Sem vložte zadání Vaší práce.

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra počítačových systémů

Diplomová práce

Kryptoanalýza šifry Baby Rijndael

Bc. Josef Kokeš

Vedoucí práce: prof. Ing. Róbert Lórencz, CSc.

6. května 2013

Poděkování

Rád bych na tomto místě poděkoval především vedoucímu mé práce, prof. Ing. Róbertu Lórenczovi, CSc., za jeho vedení a veškerý čas, který mi při zpracování této práce věnoval. Dále děkuji mému bratru Janu Kokešovi za překreslení mých neumělých obrázků. V neposlední řadě děkuji mému nadřízenému Mgr. Janu Šípkovi za jeho porozumění a ochotu při kombinování mého studia a zaměstnání.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 6. května 2013 .....

Odkaz na tuto práci

Kokeš, Josef. *Kryptoanalýza šifry Baby Rijndael*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2013.

Abstract

This work presents Baby Rijndael, a cipher designed as a reduced Rijndael to facilitate its cryptanalysis. It shows that Baby Rijndael is a good approximation of Rijndael, performs linear cryptananalysis and extends the results to the full Rijndael.

Keywords AES, Rijndael, Baby Rijndael, linear cryptanalysis, cipher, key, key recovery.

Abstrakt

Práce prezentuje šifru Baby Rijndael, zmenšenou variantu Rijndael (AES) vhodnou pro kryptoanalýzu. Ukazuje, že je dobrou aproximací Rijndael, a poté na ní provádí útok technikou lineární kryptoanalýzy. Z výsledků vyvozuje závěry pro použití lineární kryptoanalýzy na Rijndael.

Klíčová slova AES, Rijndael, Baby Rijndael, lineární kryptoanalýza, šifra, klíč, nalezení klíče.

Obsah

Seznam obrázků

Úvod

Od svého vyhlášení vítězem soutěže o nový Advanced Encryption Standard (AES) v roce 2000 začala být šifra Rijndael používána v mnoha hardwarových i softwarových řešeních. Z toho důvodu se na ni soustředí pozornost mnoha kryptologů, kteří se ji snaží prolomit. Byla publikována řada prací o různých aspektech šifry a nalezených slabých místech, včetně několika návrhů možných nových útoků.

I já jsem si chtěl ověřit, že je Rijndael silná šifra, které lze důvěřovat. Bohužel však její síla činí důkladné prozkoumání jejích vlastností velmi výpočetně náročným. Nalezl jsem však šifru Baby Rijndael, která byla podle jejího autora C. Bergmana navržena podle stejných principů, ale s mnohem menší velikostí klíče, bloku i stavu, což umožňuje prověřit její vlastnosti i hrubou silou.

V této práci popíšu šifru Rijndael i šifru Baby Rijndael a ukážu, že Baby Rijndael skutečně je dobrým modelem šifry Rijndael a lze ji tedy použít pro efektivní kryptoanalýzu tak, aby výsledky bylo možné aplikovat i na plný Rijndael. Následně na této šifře provedu lineární kryptoanalýzu včetně všech jejích komponent, od analýzy S-boxu přes návrh lineární aproximace až po vyhledání posledního rundovního klíče pro různé varianty šifry, a ukážu případná slabá místa. Získané poznatky aplikuji i na šifru Rijndael.

Kapitola 1

DES a AES

Šifrovací algoritmus DES vzniknul na počátku 70. let 20. století v laboratořích IBM a po standardizačním procesu v NBS (National Bureau of Standards, Národní úřad pro standardy) se v roce 1977 stal platným americkým standardem pro šifrování dat. Po svém zveřejnění se rychle rozšířil i mimo Spojené státy a začal být používán pro mnoho aplikací.

V druhé polovině 90. let 20. století už bylo zřejmé, že DES je třeba nahradit. Hlavní příčiny byly tři:

- Prudce rostoucí výkon počítačů způsobil, že se délka klíče 56 bitů, jak ji používal DES, začala stávat kritickou slabinou DESu. Projekt DESCHALL, který prováděl distribuovaný útok hrubou silou, prolomil klíč v roce 1997 za 3 měsíce a získal vyhlášenou odměnu 10.000 USD; tím definoval čas a částku potřebnou pro prolomení DESu. V dalších letech výkon dále rostl, na 56 hodin v roce 1998 (DES Cracker) a 22,25 hodin na začátku roku 1999.[13] Rostoucí kapacity paměťových médií navíc začaly ohrožovat zvolenou délku bloku 64 bitů.

- Pokrok v kryptoanalýze umožnil nové, rychlejší útoky na DES než hrubou sílu: Diferenciální kryptoanalýzu zjevně tvůrci DESu znali a počítali s ní v návrhu šifry, lineární kryptoanalýza ale proti DESu fungovala.[9]

- Zatímco algoritmus DES byl veřejně znám, způsob jeho návrhu byl utajen. Toto utajení a další okolnosti provázející vznik a standardizaci DESu, zejména konzultace s NSA (National Security Agency, Národní bezpečnostní agentura), vzbuzovaly pochybnosti o tom, jestli DES neobsahuje backdoor (zadní vrátka), která umožní jeho snadné prolomení.

V reakci na tyto slabiny spustil americký NIST (National Institute of Standards and Technology, Národní institut pro standardy a technologii) veřejnou soutěž na nástupce DES, zvaného AES. Soutěž byla vyhlášena v září 1997 a zúčastnilo se jí 15 kandidátů, kteří byli hodnoceni z mnoha hledisek, včetně bezpečnosti, výkonu na různých hardwarových platformách nebo i jednoduchosti hardwarové implementace. Pět nejlepších postoupilo do druhého kola, kde byli podrobeni dalším náročným analýzám, jak vlastním tak autorů ostatních šifer a kryptologické komunity. 2. října 2000 byl oznámen výsledek: Pro AES byla zvolena šifra Rijndael, kterou navrhli Joan Daemen a Vincent Rijmen.[12]

Kapitola 2

Rijndael

Rijndael je symetrická bloková šifra. Její specifikace umožňují variabilitu v délce bloku[1] a v délce šifrovacího klíče[2], které mají vliv na konkrétní implementaci šifry, například na počet rund nebo detaily jednotlivých rundovních operací. Struktura šifry však zůstává stejná. Podrobná specifikace je uvedena v [2], zde se budu zabývat pouze variantou se 128bitovým blokem a 128bitovým klíčem, která je nejbližší šifře Baby Rijndael.

2.1 Struktura šifry

Struktura Rijndael je uvedena na obrázku 2.1:

Nejprve se z použitého šifrovacího klíče mechanismem expanze klíčů (Key expansion) vygeneruje $n + 1$ rundovních klíčů, kde n je počet rund a délka rundovního klíče je shodná s délkou bloku. Poté se na otevřený text aplikuje první vygenerovaný klíč v rámci operace `AddRoundKey`. Na vzniklý blok se postupně aplikuje n rund; ty se skládají z identické posloupnosti operací `SubBytes`, `ShiftRows`, `MixColumns`, `AddRoundKey`, kromě poslední rundy, ve které chybí operace `MixColumns`[3]. Výstup z poslední rundy pak tvoří šifrový text.

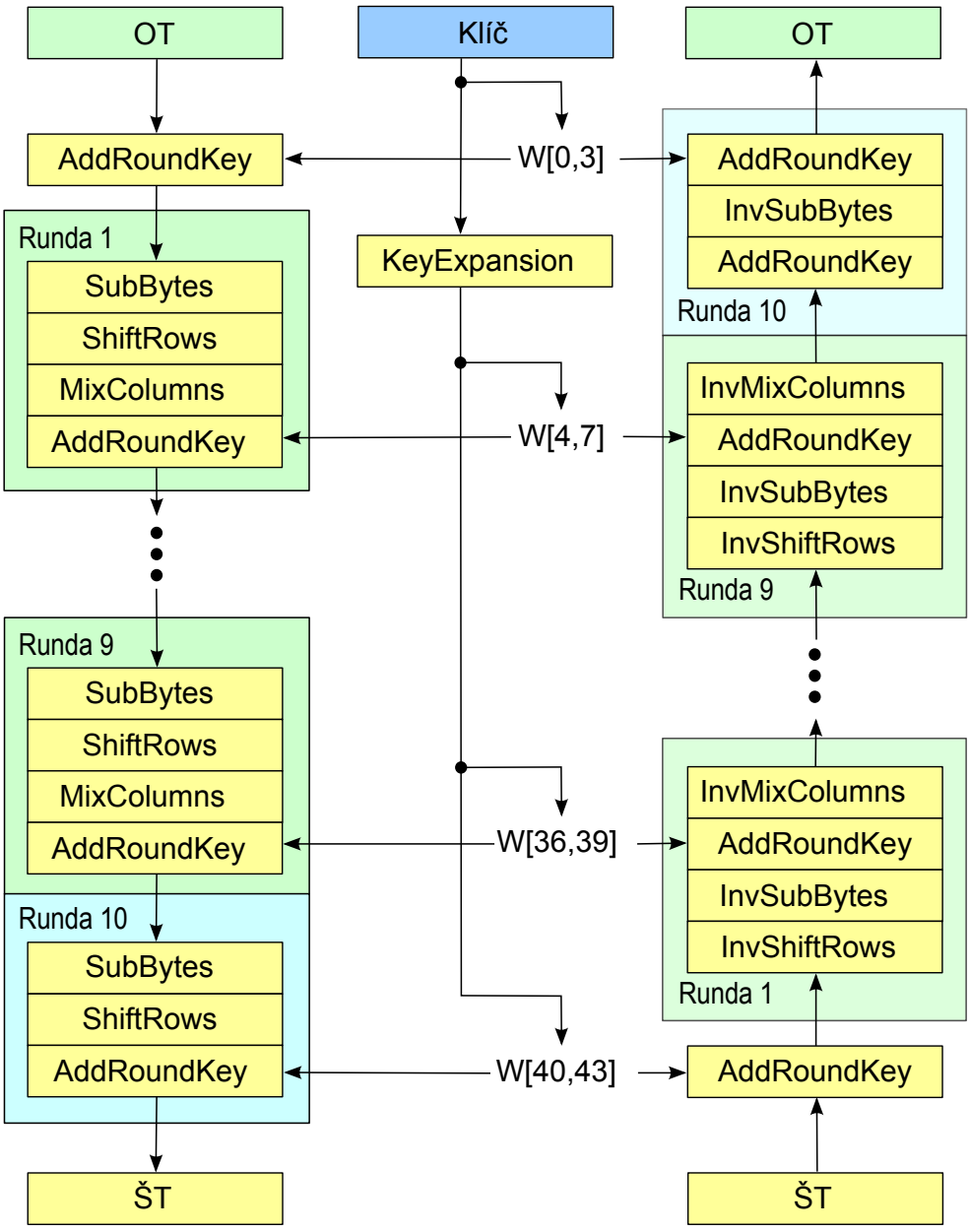
Počet rund závisí na délce klíče podle vztahu $n = 6 + DélkaKlíče/32$, pro šifru se 128bitovým klíčem jde tedy o 10 rund.

Ve všech operacích je blok uspořádán do matice bajtů o čtyřech řádcích a $DélkaBloku/32$ sloupcích, kde matice je vyplňována po sloupcích zleva

[1] 128 až 256 bitů v násobcích 32, tzn. 128, 160, 192, 224 nebo 256 bitů.

[2] Také 128 až 256 bitů v násobcích 32, nezávisle na délce bloku.

[3] Operace `MixColumns` je invertovatelná bez znalosti klíče a v poslední rundě šifry tak nepřidává žádnou bezpečnost.



Obrázek 2.1: Struktura šifry Rijndael. Převzato z [7, str. 28]

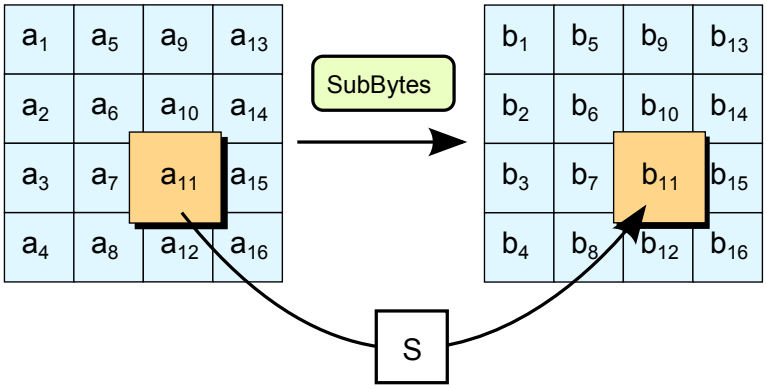
doprava a v rámci sloupce shora dolů:

$$A = \begin{pmatrix} a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \\ a_4 & a_8 & a_{12} & a_{16} \end{pmatrix}, a_i \in GF(2^8) \tag{2.1}$$

Tuto matici označujeme jako stav šifry, jednotlivé rundovní operace slouží k převedení původního stavu na stav nový.

2.2 SubBytes

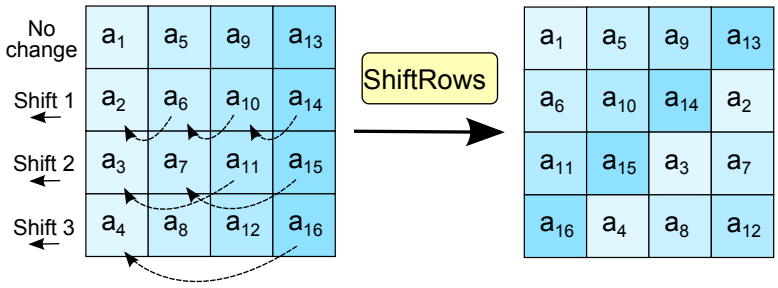
Operace `SubBytes` je jedinou nelineární komponentou šifry Rijndael. Spočívá v nahrazení každého bajtu stavu jiným bajtem. Přesný mechanismus náhrady je uveden v sekci srovnávající Rijndael s Baby Rijndael, v této fázi ho můžeme chápat jako náhradu pomocí substituční tabulky. Tak ostatně bývá běžně implementován.



Obrázek 2.2: Operace SubBytes v šifře Rijndael. Převzato z [7, str. 24]

2.3 ShiftRows

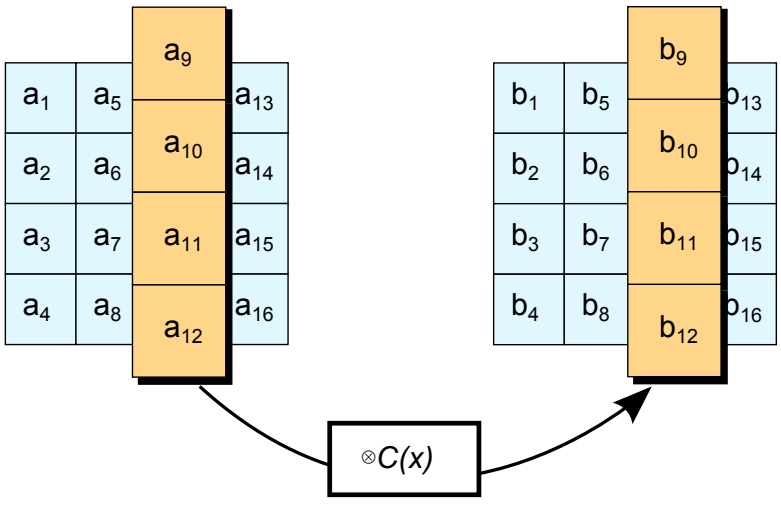
Operace `ShiftRows` provádí cyklický posuv jednotlivých řádků stavu o určený počet pozic doleva, kde velikost posuvu je závislá na pořadí řádku a na počtu sloupců: první řádek se nemění, druhý řádek rotuje o jednu pozici doleva, druhý řádek o dvě (4 a 5 sloupců) resp. tři (6 sloupců) pozice doleva a třetí řádek o tři (4 a 5 sloupců) resp. čtyři (6 sloupců) pozice doleva.



Obrázek 2.3: Operace ShiftRows v šifře Rijndael. Převzato z [7, str. 25]

2.4 MixColumns

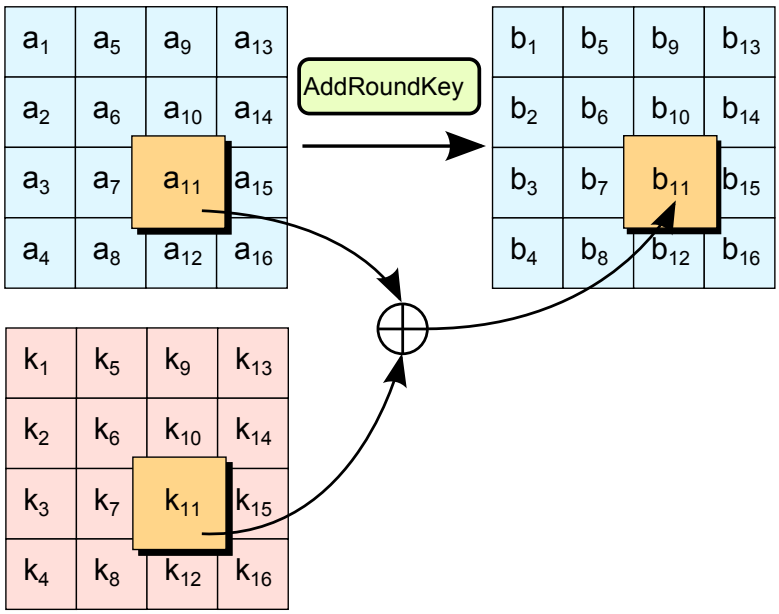
Operace `MixColumns` slouží k promíchání bitů stavu v rámci jednoho sloupce: Každý sloupec vstupního stavu je chápán jako vektor koeficientů polynomu $b(x) = b_0 + b_1x + b_2x^2 + b_3x^3$, sloupec výstupního stavu vznikne násobením $b(x)c(x) \mod x^4 + 1$, kde $c(x) = \texttt{02} + \texttt{01}x + \texttt{01}x^2 + \texttt{03}x^3$ a všechny koeficienty jsou chápány jako polynomy nad $GF(2^8)$. Přesný mechanismus bude popsán v sekci srovnávající Rijndael s Baby Rijndael.



Obrázek 2.4: Operace MixColumns v šifře Rijndael. Převzato z [2, str. 15]

2.5 AddRoundKey

V operaci `AddRoundKey` se ke stavu přičte (modulo 2) příslušný rundovní podklíč.



