1$, kde P_a je zašifrované kľúčom K_a a P_b je zašifrované kľúčom K_b. Vlož každý pár do tabuľky indexovanej podľa hešovacej funkcie $(C_{aRL}, C_{aRR}, C_{bRL}, C_{bRR})$.

   (b) Požiadaj o zašifrovanie 2^{51} párov otvorených textov (P_c, P_d), kde $P_d = P_c \oplus \alpha, P^0_{c_{LL}} = 0$ a $P^1_{c_{LR}} = 1$, kde P_c je zašifrované kľúčom K_c a P_d je zašifrované kľúčom K_d. Pre každý pár skontroluj hešovaciu tabuľku v zázname korešpondujúcom hodnote $(C_{cRL} \oplus 0020_x, C_{cRR}, C_{dRL} \oplus 0020_x, C_{dRR})$.

2. Analýza štvoríc

   (a) Pre každú štvoricu (C_a, C_b, C_c, C_d) tipni 32 bitov $KO_{8,1}$ a $KI_{8,1}$. Pre páry (C_a, C_c) a (C_b, C_d) použi hodnotu tipnutého kľúča na výpočet vstupnej a výstupnej diferencie OR funkcie v poslednom kole pre obidva páry. Pre každý bit OR funkcie časi FL_8, možné hodnoty korešpondujúceho bitu $KL_{8,2}$ sú v obrázku číslo 14. V priemere $(8/16)^{16} = 2^{-16}$ hodnôt $KL_{8,2}$ sú navrhnuté každou štvoricou a korešpondujúcim tipom $KO_{8,1}$ a $KI_{8,1}$.

   (b) Pre každú štvoricu a hodnoty $KO_{8,1}, KI_{8,1}$ a $KL_{8,2}$ podľa kroku 2(a), tipni 32 bitov kľúča $KO_{8,3}$ a $KI_{8,3}$ a použi túto informáciu na výpočet vstupnej a výstupnej diferencie funkcie AND v oboch párach. Pre každý zo 16 bitov funkcie AND, možné hodnoty korešpondujúcich bitov $KL_{8,1}$ sú v obrázku číslo 14. V priemere 2^{-16} hodnôt $KL_{8,1}$ sú navrhnuté každou štvoricou a tipom $KO_{8,1}, KI_{8,1}$ a $KO_{8,3}$ a vypočítanej hodnoty $KL_{8,2}$.

3. Nájdenie správneho kľúča prebieha nasledovne: Pre každú štvoricu a hodnoty $KO_{8,1}, KI_{8,1}, KO_{8,3}, KI_{8,3}, KL_{8,1}$ a $KL_{8,2}$ navrhnuté v kroku 2, uhádni zvyšných 32 bitov kľúča a vykonaj skúšobné šifrovanie.

OR — $KL_{8,2}$

(X'_1, Y'_1)	(X'_2, Y'_2) (0,0)	(0,1)	(1,0)	(1,1)
(0,0)	{0,1}	—	1	0
(0,1)	—	—	—	—
(1,0)	1	—	1	—
(1,1)	0	—	—	0

AND — $KL_{8,1}$

(X'_1, Y'_1)	(X'_2, Y'_2) (0,0)	(0,1)	(1,0)	(1,1)
(0,0)	{0,1}	—	0	1
(0,1)	—	—	—	—
(1,0)	0	—	0	—
(1,1)	1	—	—	1

Obr. 14: Možné hodnoty $KL_{8,2}$ a $KL_{8,1}$ [15].

3.5.4 Analýza útoku

Najprv budeme analyzovať krok 2(a) a ukážeme, že vzhľadom na danú vstupnú a výstupnú diferenciu funkcie OR v dvoch párach štvorice, očakávaných počet navrhovaných kľúčov $KL_{8,2}$ je 2^{-16}. To znamená, že $2^{38} \cdot 2^{32} = 2^{70}$ štvoríc spolu s tipnutým kľúčom navrhuje až $2^{70} \cdot 2^{-16} = 2^{54}$ možností pre 48 bitov [15]. Ak sa pozrieme na diferenciu v nejakom bite j, existujú štyri kombinácie vstupnej a výstupnej diferencie pre daný bit pre každý pár [15]. V obrázku 14 sú bity kľúča, ktorý navrhujú dva páry pre daný bit. V danej tabuľke je deväť hodnôt ktoré nemajú žiadnu hodnotu. V týchto prípadoch dochádza k sporu a diferencia vzhľadom na vstupnú a výstupnú diferenciu je nemožná. Napríklad, diferencia 0 nemôže nikdy spôsobiť diferenciu 1 pre žiadnu z funkcií OR a AND. Ďalší príklad je, že kým jeden pár navrhuje, že diferencia v kľúči je 0, druhý pár navrhuje diferenciu v kľúči s hodnotou 1 - takáto možnosť je v spore s definíciou funkcie AND a OR. Celkový počet možných bitov kľúča je 8 zo 16 možných, z čoho vyplýva, že celkovo $(1/2)^{16} = 2^{-16}$ možností prichádza v priemere do úvahy [15].

Celkový počet očakávaných štvoríc a podkľúčov, ktoré vstúpia do kroku 2(b) je 2^{54}. Pre každú štvoricu uhádneme ďalších 32 bitov podkľúča, s celkovým výsledkom $2^{54} \cdot 2^{32} = 2^{86}$ štvoríc [15]. Očakávaný počet štvoríc s podkľúčom potom je $2^{86} \cdot 2^{-16} = 2^{70}$, pričom hádame len 96 bitov kľúča.

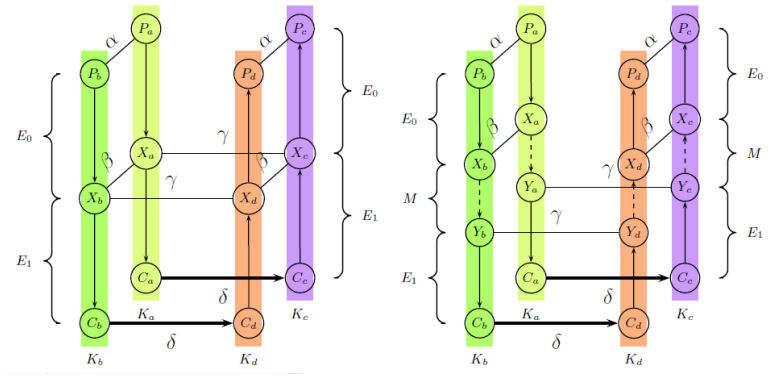
Krok číslo 2(a) môže byť implementovaný iba použitím niekoľkých logických operácií. Test, či daný pár navrhuje spor (nulovú diferenciu vo vstupe s korešpondujúcou nenulovou diferenciou vo výstupe) môže byť vykonaný tak, že sa vypočíta $Z = \bar{X}' \wedge Y'$, kde X' je vstupná diferencia, $\bar{X}'$ je jej bitový komplement a Y' je výstupná diferencia. Ak Z je nenulové, potom existuje bit v X', ktorý je nula, kým korešpondujúci bit v Y' je 1 [15]. Potom môžeme skontrolovať pomocou logických obvodov, či daný pár navrhuje spor s obrázkom číslo 14. Taktiež môžeme nájsť ktoré bity kľúča naznačuje kľúč [15]. Pre operáciu OR, bity ktoré nejaký pár naznačuje sú bity, pre ktoré X' má hodnotu 1 a hodnota $KL_{8,2}$ v týchto bitoch je rovnaká s hodnotou $\bar{Y}'$ [15]. Na to, aby

sme skontrolovali, či daný pár naznačuje nejaký spor v hodnote kľúča je dostačujúce skontrolovať, či $(X_1' \wedge X_2') \wedge (Y_1' \oplus Y_2') \neq 0$. Podobná metóda sa použije aj v kroku 2(b). V kroku 3 nakoniec útočník vyskúša všekých 2^{70} navrhovaných kľúčov pre 96 bitov kľúča a vyskúša zvyšných 32 bitov útokom hrubou silou. Celková časová zložitosť je potom 2^{102} šifrovaní [15].

Optimalizáciou útoku je možné niektoré vlastnosti šifry KASUMI využiť. Napríklad je možné krok 2(b) implementovať v dvoch rôznych podkrokoch. Ak najprv uhádneme $KO_{8,3}$, a potom iba deväť bitov $KI_{8,3,2}$ a nájdeme hodnotu deviatich bitov $KL_{8,1}$ [15], vieme sa s časovou zložitosťou dostať na $2^{78,1}$ šifrovaní s $2^{43,2}$ pamäťovou zložitosťou [15]. Týmto optimalizáciam sa venovať v tejto práci nebudeme, keďže nemajú vplyv na to, čo plánujeme v práci ďalej popísať.

3.6 Related key Sandwich attack

Posledný útok, ktorý spomenieme je takzvaný Related key Sandwich attack [14] [4]. Tento útok je zo všetkých vyššie popísaných útokov na šifru KASUMI najefektívnejší, no pritom aj najzložitejší. Nevyužíva diferenciu pre kryptoanalýzu v poslednom kole no naopak, kryptoanalýza sa deje niekde uprostred šifry KASUMI.



Obr. 15: Porovnanie Related key Boomerang attack v ľavo a Sandwich attack v pravo [15].

V našej práci sa týmto útokom podrobne zaoberať nebudeme, keďže nie je predmetom ďalšej analýzy. Preto len veľmi zhruba popíšeme postup kryptoanalýzy pomocou tohto útoku a časovú zložitosť. Útok prebieha podobne, ako v prípade útoku z predchádzajúcej časti [14]. Na začiatku je potrebné zašifrovať niekoľko otvorených textov podľa vopred danej diferencie (diferencie sa nemenia, no je väčšia šanca, že daná diferencia bude splnená). Identifikácia prebieha podobne ako predtým a to pomocou hešovacej tabuľky vzhľadom na zvolenú diferenciu. Analýza je trocha zložitejšia, opiera

sa o vlastnosti šifry, ktoré vyplývajú z toho, že v danom útoku sú rovnaké kolá v prípade posledného (predposledného) kola a taktiež aj štvrtého kola šifrovania. Tým pádom je možné v prípade Sandwich attack využiť poznatky z predchádzajúceho útoku a dokončiť analýzu v štvrtom kole šifrovania s lepšou pravdepodobnosťou úspechu. Celá myšlienka spočíva v tom, že keď sme v prípade predchádzajúceho útoku potrebovali pridať jedno kolo, aby sme našli dostatočne dlhu diferenciu, pri Sandwich attack daný fakt potrebovať nebudeme. Totižto bude stačiť nájsť vhodné diferencie tak, aby sme štvrté kolo vedeli ohraničiť danými diferenciami z oboch strán. Zvyšok sa prevedie veľmi podobne ako v útoku predtým. Pre bližšie informácie o danom útoku aj s konkrétnym popisom daného útoku odporúčame pozrieť [14].