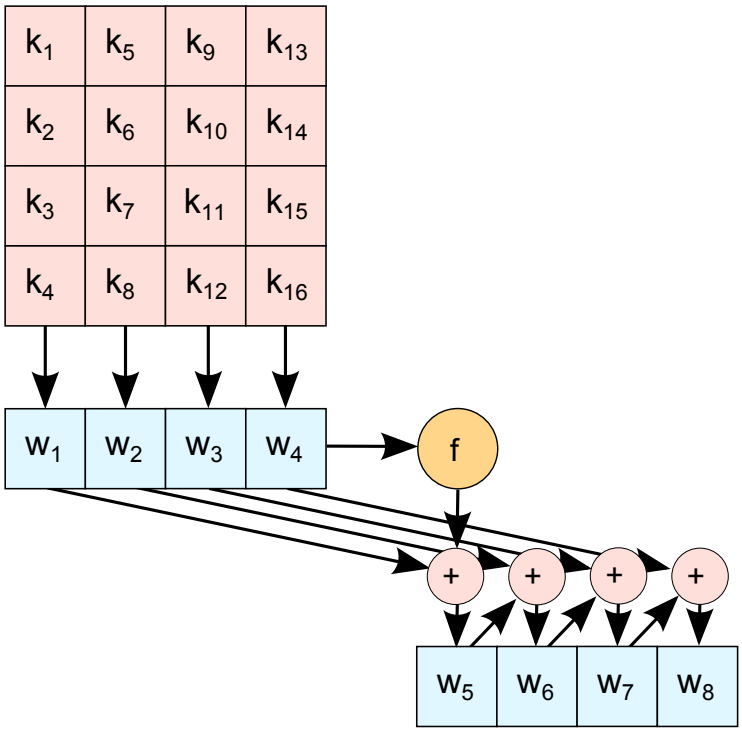
Obrázek 2.5: Operace AddRoundKey v šifře Rijndael. Převzato z [7, str. 27]

2.6 Expanze klíče

Expanze klíče slouží k vygenerování $n+1$ rundovních podklíčů požadované délky[4] z jednoho pevně daného hlavního klíče obecně odlišné délky[5]. Generování probíhá iterativně podle schématu daného obrázkem 2.6: Hlavní klíč přímo určuje prvních *DélkaKlíče* bitů klíčové posloupnosti. Další požadované bity posloupnosti se generují vždy z posledních 16 bajtů (128 bitů) k_1 až k_{16} posloupnosti, které se složí do 4 DWORDů (také 128 bitů) w_1 až w_4 a tyto za pomocí funkce f, která bude popsána ve srovnání s Baby Rijndael, vygenerují nové 4 DWORDy w_5 až w_8, které po rozložení na jednotlivé bajty vytvoří dalších 16 bajtů klíčové posloupnosti. Rundovní podklíče jsou

[4] dané zvolenou délkou bloku
[5] dané zvolenou délkou klíče

pak tvořeny odebráním příslušného počtu bajtů z vygenerované klíčové posloupnosti.



Obrázek 2.6: Key expansion šifry Rijndael. Převzato z [7, str. 30]

2.7 Dešifrování

Dešifrování probíhá obdobně jako šifrování, pouze jsou rundovní operace aplikovány v opačném pořadí a v podobě svých vlastních inverzí. Přitom:

- Expanze klíče je totožná jako v případě šifrování (obr. 2.6).
- `AddRoundKey` je sama svojí vlastní inverzí (obr. 2.5).
- `MixColumns` lze invertovat násobením polynomem $d(x) = \texttt{0E} + \texttt{09}x + \texttt{0D}x^2 + \texttt{0B}x^3$ (který je vůči $c(x)$ výše inverzní modulo $x^4 + 1$).
- `ShiftRows` invertujeme použitím rotování vpravo místo rotování vlevo, přičemž počet pozic zůstane shodný.

- `SubBytes` lze řešit inverzní tabulkou.

Pozn.: Proces dešifrování není pro řešený problém podstatný, proto je uveden jen v hrubých rysech. Podrobný popis nalezneme ve specifikaci Rijndael [2].

Kapitola 3

Baby Rijndael

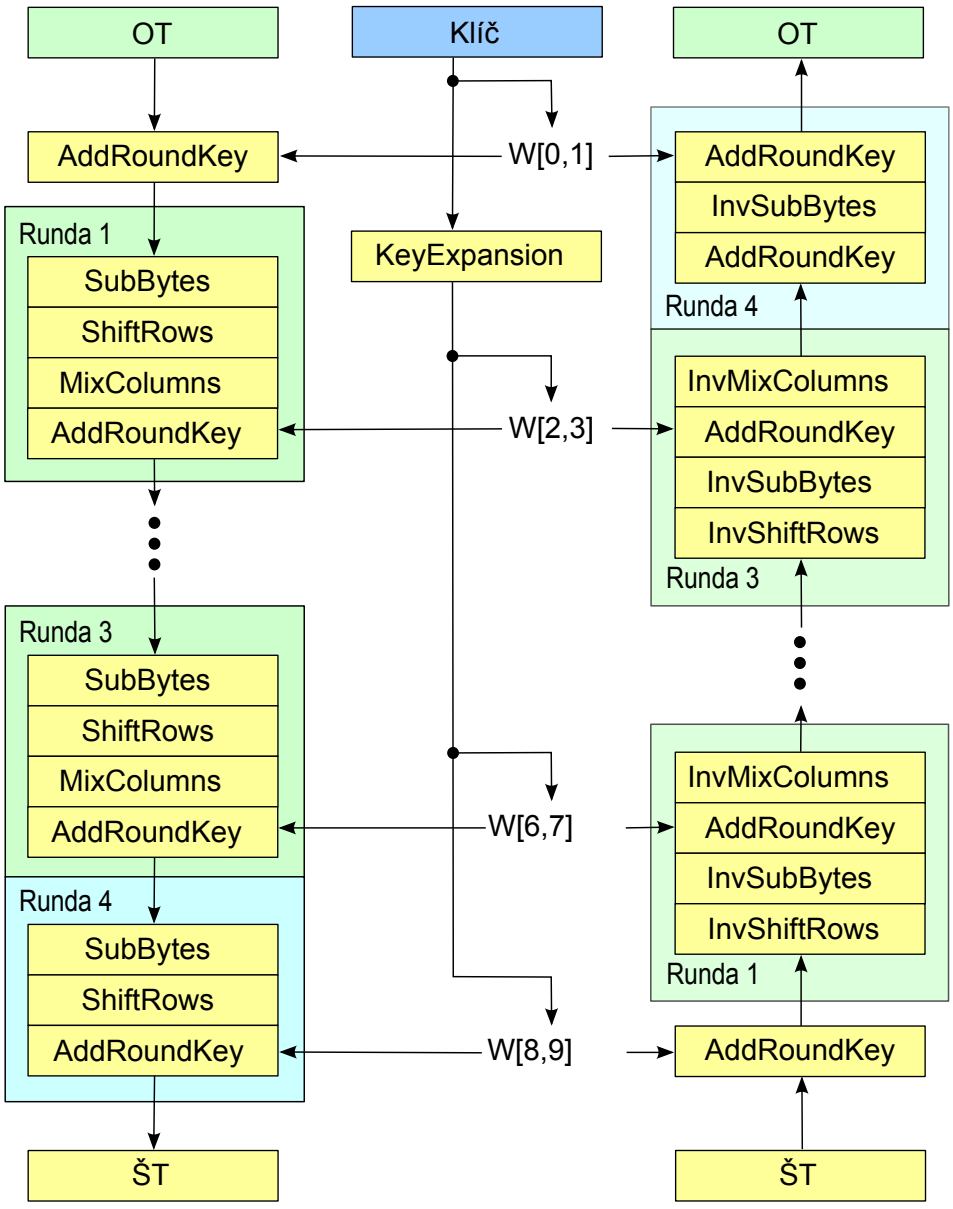
Rijndael, jak byl popsán v předchozí kapitole, se velmi dobře osvědčuje v reálných aplikacích a dobře odolává všem pokusům o prolomení. Tato vlastnost ho však také činí velmi obtížně studovatelným vzhledem k teoreticky možným útokům, které by podle předpokladů měly být rychlejší než útok hrubou silou, nicméně vyžadují takové množství operací, že i snížená výpočetní náročnost je stále příliš velká na to, aby se útok dal prakticky ověřit. To je například případ útoků na algebraickou podstatu Rijndael popsaných v [10] a [3].

Jednoduchá struktura Rijndaelu a podrobná dokumentace toho, jak byly vybírány jednotlivé parametry (viz [2]) však umožňuje poměrně snadno vytvářet odvozené varianty Rijndaelu s obdobnou strukturou a vlastnostmi, ale jinou velikostí. Jednou z těchto odvozených variant je i Baby Rijndael [1], který navrhl profesor Cliff Bergman z Iowa State University právě za účelem snazší analýzy vlastností šifry: Zatímco Rijndael používá nejméně 128bitový blok a 128bitový klíč, Baby Rijndael využívá mnohem menší 16bitové bloky a 16bitové klíče a tím umožňuje mimo jiné snadno vyzkoušet všechny možné kombinace otevřeného textu, klíče a šifrového textu.

3.1 Struktura šifry

Struktura Baby Rijndael (viz obr. 3.1) je totožná jako struktura Rijndael v tom, že se skládá z rund, ve kterých se provádí stále stejná posloupnost transformací. Rozdíl je, vedle mnohem menšího prostoru klíčů a otevřených i šifrových textů, jen v tom, že Baby Rijndael používá pouze 4 rundy. V případě potřeby však můžeme snadno přidávat rundy další.

Menší prostor otevřených textů je reprezentován menším stavem šifry: Zatímco Rijndael používá matici o čtyřech řádcích a čtyřech sloupcích, kde



Obrázek 3.1: Struktura šifry Baby Rijndael

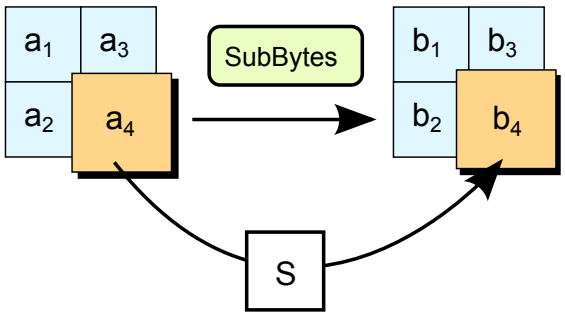
jednotlivými prvky jsou bajty (tzn. celkem $4 \times 4 \times 8 = 128$ bitů, Baby Rijndael používá matici o dvou řádcích a dvou sloupcích s prvky o délce čtyř bitů (tzn. celkem $2 \times 2 \times 4 = 16$ bitů). Struktura matice je však stejná, zejména v tom, že je naplňována po sloupcích zleva a v rámci sloupce po řádcích shora:

$$A = \begin{pmatrix} a_1 & a_3 \\ a_2 & a_4 \end{pmatrix}, a_i \in GF(2^4) \tag{3.1}$$

Pozn.: Odlišná velikost matice stavu Baby Rijndael je důsledkem kompromisu, kdy na jedné straně potřebujeme malý prostor klíčů a na straně druhé velkou variabilitu v `SubBytes`. Pokud by matice stavu měla mít čtyři řádky a čtyři sloupce jako v Rijndaelu, nutně narazíme na to, že buď bude prostor klíčů příliš velký (2^{64} v případě čtyřbitových prvků matice, 2^{48} v případě tříbitových prvků) nebo variabilita mezi prvky příliš malá (dva bity na prvek v případě prostoru 2^{16} klíčů); to by vedlo na náročnost výpočtů v prvním případě respektive falešné linearity v `SubBytes` v případě druhém. Zmenšená matice sice ovlivňuje operace `ShiftRows` a `MixColumns`, z hlediska kryptoanalýzy ale toto ovlivnění není podstatné, protože obě operace jsou plně lineární.

3.2 SubBytes

Operace `SubBytes` se chová v Baby Rijndaelu i v Rijndaelu stejně v tom, že jde o substituční tabulku, která transformuje samostatně každý prvek stavu na novou hodnotu.



Obrázek 3.2: Operace SubBytes v šifře Baby Rijndael

Podstatné ovšem je, jak tato substituční tabulka vzniká. Vzhledem k tomu, že operace `SubBytes` je v Rijndaelu jedinou nelineární komponentou, je dokonce její konstrukce naprosto klíčovou vlastností celé šifry. Musíme proto

ověřit, že `SubBytes` v Baby Rijndaelu má stejné vlastnosti jako `SubBytes` v Rijndaelu.

V Rijndaelu je `SubBytes` definována následujícím předpisem:[2, str. 13]

1. Označme vstupní bajt $a = a_0+2a_1+2^2a_2+...+2^7a_7, a_i \in \{0,1\}$. Tento bajt budeme chápat jako polynom $a(x) = a_0 + a_1x + a_2x^2 + ... + a_7x^7$.

2. Spočítáme multiplikativní inverzi $a(x)$ v $GF(2^8)$ vzhledem k ireducibilnímu polynomu $m(x) = 1 + x + x^3 + x^4 + x^8$. Bajt s hodnotou $a =$ `0x00` se mapuje sám na sebe.

3. Na tuto inverzi aplikujeme afinní transformaci nad GF(2) definovanou předpisem

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \quad (3.2)$$

kde $b \equiv a^{-1} \mod m(x)$ je výstup předchozího kroku. Tuto afinní transformaci lze vyjádřit také jako násobení polynomů:

$$b(x) = a(x)(1 + x^4 + x^5 + x^6 + x^7) + (x + x^2 + x^6 + x^7) \mod (x^8 + 1). \quad (3.3)$$

Autoři Rijndael uvádějí [2, str. 27, 28] svoji motivaci pro volbu těchto konkrétních parametrů:

- Polynom $m(x)$ je první v seznamu ireducibilních polynomů stupně osm v použité literatuře; to odpovídá nejmenšímu ireducibilnímu polynomu stupně 8, pokud tyto ireducibilní polynomy seřadíme vzestupně podle stupňů těch členů, které mají nenulový koeficient.

- Operace `SubBytes` musí být invertovatelná, měla by mít co nejmenší maximální netriviální korelaci mezi lineární kombinací vstupních a lineární kombinací výstupních bitů (obrana proti lineární kryptoanalýze) a co nejmenší maximální netriviální hodnotu v XOR tabulce (obrana proti diferenciální kryptoanalýze). Tomuto požadavku dobře vyhovuje operace inverze.

vstup	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
výstup	a	4	3	b	8	e	2	c	5	7	6	f	0	1	9	d

Tabulka 3.1: `SubBytes` pro Baby Rijndael

- Algebraický popis operace v $GF(2^8)$ by měl být co nejsložitější. Samotná inverze by se vyjádřila snadno, v kombinaci s afinní transformací uvedenou výše už ne.
- Operace by měla mít jednoduchý slovní popis.

Definice Baby Rijndael [1, str. 2] nepopisuje způsob, jak je tamní operace `SubBytes` konstruována, uvádí pouze tabulku výstupů pro všechny možné vstupní hodnoty (tabulka 3.1).

Podrobnější popis konstrukce lze nalézt v [14, str. 23]:

1. Označme vstupní čtveřici bitů $a = a_0 + 2a_1 + 2^2a_2 + 2^3a_3, a_i \in \{0, 1\}$. Tuto čtveřici budeme chápat jako polynom $a(x) = a_0 + a_1x + a_2x^2 + a_3x^3$.
2. Spočítáme multiplikativní inverzi $a(x)$ v $GF(2^4)$ vzhledem k ireducibilnímu polynomu $m(x) = 1 + x + x^4$. Čtveřice bitů s hodnotou $a = (0000)_2$ se mapuje sama na sebe.
3. Na tuto inverzi aplikujeme afinní transformaci nad GF(2) definovanou předpisem

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \tag{3.4}$$

   kde b je výstup operace `SubBytes`. Tuto afinní transformaci lze vyjádřit také jako násobení polynomů:

$$b(x) = a(x)(x + x^2 + x^3) + (x + x^3) \mod (x^4 + 1). \tag{3.5}$$

Je zřejmé, že postup výpočtu je velmi obdobný, musíme však ověřit rozdíly:

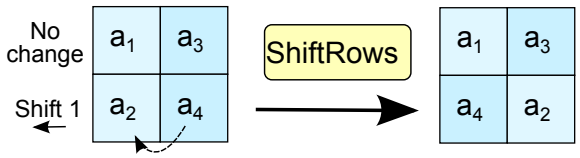
- Práce v $GF(2^4)$ místo $GF(2^8)$ je přirozeným důsledkem toho, že pracujeme s mnohem menším stavem, jehož prvky jsou z menšího tělesa. Tomu se nedá vyhnout, pokud chceme použité těleso zmenšit.

- Ireducibilní polynom $m(x) = 1 + x + x^4$ je první v seznamu ireducibilních polynomů čtvrtého stupně, tzn. byl vybrán podle stejných kritérií jako polynom v Rijndaelu.
- Operace inverze v `SubBytes` je totožná jako v případě Rijndaelu, tudíž by měla splňovat i stejné vlastnosti.
- Afinní transformace použitá v Baby Rijndael je invertovatelná ($(x + x^2 + x^3)(x + x^2 + x^3) = x^6 + x^4 + x^2 \equiv 1 \mod x^4 + 1$), multiplikativní složka má velké zastoupení vysokých mocnin a aditivní složka mění právě polovinu bitů, což jsou stejné charakteristiky jako u Rijndael.
- Slovní popis je totožný jako u Rijndaelu.

Výpočtem jsem také ověřil, že konverzní tabulka uvedená v [1, str. 2] je vytvořena podle předpisu uvedeného výše.

3.3 ShiftRows

Operace `ShiftRows` v Baby Rijndael vypadá na první pohled odlišně než u Rijndael, protože první řádek stavu zachová a v druhém prohodí sloupečky.



Obrázek 3.3: Operace ShiftRows v šifře Baby Rijndael

Snadno si ale uvědomíme, že pokud máme dvě hodnoty (a, b), pak rotace vlevo o jednu pozici, rotace vpravo o jednu pozici a prohození obou pozic jsou totožné operace s výsledkem (b, a). `ShiftRows` je tedy v Baby Rijndael definována stejně a také vyhovuje stanoveným požadavkům:[2, str. 29]

- První řádka stavu se nemění. Ostatní řádky rotují, a to každý o jiný počet pozic.
- Odolnost vůči útokům se zkrácenými diferenciály.
- Odolnost vůči Square útoku.

- Jednoduchost.

„Ostatní řádky“ jsou v kontextu Baby Rijndael, který má jen dva řádky, právě jen druhý řádek. Ten tedy musí rotovat, a to o odlišný počet pozic než řádek první (který „rotuje“ o nula pozic). To si vynucuje rotaci právě o jednu pozici. Jiná možnost neexistuje a ostatní požadavky jsou tak irelevantní. V případě analýzy Baby Rijndael na útoky se zkrácenými diferenciály a Square útoku je však třeba mít na paměti, že se zde nachází potenciální odlišnost od Rijndaelu a výsledky analýzy nemusí být pro plný Rijndael relevantní.

3.4 MixColumns

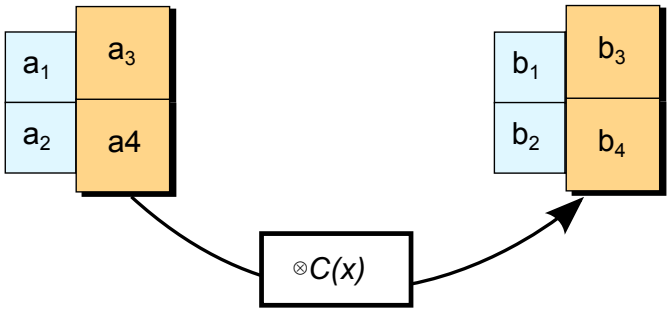
Operace `MixColumns` je v Baby Rijndael definována poněkud odlišně od definice v Rijndael, ale jen na první pohled: Rijndael definuje `MixColumns` jako násobení dvou polynomů modulo $x^4 + 1$ a ekvivalentně jako maticové násobení, s koeficienty polynomu resp. prvky matice z $GF(2^8)$. Baby Rijndael definuje `MixColumns` jako maticové násobení s prvky matice z $GF(2)$:

$$\begin{pmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix}, \tag{3.6}$$

kde vektor b je výsledný sloupec stavu šifry, vektor a je zdrojový sloupec stavu šifry ve tvaru $(a_{0,3}, a_{0,2}, a_{0,1}, a_{0,0}, a_{1,3}, a_{1,2}, a_{1,1}, a_{1,0})^T$, kde $a_{i,j}$ značí j-tý bit v i-tém řádku počítaného sloupce, a matice (označíme ji t) vyjadřuje operaci `MixColumns`.

Lze ale snadno ukázat, že násobící matici Baby Rijndael lze ekvivalentně vyjádřit jako polynom s koeficienty z $GF(2^4)$ i jako násobící matici s prvky z $GF(2^4)$, a obdobně polynom i matici Rijndael lze vyjádřit ve formě matice s prvky z $GF(2)$.

Buď $g(x) = g_0 + g_1 x$ s $g_0, g_1 \in GF(2^4)$ pevně daný polynom a $h(x) = h_0 + h_1 x$ s $h_0, h_1 \in GF(2^4)$ polynom. Pak:



Obrázek 3.4: Operace MixColumns v šifře Baby Rijndael

$$\begin{aligned} g \cdot h &= (g_0 + g_1 x)(h_0 + h_1 x) \\ &= (g_0 h_0) + (g_0 h_1 + g_1 h_0)x + (g_1 h_1)x^2 \\ &\equiv (g_0 h_0 + g_1 h_1) + (g_0 h_1 + g_1 h_0)x \mod (x^2 + 1). \end{aligned} \tag{3.7}$$

Buď matice $M = \begin{pmatrix} m_0 & m_1 \\ m_2 & m_3 \end{pmatrix}$ a vektor $h = \begin{pmatrix} h_0 \\ h_1 \end{pmatrix}$. Pak:

$$M \cdot h = \begin{pmatrix} m_0 h_0 + m_1 h_1 \\ m_2 h_0 + m_3 h_1 \end{pmatrix} \tag{3.8}$$

Má-li matice M reprezentovat operátor násobení pevným polynomem $g(x)$ modulo $x^2 + 1$, pak nutně musí platit $m_0 = m_3 = g_0$ a $m_1 = m_2 = g_1$ a tedy $M = \begin{pmatrix} g_0 & g_1 \\ g_1 & g_0 \end{pmatrix}$.

Vidíme, že matice t má skutečně tento tvar, kde:

$$g_0 = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{pmatrix}, g_1 = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{pmatrix} \tag{3.9}$$

g_0 a g_1 ovšem máme vyjádřené jako prvky z $A = GF(2)^4$, přičemž pro ukázání podobnosti s Rijndaelem potřebujeme prvky z $B = GF(2^4)$. Klíčem k převodu z A do B a opačně je věta 94 z [5, str. 30]: Buď $f \in K[x]$ ireducibilní polynom nad tělesem K. Potom existuje jednoduché algebraické rozšíření K s kořenem polynomu f jakožto definujícím prvkem.

Navazující příklad 95 [5, str. 30] ukazuje, jak toto jednoduché algebraické rozšíření sestrojit: Uvažujeme těleso K a ireducibilní polynom $f(x) \in K[x]$.

Algebraické rozšíření T bude těleso $K[x]/(f)$. Kořen polynomu $f(x)$ v $K[x]/(f)$ označíme θ. Násobící matice T vzhledem k bázi $(1, \theta, \theta^2, ...)$ bude mít pro pevně daný prvek $g = (\alpha_1, \alpha_2, ...)^T \in T$ tvar $M = ((g), (g\theta), (g\theta^2), ...)$.

V případě Baby Rijndael víme, že $K = B, T = A, f(x) = x^4 + x + 1$ a $g_i(x) = g_{i,0} + g_{i,1}x + g_{i,2}x^2 + g_{i,3}x^3$, kde $g_{i,j} \in GF(2), i \in \{0, 1\}, j \in 0, 1, 2, 3$ a $g_i = g_{i,0} + 2g_{i,1} + 2^2 g_{i,2} + 2^3 g_{i,3}$. Z tvaru matice M víme, že některý sloupec s submatic g_i je tvořen přímo jednotlivými bity hledaného koeficientu g_i a že $g_{i,3}$ se nachází v prvním řádku submatic g_i[6]. Nevíme ovšem, *který* sloupec je ten hledaný, protože nevíme, jakou bázi autor šifry zvolil[7]. Stačí ale zkoušet jednotlivé polynomy reprezentované sloupečky M postupně násobit θ, θ^2, θ^3 (vše modulo $f(x)$) a ověřovat, že se nalezené mezivýsledky v testované submatici nacházejí. Tímto postupem jsem zjistil, že submatice g_0 a g_1 jsou skutečně spočítané pro bázi $(1, \theta, \theta^2, \theta^3)$ a hledané koeficienty jsou tedy v posledním sloupečku; $g_0 = \texttt{5}, g_1 = \texttt{D}$. Zadaná matice t tedy popisuje operaci násobení zvoleného polynomu $h(x)$ polynomem $g(x) = \texttt{D}x + \texttt{5}$ mod $(x^4 + x + 1)$.

Tím jsem ukázal, že operace `MixColumns` je v Baby Rijndael i Rijndael zavedena stejně. Zbývá ukázat, že během transformace na menší těleso nebyly porušeny žádné z požadavků, které autoři Rijndaelu stanovili:[2, str. 29]

- Invertibilita: Je splněna, protože matice t je regulární a tedy má inverzi.
- Linearita v $GF(2)$, symetrie a jednoduchost popisu: Rijndael pro splnění těchto požadavků zvolil násobení polynomů modulo $x^4 + 1$ („na čtvrtou“ proto, že matice stavu šifry má čtyři řádky). Baby Rijndael používá násobení polynomů modulo $x^2 + 1$ (jeho stav má dva řádky), tedy obdobnou operaci.
- Rychlost na 8bitových procesorech: Pro Baby Rijndael není relevantní.
- Relevantní difúzní síla: Difúzní síla v šifře Rijndael je měřena pomocí charakteristiky *branch number*. Ta je definována jako $BN = min_{a \neq 0}(W(a) + W(F(a)))$, kde a je sloupec vstupního stavu, $F(a)$ odpovídající sloupec výstupního stavu a $W(x)$ je bajtová váha vektoru x^8 a $F(x)$. Pro Rijndael je $BN = 5$, protože v nejhorším možném případě, kdy na vstupu (resp. na výstupu) je jediný nenulový bajt,

[6] Vyplývá to z konstrukce vektoru a.

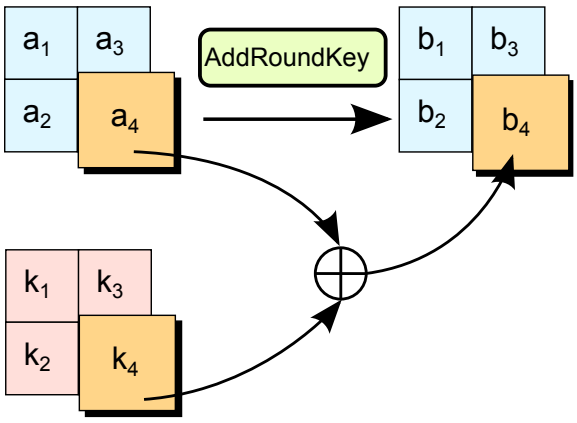
[7] Vztahy výše platí i pro libovolnou permutaci uvedené báze.

[8] Bajtová váha vektoru $x = (x_1, x_2, ..., x_n)$ je definována jako počet nenulových komponent vektoru, tzn. $W(x) = \sum_{x_i \neq 0} 1$.

jsou na výstupu (resp. na vstupu) všechny čtyři bajty nenulové. Pro Baby Rijndael jsem napsal jednoduchý prográmek `BranchNum` (k dispozici na přiloženém CD), který hodnotu BN spočítá. Výsledkem je $BN = 3$, což odpovídá tomu, že pro nejhorší možný případ, kdy na vstupu (resp. na výstupu) je jediný nenulový půlbajt, jsou na výstupu (resp. na vstupu) oba dva půlbajty nenulové. Podmínka je tedy v případě Baby Rijndael splněna.

3.5 AddRoundKey

Operace `AddRoundKey` je u Baby Rijndael přesně stejná jako u Rijndael, jen na menším stavu.



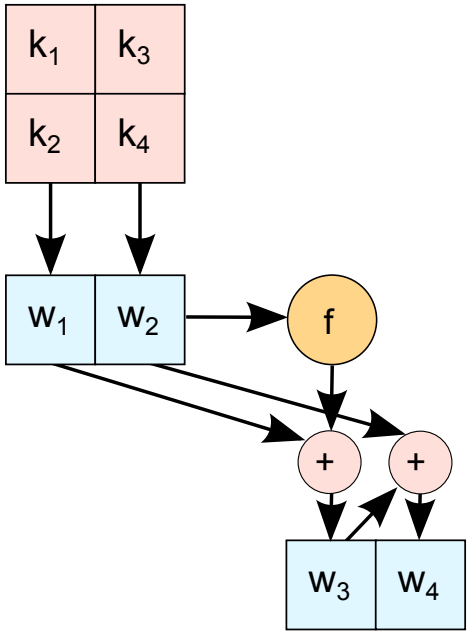
Obrázek 3.5: Operace AddRoundKey v šifře Baby Rijndael

3.6 Expanze klíče

Také expanze klíče probíhá u Baby Rijndael obdobně jako u Rijndael.

Rozdíl je v tom, že k_i jsou jen čtyřbitové hodnoty a jsou jen čtyři, uspořádáné v matici 2×2, zatímco u Rijndael jsou osmibitové a v matici 4×4. V důsledku toho jsou w_i u Baby Rijndael jen dvě a jen osmibitové, zatímco u Rijndael jsou čtyři šestnáctibitové.

Funkce f je definována v obou šifrách obdobně, s rozdílem daným odlišnou velikostí w_i: Zpracovávaná hodnota w_i (`dword` v případě Rijndael, resp. `byte` v případě Baby Rijndael) je rotována o bajt (Rijndael) resp. půlbajt (Baby Rijndael) doleva a následně je na ni uplatněna transformace



Obrázek 3.6: Key expansion šifry Baby Rijndael

`SubBytes`. K výsledku je pak přixorován výsledek funkce $Rcon(i)$, kde i je číslo zpracovávané rundy. $Rcon(i)$ je v obou případech definováno jako (x^{i-1}) redukováno polynomem $m(x)$[9].

Nakládání s vygenerovanou klíčovou posloupností je v obou šifrách zcela totožné.

3.7 Dešifrování

Dešifrování není pro zamýšlené použití Baby Rijndael podstatné, protože šifra není určena k reálnému používání ale jen k analýze. Nicméně dešifrovací proces v ní samozřejmě funguje obdobně jako u Rijndael. Pro úplnost uvádím, že násobící polynom pro inverzi `MixColumns` je $\mathtt{B}x + \mathtt{4}$.

3.8 Struktura – vyhodnocení

V předchozích kapitolách jsem ukázal, že Baby Rijndael používá v rámci možností stejné mechanismy jako Rijndael; kroky, ve kterých se obě šifry

[9]Ten je ovšem u Rijndael a Baby Rijndael odlišný a byl uveden u analýzy `SubBytes`.

liší, jsou vynuceny tím, že Baby Rijndael používá menší stav i menší klíč, ale v tom případě je pečlivě dodržována zásada, že parametry jsou voleny podle stejných kritérií jako u Rijndaelu, byť v menším tělese. Z toho lze usuzovat, že Baby Rijndael je dobrou aproximací Rijndael pro účely analýzy jeho vlastností, s možnou výjimkou útoku se zkrácenými diferenciály a čtvercového útoku.

Toto je nesmírně důležitý výsledek, protože díky němu máme model, na kterém můžeme prakticky ověřovat odolnost šifry proti různým typům útoků: zatímco Rijndael používá natolik rozsáhlé komponenty, že je výpočetně nezvladatelné některé typy útoků provést a musíme se spolehnout jen na teoretický přístup, menší rozměr Baby Rijndael umožňuje teorii podpořit praxí, např. ověřením předpokladů pomocí hrubé síly. To vedle lineární kryptoanalýzy, kterou se zabývají následující kapitoly, platí zejména pro algebraické útoky, např. [10] nebo [3].

3.9 Implementace

Šifru Baby Rijndael jsem implementoval v prostředí Delphi (`http://www.embarcadero.com/products/delphi`), konkrétně ve verzi 5, lze ale použít i libovolnou novější verzi (testováno do Delphi XE2 včetně) a nebo nekomerční FreePascal (`http://www.freepascal.org`). Kompletní zdrojové kódy i přeložený program jsou k dispozici na přiloženém CD.

Klíčovou částí implementace je třída `TBabyRijndael` ze souboru `uBabyRijndael.pas`. Ta nabízí následující veřejné metody:

- `constructor Create` - Vytvoří instanci `TBabyRijndael`.
- `destructor Destroy` - Zruší instanci `TBabyRijndael`.
- `property NumberOfRounds: integer` - Definuje počet rund, které se budou používat pro šifrování a dešifrování. Standardně je nastavena na 4, lze ale použít i jiný počet rund, například pro analýzu toho, jak se zeslabí šifra, pokud použijeme 3 nebo dokonce jen 2 rundy, nebo naopak, jestli zjištěné principy budou fungovat i pro případný větší počet rund.
- `function Encrypt(const Block, Key: TKey): Word;` - Zašifruje otevřený text `Block` klíčem `Key` a vrátí šifrový text.
- `function Decrypt(const Block, Key: TKey): Word;` - Dešifruje šifrový text `Block` klíčem `Key` a vrátí otevřený text.

V chráněné (`protected`) části třídy se nachází implementace jednotlivých dílčích operací, tedy `SubBytes`, `MixColumns`, `ShiftRows` včetně jejich inverzí, vygenerování sekvence rundovních klíčů v rámci key expansion a několik pomocných metod. Implementace je orientována na shodu s definicí šifry spíš než na výkon, pouze `SubBytes` je implementováno substituční tabulkou a ne hledáním inverze v $GF(2^8)$ a následným maticovým násobením.

Způsob používání `TBabyRijndael` demonstruji v programu `BabyRijndael.dpr`, který se skládá ze tří částí:

- Napřed zkouším vzorové šifrování uvedené v dokumentaci šifry [1, str. 4]. Ověřuji, jestli pro zadaný klíč a otevřený text dostanu určený šifrový text.
- Poté ověřuji dešifrování tím, že tento získaný šifrový text dešifruji stejným klíčem a ověřuji, že získám původní otevřený text.
- Nakonec ve dvou vnořených smyčkách ověřuji, že když každý otevřený text zašifruji každým klíčem a následně dešifruji stejným klíčem, dostanu opět původní otevřený text.

Kapitola 4

Lineární kryptoanalýza

Lineární kryptoanalýza je jednou ze dvou základních kryptoanalytických metod[10]. Poprvé ji publikoval Mitsuru Matsui v roce 1993 jako teoretický koncept útoku na DES, o rok později pak v [9] ukázal praktickou implementaci.

Následující popis je založen na [4] a [8], který z [4] vychází. Protože popis principů lineární kryptoanalýzy není cílem této práce, omezil jsem se jen na stručné shrnutí principů a postupů; detailnější informace naleznete v uvedených materiálech.

4.1 Základní princip

Šifra je standardně popisována jako posloupnost transformací, např. `SubBytes`, `ShiftRows`, `MixColumns` a `AddRoundKey` u Rijndaelu. Tyto transformace mají typicky lineární složku (`ShiftRows`, `MixColumns`), nelineární složku (`SubBytes`) a aplikaci klíče (`AddRoundKey`). Lineární kryptoanalýza, která patří mezi tzv. *plaintext útoky*[11], se snaží vyjádřit určitou část analyzované šifry v podobě lineární funkce, která dává do souvislosti bity otevřeného textu a bity některého vnitřního stavu šifry. Dokonale to udělat, to vzhledem k nelineární složce není možné; pokud je však nelineární složka navržena špatně, může být možné ji s velkou pravděpodobností lineární funkcí aproximovat. Techniky lineární kryptoanalýzy nám pomáhají najít takové aproximace, které budou mít co nejvyšší pravděpodobnost, a následně je využít k nalezení části šifrovacího klíče.