Tento útok bol aj experimentálne overený. Je možné pomocou neho prelomiť šifrovanie pomocou šifry KASUMI s časovou zložitosťou 2^{32}, na bežnom počítači je to približne 112 minút. Pamäťová zložitosť daného útoku je 2^{30} [14]. Vzhľadom na to, že daný útok požaduje šifrovanie otvoreného textu pod rovnakým kľúčom a s danou diferenciou, nie je možné tento útok použiť na praktickú kryptoanalýzu šifry používanej v sieťach 3G.

4 Návrh kryptoanalytických metód a vylepšenia kryptografických systémov

V tejto kapitole sa budeme venovať analýze jednotlivých útokov. Najprv sa pozrieme na problémy a riešenia ochrany voči kryptoanalýze pomocou dúhových tabuliek šifry A5/1 a popíšeme, ako je možné dané kryptoanalytické útoky ešte viac vylepšiť. Najväčší problém spočíva v tom, ako je generované frame number a aké bity sú v danej šifre šifrované, to znamená, že potrebujeme uhádnuť to, čo daná šifra šifruje, takzvaný otvorený text. V prípade šifry KASUMI sa pozrieme na možnosti filtrovania v prípade využitia Related key Rectangle attack, ktoré nie sú popísané v pôvodnom útoku. Tieto možnosti filtrovania ukážu, že v časti, kde hádame jednotlivé kľúče, je možné pridať aj nepriamo filtrovanie, ktoré vie zabezpečiť to, že kľúče posledného kola vieme odhadnúť s lepšou pravdepodobnosťou.

4.1 Vlastnosti šifry A5/1 a útokov na šifru A5/1

Celá myšlienka útoku na šifru A5/1 z praktického hľadiska je založená na tom, že poznáme otvorený text o dĺžke 114 bitov a využitím dúhových tabuliek vieme odhaliť vnútorný stav šifry. Dúhové tabuľky popísane v 2. kapitole pracujú na princípe získania vnútorného stavu z výstupu šifry. Zhrnieme si základné poznatky druhej kapitoly:

1. Ak poznáme vnútorný stav, vieme šifru A5/1 točiť späť skoro ľubovoľne. Význam slova skoro v tomto prípade znamená, že ak šifru A5/1 budeme točiť príliš dlho späť, vznikne nám príliš veľká množina riešení, ktoré je potrebné ďalej spracovať, čo nemusí byť efektívne.

2. Pri spätnom točení o 1 až 2 bity dostávame v priemere 1 až 2 možné riešenia predchádzajúcich stavov. Pri spätnom točení o 100 bitov dostávame v priemere až 14 možných riešení.

3. Funkciu f definujú v prípade kryptoanalýzy rôzni autori rôzne. Podľa [32] je vhodné ako funkciu f zvoliť stav po inicializácii šifry kľúčom, respektíve frame number. Podľa [30] je ako funkcia f považovaný stav po zamiešaní kľúča. V prípade [32] dúhové tabuľky zverejnené neboli, takže s nimi pracovať nevieme.

4. Šifra A5/1 má len približne $2^{61,2}$ vnútorných stavov, pokrytie dúhovými tabuľkami vieme zväčšiť, ak použijeme tabuľky podľa [32].

5. Nevieme z výstupu efektívne vygenerovať, získať vnútorný stav šifry A5/1 - to je problém, na ktorý slúžia dúhové tabuľky.

6. Pri generovaní známych správ v mobilných sieťach, operátori častokrát vkladajú do vnútra náhodné zarovnanie, aby útok popísaný v [30, 32] nebol efektívny.

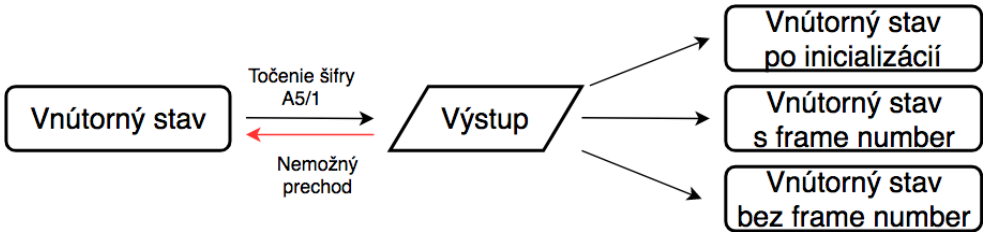
7. Ľubovoľný vnútorný stav môžeme chápať ako stav s ľubovoľným frame number a k tomu s fixným kľúčom. Opačne to ale neplatí. Pre ľubovoľný kľúč nevieme nájsť frame number taký, aby odpovedal vnútornému stavu. Dôkaz daného tvrdenia je jednoduchý a spočíva v tom, aká je veľkosť, tazvaný keyspace, kľúča a aká je veľkosť množiny frame number. Totižto frame number je o veľkosti 2^{22} bitov. V najhoršom prípade je vo vnútornom stave šifry A5/1 jeden bit, ktorý nie je určený frame number, a to je bit registra 3 na pozícií 23. Z daného vnútorného stavu šifry vieme spätným točením frame number odstrániť. Ak by sme chceli odstrániť kľúč, ktorý sme si zafixovali, potrebovali by sme nájsť taký frame number, ktorý po odstránení zo šifry A5/1 spôsobí to, že v šifre zostane taký stav, ktorý spätným točením vygeneruje kľúč, ktorý sme si zafixovali. Keďže veľkosť vnútorného stavu šifry A5/1 je 2^{64} a veľkosť množiny frame number je len 2^{22}, tak odstránením frame number dostávame pri najlepšom odhade množinu stavov o veľkosti 2^{42}, odpočítaním bitu, ktorý nebol pridaným frame number ovplyvnený je to 2^{41} a ak daný kľúč spadá do tejto množiny, potom ho nie je možné získať s tým, že odstránime aj všetkých 2^{22} možností frame number.

8. Výstup z funkcie f v prípade dúhových tabuliek vieme považovať ako vnútorný stav šifry v ľubovoľnej časti šifrovania (okrem toho, aký kľúč si zašifrujeme). Dané tvrdenie vyplýva z toho, že šifrou A5/1 je možné točiť ľubovoľne dlho dopredu a späť, a tým vygenerovať potrebné výstupy, respektíve vieme začať inicializáciu šifry novým frame number alebo kľúčom.

Najdôležitejší problém, ktorý by sme chceli vyriešiť je popísanie útoku, ktorý by umožňoval aj pokrytie, respektíve odhadnutie správ a ich náhodného zarovnania. Totižto, v pôvodnom útoku boli použité v prípade kryptoanalýzy správy typu NOP,

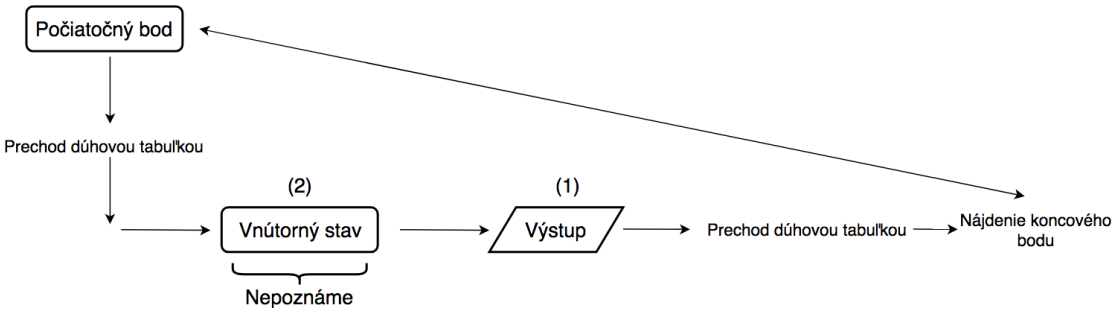
ktoré boli dorovnané na 114 bitov konštantou. Tým pádom vedel útočník presný obsah správ a vedel pomocou funkcie XOR vygenerovať výstup, ktorý považoval za výstup z nejakého vnútorného stavu šifry A5/1 a snažil sa tento výstup nájsť v dúhových tabuľkách. Potom, ako boli dané útoky zverejnené, operátori pristúpili na to, že začali dorovnanie správ na 114 bitov maskovať náhodnými postupnosťami.

Takýmto spôsobom nie je možné pôvodne odhadnúť, aké správy sa posielajú sieťou, a tým pádom efektívne odhadnúť výstup zo šifry A5/1. Existujú možnosti, napríklad vyskúšanie všetkých možných kombinácií otvoreného textu (respektíve jeho dorovnania), no tie nám neprinášajú možnosť kryptoanalýzy v reálnom čase. Totiž, ak by sme chceli preskúmať všetky možnosti napríklad len 50 bitov, ktoré sú zvolené náhodne, potrebujeme 2^{50} kombinácií, ktoré je potrebné vyskúšať spolu s náročným prehľadávaním dúhovej tabuľky a následným spätným točením šifry A5/1.



Obr. 16: Príklad uvažovania výstupu zo šifry A5/1.

V obrázku 16 ukazujeme príklad možnosti, ako je možné vnímať výstup zo šifry A5/1 v ďalšom kole dúhovej tabuľky a v obrázku 17 popisujeme bod, ktorý vstupuje do kryptoanalýzy. Ak by bolo možné z bodu (1) prejsť späť o jeden prechod, dúhové tabuľky nie sú vo všeobecnosti potrebné. Takéto útoky existujú, ale je dokázané, že majú časovú zložitosť takú, že ich nie je možné riešiť v reálnom čase [18].



Obr. 17: Schéma pôvodného útoku.

V danom prípade je nutné nájsť koncový bod, ktorým sa vieme dostať na počiatočný bod dúhovej tabuľky, a tú prejsť celým algoritmom, aby sme našli neznámu podľa

obrázka 17.

Teraz uvedieme niekoľko viet, ktoré sú potrebné pri popise nových utokov.