[10] Tou druhou je diferenciální kryptoanalýza, lineární kryptoanalýze velmi blízká

[11] Útok se znalostí otevřeného textu, který si ale útočník nemůže zvolit

Klíčový termín pro lineární kryptoanalýzu je tzv. *bias*, odchylka pravděpodobnosti. Řekněme, že máme aktivní prvek (S), do kterého vstupuje jeden bit (X) a vystupuje také jeden bit ($Y = S(X)$). Pokud by tento prvek byl zcela náhodný, bude

$$Pr(Y = X) = Pr(Y = 0) = Pr(Y = 1) = \frac{1}{2} \tag{4.1}$$

nezávisle na hodnotě X. Reálné šifry ovšem zcela náhodné nebývají, vztah 4.1 nabývá podoby

$$Pr(Y = X) = \frac{1}{2} + \epsilon, \tag{4.2}$$

$$Pr(Y = X \oplus 1) = 1 - Pr(Y = X) = \frac{1}{2} - \epsilon, \tag{4.3}$$

kde $\epsilon \in \langle -1/2; 1/2 \rangle$ je bias prvku S. Lineární kryptoanalýza využívá toho, že čím je bias vzdálenější od nuly, tím spíše lze nelineární prvek S nahradit lineárním výrazem $Y = X$ (pro kladné ϵ) resp. $Y = X \oplus 1$ (pro záporné ϵ).

Uvažujme jednoduchou šifru se strukturou podle obrázku 4.1. Tato šifra, tzv. SPN (Substitution-Permutation Network, Substituční a permutační síť), pracuje se 16bitovým blokem a 16bitovým klíčem a se skládá z několika rund, v nichž každá provádí operaci přičtení klíče, aplikaci S-boxu a permutaci bitů. Přičtení klíče a permutace bitů jsou lineární operace, S-box je nelineární.

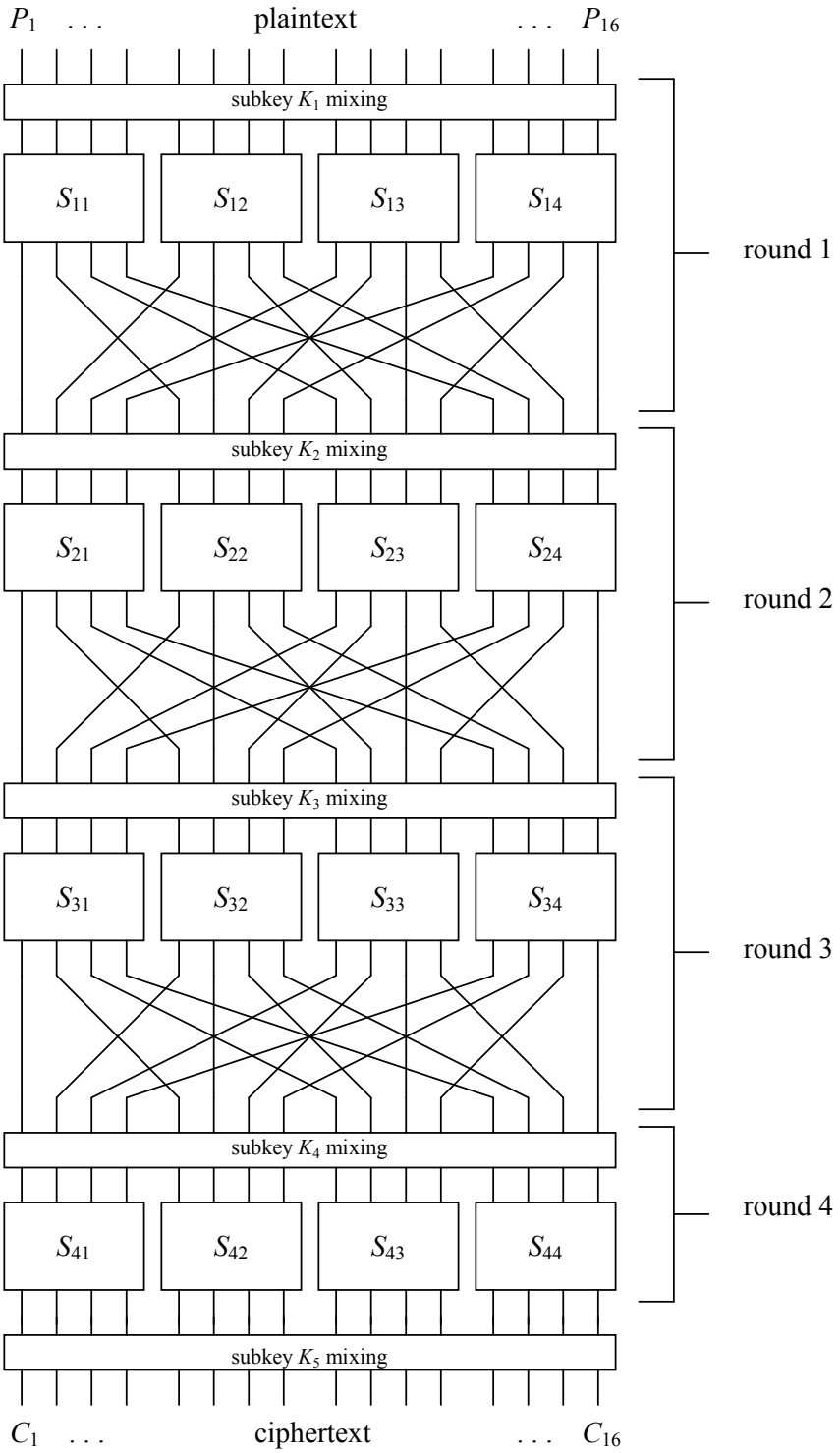
Lineární kryptoanalýza šifry pracuje v několika krocích:

- Analýza S-boxů s cílem najít v nich slabiny, které je následně umožní aproximovat lineární funkcí.
- Sestavení lineární aproximace pro část šifry začínající otevřeným textem a končící před poslední sadou S-boxů tak, aby v poslední sadě S-boxů byla aktivní pokud možno polovina[12] S-boxů a celkový bias celé aproximace byl v absolutní hodnotě co největší.
- Zjištění posledního rundovního klíče.

4.2 Analýza S-boxu

S-box typicky funguje jako překladová tabulka: Na jedné straně do něj vstupuje n bitů, na druhé straně m bitů vystupuje. Označme vstup jako $X = (X_1, X_2, ..., X_n)$ a výstup jako $Y = (Y_1, Y_2, ..., Y_m)$.

[12]Může jich být i méně. Polovina S-boxů nejvíce usnadní hledání zbytku klíče hrubou silou, méně S-boxů je výhodných v případě, kdy i další části klíče hledáme pomocí lineární kryptoanalýzy.



Obrázek 4.1: Substituční a permutační síť pro lineární kryptoanalýzu. Převzato z [4, str. 4]

Uvažujme nyní rovnici např.

$$X_2 \oplus X_3 = Y_1 \oplus Y_3 \oplus Y_4. \tag{4.4}$$

Ke všem možným hodnotám X, kterých je 2^n, dosazením do S-boxu určíme odpovídající Y, a dosadíme do rovnice výše. Sledujeme přitom, v kolika případech bude rovnice splněna; tento počet označíme k. Pokud by S-box byl vůči rovnici 4.4 zcela nelineární, měla by rovnice být splněna v právě 2^{n-1} případech. Tomu by odpovídal bias nula. Ve skutečnosti pravděpodobně k bude mít odlišnou hodnotu a skutečný bias tedy bude $\epsilon = \frac{k-2^{n-1}}{2^n} = \frac{k}{2^n} - 1/2$; protože 2^n je pro danou šifru konstantní, často se pro zvýšení přehlednosti bias uvádí jen ve tvaru $k' = k - 2^{n-1}$ s tím, že dělení 2^n se provádí implicitně.

Toto můžeme opakovat pro všechny možné lineární rovnice. Těch je 2^{m+n}. Vznikne tzv. lineární aproximační tabulka podobná jako na obrázku 4.2. Tabulka má n řádků pro každou kombinaci aktivních bitů X a m sloupců pro každou kombinaci aktivních bitů Y, prvky jsou pak biasy k (ve zkráceném podání, tzn. bez dělení 2^n). Aktivní bity jsou zakódovány jako binární číslo $N = X_1 \cdot 2^{n-1} + X_2 \cdot 2^{n-2} + ... + X_{n-1} \cdot 2 + X_n = (X_1, X_2, ..., X_n)_2$.[13] Hodnotu +4 v řádku `B` a sloupci hexx6 tedy intepretujeme tak, že „rovnice $X_1 \oplus X_3 \oplus X_4 = Y_2 \oplus Y_3$ je splněna s biasem $\frac{4}{16} = \frac{1}{4}$, tedy pravděpodobností $\frac{4+8}{16} = \frac{3}{4}$“: $\texttt{B} = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2 + 1 = (1011)_2$, aktivní bity odpovídají pozicím, které mají koeficient 1, tedy aktivní je X_1, X_3 a X_4. Obdobně je určena i pravá strana rovnice.

4.3 Sestavení lineární aproximace

V okamžiku, kdy máme k dispozici lineární aproximace S-boxů, můžeme přistoupit k sestavení lineární aproximace šifry. Označme v obrázku4.1 vstupy do S-boxů symbolem $U_{r,i}$ a výstupy z S-boxů $V_{r,i}$, kde r je číslo rundy a $i \in 1..16$ označuje bity ve stejném pořadí jako bity otevřeného textu (tzn. druhý S-box v rundě pracuje s $U_{r,5}$ až $U_{r,8}$ a $V_{r,5}$ až $V_{r,8}$). Pak platí:

- $U_{1,i} = P_i \oplus K_{1,i}$
- $U_{r,i} = V_{r-1,j} \oplus K_{r,i}$, kde $r > 1$ a vztah mezi i a j je dán použitou permutací v šifře (v případě SPN tedy $i = 4j + 12 - 15\lfloor \frac{j+3}{4} \rfloor$).
- Vztah mezi $U_{r,*}$ a $V_{r,*}$ je dán zvolenou aproximací příslušného S-boxu, jak byla určena v minulé kapitole. Na rozdíl od předchozích vztahů

[13] Pozor! Toto vyjádření se týká pouze toho, které bity vstupu resp. výstupu jsou aktivní. Neříká nic o tom, jakou hodnotu tyto bity mají.

		Output Sum															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Input Sum	0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	0	–2	–2	0	0	–2	+6	+2	+2	0	0	+2	+2	0	0
	2	0	0	–2	–2	0	0	–2	–2	0	0	+2	+2	0	0	–6	+2
	3	0	0	0	0	0	0	0	0	+2	–6	–2	–2	+2	+2	–2	–2
	4	0	+2	0	–2	–2	–4	–2	0	0	–2	0	+2	+2	–4	+2	0
	5	0	–2	–2	0	–2	0	+4	+2	–2	0	–4	+2	0	–2	–2	0
	6	0	+2	–2	+4	+2	0	0	+2	0	–2	+2	+4	–2	0	0	–2
	7	0	–2	0	+2	+2	–4	+2	0	–2	0	+2	0	+4	+2	0	+2
	8	0	0	0	0	0	0	0	0	–2	+2	+2	–2	+2	–2	–2	–6
	9	0	0	–2	–2	0	0	–2	–2	–4	0	–2	+2	0	+4	+2	–2
	A	0	+4	–2	+2	–4	0	+2	–2	+2	+2	0	0	+2	+2	0	0
	B	0	+4	0	–4	+4	0	+4	0	0	0	0	0	0	0	0	0
	C	0	–2	+4	–2	–2	0	+2	0	+2	0	+2	+4	0	+2	0	–2
	D	0	+2	+2	0	–2	+4	0	+2	–4	–2	+2	0	+2	0	0	+2
	E	0	+2	+2	0	–2	–4	0	+2	–2	0	0	–2	–4	+2	–2	0
	F	0	–2	–4	–2	–2	0	+2	0	0	–2	+4	–2	–2	0	+2	0

Obrázek 4.2: Lineární aproximační tabulka pro demonstrační šifru. Převzato z [4, str. 11]

je ovšem tento vztah jen pravděpodobnostní, protože typicky aproximace S-boxu platí jen v určitém procentu případů — kdyby platily vždy, signalizovalo by to katastrofální chybu v S-boxu a naprosté prolomení šifry.

S využitím těchto vztahů potřebujeme vytvořit *vhodnou* aproximaci od začátku šifry až před poslední sadu S-boxů. Která aproximace je vhodná? Taková, která má maximální *celkový* bias (v absolutní hodnotě). Důležitý je také požadavek, aby v poslední rundě měla polovinu nebo méně aktivních S-boxů. Tzv. *piling-up lemma* [9] nám dává návod, jak tento celkový bias spočítat:

Pro n nezávislých náhodných binárních[14] proměnných $X_1, X_2, ..., X_n$ platí:

$$Pr(X_1 \oplus X_2 \oplus ... \oplus X_n) = \frac{1}{2} + 2^{n-1} \prod_{i=1}^{n} \epsilon_i, \tag{4.5}$$

[14] Tzn. $X_i \in \{0, 1\} \forall i$.

a také

$$\epsilon_{1,2,\ldots,n} = 2^{n-1} \prod_{i=1}^{n} \epsilon_i, \tag{4.6}$$

kde ϵ_i je bias proměnné X_i a $\epsilon_{1,2,\ldots,n}$ je celkový bias výrazu $X_1 \oplus X_2 \oplus \ldots \oplus X_n$.[4, str. 8]

Nechť uvažovaná celková lineární aproximace A prochází n různými S-boxy $B_i, i = 1..n$, kde A_i označuje lineární aproximaci S-boxu B_i. Označme ϵ_i bias A_i. Náhodná binární proměnná E_i bude nabývat hodnoty 1 v případě, že aproximace A_i je splněna, a hodnoty 0 v případě opačném. Pak A_i jsou binární náhodné proměnné. Piling-up lemma ovšem navíc vyžaduje, aby tyto binární proměnné byly *nezávislé*. To nedokážeme při lineární kryptoanalýze zaručit a naopak se ukazuje, že jak Rijndael konkrétně (viz [6]), tak i řada dalších symetrických šifer (viz [11]) závislosti mezi A_i reálně vykazuje. Přesto lze piling-up lemma použít, je však nutné ho chápat pouze jako *odhad biasu*, ne jako *přesný výpočet*.

Je zřejmé, že vhodná aproximace bude taková, která použije co nejmenší počet S-boxů s co nejvyššími biasy dílčích aproximací těchto S-boxů. Zbývá už ji jen nalézt. Nepodařilo se mi nalézt žádný přesný algoritmus, který by tento problém v uspokojivém čase řešil, poměrně úspěšné však jsou heuristické metody, ve kterých si v analýze S-boxů vytipujeme ty dílčí aproximace, které mají co nejvyšší bias a současně co nejmenší počet aktivních bitů na výstupu, a z těchto se pokusíme sestavit celkovou aproximaci. Například [4] uvádí jako vhodnou aproximaci pro šifru z obrázku 4.1 aproximaci na obrázku 4.3 s celkovým biasem $-\frac{1}{32}$.

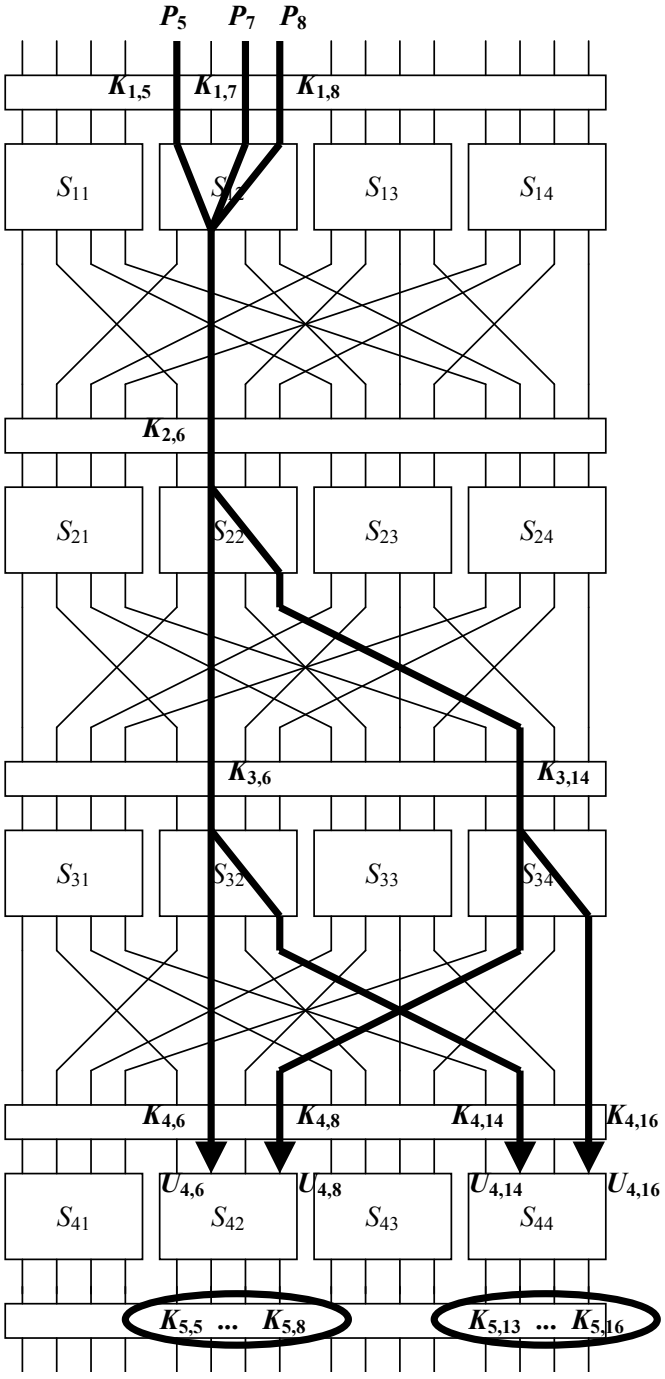
Jak zformulovat tuto lineární aproximaci symbolicky?