Veta 1: *Nech R' je stav šifry A5/1, ktorý vznikol zo stavu R tak, že bol do šifry inicializačnou metódou vložený bit v. Potom je možné jednoznačne zo stavu R' získať stav R, ak je v známe.*

Dôkaz: Označme vnútorny stav šifry A5/1 R takto $R1 = r^1_{18}, \ldots, r^1_0, R2 = r^2_{21}, \ldots, r^2_0, R3 = r^3_{22}, \ldots, r^3_0$, kde $R1, R2, R3$ je označenie pre registre číslo $1, 2, 3$ a vnútorný stav šifry R' ako $R'1 = r'^1_{18}, \ldots, r'^1_0, R'2 = r'^2_{21}, \ldots, r'^2_0, R'3 = r'^3_{22}, \ldots, r'^3_0$. Potom platí, že $r'^1_{18} = r^1_{17}, \ldots, r'^1_1 = r^1_0, r'^1_0 = v \oplus r^1_{13} \oplus r^1_{16} \oplus r^1_{17} \oplus r^1_{18}, r'^2_{21} = r^2_{20}, \ldots, r'^2_1 = r^2_0, r'^2_0 = v \oplus r^2_{20} \oplus r^2_{21}, r'^3_{22} = r^3_{21}, \ldots, r'^3_1 = r^3_0, r'^3_0 = v \oplus r^3_7 \oplus r^3_{20} \oplus r^3_{21} \oplus r^3_{22}$. V našom prípade poznáme hodnoty registor v stave R'. Z toho vyplýva, že priamo poznáme hodnoty $r^1_{17}, \ldots, r^1_0, r^2_{20}, \ldots r^2_0, r^3_{21}, \ldots, r^3_0$. Jedinou neznámou sú pre nás hodnoty $r^1_{18}, r^1 21, r^1_{22}$, ktoré ale vieme vypočítať z rovníc

$$r'^1_0 = v \oplus r^1_{13} \oplus r^1_{16} \oplus r^1_{17} \oplus r^1_{18},$$

$$r'^2_0 = v \oplus r^2_{20} \oplus r^2_{21},$$

$$r'^3_0 = v \oplus r^3_7 \oplus r^3_{20} \oplus r^3_{21} \oplus r^3_{22},$$

totižto jedine neznáme v týchto rovniciach sú r^1_{18}, r^2_{21} a r^3_{22} a operácia $\oplus$ je asociatívna a komutatívna, z toho vyplýva, že

$$r'^1_0 \oplus v \oplus r^1_{13} \oplus r^1_{16} \oplus r^1_{17} = r^1_{18},$$

$$r'^2_0 \oplus v \oplus r^2_{20} = r^2_{21},$$

$$r'^3_0 \oplus v \oplus r^3_7 \oplus r^3_{20} \oplus r^3_{21} = r^3_{22},$$

□

Veta 2: *Pre ľubovoľný vnútorný stav šifry po inicializácií kľúčom K_c a s ľubovoľným frame number párnym, vieme prejsť k vnútornému stavu po inicializácií s kľúčom K_c, a s frame number o jedna väčším.*

Dôkaz: Označme si frame number ako FN, pričom v binárnej forme má FN tvar $FN = b_{21}, b_{20}, \ldots, b_0$. Keďže ide o párne číslo, tak platí, že $b_0 = 0$. Pre $FN + 1$ platí, že $FN + 1 = b_{21}, b_{20}, \ldots, 1$. Keďže inicializácia šifry A5/1 je lineárna, tak je potrebné

zmeniť posledný bit po inicializácií FN na $b_0 = 1$. Podľa vety 1 je spätné točenie šifry A5/1 o jeden bit za predpokladu, že je známy vstup pri kroku inicializácií známy, jednoznačné. Z toho vyplýva, že je možné šifrou A5/1 zatočiť späť za predpokladu, že vstup bol 0 (čo je v prípade párneho FN), jednoznačne a následne vložiť a zatočiť šifrou A5/1 dopredu so vstupom 1 o jedno otočenie. □

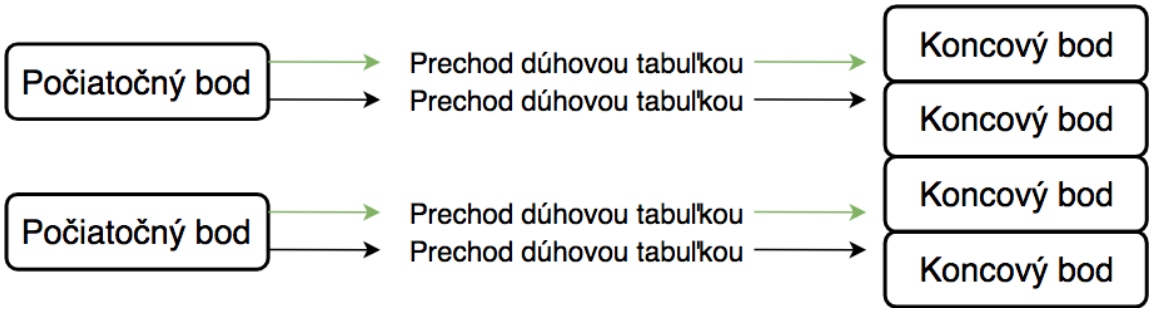
Veta 3: *Po inicializácií šifry A5/1 je možné jednoznačne šifrou A5/1 točiť spätne o ľubovoľný počet bitov vzhľadom na ľubovoľný frame number.*

Dôkaz: Túto vetu dokážeme indukciou veľmi jednoducho, podľa n - počtu točení. Podľa vety 1 je možné šifrou A5/1 zatočiť späť jeden bit, čím sme dokázali, že dané tvrdenie platí pre $n = 1$, ak si zvolíme ľubovoľné v. Predpokládajme, že tvrdenie platí pre ľubovoľné $n = k$ a ukážeme, že platí aj pre $k + 1$ točenie. Keďže točenie šifry A5/1 pri inicializácií, kde nie je zapnutá majoritná funkcia je lineárne, to znamená, že platí, že $k + 1$ točení je to isté, ako k točení a potom 1 točenie, potom podľa vety 1 a za predpokladu, že tvrdenie platí pre k točení, tak platí aj pre $k + 1$ točení. Ak položíme hodnotu v v každom kroku podľa vety 1 ľubovoľne, čo nie je predpokladom tvrdenia vety 1, vieme odstrániť ľubovoľný frame number. □

4.2 Vnútorné stavy podľa frame number

Jedným z možných vylepšení dúhových tabuliek je to, aby sme danú šifru prechádzali na základe fixovaného frame number. Podľa vlastnosti číslo 7 a 8 popísaných v kapitole 4.1, vieme v rámci ľubovoľného vnútorného stavu zafixovať frame number. Ak by sme v dúhových tabuľkách pridali dva koncové body také, že odpovedajú dvom po sebe nasledujúcim frame number a aby dúhové tabuľky boli generované tak, že vnútorný stav šifry je potrebný ešte doplniť o frame number, ktorý je pre našu kryptoanalýzu fixovaný. Tým pádom by sme dosiali to, že ak by sme uhádli bity dvoch po sebe nasledujúcich otvorených textov, ktoré boli šifrované šifrou A5/1, ale frame number by bol rozdielny (hádame napríklad dve rôzne, po sebe nasledujúce správy) a v našom prípade by sme mali pripravené dúhové tabuľky také, že sú tvorené na presne tento daný frame number, vedeli by sme tým, že dôjdeme na koniec v prípade oboch uhádnutých správ overiť, či daný kľúč je zhodný (oba konce, ktoré vygenerujeme sú v jednom riadku). Nepotrebovali by sme v tomto prípade nájsť počiatočný stav, vytvo-

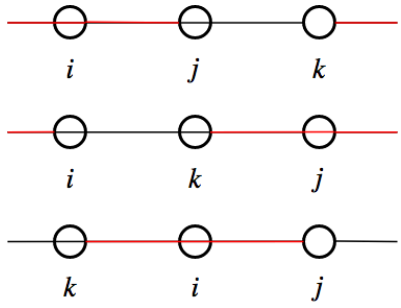
riť prechod k stavu pred generovaním výstupu (k stavu z obrázka 17 označeného ako (2)), vygenerovať, respektíve spätným točením šifry overiť, že vygenerujeme rovnaký kľúč. Totižto všetky správy, ktoré sú šifrované, sú šifrované rovnakým kľúčom, s rozdielnym frame number. Ak by sme si tipli (hádali) zlý otvorený text, tak v dúhovej tabuľke by sme nenašli zhodu v oboch koncových bodov. Ak by sme si tipli dobre (aspoň by sme to predpokládali), vedeli by sme tým, že prídeme na koniec dúhových tabuliek a pozrieme sa do odpovedajúceho riadku, overiť náš tip bez toho, aby sme museli tabuľky prechádzať znova. Schéma útoku je zobrazená v obrázku číslo 18.



Obr. 18: Schéma dúhovej tabuľky pre dva koncové stavy.

Ako funkciu redukcie by sme v tomto prípade zvolili zvažovanie výstupu zo šifry A5/1 ako vnútorný stav po inicializácií kľúčom, ku ktorému je potrebné pridať frame number taký, aký je v dúhových tabuľkách. Týmto útokom by sme oslabili silu dúhových tabuliek v tom zmysle, že z ľubovoľného vnútorného stavu vieme odstrániť ľubovoľný frame number, čo zvyšuje pokrytie dúhových tabuliek. Takýmto spôsobom by sme mali pokryté dúhové tabuľky len na 1 frame number a keďže množina frame number je o veľkosti 2^{22} a jeden frame number trvá približne 4,615 ms, celkový čas, ktorým by v najhoršom prípade musel útočník čakať, aby dospel k vhodnému frame number, je približne 5 hodín. Veľkosť dúhových tabuliek by samozrejme bola iná, potrebovali by sme pridať dva koncové stavy, ktoré by odpovedali točením v rámci iného kľúča. V obrázku 17 sme zelenou farbou označili jeden frame number a čiernou farbou sme označili druhý frame number. Netvrdíme, že musí platiť, že druhý frame number je o 1 väčší, tj. analyzujeme dve po sebe nasledujúce správy, no je možné frame number zvoliť aj s väčším odstupom, čo ale čas čakania na vhodný frame number v najhoršom prípade nezlepší. Predpokladajme, že máme v dúhovej tabuľke i-tý frame number a j-tý frame number. V pripade, že prvú správu, ktorú zachytíme je s frame number $j+1$ a platí, že $i < j$, potom budeme musieť čakať celých 2^{22} frame number v najhoršom prípade.

4.2.1 Algoritmus a analýza útoku dúhových tabuliek pre fixný frame number



Obr. 19: Schéma výpočtu vzdialeností.