Vidíme, že aktivní bity otevřeného textu jsou P_5, P_7 a P_8, a podle vztahů výše víme, že:

$$U_5 = P_5 \oplus K_{1,5} \tag{4.7}$$
$$U_7 = P_7 \oplus K_{1,7} \tag{4.8}$$
$$U_8 = P_8 \oplus K_{1,8}. \tag{4.9}$$

Pro S-box $S_{1,2}$ byla zvolena aproximace $Y_2 = X_1 \oplus X_3 \oplus X_4$ s biasem $\frac{1}{4}$ a tedy:

$$\begin{aligned} V_{1,6} &= U_{1,5} \oplus U_{1,7} \oplus U_{1,8} \\ &= P_5 \oplus K_{1,5} \oplus P_7 \oplus K_{1,7} \oplus P_8 \oplus K_{1,8} \\ &= P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_{1,8}. \end{aligned} \tag{4.10}$$



Obrázek 4.3: Lineární aproximace demonstrační šifry. Převzato z [4, str. 13]

Ve druhé rundě:

$$U_{2,6} = V_{1,6} \oplus K_{2,6}. \tag{4.11}$$

V S-boxu $S_{2,2}$ byla použita aproximace $Y_2 \oplus Y_4 = X_2$ s biasem $-\frac{1}{4}$ a tedy:

$$\begin{aligned} V_{2,6} \oplus V_{2,8} &= U_{2,6} \\ &= P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6}. \end{aligned} \tag{4.12}$$

Ve třetí rundě:

$$U_{3,6} = V_{2,6} \oplus K_{3,6} \tag{4.13}$$

$$U_{3,14} = V_{2,8} \oplus K_{3,14}. \tag{4.14}$$

Aktivní jsou dva S-boxy, $S_{3,2}$ a $S_{3,4}$, které oba používají stejnou aproximaci $Y_2 \oplus Y_4 = X_2$ s biasem $-\frac{1}{4}$, tedy:

$$\begin{aligned} V_{3,6} \oplus V_{3,8} \oplus V_{3,14} \oplus V_{3,16} \\ = U_{3,6} \oplus U_{3,14} \\ = V_{2,6} \oplus K_{3,6} \oplus V_{2,8} \oplus K_{3,14} \\ = P_5 \oplus P_7 \oplus P_8 \oplus K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14}. \end{aligned} \tag{4.15}$$

Konečně před poslední sadou S-boxů ve čtvrté rundě máme:

$$U_{4,6} = V_{3,6} \oplus K_{4,6} \tag{4.16}$$

$$U_{4,8} = V_{3,8} \oplus K_{4,8} \tag{4.17}$$

$$U_{4,14} = V_{3,14} \oplus K_{4,14} \tag{4.18}$$

$$U_{4,16} = V_{3,16} \oplus K_{4,16}. \tag{4.19}$$

Pak:

$$\begin{aligned} U_{4,6} \oplus U_{4,8} \oplus U_{4,14} \oplus U_{4,16} \\ = V_{3,6} \oplus V_{3,8} \oplus V_{3,14} \oplus V_{3,16} \oplus K_{4,6} \oplus K_{4,8} \oplus K_{4,14} \oplus K_{4,16} \\ = P_5 \oplus P_7 \oplus P_8 \oplus \sum K, \end{aligned} \tag{4.20}$$

kde:

$$\sum K \ = K_{1,5} \oplus K_{1,7} \oplus K_{1,8} \oplus K_{2,6} \oplus K_{3,6} \oplus K_{3,14} \oplus K_{4,6} \oplus K_{4,8} \oplus K_{4,14} \oplus K_{4,16}. \tag{4.21}$$

Toto je hledaná aproximace šifry až před poslední S-box. Celkový bias nám určí piling-up lemma: Aproximace prošla celkem čtyřmi S-boxy s dílčími biasy $\frac{1}{4}$, $-\frac{1}{4}$, $-\frac{1}{4}$ a $-\frac{1}{4}$, a celkový bias tedy je:

$$\begin{aligned} \epsilon_{1,2,3,4} &= 2^{4-1} \cdot \frac{1}{4} \cdot (-\frac{1}{4}) \cdot (-\frac{1}{4}) \cdot (-\frac{1}{4}) \\ &= 2^3 \cdot (-1)^3 \cdot 2^{-8} \\ &= -2^{-5} \\ &= -\frac{1}{32}. \end{aligned} \tag{4.22}$$

Poznámka: Určení celkové aproximace lze zjednodušit a zkrátit, když si uvědomíme, že do ní budou zasahovat pouze bity otevřeného textu P_i, bity vstupující v poslední rundě do S-boxů $U_{r,i}$ a všechny bity klíče, které cesta šifrou protne. Navíc, jak bude ukázáno v příští kapitole, na jednotlivých bitech klíče vůbec nezáleží. Tudíž pokud máme grafické znázornění cesty, můžeme na začátku rovnou číst bity plaintextu, na konci bity vstupující do posledních S-boxů a všechny tyto údaje zxorovat navzájem a s XORem všech zúčastněných bitů klíče $\sum K$.

4.4 Zjištění posledního rundovního klíče

Nalezenou lineární aproximaci a její bias využijeme při zjišťování posledního rundovního klíče. K tomu potřebujeme dostatečně velký vzorek otevřených textů a jim odpovídajících šifrových textů zašifrovaných neznámým, ale pevným, šifrovacím klíčem. Pokud je totiž klíč pevně daný, lze snadno ukázat, že $\sum_{r,i} K_{r,i}$ je konstantní pro libovolně volená r a i: Expanze klíče typicky (např. v Rijndaelu) závisí *pouze* na klíči, velikosti bloku a na počtu rund. V lineární kryptoanalýze máme velikost bloku i počet rund pevně daný a předem známý, pevně daný je také neznámý šifrovací klíč K_0. Z toho vyplývá, že i všechny expandované klíče K_1, K_2 atd. jsou pevně dané (i když neznámé), a tudíž XOR jejich libovolných pevně zvolených bitů je také konstantní (i když neznámý).

Označme tento neznámý součet $\sum K$. Pak lineární aproximace bude mít tvar

$$P_{i_1} \oplus P_{i_2} \oplus ... \oplus P_{i_r} \oplus U_{r,j_1} \oplus U_{r,j_2} \oplus ... \oplus U_{r,j_s} \oplus \sum K = 0 \tag{4.23}$$

s biasem $\epsilon_{1,2,...,t}$ pro množinu $\{i_1, i_2, ..., i_r\}$, množinu $\{j_1, j_2, ..., j_s\}$, které jsou dané zvolenou lineární aproximací. Vztah 4.23 můžeme zapsat pravděpodobnostně jako

$$Pr(P_{i_1} \oplus P_{i_2} \oplus ... \oplus P_{i_r} \oplus U_{r,j_1} \oplus U_{r,j_2} \oplus ... \oplus U_{r,j_s} \oplus \sum K = 0) = 1/2 + \epsilon_{1,2,...,t}. \tag{4.24}$$

Podstatné je, že když jednotlivé bity expandovaného klíče jsou z množiny $\{0, 1\}$, musí i xor těchto bitů, tedy $\sum K$, být z $\{0, 1\}$. Tudíž mohou nastat jen dva případy: případ hlavní, kdy uhádneme hodnotu $\sum K$, a případ alternativní, kdy uhádneme jeho opak. Tento případ pak musí mít pravděpodobnost

$$\begin{aligned} &Pr(P_{i_1} \oplus P_{i_2} \oplus ... \oplus P_{i_r} \oplus U_{r,j_1} \oplus U_{r,j_2} \oplus ... \oplus U_{r,j_s} \oplus \sum K \ = 1) \\ = 1 - &Pr(P_{i_1} \oplus P_{i_2} \oplus ... \oplus P_{i_r} \oplus U_{r,j_1} \oplus U_{r,j_2} \oplus ... \oplus U_{r,j_s} \oplus \sum K \ = 0) \\ &= 1 - (1/2 + \epsilon_{1,2,...,t}) \\ &= 1/2 - \epsilon_{1,2,...,t}. \end{aligned} \tag{4.25}$$

Je zřejmé, že absolutní hodnota $\epsilon_{1,2,...,t}$ je v obou případech stejná. Pokud tedy budeme bias uvažovat v absolutní hodnotě, je pro určení klíče nepodstatné, jaká je skutečná hodnota $\sum K$[15] a můžeme lineární aproximaci zapisovat ve tvaru

$$\begin{aligned} &Pr(P_{i_1} \oplus P_{i_2} \oplus ... \oplus P_{i_r} \oplus U_{r,j_1} \oplus U_{r,j_2} \oplus ... \oplus U_{r,j_s} = 0) \\ &\qquad \in \{1/2 + \epsilon_{1,2,...,t}, 1/2 - \epsilon_{1,2,...,t}\}. \end{aligned} \tag{4.26}$$

Poslední rundovní klíč zjistíme následovně: Určíme množinu všech možných částí klíče, které ovlivňují aktivní S-boxy v poslední rundě. V příkladu z minulé kapitoly byly aktivní druhý a čtvrtý S-box, tzn. zkoumáme bity $K_{5,5}$ až $K_{5,8}$ a $K_{5,13}$ až $K_{5,16}$. Celkem jde o osm bitů, tedy 256 různých možností.

Pro každý z těchto zkoumaných podklíčů provedeme následující posloupnost operací:

- Nastavíme počitadlo na nulu.

[15] Ovšem musí být konstantní, jak bylo uvedeno na začátku kapitoly, což lze zaručit pouze v případě, kdy všechny vzorky otevřených textů byly zašifrovány stejným klíčem.

- Určíme kandidátní klíč K tak, že bity odpovídající aktivním S-boxům v poslední rundě jsou zkopírovány z odpovídajících bitů podklíče a ostatní bity jsou nastaveny na nějakou pevnou hodnotu, např. na nulu.

- Pak pro každou dvojici (P, C), kde P značí otevřený text a C značí odpovídající šifrovací text, kterou máme k dispozici, provedeme tuto posloupnost kroků:

  - Spočítáme $V = C \oplus K$.
  - Spočítáme $U = \text{InvSBox}(V)$.[16]
  - Příslušné bity z P a z U dosadíme do lineární aproximace a ověříme, jestli je rovnice splněna, tzn. jestli se $P_{i_1} \oplus P_{i_2} \oplus ... \oplus P_{i_r} \oplus U_{r,j_1} \oplus U_{r,j_2} \oplus ... \oplus U_{r,j_s}$ rovná nule.
  - Pokud ano, zvýšíme počítadlo o 1.

- Určíme očekávaný počet M úspěšných splnění rovnice. Pokud pracujeme s N vzorky dvojic otevřeného a šifrového textu, tak v souladu s výrazem pro pravděpodobnost splnění lineární aproximace výše dostaneme, že $M \in \{N \cdot (1/2 + \epsilon_{1,2,...,t}), N \cdot (1/2 - \epsilon_{1,2,...,t})\}$.

- Seřadíme všechny zkoumané klíče podle blízkosti jejich počtu splnění lineární aproximace k hodnotám M.

- Hodnota, která má nejbližší počet splnění lineární aproximace k M by měl být hledaný klíč.

Zbytek posledního rundovního klíče můžeme určit obdobně s využitím jiné lineární aproximace (která končí u jiných S-boxů) nebo i pomocí hrubé síly.

Hlavní šifrovací klíč pak lze buď určit iterativně (klíč v poslední rundě už známe, můžeme tedy zpětný chod od C k U prodloužit o jednu rundu a určit tak klíč v předposlední rundě, atd.), nebo ho může jít spočítat přímo: například šifra Rijndael má expanzi klíče definovanou tak, aby byla invertovatelná, tzn. aby z libovolného rundovního klíče šlo dopočítat libovolný jiný rundovní klíč, včetně klíče hlavního.

[16] Operace InvSBox značí inverzi S-boxu, tzn. pro zadaný výstup nalezne odpovídající vstup.

Kapitola 5

Lineární kryptoanalýza Baby Rijndael

Poznatky z předchozích dvou kapitol jsem uplatnil v lineární kryptoanalýze šifry Baby Rijndael. Za tímto účelem jsem vytvořil program `LinearCryptanalysis`, dostupný na přiloženém CD, který provádí jednotlivé výpočetně náročné dílčí části analýzy a poskytuje data, která dále interpretuji. Program je založený na už dříve vytvořené implementaci Baby Rijndael, je tedy také napsaný v Delphi a také používá třídu `TBabyRijndael`, konkrétně v podobě třídy `TBabyRijndaelLC`, která je potomkem `TBabyRijndael`. Na rozdíl od předchozího programu je ale z větší části řiditelný pomocí parametrů na příkazové řádce, aby v něm šlo pokud možno pohodlně provádět různé typy analýzy. Podrobnosti k ovládání a případným použitým programovým komponentám naleznete u popisu jednotlivých dílčích analýz.

5.1 Převod na tvar SPN

Postup lineární kryptoanalýzy byl demonstrován na šifře typu SPN podle obrázku 4.1. Baby Rijndael má však poněkud odlišnou strukturu, která je zachycena na obrázku 5.1. Je tedy třeba buď upravit postup lineární kryptoanalýzy, nebo převést Baby Rijndael na SPN. Zvolil jsem druhou možnost, protože mi připadala jednodušší.

Základní problém leží v operaci `MixColumns`: Zatímco `SubBytes` lze přímo mapovat na S-box, `AddRoundKey` je v obou šifrách totožný a `ShiftRows` je dokonce jednodušší varianta permutace, protože nemění pořadí bitů, `MixColumns` žádnou obdobu v SPN nemá. Operace je sice lineární, takže by

ji jistě šlo vhodnou lineární funkcí popsat; bohužel však je výpočet definován tak, že hodnota výstupu závisí na konkrétních *hodnotách* jednotlivých bitů vstupu, které lineární kryptoanalýza nedokáže určit, protože se zabývá pouze *vztahy* mezi jednotlivými bity.[17]

Nalezl jsem celkem tři možnosti konverze:

- Využít toho, že v `MixColumns` dokážeme kterýkoliv jednotlivý bit výstupu vyjádřit pomocí součtu (modulo 2) určitých předem známých bitů vstupu, a nahradit celou operaci `MixColumns` tímto součtem.
- Operaci `MixColumns` chápat jako druhý nezávislý S-box a nahrazovat ji lineárními aproximacemi s danou pravděpodobností.
- Spojit operace `SubBytes` a `MixColumns` do jednoho většího S-boxu. Výslednou operaci budu nadále označovat jako `SubBytes+MixColumns`.

5.1.1 MixColumns jako vyjádření výstupního bitu

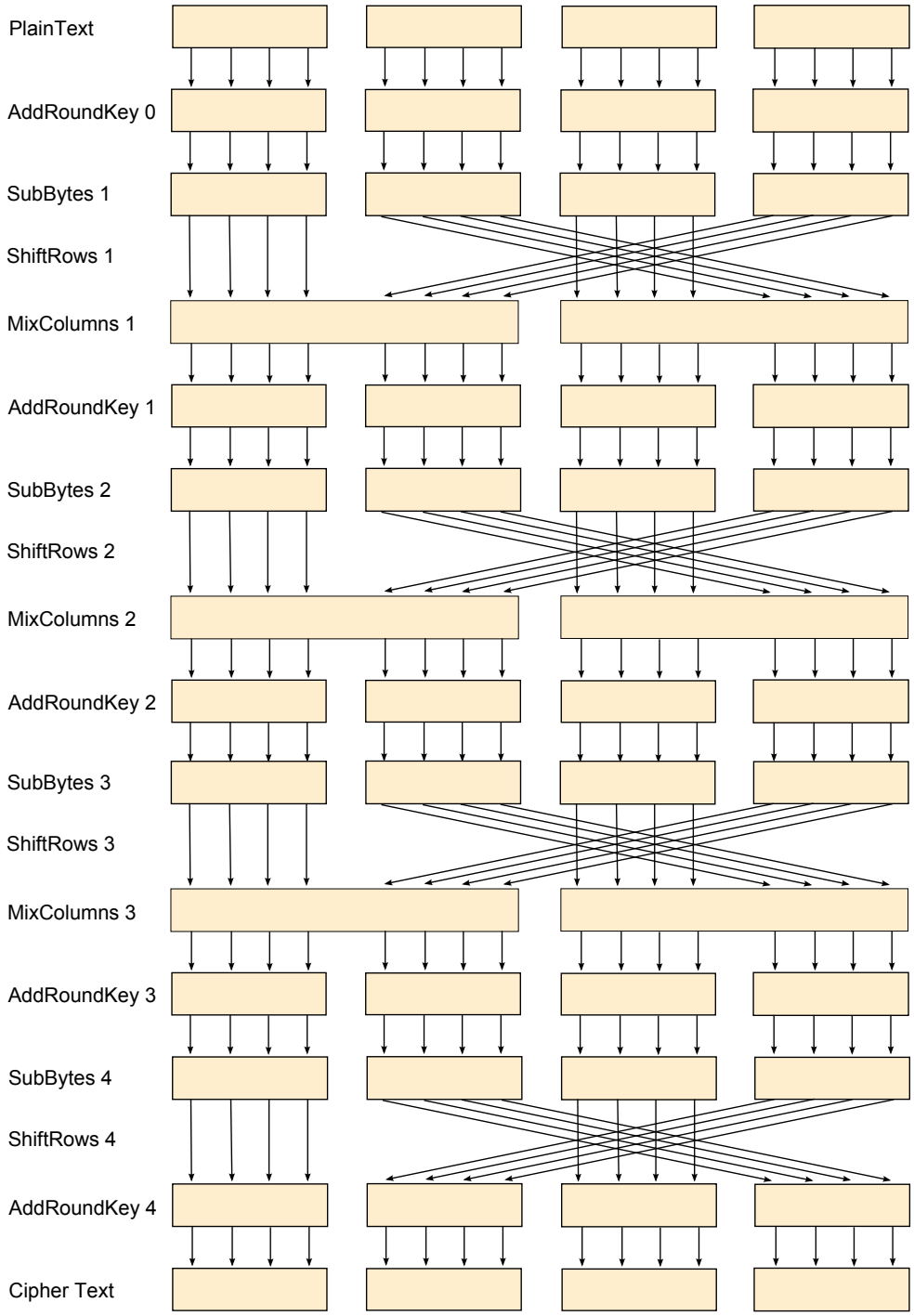
Z popisu šifry Baby Rijndael víme, že operace `MixColumns` je definována jako maticové násobení transformační matice a osmibitového vstupního vektoru a jejím výstupem je opět osmibitový vektor (viz vztah 3.6). Rozepsáním maticového násobení můžeme vyjádřit vztah mezi jedním bitem výstupu a všemi bity vstupu, např. pro bit 0:

$$b_0 = a_0 \oplus a_2 \oplus a_6 \oplus a_7 \tag{5.1}$$

Je zřejmé, že každý bit výstupu je vyjádřen jako součet (modulo 2) určitých bitů vstupu, což je také obecně tvar lineární aproximace S-boxu. Pokud bychom tedy v předcházejících S-boxech použili takové aproximace, které na výstupu mají tvar $Y_{0,1} \oplus Y_{0,3}$ (tedy maska `0xA`) a $Y_{1,3} \oplus Y_{1,4}$ (tedy maska `0x3`), můžeme přímo vyjádřit bit 0 vystupující z `MixColumns` jako $Z_1 = Y_{0,1} \oplus Y_{0,3} \oplus Y_{1,3} \oplus Y_{1,4}$ (tedy maska `0x8`).