V priemernom prípade vieme očakávaný čas odhadnúť približne takto: Zvolíme očakávanú hodnotu pre ľubovoľný k-tý frame number ako $2^{22} + j - k$ v prípade, že $k - j > 0$, $2^{22} - k + i$ ak $i < k < j$, inač $j - k$, ak predpokladáme, že $i < j$, pridaným času potrebného na vygenerovanie jedného frame number, približne 4,615 ms a pri predpoklade, že je rovnako pravdepodobné, že daný frame number sa vyskytne, tj. $1/2^{22}$. Pre bližšie pochopenie prikladáme obrázok, ktorý dané výpočty znázorňuje ako obrázok číslo 19, kde je červenou farbou znázornený interval, dĺžka čakania vzhľadom na konštanty i, j a k. Nakoniec spočítame očakávanú hodnotu ako:

$$E(i,j) = \sum_{k=1}^{2^{22}} \frac{4,615}{2^{22}} p(k,i,j), \quad \text{kde} \quad p(k,i,j) = \begin{cases} 2^{22} - k + j & \text{ak} \quad k - j > 0 \\ 2^{22} - k + i & \text{ak} \quad i < k < j \\ j - k & \text{inač.} \end{cases}$$

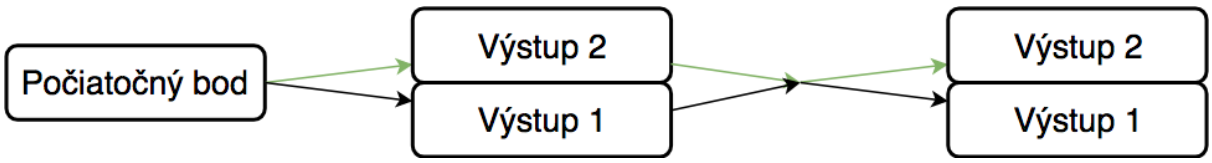
ktorú môžeme vyjadriť len pomocou dĺžky, to znamená vzdialeností medzi i a j ako:

$$E(l) = \sum_{k=1}^{2^{22}} \frac{4,615}{2^{22}} p(k,l), \quad \text{kde} \quad p(k,l) = \begin{cases} 2^{22} - k + l & \text{ak} \quad k > l \\ 2^{22} - k & \text{inač.} \end{cases}$$

pretože pravdepodobnosť je rovnaká pre každý jav, takže daný problém vieme vyjadriť aj ako vzdialenosť j od 0, keď budeme predpokladať, že $i = 0$. Odhadované časi čakania uvádzame v prílohe pod písmenom E. Celková kapacita na disku takýchto tabuliek bude maximálne 1.5 násobne väčšia ako pôvodná veľkosť tabuliek - pretože potrebujeme pridať 1/2 veľkosti tabuľky - tj. dvojnásobná veľkosť bitov koncovej časti, ktorá je maximálne 1/2 z pôvodnej, čo je odhadom 2,7 TB. Pokrytie danej tabuľky,

v prípade, že bude generovaná ako [32] sa nezmení a podľa [32] je 50 % úspešnosť útoku, a keďže ide o dva nezávisle javy, no stále hádame len jeden a ten istý kľúč, aj keď prechádzame tabuľkami dva krát, pravdepodobnosť úspechu sa nezmení, to znamená, že aj v tomto prípade je 50 %. Týmto spôsobom môžeme rýchlejšie odhadnúť tip otvoreného textu, pretože v porovnaní s pôvodným útokom nie je potrebné prechádzať celý riadok dúhovej tabuľky znova a nie je potrebné nájsť pôvodný kľúč. Stačí, ak dané koncové body budú odpovedať jednému a tomu istému riadku. Tým je zabezpečené to, že s určitosťou vieme, že existuje vnútorný stav, ktorý predchádzal výstupu.

Jediný problém je pri prechode dúhovou tabuľkou. Princíp dúhovej tabuľky je taký, že vychádzame z bodu a generujeme pomocou funkcie f ďalší stav. V tomto prípade máme ale dva prechody tabuľkou v rámci jedného riadku. Na to, aby sme vedeli nájsť pôvodný kľúč, ktorý vygeneruje oba prechody na fixovaný frame number, je potrebné zlučovanie, ktoré je možné implementovať ako funkciu XOR, v ktorej ale môže dôjsť k nedosiahnuteľnému vnútornému stavu alebo tak, že výstup pre jeden frame number aj pre druhý frame number vložíme do šifry A5/1 ako pri inicializácií. Nasledujúci obrázok znázorňuje riešenie daného problému - výstup 2 bude považovaný ako ďalší vnútorný stav šifry podľa toho, ako to bolo v pôvodnom útoku a výstup 1 bude pridaný do šifry A5/1 inicializačnou metódou podľa toho, ako je inicializovaný kľúč šifry A5/1.



Obr. 20: Spájanie výstupných bodov.

Celková časová zložitosť podľa [32] bude [32] $\sum_{j=1}^{s}[q+(j-1)qv+(r-1)qsv]v = 2(tsv+\frac{t}{r}-\frac{tsv}{2r}-\frac{tv}{2r})$, výpočtov funkcie f, podľa označenia z kapitoly 2, kde f je funkcia mixovania kľúča a pridania hodnoty frame number, dva krát. Z toho vyplýva, že celkovo bude pridaných len konštantne viac operácií vzhľadom na pôvodné dúhové tabuľky. Na druhej strane, hľadanie v tabuľkách zostáva rovnaké ako pri pôvodnom útoku. Daný algoritmus sme implementovali a otestovali na 1000, vopred pripravených riadkoch tabuľky, ktorých generovanie trvalo bez využitia grafických kariet a výkonnej výpočtovej sily približne 12 minút. Presnou implementáciou a generovaním celej tabuľky sa venovať nebudeme, pretože nemáme k dispozícií dostatočnú výpočtovú silu a vytvoriť danú tabuľku nie je časovo reálne.

Algoritmus 2 Vytvorenie tabuľky (riadka) bez použitia farby.

```
1:  procedure NÁJDI SP A EP(FN1, FN2)   ▷ FN označuje fixované frame number
2:      x ← random(64)                  ▷ Vygeneruje náhodné 64 bitové číslo
3:      while x ∉ EP do                 ▷ EP je množina endpoint-ov, 12 núl nakonci
4:          s ← x                       ▷ s je vnútorný stav šifry
5:          A51Append(&s, FN1)          ▷ Inicializačná funkcia šifry, & značí referenciu
6:          A51(&s, 100)                ▷ Mixovanie kľúča
7:          m ← A51(&s, 64)             ▷ Vygenerovanie výstupu o veľkosti 64 bitov
8:          s ← x                       ▷ s je vnútorný stav šifry
9:          A51Append(&s, FN2)          ▷ Inicializačná funkcia šifry
10:         A51(&s, 100)                ▷ Mixovanie kľúča
11:         m' ← A51(&s, 64)            ▷ Vygenerovanie výstupu o veľkosti 64 bitov
12:         x ← A51Append(&m, m')
13:     return x
```

Algoritmus 2 popisuje problém generovania jedného riadka bez použitia farby. Pridanie farby je len zopakovanie celého algoritmu s novým vstupom x a konštanty ako farby. Nakoniec je možné ako výstup z algoritmu číslo 2 vytiahnuť buď jeden koncový bod alebo miernou modifikáciou dva koncové body, podľa toho, ktorú časť dúhovej tabuľky generujeme.

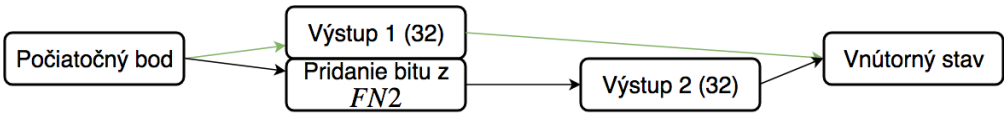
Výhoda takého prístupu spočíva v tom, že ak uhádneme, tipneme si ľubovoľnú časť otvorenej správy, stačí nám dôjsť na koniec dúhových tabuliek, prehľadať ich a zistiť, či v nájdenom riadku sú oba koncové body, ktoré sme vygenerovali. Ak nie, musíme zmeniť tip.

4.3 Rozdelenie výstupu dúhových tabuliek

V tejto časti popíšeme útok, ktorým je možné zlepšiť časovú zložitosť predchádzajúceho útoku tým, že nie je potrebné uhádnuť toľko bitov ako v pôvodnom útoku. V princípe je možný daný útok previesť tak, že výstupy oboch frame number budeme zlučovať, podľa toho, ktorý bit je potrebný, respektíve ktorý bit poznáme. Predpokladajme, že sme zachytili 2 správy, jednu s frame number, označeným ako $FN1$ a druhú správu s frame number, označeným ako $FN2$. Z princípu inicializácie šifry vyplýva, že ak platí, že $FN1 = FN2 - 1$ a $FN1$ je párne číslo, tak v binárnej forme má ľubovoľné $FN1$ na konci 0. Tým pádom je možné šifru, ktorá bola inicializovaná ľubovoľným $FN1$ točiť späť a získať $FN2$ tým, že pridáme na koniec šifry 1 bit na pozíciu registrov $R1(0), R2(0)$ a $R3(0)$, podľa vety 1 a 2. Z daného a z vlastností šifry vyplýva,

že ak predpokladáme vnútorný stav s ľubovoľným frame number, vieme sa dostať k vnútornému stavu šifry pred zamiešaním kľúča s frame number o jeden väčším.

Ďalším pozorovaním, ktoré budeme využívať je to, že výstup zo šifry A5/1 je funkciou XOR pridaný k správe, ktorú chceme šifrovať. Funkcia XOR je bitová, to znamená, že ľubovoľný bit je ovplyvňovaný len samostatne, bez toho, aby sme potrebovali poznať bit pred, respektíve bit po danom bite. Teraz predpokladajme, že poznáme 32 bitov správy, ale nepoznáme zvyšných 32 bitov správy. Takýto stav nastáva napríklad pri posielaní správ NOP, kde je za správu NOP pridané náhodné dorovnanie. V práci netvrdíme, že musí ísť práve o 32 bitov, no uvádzame to ako príklad, ktorý je možné ľahko rozšíriť ďalej. NOP správy sa častokrát posielajú aj niekoľko krát po sebe. Tým pádom máme dve po sebe nasledujúce správy, ktoré sú šifrované rovnakým kľúčom, s frame number rozdielom o veľkosti 1, a z ktorých poznáme len 32 bitov. Následne vieme vytvoriť dúhové tabuľky, ktoré z ľubovoľného vnútorného stavu vygenerujú dva postupnosti také, že jedná postupnosť bude výstup z funkcie A5/1 s frame number $FN1$ a druhá bude postupnosť výstupu z funkcie $A5/1$ s frame number $FN2$. Tieto postupnosti spojíme tak, že konkatenujeme prvú časť výstupu podľa $FN1$ o veľkosti 32 bitov s druhou časťou výstupu podľa $FN2$ o veľkosti 32 bitov. Tým pádom získame výstup, ktorý budeme ďalej redukčnou funkciou považovať za vnútorný stav šifry A5/1 a pokračovať do bodu, kedy získame koncový bod, takzvaný ENDPOINT.



Obr. 21: Príklad generovania tabuliek, kde veľkosť výstupu je v zátvorke.

Obrázok 21 schematický znázorňuje princíp generovania tabuliek. Ak by vnútorný stav, ktorý vznikne po konkatenácií oboch výstupov nebol validný, to neznamená, že ďalší vnútorný stav nebude validný - je malá pravdepodobnosť, že pre ľubovoľný vnútorný stav tento stav nebude validný.

4.3.1 Algoritmus a analýza útoku dúhových tabuliek pre rozdelený výstup

Algoritmus 3 popisuje postup generovania jedného riadka dúhovej tabuľky. V prípade útoku, ak poznáme len prvých 32 bitov dvoch po sebe nasledujúcich správ vieme postupovať tak, že použijeme konkatenáciu po použití funkcie XOR na zachytenú správu a následným točením šifry A5/1, kým nenarazíme na koncový bod.

Algoritmus 3 Vytvorenie tabuľky (riadka) bez použitia farby.

```
1:  procedure NÁJDI SP A EP(FN1, FN2)   ▷ FN označuje fixované frame number
2:      x ← random(64)                  ▷ Vygeneruje náhodné 64 bitové číslo
3:      while x ∉ EP do                 ▷ EP je množina endpoint-ov, 12 núl nakonci
4:          s ← x
5:          A51(&s, 100)                ▷ Mixovanie kľúča
6:          m ← A51(&s, 32)             ▷ Vygenerovanie výstupu o veľkosti 32 bitov
7:          s ← x                       ▷ s je vnútorný stav šifry
8:          A51AppendDel(&s, 1)         ▷ Odstránenie jedného bitu a pridanie 1 na koniec
9:          A51(&s, 100)                ▷ mixovanie kľúča
10:         m' ← A51(&s, 32)            ▷ Vygenerovanie výstupu o veľkosti 32 bitov
11:         x ← m||m'                   ▷ Označením || myslíme konkatenáciu
12:     return x
```

Analýza takého útoku spočíva v tom, že potrebujeme vyjadriť jeho korektnosť a časovú zložitosť. V princípe, podľa vlastností šifry A5/1, popísaných v kapitole 4.1, respektíve dúhových tabuliek popísaných v tejto kapitole a vety 3 vieme, že v rámci ľubovoľného vnútorného stavu vieme odstrániť ľubovoľné frame number. Tým máme zaručené to, že vieme odstrániť aj jeden bit frame number. Nech šifra A5/1 je inicializovaná kľúčom K_c a frame number $FN1$, ktoré je párne číslo. Spätným točením šifry A5/1 o jeden bit, tak, že predpokládamé, že $FN1$ je párne a následným pridaním jednotky vieme, podľa vlastnosti dostať vnútorný stav šifry A5/1 inicializovanú ako K_c s frame number $FN1+1$ podľa vety 2. Tým máme zaručené, že výstup zo šifry A5/1 pomocou $FN1$ aj $FN1+1$ je spôsobený rovnakým kľúčom. V danom útoku môžeme považovať za vnútorný stav, stav s ľubovoľným $FN1$. Totižto, ak $FN1$ zvolíme ľubovoľne, tak pomocou vlastnosti 7 vieme prejsť k inému kľúču, fixnému pre daný frame number. Ako funkciu f zvolíme vnútorný stav šifry pred zamiešaním kľúča a konkatenáciu dvoch výstupov podľa toho, ako je to navrhnuté v algoritme 3.

Takýmto útokom získame možnosť kombinovať rôzne známe pozície bitov. Napríklad nemusí isť vo funkcií f o konkatenáciu, no môžeme použiť aj striedanie bitov z výstupov. Označme si c_i ako i-tý výstup vzhľadom na $FN1$ a d_i ako i-tý výstup vzhľadom na $FN1+1$. Potom môžeme použiť striedanie bitov takto: c_1, d_1, c_2, d_2. Takýto útok je možné pripraviť vopred na danú postupnosť známych bitov. Napríklad podvrhnutá SMS správa $m_1, m_2, m_3, \ldots$ je práve posielaná ako postupnosť niekoľkých správ s frame number $FN1, FN1+1, FN1+2, \ldots, FN1+n$ s tým, že vzhľadom na prvý frame number obsahuje šifrovaný text tieto bity: $m_1, m_{1+k}, m_{1+2k}, \ldots$, pre nejaké fixné k. Takýmto spôsobom je možné pripraviť dúhové tabuľky podľa prístupu ku kryptoanalýze al-

goritmom, ktorý sme popísali v danej časti práce. Zaujímave je podľa vety 3 ešte preskúmať tento algoritmus s tým, že je možné zvoliť 64 po sebe nasledujúcich frame number. Podľa vety 3 vieme odstrániť ľubovoľný frame number. Na to, aby sme zachovali princíp dúhových tabuliek potrebujeme zabezpečiť to, aby sme odstránili len rozdiely vo frame number, nie konkrétny frame number, preto budeme potrebovať odstrániť 6 bitov a následne inicializovať šifry A5/1 postupne $FN+1, \ldots, FN+64$. Aby táto inicializácia nespôsobila zmenu v inom bite frame number, potrebujeme aby FN bol deliteľný číslom 64 (tj. mal nakonci 6 bitov rovných 0).

Implementovali sme vytvorenie takého riadku tabuľky, kde sme si zapamätali jednu hodnotu kľúča, čo slúži na overenie správnosti výpočtu. Vytvorenie 1000 takýchto riadkov trvalo približne rovnako dlho ako v predchádzajúcom prípade, to znamená približne 12 minút neoptimalizovanou metódou. Následne overenie dosiahnutia koncového bodu z vopred pripraveného výstupu (ktorý sa vyskytol v dúhových tabuľkách) dopadol úspešne. Keďže nemáme k dispozícií dostatočne výkonné výpočtové zariadenie, generovanie celej tabuľky považujeme za bezpredmetné.

Časová zložitosť výpočtu je veľmi podobná časovej zložitosti predchádzajúceho útoku, keďže aj v tomto prípade generujeme dva výstupy, funkciu konkatenácie máme zadarmo a jednými problémom môže byť spätné točenie. No v prípade spätného točenia o 1, respektíve pár bitov, je tento problém zanedbateľne malý. Tým pádom je celková časová zložitosť podľa [32] $\sum_{j=1}^{s}[q+(j-1)qv+(r-1)qsv]v = tsv + \frac{t}{r} - \frac{tsv}{2r} - \frac{tv}{2r}$, pri označení z kapitoly 2, výpočtov funkcie f, ktorá odpovedá spätnému točeniu šifry A5/1 o jeden bit, pridanie jedného bitu a zamiešanie šifry A5/1, takzvaným mixovaním kľúča uskutočnením dva krát a vygenerovaním výstupu o veľkosti 64 bitov. V tomto prípade ale nepotrebujeme generovať dva výstupy plnej dĺžky ako to bolo v prípade predchádzajúceho útoku, pretože dĺžka 64 bitov je generovaná len jeden krát.

Posledný problém, ktorý je potrebné vyriešiť je, ako je potrebné vybrať predchádzajúci vnútorný stav pri spätnom točení šifry, keďže podľa druhej kapitoly vieme, že spätné točenie môže vygenerovať niekoľko výstupov. V našom prípade potrebujeme vyberať stále ten istý stav zo spätného točenia, čo je možné dosiahnuť tým, že všetky možné stavy si usporiadame a vyberieme ten najmenší.

Hlavnou výhodou daného útoku je, že je potrebné poznať len časť správy v danom frame number, čo je obvyklý prípad v mobilných sieťach. Taktiež, v porovnaní s predchádzajúcim útokom nie sme viazaní na fixované frame number, pričom stačí len nájsť dve po sebe nasledujúce správy, ktorých časť poznáme. Okrem toho je možné daný útok rozšíriť pre ľubovoľnú vhodnú časť šifrovaného textu zašifrovaného jedným frame number.

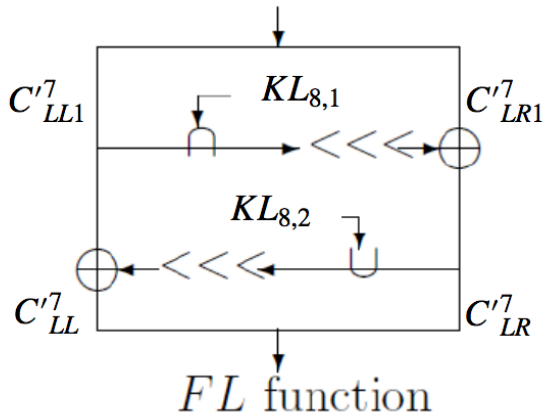
4.4 Vylepšenie filtrovania diferenčnej kryptoanalýzy šifry KASUMI

Autor útoku známeho ako Related key Rectangle attack [15] nepopisuje možnosti filtrovania na základe ľavej strany šifrovaného textu po kole číslo 7. Totižto, podľa predpokladov, predpokladáme, že diferenčná kryptoanalýza v danom útoku spĺňa poslednú diferenciu $\Delta K_{ac} = (0,0,0,0,0,0,1,0)$ a $\gamma = (0_x, 00200000_x) \rightarrow (0_x, 00200000_x) = \delta$. Ak si označíme šifrovaný text, ktorý máme k dispozícií po útoku zo sekcie 3.5.3, časti 1, ako $(C_{iLL}, C_{iLR}, C_{iRL}, C_{iRR})$ a šifrovaný text po siedmom kole ako $(C^7_{iLL}, C^7_{iLR}, C^7_{iRL}, C^7_{iRR})$, pre $i \in \{a,b,c,d\}$, potom šifrovaný text (C^7_{iLL}, C^7_{iLR}) musí so šifrovaným textom (C^7_{jLL}, C^7_{jLR}) spĺňať diferenciu rovnú 0 pre nejaké, podľa útoku popísane i a j. Tento predpoklad nie je v celom algoritme útoku popísaný, a preto je ho možné využiť na filtrovanie takým spôsobom, že budeme pri odhadovaní kľúča v časti 2 útoku Related Key Rectangle attack postupovať tak, že budeme overovať jednotlivé uhádnuté kľúče, či spĺňajú potrebnú diferenciu.