Toto řešení má ovšem zásadní nevýhodu v tom, že *každý* bit výstupu operace `MixColumns` je definován pomocí vstupních bitů z obou čtveřic vstupních bitů a tedy každá operace `MixColumns` potřebuje dva S-boxy před sebou, ale umožní definovat jen jeden S-box za sebou. To znamená, že pokud ve čtvrté rundě potřebujeme aspoň jeden aktivní S-box, musely ve třetí rundě být aktivní S-boxy dva, ve druhé rundě čtyři a v první rundě

[17] V tomto ohledu je na tom mnohem lépe diferenciální kryptoanalýza, která pracuje s diferencemi bitů: Ty totiž operací `MixColumns` beze změny procházejí. Podrobněji viz [14].



Obrázek 5.1: Struktura Baby Rijndael pro lineární kryptoanalýzu

dokonce osm, což je víc, než vůbec má šifra k dispozici. Tento způsob konverze `MixColumns` je tedy omezen na maximálně tři rundy. Navíc vynucuje aktivitu ve velmi mnoha S-boxech a tím zhoršuje celkový bias.

5.1.2 MixColumns jako nezávislý S-box

Toto je asi nejpřirozenější řešení konverze BabyRijndael na SPN. Vychází z myšlenky, že v principu libovolnou operaci v libovolné šifře lze chápat jako S-box a následně ji nahradit lineární aproximací s danou pravděpodobností, pokud nevadí právě ta skutečnost, že půjde o pravděpodobnostní vyjádření. V případě `MixColumns` to například nevadí, protože operace `MixColumns` je lineární, tudíž její „S-box“ vykazuje extrémní linearitu a její pravděpodobnost tedy je rovna jedné.

Analýzu `MixColumns` z tohoto hlediska provádí program `LinearCryptanalysis` s parametrem `MIXCOLUMNS`. Jeho výstupem je CSV soubor, kde řádky jsou masky vstupů do „S-boxu“ `MixColumns`, sloupce jsou masky výstupů z „S-boxu“ `MixColumns` a jednotlivé položky vyjadřují bias pro příslušnou kombinaci vstupu a výstupu. Jde tedy o strukturu obdobnou obrázku 4.2, pouze s jinými rozměry — `MixColumns` v Baby Rijndael pracuje s osmibitovými hodnotami a matice má tedy 256 řádků i sloupců.

Podíváme-li se na tuto tabulku, zjistíme velmi rychle, že skutečně potvrzuje očekávanou linearitu operace `MixColumns`: V každém z 256ti řádků (resp. sloupců) jsou samé nuly, kromě jednoho sloupce (resp. řádku), který obsahuje hodnotu 128, tzn. jeho bias je $\frac{128}{256} = \frac{1}{2}$ a z toho pravděpodobnost příslušné lineární aproximace je 1. Operaci `MixColumns` tedy lze chápat jako S-box, přičemž *bias aproximace celé šifry se tím nezhorší*. Je to ovšem kompenzováno tím, že nutně musíme pracovat se značným množstvím aktivních S-boxů pro operaci `SubBytes`, protože operace `MixColumns` byla úmyslně navržena tak, aby ze čtyř čtyřbitových vstupů a výstupů byl nejvýše jeden nulový.

5.1.3 MixColumns spojený se SubBytes

Operace `ShiftRows`, která reprezentuje permutaci bitů v SPN, má oproti SPN jednu zajímavou vlastnost: permutuje pouze celé čtveřice bitů, nikdy ne jednotlivé bity. V kombinaci s tím, že v Baby Rijndael jsou všechny S-boxy totožné, z toho vyplývá, že v Baby Rijndael nezáleží na tom, v jakém pořadí proběhnou operace `ShiftRows` a `SubBytes`. Původní popis šifry jako posloupnost operací `SubBytes`, `ShiftRows`, `MixColumns`, `AddRoundKey` mů-

žeme tedy přepsat na `ShiftRows`, `SubBytes`, `MixColumns`, `AddRoundKey`, který je z hlediska vztahu mezi vstupy a výstupy ekvivalentní.

Chápeme-li navíc `MixColumns` jako samostatný S-box, jak jsem ukázal v předchozím bodu, je zřejmé, že se nyní mezi `ShiftRows` a `AddRoundKey` vyskytují dvě operace, které vždy vezmou vstupní bity a nahradí je podle určitého vztahu stejným počtem jiných bitů. Jistě tedy můžeme najít transformaci, která obě operace nahradí operací jedinou. Jediný drobný problém je v tom, že `SubBytes` pracuje se čtyřmi bity, zatímco `MixColumns` s osmi, ale jak je vidět z obrázků 5.1 a 5.2, to jednoduše znamená, že kombinujeme nikoliv jeden `SubBytes` a jeden `MixColumns`, ale dva `SubBytes` a jeden `MixColumns`.

Výsledná operace, kterou jsem nazval `SubBytes+MixColumns`, má opět charakter S-boxu[18] a převádí Baby Rijndael přesně na tvar, který má SPN. Navíc se ukazuje, že tento S-box má výhodné parametry pro lineární kryptoanalýzu (bude dokázáno dále v kapitole) a jeho větší velikost umožňuje zmenšit počet aktivních S-boxů v celé šifře při zachování počtu aktivních bitů v rundě. Proto jsem mu dal přednost před řešením s `MixColumns` jako samostatným S-boxem.

5.2 Analýza SubBytes

První krok v lineární kryptoanalýze spočívá v analýze S-boxů. Baby Rijndael má pouze jeden S-box, který je implementován v operaci `SubBytes` a který se shodně používá všude tam, kde je potřeba provést nějakou nelineární operaci.

Analýzu `SubBytes` provedeme voláním `LinearCryptanalysis` s parametrem `SUBBYTES`. Program postupně pro všechny možné lineární aproximace `SubBytes` vyzkouší všechny možné vstupy a spočítá bias jednotlivých operací. Tento bias následně vypíše v tabulce, jejíž řádky definují zúčastněné bity na vstupu, sloupce definují zúčastněné bity na výstupu a uvnitř tabulky je bias v zjednodušeném tvaru (bez dělení 2^4)[19].

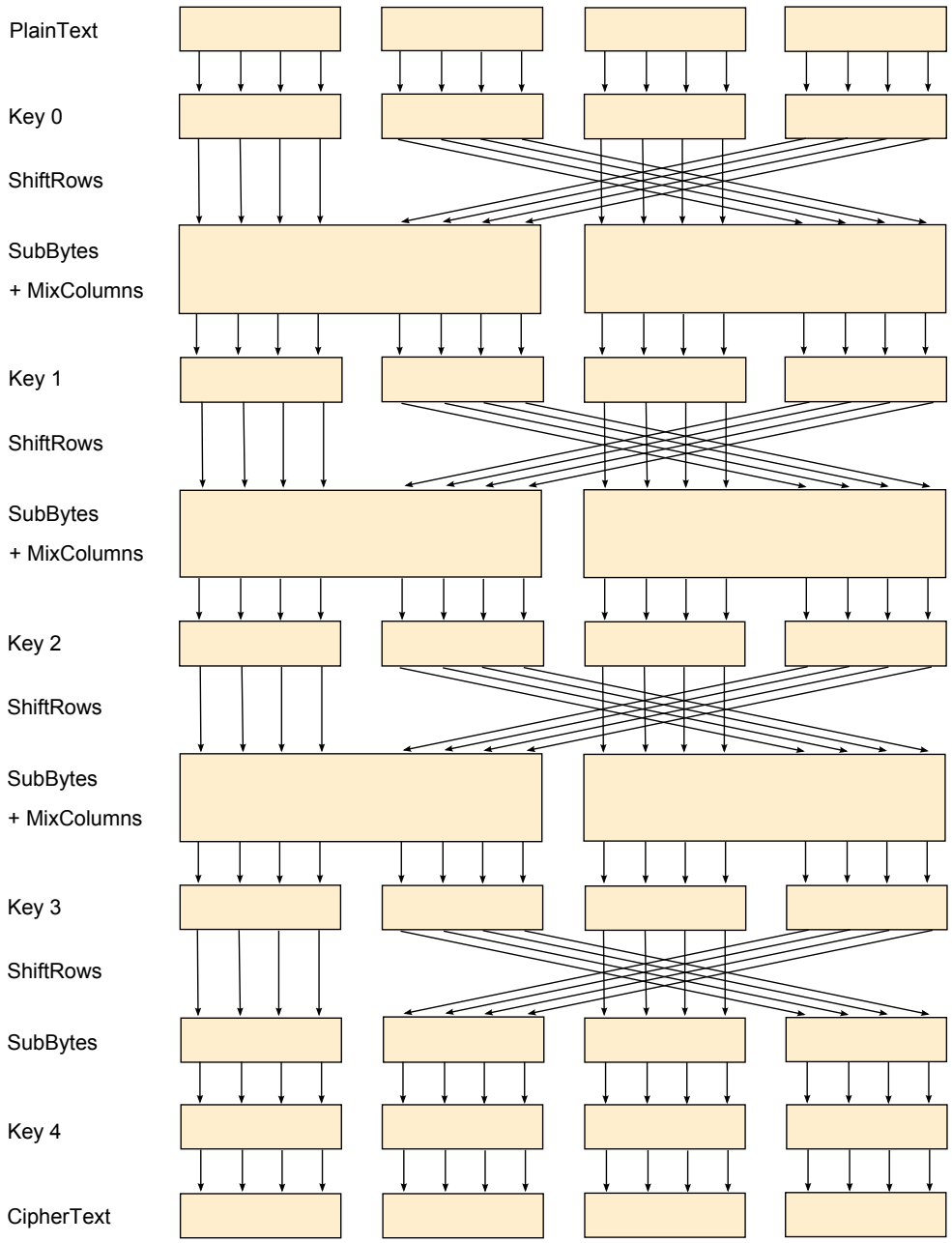
Elektronický tvar naleznete na přiloženém CD v souboru `Vysledky/S-box/SubBytes.xlsx`.

V tabulce lineárních závislostí je vidět několik podstatných skutečností:

- `SubBytes` v Baby Rijndael vykazuje jistou míru lineárních závislostí. To může být překvapující, protože `SubBytes` je definována primárně

[18] Většího, než byl ten původní — `SubBytes+MixColumns` pracuje s osmibitovými hodnotami.

[19] Jde tedy o stejný tvar, jaký je uveden v [4, str. 11].



Obrázek 5.2: Sloučený SubBytes a MixColumns pro lineární kryptoanalýzu Baby Rijndael

	Aktivní výstupy															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	+8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	+2	0	−2	+4	+2	0	+2	+2	0	−2	0	−2	0	−2	+4
2	0	+2	+2	0	0	+2	+2	0	+2	−4	+4	+2	−2	0	0	−2
3	0	0	+2	+2	0	+4	−2	+2	−4	0	+2	−2	0	0	+2	+2
4	0	−2	−4	+2	−2	0	−2	0	+2	0	+2	0	−4	−2	0	+2
5	0	0	0	−4	−2	−2	−2	+2	0	−4	0	0	+2	−2	+2	+2
6	0	0	+2	−2	−2	+2	−4	−4	0	0	−2	+2	−2	+2	0	0
7	0	−2	−2	−4	+2	0	0	−2	−2	0	+4	−2	0	+2	−2	0
8	0	+4	−2	+2	+2	−2	−4	0	−2	−2	0	0	0	0	−2	−2
9	0	−2	−2	0	+2	+4	0	−2	0	−2	−2	0	+2	−4	0	−2
A	0	+2	0	−2	−2	0	+2	0	−4	+2	0	+2	−2	−4	−2	0
B	0	0	0	0	+2	−2	+2	−2	−2	−2	−2	−2	−4	0	+4	0
C	0	−2	+2	0	+4	−2	−2	0	0	+2	+2	+4	0	−2	+2	0
D	0	0	−2	+2	0	0	+2	−2	−2	−2	0	+4	+2	+2	0	+4
E	0	+4	0	0	0	0	0	−4	+2	+2	+2	−2	+2	−2	+2	+2
F	0	+2	−4	−2	0	+2	0	+2	0	+2	0	+2	0	+2	+4	−2

Tabulka 5.1: Lineární závislosti SubBytes v Baby Rijndael

operací inverze, která jistě není lineární. V malém tělese, jako je $GF(2^4)$, se však může linearita projevit prostě proto, že pracujeme s příliš málo prvky. Jde tedy o linearitu umělou, která nám ovšem pro lineární kryptoanalýzu stačí.

- Maximální hodnota biasu je $\pm\frac{4}{16}$.

  Pozn.: Hodnotu 8 v nultém řádku a nultém sloupci neuvažujeme, protože vyjadřuje aproximaci, které se neúčastní žádné bity vstupu a žádné bity výstupu, tedy aproximaci vyjádřenou vztahem $\sum K = 0$. Ta je ovšem pro lineární kryptoanalýzu zcela neužitečná.

- Každý řádek i každý sloupec má dvě aproximace s biasem $\pm\frac{4}{16}$, osm aproximací s biasem $\pm\frac{2}{16}$ a šest aproximací s biasem $\frac{0}{16}$. Tyto aproximace jsou rovnoměrně rozprostřeny po celé tabulce, nedochází k žádným vysokobiasovým nebo naopak nízkobiasovým shlukům. To signalizuje, že `SubBytes` neobsahuje žádné zvlášť citlivé komponenty a z hlediska lineární kryptoanalýzy je dobře navržen.

5.3 Analýza SubBytes+MixColumns

Protože při konverzi vyvstala jako jedna z možností, spojit `SubBytes` a `MixColumns` v jeden velký S-box, musel jsem provést i jeho analýzu. Vlastní

výpočet provádí opět program `LinearCryptanalysis`, tentokrát s parametrem `SUBBYTESMIXCOLUMNS`. Výstupem je tabulka 65536 biasů pro všechny variace 256 vstupů a 256 výstupů, přičemž vstupem a výstupem opět rozumíme zakódované zúčastněné bity (tzn. hodnota 64 v řádku `0x03` a sloupci `0x1D` značí, že lineární aproximace $X_7 \oplus X_8 = Y_4 \oplus Y_5 \oplus Y_6 \oplus Y_8$ nastává s pravděpodobností $\frac{64}{256} = \frac{1}{4}$).

Takto velká tabulka je samozřejmě značně nepraktická pro zkoumání člověkem, proto jsem do `LinearCryptanalysis` zavedl další parametr `SUBBYTESMIXCOLUMNSHIGH`, který vypíše pouze položky s nejvyšším absolutním biasem — v tomto případě tedy položky s biasem $\frac{1}{4}$ a $\frac{1}{8}$. Ukazuje se, že tyto položky naprosto stačí pro kompletní lineární analýzu, protože mají následující vlastnosti:

1. Celkem 60 položek má bias $\frac{1}{4}$. Jejich struktura je vzájemně velmi podobná, ve všech případech platí, že:
   a) v zakódovaných vstupních bitech je jedna čtveřice bitů nulová a druhá čtveřice bitů nenulová,
   b) zastoupeno je všech 30 vstupů, které vyhovují tomuto pravidlu (`0xi0` a `0x0j` pro $i, j \in \{\mathtt{1}, \mathtt{2}, ..., \mathtt{E}, \mathtt{F}\}$), a to každý s právě dvěma různými výstupními hodnotami,
   c) obdobně je zastoupeno 30 různých výstupů, z toho každý právě dvakrát (pro dva různé vstupy),
   d) v zakódovaných výstupních bitech nikdy není nulová čtveřice, a
   e) pokud jsou čtveřice (i, j) transformovány na čtveřice (m, n), pak také čtveřice (j, i) budou transformovány na (n, m).
2. Celkem 1140 položek má bias $\frac{1}{8}$. Opět vykazují jistou míru pravidelnosti:
   a) Každý z 255 vstupů je v tabulce zastoupen.
   b) Četnost zastoupení vstupů závisí na tom, jestli vstup obsahuje nebo neobsahuje v první nebo druhé čtveřici bitů nulu: Pokud obsahuje, existuje k tomuto vstupu osm výstupů s daným biasem, v opačném případě k němu existují čtyři výstupy.
   c) Každý z 255 výstupů je v tabulce zastoupen.
   d) Četnost zastoupení výstupů záleží na tom, jestli existuje nějaký vstup s nulovou čtveřicí bitů, který se mapuje na daný výstup. Pokud existuje, bude se daný výstup vyskytovat osmkrát, v opačném případě čtyřikrát.

e) Nikdy nenastává případ, kdy by jak vstup tak výstup měly současně některou čtveřici bitů nulovou.

Zejména druhá skupina vlastností je důležitá, protože ukazuje, že jakoukoliv lineární aproximaci šifry lze zkonstruovat tak, aby jednotlivé zúčastněné aproximace S-boxů měly bias nejhůře $\frac{1}{8}$.

Elektronický tvar aproximační tabulky naleznete na přiloženém CD v souboru `Vysledky/S-box/SubBytesMixColumns.csv`. Výpis nejvyšších aproximací naleznete na CD v souborech `Vysledky/S-box-/SubBytesMixColumnsHighApproximations.txt` (nejvyšší aproximace) a `Vysledky/S-box/SubBytesMixColumnsHighApproximationsInverse.txt` (jejich inverze).

5.4 Sestavení lineární aproximace

V této části ukážu postupy pro sestavení lineárních aproximací s vysokým celkovým biasem, a to zvlášť pro dvě z variant převodu Baby Rijndael na SPN: pro variantu, ve které je MixColumns redukováno na vyjádření jednoho bitu, a pro variantu se spojeným `SubBytes` a `MixColumns`. Variantu, ve které je `MixColumns` chápáno jako samostatný S-box, neuvádím, protože je velmi podobná variantě se `SubBytes+MixColumns`.