Podľa [15] platí, že po siedmom kole $C^7_{aLL} \oplus C^7_{cLL} = C^7_{bLL} \oplus C^7_{dLL} = (0000)$. Obdobne to platí aj pre pravú časť šifrovaného textu C^7_{iLR} ľavej polovice, pre $i \in \{a,c\}$. Daná diferencia znamená, že šifrovaný text má byť rovný. Totižto ide o funkciu XOR, ktorá je rovná a XOR $b = 0$ len a len vtedy, ak platí, že $a = b$. Samozrejme, v prípade útoku je daná diferencia rovná len po siedmom kole, nie po ôsmom, tj. v časti, kde máme šifrovaný text k dispozícií. Diferencia v poslednom kole neplatí, tú nepoznáme, no vieme ju vypočítať, keďže máme k dispozícií šifrovaný text (v útoku sa ukladá len pravá časť šifrovaného textu, to je možné jednoducho zmeniť). Ďalej vieme, že (C_{iLL}, C_{iLR}) má veľkosť 32 bitov (keďže ide o ľavú polovicu celého šifrovaného textu, ktorý má veľkosť 64 bitov). Po siedmom kole následne platí, že $C^7_{iLL} \oplus C'^7_{iLL} = C_{iLL}$, kde C'^7_{iLL} je čiastočný výstup z funkcie $FL8$ a obdobne platí, že $C^7_{iLR} \oplus C'^7_{iLR} = C_{iLR}$, kde C'^7_{iLR} je čiastočný výstup z funkcie $FL8$.

Ďalej budeme používať označenie z obrázka číslo 22 a pôvodné označenie podľa pôvodného útoku. Funkcia $FL8$ počíta funkcie AND a OR a XOR kľúča $KL_{8,1}$ a $KL_{8,2}$ o veľkosti 16 bitov každý. Celkovo funkcia $FL8$ pridá 32 bitov kľúča k výstupu z funkcie FO, ktoré následne funkciou XOR pridá k (C^7_{iLL}, C^7_{iLR}). Ak vezmeme časť šifrovaných textov (C_{iLL}, C_{iLR}), ktorých je podľa [15] 2^{38} a vygenerujeme pre jeden zvolený šifrovaný text (C_{aLL}, C_{aLR}) všetky možnosti kľúčov $KL_{8,1}$ a $KL_{8,2}$, dostávame celkovo $2^{38} \cdot 2^{32} = 2^{70}$ možností. Keďže poznáme diferenciu medzi ΔK_{ac}, tak vieme aj všetkých 2^{32} kľúčov pre (C_{cLL}, C_{cLR}) (generovanie kľúčov je lineárne [15]). Obdobne vieme vygenerovať aj (C_{bLL}, C_{bLR}) z čoho okamžite získavame možné kľúče pre (C_{dLL}, C_{dLR}). Celková časová zložitosť v tomto bode je $2^{70} + 2^{70} = 2^{71}$, čo je stále menej ako celková časová

zložitosť útoku 2^{78}. V prípade, že generovanie kľúča $KL_{8,2}$ postavíme tak, že nevygenerujeme kľúč, ale vygenerujeme možnú hodnotu C'^{7}_{aLR}, ktorá má veľkosť 2^{16} a pre každú z týchto hodnôt vygenerujeme kľúč $KL_{a,8,2}$, celkovo máme $2^{38} \cdot 2^{16} \cdot 2^{16} = 2^{70}$ možných hodnôt a obdobne vygenerujeme aj hodnotu C'^{7}_{bLR} a kľúče $KL_{b,8,2}$ získavame okamžite, keďže pre dané kľúče musí platiť diferencia popísaná v útoku. Pre generovanie takýchto párov máme celkovú časovú zložitosť 2^{71}, z čoho okamžite vieme vypočítať hodnoty $KL_{c,8,2}$ a $KL_{d,8,2}$, pričom, vzhľadom na to, že poznáme šifrovaný text na ľavej strane C_{aLR} a C_{cLR}, tým aj jeho diferenciu, vieme vypočítať aj hodnotu C'^{7}_{cLR}. Dané tvrdenie vyplýva z toho, že ak platí diferencia pre 7 kolo, to znamená, že $C^{7}_{aLR} \oplus C^{7}_{cLR} = 0$ a taktiež aj platí, že $C_{aLR} \oplus C_{cLR} = C^{7}_{aLR} \oplus C'^{7}_{aLR} \oplus C^{7}_{cLR} \oplus C'^{7}_{cLR}$, pričom hodnota $C^{7}_{aLR} \oplus C^{7}_{cLR} = 0$ a hodnoty C_{aLR}, C_{cLR} sú známe, kde hodnotu C'^{7}_{aLR} sme generovali, tak z danej rovnice vieme vyjadriť aj hodnotu C'^{7}_{cLR} (funkcia XOR je komutatívna a asociatívna). Obdobne to platí aj pre druhý pár medzi častami šifrovaného textu označované ako b, d. Týmto spôsombom môžeme indexovať tabuľku vypočítaných hodnôt pre C'^{7}_{aLR} a C'^{7}_{cLR} na základe výstupnej hodnoty pre nejaký kľúč do funkcie, kde sa pridáva kľúč $KL_{8,2}$ (označujeme to všeobecne, je to zvlášť aj pre $KL_{a,8,2}$ a $KL_{b,8,2}$). Vzhľadom na krok 2(b), vieme z tejto tabuľky na základe uhadnutého kľúča (časti bitov kľúča) a hodnoty výstupnej diferencie podľa kroku 2(b) z indexovej množiny vybrať možnú hodnotu C'^{7}_{aLR} a C'^{7}_{cLR}, respektíve C'^{7}_{bLR} a C'^{7}_{dLR}. Túto hodnotu vieme na základe indexu vzhľadom na výstupnú diferenciu a kľúč, ktorý sme navrhli v kroku 2(b) získať okamžite. Týmto výpočtom získavame vstupné a hlavne výstupné diferencie do funkcie, kde sa pridáva $KL_{8,1}$ (všeobecne, pre a aj b). Totiž, pre vstupnú diferenciu medzi C'^{7}_{aLL1} a C'^{7}_{cLL1} platí, že $C'^{7}_{aLL1} \oplus C'^{7}_{cLL1} = C'^{7}_{aLL} \oplus C'^{7}_{cLL} \oplus ((C'^{7}_{aLR}$ OR $KL_{a8,2}) <<<) \oplus ((C'^{7}_{cLR}$ OR $KL_{c8,2}) <<<)$, kde $C'^{7}_{aLL} \oplus C'^{7}_{cLL}$ je známe zo šifrovaných textov.



Obr. 22: Označenie časti funkcie FL.

Takto si overíme výstupnú diferenciu z kola, kde sa pridáva kľúč $KL_{8,1}$ čím si

overíme predpoklady, či kľúč, ktorý v kole 2(b) používame z kola 2(a) je správny a spĺňa všetky diferencie. Ak nie, tento kľúč vyhodíme. Takže na základe predpokladov, že daný šifrovaný text spĺňa diferenciu na ľavej strane po 7 kole vzhľadom na odhadnutý kľúč overíme, že daný šifrovaný text spĺňa predpoklady diferencií z útoku. Toto patrí medzi základné princípy filtrovania a keďže iba $2^{38}/2^{32}$ šifrovaných textov spĺňa diferenciu na ľavej strane [15], tj. celkovo 64 šifrovaných textov spĺňa diferenciu, to znamená, že daný kľúč filtrujeme s veľkou pravdepodobnosťou. Tento útok filtruje jednu jedinú nejednoznačnosť z obrázka číslo 14. Tým pádom, celková časová zložitosť je podľa [15] $2^{38} \cdot 2^{32} = 2^{70}$ (štvoríc, možných kľúčov), ktoré navrhujú, naznačujú $2^{70} \cdot 7/16^{16} = 2^{51}$ podkľúčov pre 48 bitový kľúč, po kroku 2(a) a po kroku 2(b) je to $2^{51} \cdot 2^{32} \cdot 7/16^{16} = 2^{63}$. Optimalizáciou popísanou v práci [15] je možné dostať sa na maximálnu časovú zložitosť v kroku 2, čo je 2^{71} v našom prípade, kde potrebujeme v kroku 2 vygenerovať príslušnú tabuľku pre všetky možnosti kľúčov $KL_{a,8,2}$ a hodnoty C'^{7}_{aLR} a C'^{7}_{bLR} nezávisle, to je celkovo $(2^{38} \cdot 2^{16} \cdot 2^{16}) \cdot 2$.

Tento útok sme priamo konzultovali s autorom pôvodného útoku, ktorý potvrdil správnosť našich postupov, no na druhej strane nie je možné tento prístup využiť pri útoku typu Sandwich attack a neznižuje časovú zložitosť tak, že je možné tento útok predviesť v reálnom čase, respektíve overiť dané predpoklady experimentálne. Pôvodný útok bol predchodcom Sandwich attack, v ktorom boli čiastočne overené aj predpoklady pôvodného Related key Rectangle attack, keďže Sandwich attack využíva pôvodné vlastnosti, ktoré sú použité pri Rectangle attack s pridaním ďalších nových spôsobov.

Záver

Mobilné siete sa s príchodom nových technológií stávajú čím ďalej tým viac využívané a pokrok v moderných technológiách nám dáva väčšie možnosti využívania týchto sietí. Vzhľadom na to, že mobilné siete používajú rôzne zastaralé kryptografické primitíva, je možné vytvoriť útoky, ktorými je možné útočiť na kryptografické primitíva priamo, bez toho aby sme využívali útoky pomocou iných kanálov. Pri tom, aký je rýchly vývoj mobilných sieti, sa v súčasnosti zabúda na bezpečnosť zariadení a protokolov.

Jedným z cieľov práce bolo popísať a analyzovať súčasné možnosti kryptoanalýzy v mobilných sieťach, aj vzhľadom na praktickú implementáciu, čo sme splnili v prvých troch kapitolách. Keďže ide o komplikované procesy útoku a existuje niekoľko rôznych kryptografických primitív, tieto útoky nie je možné detailnejšie rozpracovať, poskytnúť dôkazy jednotlivých tvrdení priamo v práci a venovať sa všetkým kryptografickým primitívam, ktoré sa používajú v mobilných sieťach. Preto sme si vybrali dve najčastejšie kryptografické primitíva, ktoré sa používajú najviac. Ide o šifru A5/1 a šifru KASUMI. Šifra A5/1 je zodpovedná za šifrovanie hlasového hovoru a textových správ, pričom šifra KASUMI sa používa v štandardoch 3G, ktoré sa väčšinou používajú na internetovú komunikáciu.