5.4.1 Aproximace pro MixColumns jako vyjádření jednoho bitu

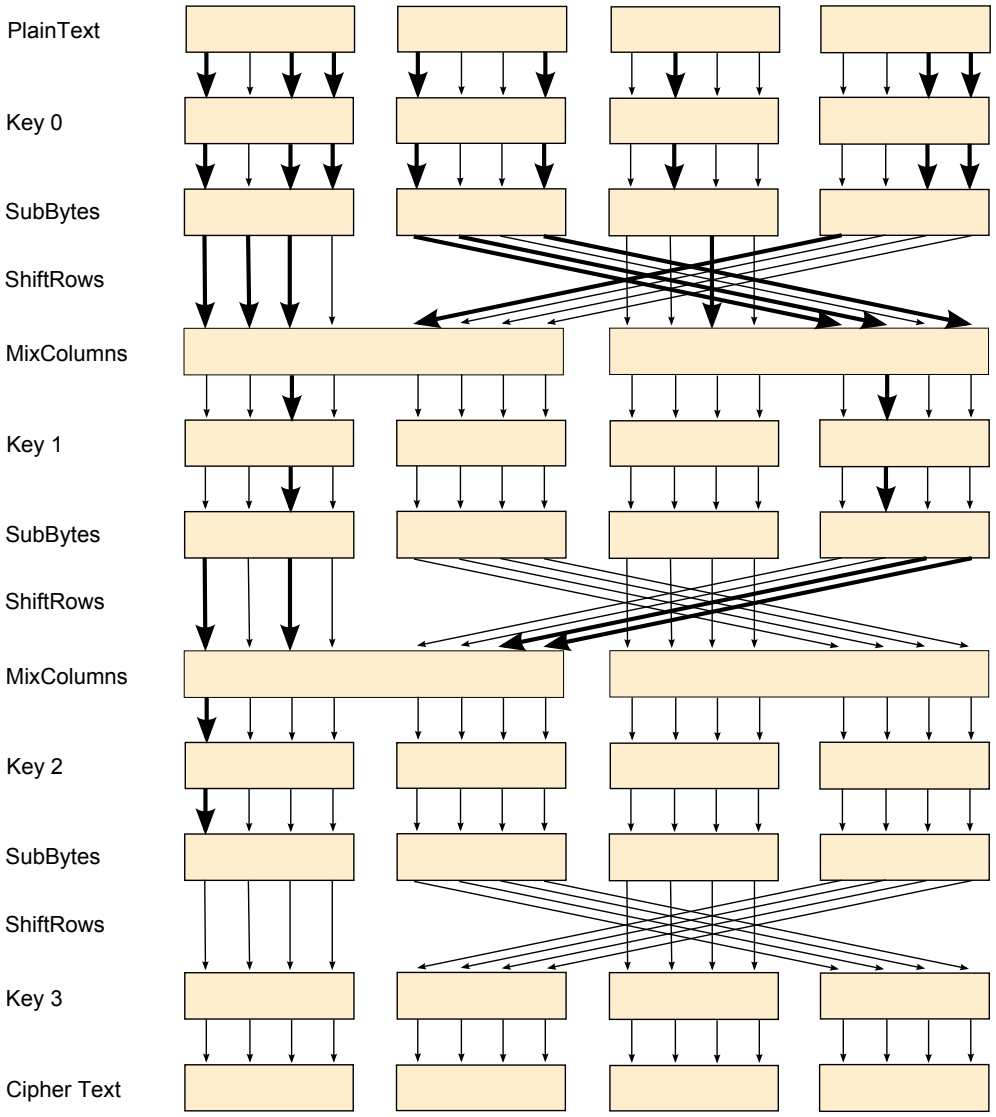
Tato aproximace je použitelná pouze pro dvě nebo tři rundy Baby Rijndael, pro větší počet rund nelze použít, protože v každé rundě šifra přichází o polovinu aktivních S-boxů. Při maximálně čtyřech aktivních S-boxech v rundě to znamená, že můžeme aproximovat maximálně tolik rund, v kolika se objeví dva výskyty `MixColumns`, což jsou tři rundy.

Postup pro sestavení aproximace s maximálním biasem je následující:

1. Zvolíme bit, který bude vystupovat z posledního `MixColumns` (na konci druhé rundy). Od tohoto výstupního bitu budeme postupovat směrem k počátku šifry.

2. Zvoleným bitem je jednoznačně určen mix bitů, které do posledního `MixColumns` vstupují.

3. Tyto vstupující bity procházejí operací `ShiftRows`, která je rozptýlí do dvou S-boxů v druhé rundě.



Obrázek 5.3: Aproximace MixColumns - 3 rundy, 1. bit

4. Pro každý z těchto S-boxů hledáme aproximaci, která
   - má požadovanou kombinaci bitů na výstupu,
   - na vstupu má aktivní právě jeden bit, a
   - při splnění předchozích dvou podmínek má nejvyšší bias.
5. Tento jeden aktivní vstupní bit `SubBytes` se stává jednoznačně definovaným výstupním bitem předchozího (v první rundě) `MixColumns`. Kroky 2 a 3 se uplatní obdobně i pro tento `MixColumns`.
6. Při aproximaci S-boxů v první rundě už se nemusíme řídit druhým požadavkem, protože už žádný další `MixColumns` procházet nebudeme a je proto lhostejné, kolik bude aktivních vstupních bitů v jednotlivých S-boxech.
7. Celý postup následně zopakujeme pro všech 7 možných dalších výstupních bitů z posledního `MixColumns`. Vzhledem k tomu, že všechny rundy šifry používají shodné základní operace `SubBytes`, `ShiftRows`, `MixColumns` a `AddRoundKey`, není třeba zkoušet bity u druhého `MixColumns` v druhé rundě, protože z nich odvozená lineární aproximace bude obdobná, pouze se prohodí pozice jednotlivých S-boxů.

Lineární aproximace pro první bit výstupu posledního `MixColumns` je uveden na obrázku 5.3. Vidíme, že aproximaci můžeme vyjádřit vztahem:

$$P_1 \oplus P_3 \oplus P_4 \oplus P_5 \oplus P_8 \oplus P_{10} \oplus P_{15} \oplus P_{16} \oplus U_{2,1} \oplus \sum K \ = 0 \qquad (5.2)$$

Použité aproximace S-boxů jsou v první rundě `0xB` na `0xE` s biasem $\frac{1}{4}$, `0x9` na `0xD` s biasem $\frac{1}{4}$, `0x4` na `0x2` s biasem $\frac{1}{4}$ a `0x3` na `0x8` s biasem $\frac{1}{4}$, ve druhé rundě pak `0x2` na `0xA` s biasem $\frac{1}{4}$ a `0x4` na `0x3` s biasem $\frac{1}{8}$.

Celkový bias bude, podle piling-up principu:

$$\begin{aligned} \epsilon_{1,\ldots,6} &= 2^{6-1} \prod_{i=1}^{6} \epsilon_i \\ &= 2^5 \cdot (2^{-2})^5 \cdot 2^{-3} \\ &= 2^{5-10-3} = 2^{-8} = \frac{1}{256} \end{aligned} \qquad (5.3)$$

Obdobně lze vyjádřit vztahy pro ostatní výstupní bity posledního `MixColumns`, rychle se však ukáže, že v nich nelze dosáhnout lepšího biasu než $\frac{1}{256}$ a že v některých případech je bias dokonce horší.

5.4.2 Aproximace pro sloučený SubBytes a MixColumns

Tato aproximace je použitelná pro libovolný počet rund Baby Rijndael, včetně variant s větším počtem rund než čtyři. Požadavky na aproximaci jsou stejné jako pro jakoukoliv jinou aproximaci v lineární kryptoanalýze:

- Vstup poloviny (tzn. dvou) `SubBytes` v poslední rundě bude aktivní.
- Celkový bias aproximace bude co nejvyšší.

Vhodný postup pro sestavení aproximace, která vyhovuje těmto požadavkům, je:

- Pro Baby Rijndael o dvou rundách:
  1. V první rundě zvolíme pro `SubBytes+MixColumns` libovolnou z aproximací s biasem $\frac{1}{4}$.
  2. Tato aproximace jednoznačně určuje aktivní bity v otevřeném textu a také aktivní bity ve vstupu dvou `SubBytes` druhé rundy.
  3. Protože je aktivní pouze jeden aproximovaný S-box, je celkový bias jednoduše $\frac{1}{4}$.
- Pro Baby Rijndael o třech rundách:
  1. V druhé rundě zvolíme jeden `SubBytes+MixColumns` jako aktivní a zvolíme pro něj libovolnou aproximaci s biasem $\frac{1}{4}$.
  2. Tím jsou jednoznačně určeny aktivní bity v `SubBytes` v poslední rundě.
  3. První runda bude nutně mít na výstupu jednu čtveřici bitů neaktivní. To nelze dosáhnout s biasem $\frac{1}{4}$ (viz bod 1d vlastností `SubBytes+MixColumns` na straně 46), volíme tedy libovolnou takovou aproximaci z těch s biasem $\frac{1}{8}$. Ta jistě existuje vzhledem k bodu 2c vlastností `SubBytes+MixColumns`.
  4. Aktivní bity otevřeného textu jsou tím jednoznačně určeny.
  5. Celkový bias bude podle piling-up principu:[20]

$$\epsilon_{1,2} = 2^{2-1} \cdot \frac{1}{4} \cdot \frac{1}{8} = \frac{1}{16} \tag{5.4}$$

[20]Pozn.: Vidíme, že to je výrazně lepší bias než u aproximace s nahrazením `MixColumns` pomocí vztahů pro jeden jeho bit.

- Pro Baby Rijndael o čtyřech a více rundách:

  1. Pro čtyřrundovou a delší verzi Baby Rijndael je mimořádně užitečnou skutečností to, že v `SubBytes+MixColumns` existují smyčky, které mají na vstupu pouze polovinu aktivních čtveřic bitů a které se po dvou rundách vrátí do výchozího stavu s tím, že použijí právě dvě aproximace s biasem $\frac{1}{4}$ a právě dvě aproximace s biasem $\frac{1}{8}$: Máme-li na vstupu `SubBytes+MixColumns` v i-té rundě aktivní bity `0x0303`, pak po substituci (dvakrát bias $\frac{1}{4}$) z nich vznikne `0x1D1D`. Ten projde operací `ShiftRows` beze změny (obě `0xD` si vymění místa) a následně má v rundě $i+1$ substituci (s biasem dvakrát $\frac{1}{8}$) zpět na `0x0303`, který opět projde operací `ShiftRows` beze změny. V $i+2$. rundě potom máme na vstupu S-boxů stejné aktivní bity jako v i-té rundě a tento postup můžeme opakovat.[21] Podstatné je, že za předpokladu, že jsou aktivní oba S-boxy, nemůže existovat žádná jiná aproximace dvou rund, která by měla nižší bias — je to dané body 1a a 1d vlastností `SubBytes+MixColumns` (str. 46).
  2. Tímto způsobem lze s maximálním možným biasem aproximovat libovolný sudý počet rund. V poslední rundě budou aktivní dvě čtveřice bitů, což je přesně ideální množství pro zjišťování posledního rundovního klíče. Zbývá ovšem otázka, jak se dostat do tohoto výhodného počátečního stavu:
     - Pro šifry s lichým počtem rund můžeme začít přímo v požadované smyčce, tzn. její počátek přímo určuje aktivní bity v otevřeném textu.
     - Pro šifry se sudým počtem rund by šlo využít bodu 2c vlastností `SubBytes+MixColumns` (str. 46) a s biasem $\frac{1}{8}$ v jediném aktivním `SubBytes+MixColumns` si tento stav jednoduše vygenerovat.
     - V obou případech může ale existovat i výhodnější cesta: Například pro sudý počet rund se aktivní bity `0x0001` otevřeného textu v jediném aktivním `SubBytes+MixColumns` přepíší na `0x93` s biasem $\frac{1}{4}$, který je následně pomocí `ShiftRows` rozložen na `0x9003`, tyto bity jdou převést (dvakrát bias $\frac{1}{4}$) na `0xD11D`, z něj po `ShiftRows` vznikne `0xDD11` a ten už jde substituovat (dvakrát bias $\frac{1}{8}$) na `0x0309`, který zahajuje smyčku. Do smyčky jsme se tedy dostali až po třech rundách,

[21] Toto není jediná podobná smyčka. Obdobně existuje smyčka pro `0x0909` a `0x1D1D` nebo pro `0x0606` a `0x5757`. Patrně existují i další.

ovšem druhá i třetí runda proběhly se stejným maximálním biasem, s jakým by proběhla smyčka, a první runda proběhla s biasem dvakrát vyšším, než by byl při aplikování předchozího bodu.

3. Celkový bias pro sudý počet rund $n = 2k, k \geq 1$, tak bude:

$$\begin{aligned}\epsilon_{1,\ldots,4k-3} &= 2^{4k-3-1} \cdot (2^{-2})^{2(k-1)+1} \cdot (2^{-3})^{2(k-1)} \\ &= 2^{4k-4-4k+2-6k+6} \\ &= 2^{-6k+4}\end{aligned}$$

a pro lichý počet rund $n = 2k + 1, k \geq 1$ maximálně:

$$\begin{aligned}\epsilon_{1,\ldots,4k} &= 2^{4k-1} \cdot (2^{-2})^{2k} \cdot (2^{-3})^{2k} \\ &= 2^{4k-1-4k+2-6k} \\ &= 2^{-6k-1}\end{aligned}$$

Speciálně pro $n = 4$ tedy $\epsilon_{1,\ldots,5} = 2^{-8} = \frac{1}{256}$.

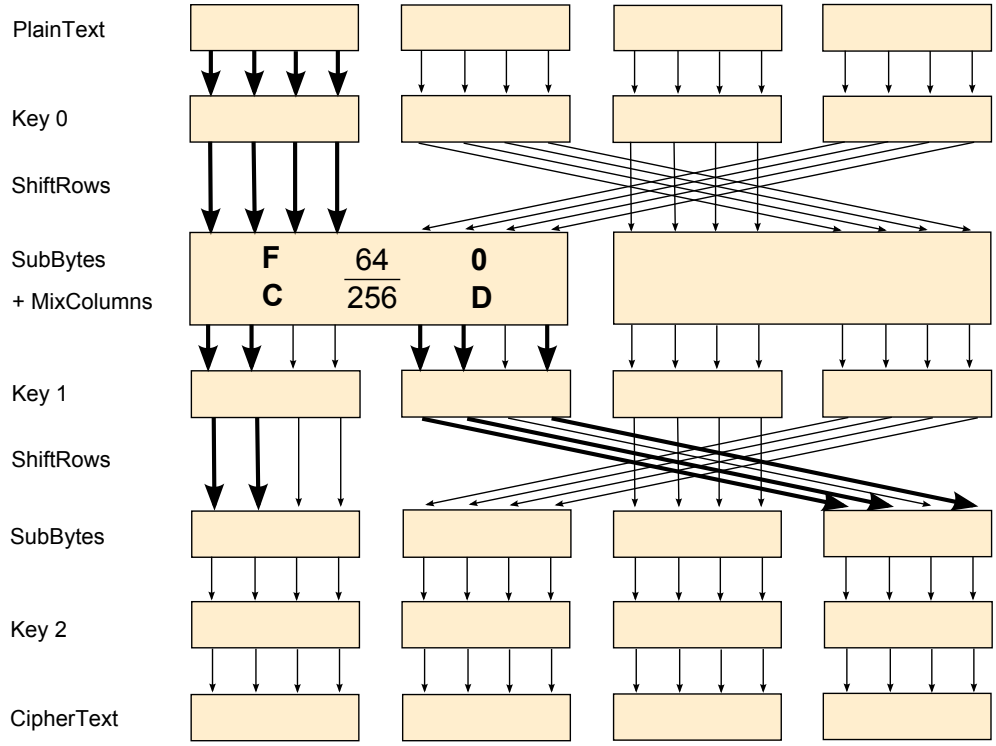
Tento poněkud ad-hoc přístup k sestavení lineární aproximace jsem následně ověřil v programu `LinearCryptanalysis`, do kterého jsem pod parametr `BESTPATH` přidal hledání optimální lineární aproximace hrubou silou, prozkoušením všech možností. Program následně vypíše jednu z optimálních aproximací z mnoha možných a uvede u ní celkový dosažený bias. Program by samozřejmě šlo snadno upravit na výpis všech těchto optimálních aproximací, to je čistě jen otázka vhodného způsobu dynamického ukládání dosud nejlepších aproximací během výpočtu.

Důležité je, že pro dvě (parametr `-a 2r`), tři (parametr `-a 3r`) i čtyři rundy (parametr `-a 4r`) mi program jako optimální nalezl aproximaci, která má přesně stejný bias jako aproximace, které jsem předem určil pomocí algoritmu uvedeného výše. Z toho vyplývá, že všechny mnou nalezené aproximace patřily mezi množinu optimálních aproximací a tedy algoritmus pro jejich určení je pravděpodobně správný. To je užitečné pro varianty šifry s větším počtem rund, kdy už hledání optimálních aproximací hrubou silou narazí na výpočetní limity.

V další práci jsem použil takto zkonstruované aproximace pro dvě rundy (obrázek 5.4), pro tři rundy (obrázek 5.5) a dvě odlišné varianty pro čtyři rundy (obrázek 5.6, 5.7).