Vybrané základné kryptografické primitíva sme ďalej analyzovali a popísali možnosť útoku na tieto kryptografické systémy, ako z praktického hľadiska, tak aj z teoretického hľadiska. V súčasnosti neexistuje efektívny spôsob kryptoanalýzy šifry KASUMI v reálnom čase tak, aby ho bolo možné využiť v takom princípe, v akom sa používa šifra KASUMI v mobilných sieťach. Na druhej strane, šifra A5/1 je z pohľadu kryptoanalytikov považovaná za prelomenú. Poskytovatelia mobilných sietí využívajú rôzne metódy, ktoré sa snažia kryptoanalýzu šifry A5/1 znemožniť, a to tým, že implementujú protokoly tak, že nie je možné efektívne uhádnuť správu, ktorá je šifrovaná šifrou A5/1.

Na základe toho sme analyzovali možnosti kryptoanalýzy šifry A5/1, pričom sme popísali nové metódy kryptoanalýzy tejto šifry, a to tak, že aj napriek snahe mobilných operátorov vieme efektívnejšie uhádnuť otvorený text, ktorý bol šifrou A5/1

zašifrovaný. Popísali sme metódy generovania a prechodu nových typov dúhových tabuliek, pričom práve tento princíp nových dúhových tabuliek je možné využiť aj v praxi pri praktickom kryptoanalytickom útoku. Tieto tabuľky nie sme schopní generovať, pretože vyžadujú dostatočný výpočtový výkon, ktorý k dispozícií nemáme. Na druhej strane by bolo vhodné dané dúhové tabuľky otestovať tak, že sa tieto tabuľky vygenerujú a útok sa bude simulovať, čím sa overia vlastnosti, ktoré sme v práci popísali pri väčšom pokrytí dúhových tabuliek.

Následne sme analyzovali možnosti kryptoanalýzy šifry KASUMI, kde sme popísali novú metódu využívania diferencie, ktorá v pôvodnom útoku nebola spomenutá. Túto metódu útoku sme konzultovali priamo s tvorcom pôvodného útoku, ktorý naše závery potvrdil. Tým sme znížili časovú zložitosť celého útoku, pričom je ale potrebné zvýšiť priestorovú zložitosť. Tento útok aj vzhľadom na to, že je časová zložitosť nižšia, nie je možné reálne implementovať, pretože čas potrebný na kryptoanalýzu je vzhľadom na súčasné technológie nevypočitáteľný v reálnom čase.

Popísaním nových útokov sme naplnili aj tretí ciel práce, tým bolo navrhnutie nových metód kryptoanalýzy šifier, používaných v mobilných sieťach. Vzhľadom na nedostatok výpočtového výkonu odporúčame implementovať a ďalej rozširovať dúhové tabuľky, ktoré sme popísali v kapitole 4 a následne overiť metódu kryptoanalýzy a časové zložitosti na dúhových tabuľkách s pokrytím aspoň 50 %.

Na to, aby vyššie popísané útoky neboli efektívne, je nutné zastaviť používanie mobilných sieti typu 2G, a zároveň zastaviť možnosť spätnej kompatibility a tieto siete prestať používať. Šifra KASUMI je v súčasnosti neprelomiteľná, takže by bolo dostačujúce používať mobilné siete typu 3G, pričom budúcnosť vidíme vo využívaný najnovších mobilných sieti 4G a 5G, kde sa využívajú šifry ako AES, ktoré sa už dlhé roky považujú za bezpečné. Na druhej strane, operátori stále využívajú siete typu 2G na mobilnú hlasovú komunikáciu a na komunikáciu pomocou textových správ, aby siete typu 3G a 4G mali dostatočnú priepustnosť a nezvyšovalo sa ich zaťaženie na komunikáciu so sieťou internet.

Zoznam použitej literatúry

[1] Anwar, N., Riadi, I., and Luthfi, A. Forensic sim card cloning using authentication algorithm. *International Journal of Electronics and Information Engineering 4*, 2 (2016), 71–81.

[2] Barkan, E., Biham, E., and Keller, N. Instant ciphertext-only cryptanalysis of gsm encrypted communication. *Journal of Cryptology 21*, 3 (2008), 392–429.

[3] Benchoff, B. Cracking gsm with rtl-sdr for thirty dollars. Dostupné na internete: `https://hackaday.com/2013/10/22/cracking-gsm-with-rtl-sdr-for-thirty-dollars/`, 2013. [cit. 22. 02. 2018].

[4] Biham, E., Dunkelman, O., and Keller, N. A related-key rectangle attack on the full kasumi. In *International Conference on the Theory and Application of Cryptology and Information Security* (2005), Springer, pp. 443–461.

[5] Biham, E., and Shamir, A. *Differential cryptanalysis of the data encryption standard*. Springer Science & Business Media, 2012.

[6] Biryukov, A., Shamir, A., and Wagner, D. Real time cryptanalysis of a5/1 on a pc. In *International Workshop on Fast Software Encryption* (2000), Springer, pp. 1–18.

[7] Blunden, M., and Escott, A. Related key attacks on reduced round kasumi. In *International Workshop on Fast Software Encryption* (2001), Springer, pp. 277–285.

[8] Bogdanov, A., Khovratovich, D., and Rechberger, C. Biclique cryptanalysis of the full aes. In *International Conference on the Theory and Application of Cryptology and Information Security* (2011), Springer, pp. 344–371.

[9] Briceno, M., Goldberg, I., and Wagner, D. A pedagogical implementation of the gsm a5/1 and a5/2 "voice privacy" encryption algorithms. *Originally published at http://www. scard. org, mirror at http://cryptome. org/gsm-a512. htm 26* (1999).

[10] Casanovas, F. Cracking and sniffing gsm with a rtl-sdr. Dostupné na internete: https://ferrancasanovas.wordpress.com/cracking-and-sniffing-gsm-with-rtl-sdr-concept/, 2014. [cit. 22. 02. 2018].

[11] Daemen, J., and Rijmen, V. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013. ISBN: 978-3-662-04722-4.

[12] Dahlman, E., Parkvall, S., Skold, J., and Beming, P. *DAHLMAN, Erik, et al. 3G evolution: HSPA and LTE for mobile broadband*. Academic press, 2010. ISBN: 978-01-237-2533-2.

[13] Dohmen, J. R., and Olaussen, L. S. Umts authentication and key agreement. Master's thesis, Agder University College, Universitetet i Agder - Fakturamottak, Postboks 383 Alnabru, 0614 Oslo, 2001. Dostupné na internete: https://brage.bibsys.no/xmlui/bitstream/handle/11250/137418/master_ikt_2001_dohmen.pdf?sequence=1, [cit. 12. 01. 2018].

[14] Dunkelman, O., Keller, N., and Shamir, A. A practical-time related-key attack on the kasumi cryptosystem used in gsm and 3g telephony. In *Annual Cryptology Conference* (2010), Springer, pp. 393–410.

[15] Dunkleman, O., and Biham, E. *Techniques for Cryptanalysis of Block Ciphers*. Springer, 2018. ISBN: 978-3-642-17232-8.

[16] Gendrullis, T., Novotnỳ, M., and Rupp, A. A real-world attack breaking a5/1 within hours. In *International Workshop on Cryptographic Hardware and Embedded Systems* (2008), Springer, pp. 266–282.

[17] Gendrullis, T., Novotnỳ, M., and Rupp, A. A real-world attack breaking a5/1 within hours. 266–282.

[18] Golić, J. D. Cryptanalysis of alleged a5 stream cipher. 239–255.

[19] Hellman, M. A cryptanalytic time-memory trade-off. *IEEE transactions on Information Theory 26*, 4 (1980), 401–406.

[20] Hrach, J. Útoky na lfsr šifry. Dostupné na internete: https://brmlab.cz/project/gsm/deka/attack-theory, 2016. [cit. 22. 02. 2018].

[21] Hussain, S. R., Chowdhury, O., Mehnaz, S., and Bertino, E. Lteinspector: A systematic approach for adversarial testing of 4g lte. In *Network and Distributed Systems Security (NDSS) Symposium 2018* (2018).

[22] Kelsey, J., Kohno, T., and Schneier, B. Amplified boomerang attacks against reduced-round mars and serpent. In *International Workshop on Fast Software Encryption* (2000), Springer, pp. 75–93.

[23] Kitsos, P., Selimis, G., and Koufopavlou, O. High performance asic implementation of the snow 3g stream cipher. *IFIP/IEEE VLSI-SOC* (2008), 13–15.

[24] Kitsos, P., Sklavos, N., Provelengios, G., and Skodras, A. N. Fpga-based performance analysis of stream ciphers zuc, snow3g, grain v1, mickey v2, trivium and e0. *Microprocessors and Microsystems 37*, 2 (2013), 235–245.

[25] Knudsen, L. R., and Robshaw, M. *The block cipher companion.* Springer Science & Business Media, 2011.

[26] Kröll, H., Benkeser, C., Zwicky, S., Weber, B., and Huang, Q. Baseband signal processing framework for the osmocombb gsm protocol stack. In *Wireless innovation forum European conference on communication technologies and software defined radio* (2012).

[27] Li, Z. Optimization of rainbow tables for practically cracking gsm a5/1 based on validated success rate modeling. In *Cryptographers' Track at the RSA Conference* (2016), Springer, pp. 359–377.

[28] Lu, J., Li, Z., and Henricksen, M. Time–memory trade-off attack on the gsm a5/1 stream cipher using commodity gpgpu. In *International Conference on Applied Cryptography and Network Security* (2015), Springer, pp. 350–369.

[29] Neethu, R., Sindhu, M., and Srinivasan, C. Xuba: An authenticated encryption scheme. In *Data Engineering and Intelligent Computing.* Springer, 2018, pp. 647–655.

[30] Nohl, K. Attacking phone privacy. *Black Hat USA* (2010), 1–6. Dostupné na internete: `https://opensource.srlabs.de/projects/a51-decrypt/`, [cit. 3. 01. 2018].

[31] Nohl, K., and Munaut, S. Wideband gsm sniffing. 27th Chaos Communication Congress. Dostupné na internete: `https://events.ccc.de/congress/2010/Fahrplan/events/4208.en.html`, [cit. 13. 03. 2018], 2010.

[32] Oechslin, P. Making a faster cryptanalytic time-memory trade-off. In *Annual International Cryptology Conference* (2003), Springer, pp. 617–630.

[33] OpenSource. Osmocombb. Dostupné na internete: `https://osmocom.org/projects/baseband`, 2010. [cit. 22. 02. 2018].

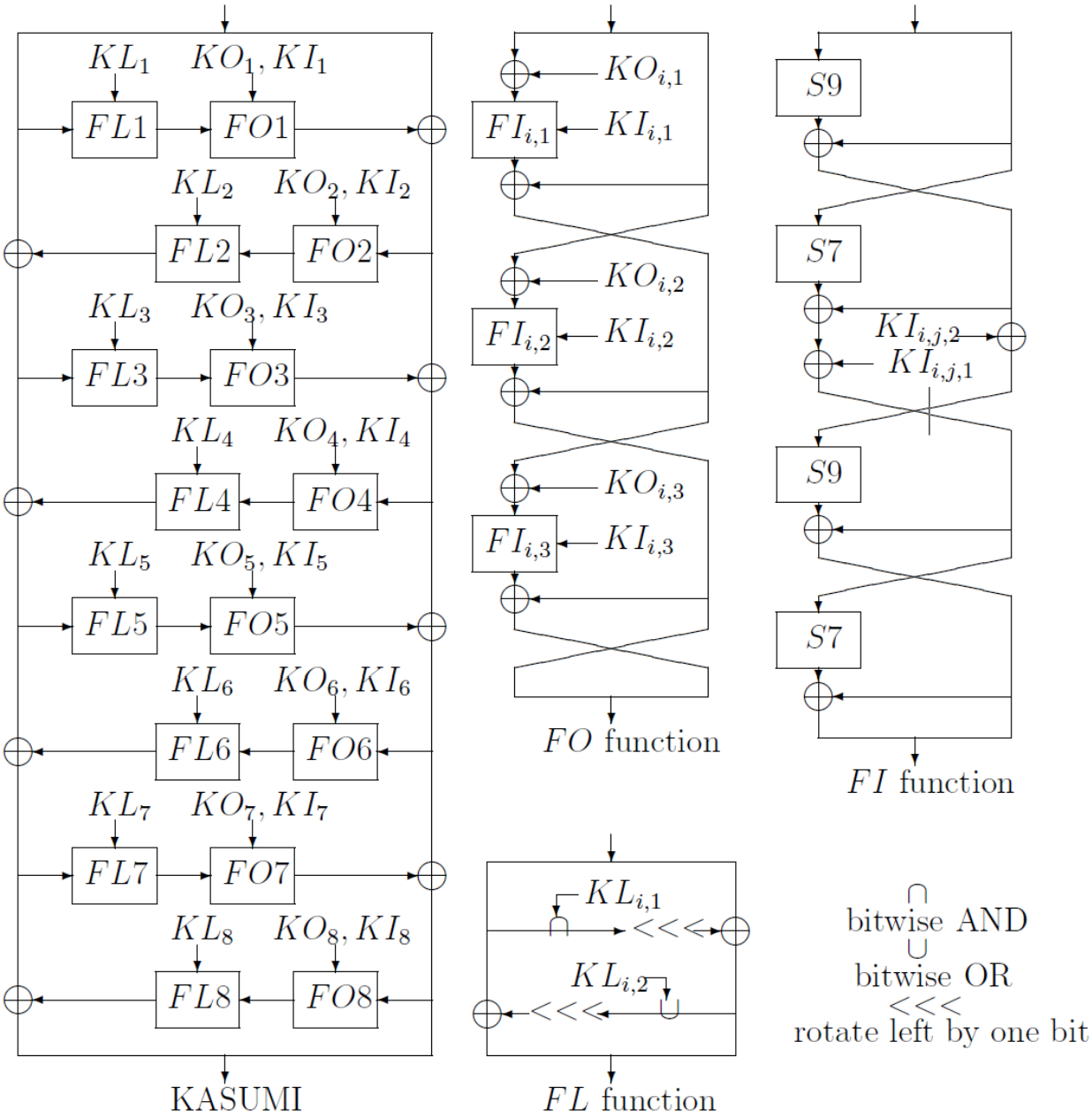
[34] Quirke, J. Security in the gsm system. *AusMobile, May* (2004), 1–26.

[35] Rhee, M. Y. *Mobile communication systems and security.* John Wiley and Sons, Reading, Massachusetts, 2009. ISBN: 978-0-470-82336-1.

[36] Wagner, D. The boomerang attack. In *International Workshop on Fast Software Encryption* (1999), Springer, pp. 156–170.

Prílohy

Príloha A: Schéma šifry KASUMI [15].



Príloha B: Schéma generovania kľúča šifry KASUMI [15].

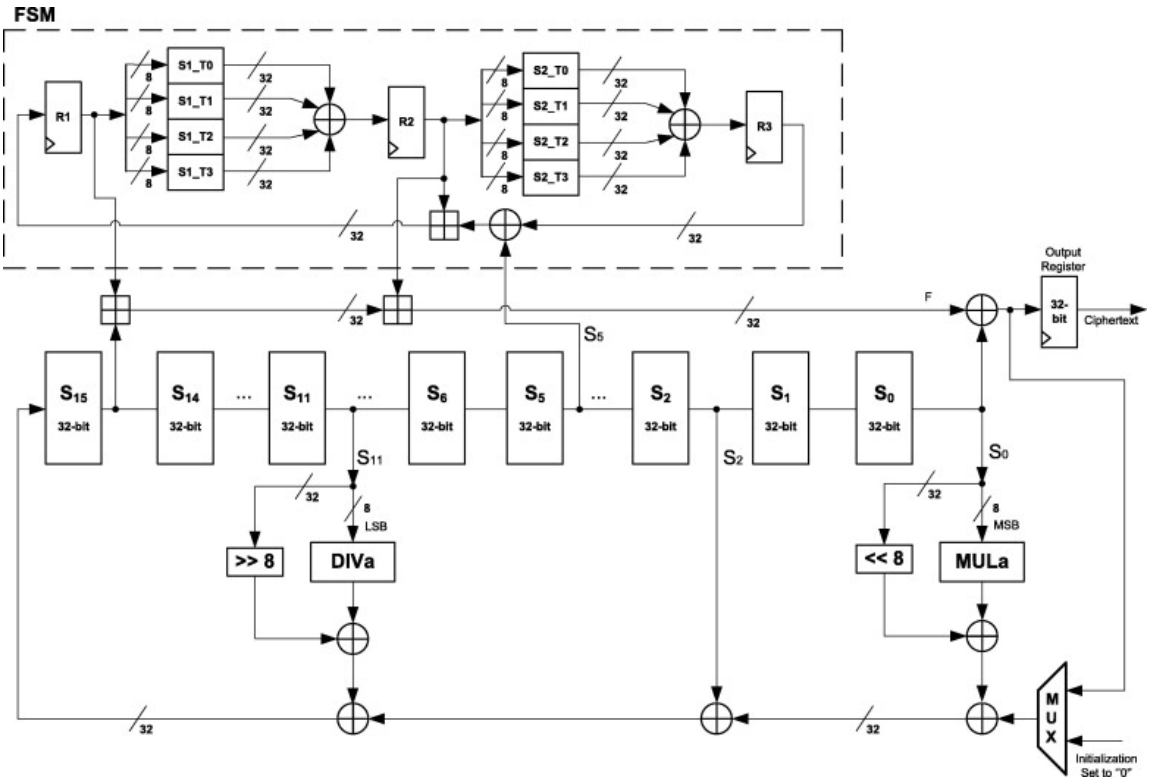
Round	$KL_{i,1}$	$KL_{i,2}$	$KO_{i,1}$	$KO_{i,2}$	$KO_{i,3}$	$KI_{i,1}$	$KI_{i,2}$	$KI_{i,3}$
1	$K_1 \lll 1$	K'_3	$K_2 \lll 5$	$K_6 \lll 8$	$K_7 \lll 13$	K'_5	K'_4	K'_8
2	$K_2 \lll 1$	K'_4	$K_3 \lll 5$	$K_7 \lll 8$	$K_8 \lll 13$	K'_6	K'_5	K'_1
3	$K_3 \lll 1$	K'_5	$K_4 \lll 5$	$K_8 \lll 8$	$K_1 \lll 13$	K'_7	K'_6	K'_2
4	$K_4 \lll 1$	K'_6	$K_5 \lll 5$	$K_1 \lll 8$	$K_2 \lll 13$	K'_8	K'_7	K'_3
5	$K_5 \lll 1$	K'_7	$K_6 \lll 5$	$K_2 \lll 8$	$K_3 \lll 13$	K'_1	K'_8	K'_4
6	$K_6 \lll 1$	K'_8	$K_7 \lll 5$	$K_3 \lll 8$	$K_4 \lll 13$	K'_2	K'_1	K'_5
7	$K_7 \lll 1$	K'_1	$K_8 \lll 5$	$K_4 \lll 8$	$K_5 \lll 13$	K'_3	K'_2	K'_6
8	$K_8 \lll 1$	K'_2	$K_1 \lll 5$	$K_5 \lll 8$	$K_6 \lll 13$	K'_4	K'_3	K'_7

$X \lll i$ — X rotated to the left by i bits

Príloha C: Konštanty generovania kľúča šifry KASUMI [15].

Round	1	2	3	4	5	6	7	8
Constant	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8
Value	0123_x	4567_x	$89AB_x$	$CDEF_x$	$FEDC_x$	$BA98_x$	7654_x	3210_x

Príloha D: Schéma šifry Snow3G [24].



Príloha E: Očakávaný čas čakania pri rozdielných frame number.

Vzdialenosť frame number	Čas v ms	Čas v h
1	$9,678358791 * 10^6$	$2,68$
2	$9,678363406 * 10^6$	$2,68$
14	$9,678418786 * 10^6$	$2,68$
200	$9,679277132 * 10^6$	$2,68$
2000	$9,687579775 * 10^6$	$2,69$
10000	$9,724394146 * 10^6$	$2,7$
20000	$1,055734211 * 10^7$	$2,8$
200000	$9,678363406 * 10^6$	$2,9$
$2^{21} - 1$	$1,451753242 * 10^7$	4
$2^{21} + 500$	$1,451753214 * 10^7$	4
$2^{21} + 5000$	$1,451750491 * 10^7$	4
$2^{21} + 5000$	$1,451742239 * 10^7$	4

Tabuľka 5: Prehľad značenia [15].