5.5 Hledání posledního rundovního klíče

Pro hledání posledního rundovního klíče jsem do programu `LinearCryptanalysis` a speciálně do třídy `TBabyRijndaelLC` napro-



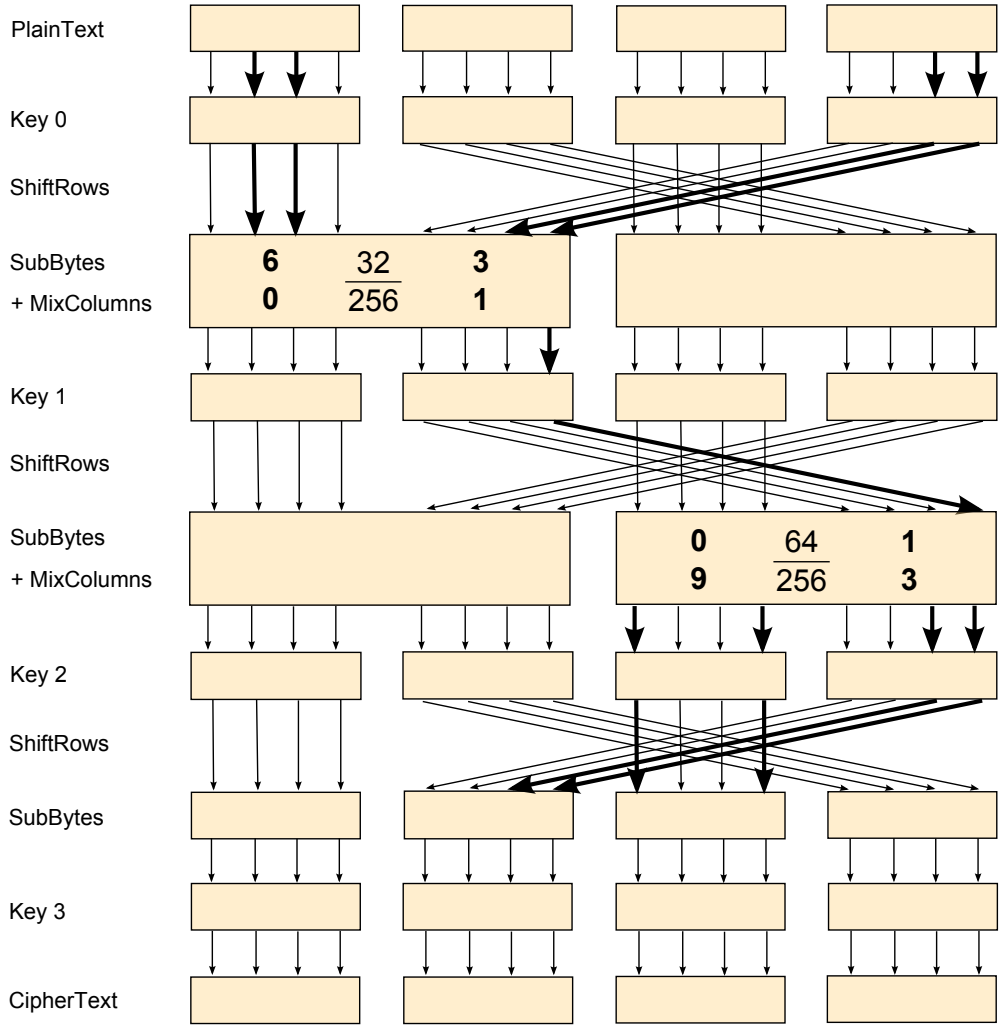
Obrázek 5.4: Aproximace pro 2 rundy pomocí sloučeného `SubBytes` a `MixColumns`

gramoval sadu funkcí, které umožňují provádět lineární kryptoanalýzu pro obecně libovolnou kombinaci vstupů. Ne všechny možnosti jsou uživatelsky přístupné, některé části analýzy jsou napsané přímo v kódu a jejich změna vyžaduje rekompilaci programu, mnoho úloh lze ale řešit i bez programátorských zásahů jen prostřednictvím parametrů na příkazové řádce.

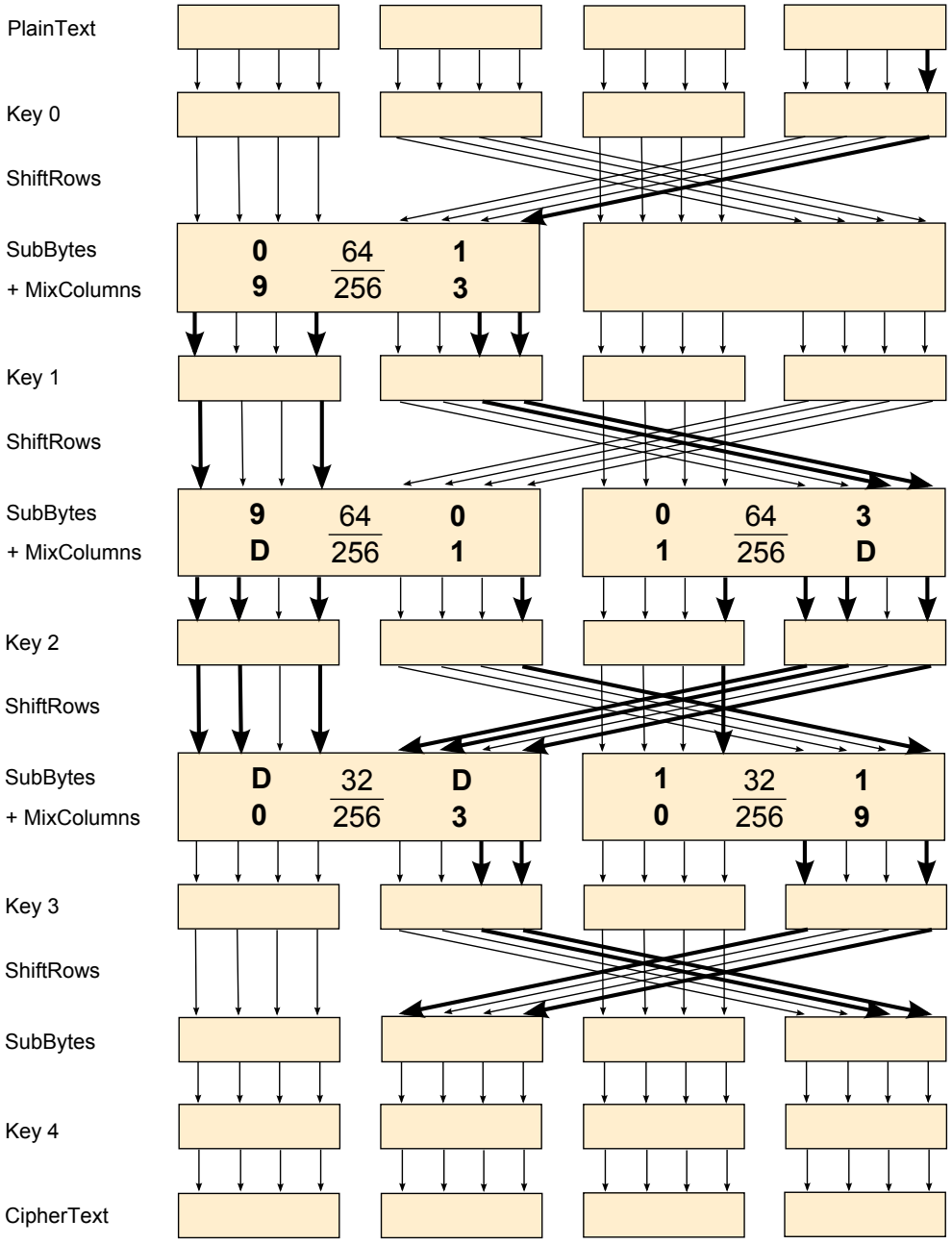
5.5.1 Programátorské řešení

Třída `TBabyRijndaelLC` nabízí pro lineární kryptoanalýzu zejména tyto metody a vlastnosti:

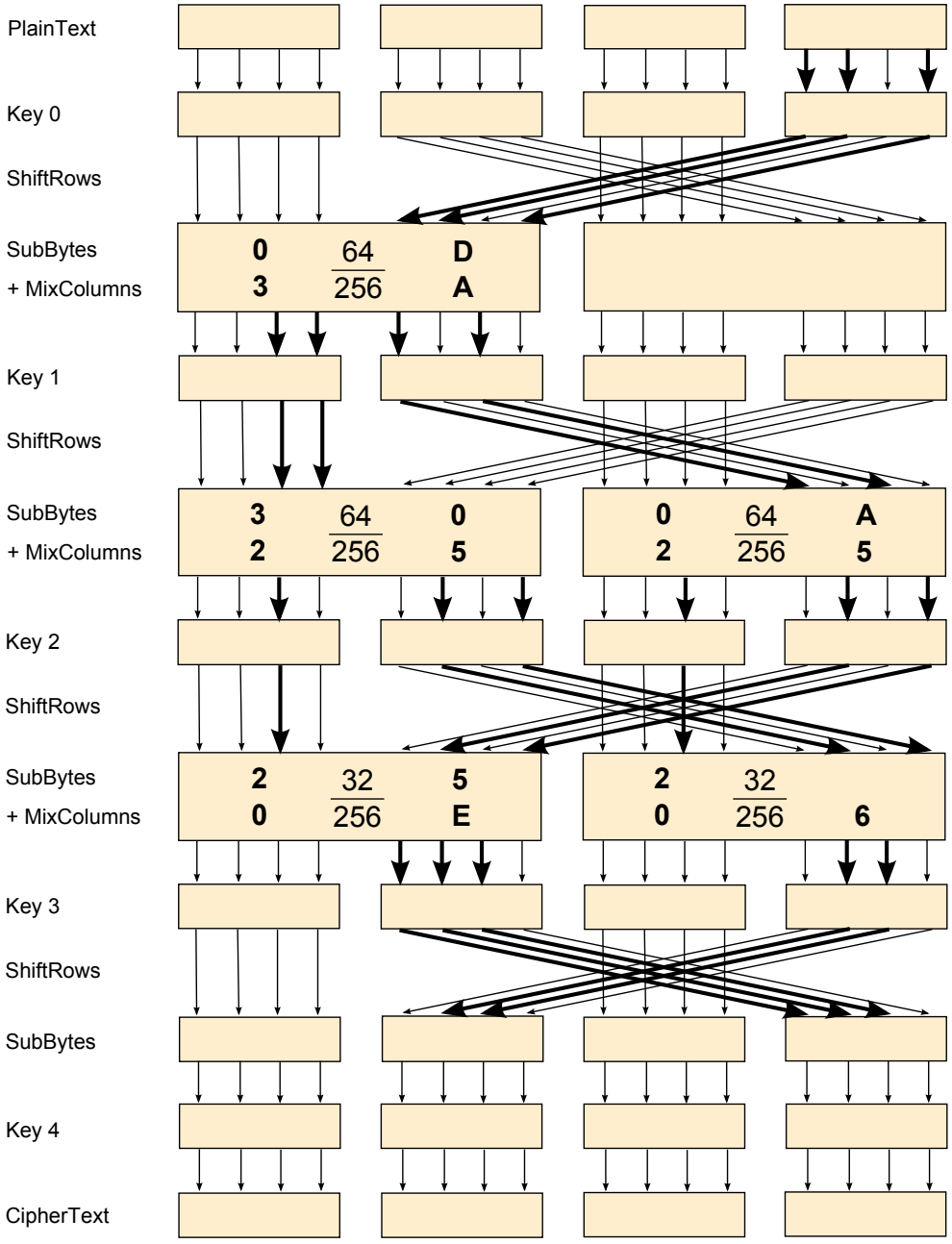
- `property SampleCount: integer`: Určuje velikost množiny vzorků, která bude použita pro kryptoanalýzu. Vzorky samotné jsou nadefinovány jako vlastnosti `PlainTexts[Index: integer]: Word` a `CipherTexts[Index: integer]`. Volající aplikace si je tedy může zcela libovolně nastavit. Měla by ovšem dodržet požadavky lineární



Obrázek 5.5: Aproximace pro 3 rundy pomocí sloučeného `SubBytes` a `MixColumns`



Obrázek 5.6: Aproximace pro 4 rundy (verze A) pomocí sloučeného `SubBytes` a `MixColumns`



Obrázek 5.7: Aproximace pro 4 rundy (verze B) pomocí sloučeného `SubBytes` a `MixColumns`

kryptoanalýzy, tzn. zejména aby položka z `CipherTexts` vznikla z odpovídající položky z `PlainTexts` zašifrováním konstantním klíčem.

- `procedure GenerateSamples(Count: integer; Key: TKey)`: Metoda umožňuje vygenerovat s daným klíčem `Key` požadovaný počet `Count` dvojic $(OT, \check{S}T)$. Generování probíhá tak, že napřed se vytvoří pole 65536 otevřených textů, kde na i-té pozici je text i, následně se toto pole náhodně zamíchá a ze zamíchaného pole se vezme prvních `Count` prvků. Tyto se následně zašifrují dodaným klíčem a uloží do pole šifrových textů.

- `procedure GenerateManySamples(MultiplesOf64K: integer; Key: TKey)`: Metoda opět generuje množinu otevřených a šifrových textů, tentokrát ale mnohem větší, než šifra vůbec připouští: Prvních 65536 vzorků je vytvořeno standardním způsobem jako u předchozí metody, další vzorky pak vznikají tak, že se k šifrování místo zadaného klíče `Key` použije klíč od něj odvozený, a to tak, aby aktivní S-boxy v poslední rundě nebyly dotčeny (tzn. pokud aktivní S-boxy mají masku `0xF00F`, budou se v klíči měnit bity odpovídající masce `0x0FF0`. To ovšem porušuje základní požadavek lineární kryptoanalýzy, aby totiž klíč zůstal konstatní, takto vygenerované vzorky tedy lze použít pouze k některým doplňkovým analýzám, nikdy ne pro hlavní funkce.

- `property Approximation: TBabyRijndaelLCApproximation`: Vlastnost určuje, která lineární aproximace se má použít pro lineární kryptoanalýzu. Aproximace je definována jako potomek abstraktní statické třídy `TBabyRijndaelLCApproximation`, která definuje nutné parametry pro využití aproximace:

  - `function ShortcutName: string;`: Vrací stručný popis aproximace pro použití v příkazové řádce s parametrem `-a`.
  - `function NumberOfRounds: integer`: Vrací počet rund, pro který je aproximace určená. Pro aproximaci 5.6 tedy vrací 4.
  - `function GetNumberOfPossibleKeys: Word`: Vrací počet klíčů, které se ověřují v rámci lineární kryptoanalýzy: pokud by byl v poslední rundě aktivní jeden S-box, hodnota je 16, pro dva aktivní S-boxy to je 256, atd.[22] Pro aproximaci 5.6 tedy vrací 256.

[22] Údaj by ovšem šel dopočítat z `KeyPartMask`.

- `function GetKeyByIndex(Index: Word): TKey`: Vrací klíč, který odpovídá danému indexu, v podstatě tedy jde o „rozptýlení“ indexu do aktivních S-boxů v poslední rundě. Pro aproximaci 5.6 tedy vrací pro $Index = 0$ hodnotu `0x0000`, pro $Index = 1$ hodnotu `0x0001` a pro $Index = 255$ hodnotu `0x0F0F`.
- `function KeyPartMask: TKey`: Vrací masku pro bity klíče v poslední rundě. Pro aproximaci 5.6 tedy vrací `0x0F0F`, protože aktivní je druhý a čtvrtý S-box.
- `function ExpectedBias: double`: Vrací očekávaný absolutní bias lineární aproximace, pro aproximaci 5.6 tedy $\frac{1}{256}$.
- `function PlainTextMask: Word`: Vrací masku pro aktivní bity otevřeného textu, pro aproximaci 5.6 tedy `0x0001`.
- `function LastRoundTextMask: Word`: Vrací masku pro aktivní bity vstupující do posledního S-boxu. Pro aproximaci 5.6 tedy vrací `0x0903`.
- Aby program `LinearCryptanalysis` uměl s nově nadefinovanou aproximací pracovat, je třeba ji do něj zaregistrovat pomocí metody `TBabyRijndaelLCApproximation.Register` podobně, jako je to pro předdefinované aproximace provedeno na konci jednotky `uLinearCryptanalysisApproximations.pas` v sekci `initialization`. Potom se aproximace objeví například v nápovědě pro příkazovou řádku a uživatel si ji bude moci vybrat pomocí parametru `-a`.

`function Solve: TKey`: Toto je hlavní metoda pro nalezení klíče pomocí lineární kryptoanalýzy. Využije nadefinovanou aproximaci a množinu otevřených a šifrových textů z předchozích vlastností a použije na ni algoritmus popsaný v kapitole o lineární analýze. V průběhu práce vyzkouší všechny možné podklíče, zjistí jejich skutečný bias a porovnáním s teoretickým biasem určí nejlepší klíč. Tento klíč také vrátí volajícímu. Všechny vyzkoušené klíče, frekvence úspěšnosti lineární aproximace pro každý z nich a také skutečné biasy uloží do proměnných instance, odkud si je volající může přečíst prostřednictvím vlastností `property BestKeys[Index: integer]: Word`, `property BestKeyBiases[Index: integer]: Double` a `property BestKeyFrequencies[Index: integer]: integer`.

•

- Metody `procedure AnalyzeKeyBits`, `procedure AnalyzeKeyBits2` a `procedure AnalyzeKeyBitsDynamic` slouží k pokročilejší analýze klíče po jednotlivých bitech a jejich použití je nad rámec této práce.

5.5.2 Uživatelské rozhraní

Uživateli nabízí program`LinearCryptanalysis` dvě základní operace a několik parametrů pro jejich řízení:

- `-n` *`počet`*: Kryptoanalýza se bude provádět na *počet* náhodně vygenerovaných dvojicích otevřeného a šifrového textu. Výchozí hodnota je 65535 dvojic.

- `-a` *`id`*: Kryptoanalýza použije aproximaci pojmenovanou *id*. V programu je natvrdo zapsaných několik aproximací: `4R` (obrázek 5.6), `4R_V2` (obrázek 5.7), `3R` (obrázek 5.5), `3R_V2` (obrázek 5.3) a `2R` (obrázek 5.4). Výchozí aproximace je `4R`.

5.5.2.1 Příkaz `ONEKEY`

Příkaz `ONEKEY` spustí lineární kryptoanalýzu pro jeden zvolený šifrovací klíč a následně vypíše na standardní výstup podrobné informace o průběhu. Klíč je ve výchozím stavu nastaven na `0x6b5d`[23], lze ale zvolit jiný klíč pomocí parametru `-k`, za kterým následují čtyři hexadecimální číslice, např. `-k 1234`.

Výstupem je CSV soubor (oddělovačem je středník, aby šel soubor snadno načíst do Excelu), ve kterém je postupně vypsáno:

- Základní informace o použité aproximaci a jejích nastaveních, tzn. popis aproximace, velikost množiny vzorků, hlavní klíč (tzn. klíč, ze kterého se v rámci key expansion vygenerují všechny rundovní klíče), hledaný klíč v poslední rundě, očekávaný bias, očekávaný počet vzorků, ve kterých by měla být lineární aproximace splněna.

- Hodnota kandidátního klíče, jehož bias nejlépe odpovídá teoretickému biasu.

- Seznam všech kandidátních klíčů seřazený podle blízkosti jejich skutečného biasu k teoretickému biasu, včetně velikosti skutečného biasu a absolutního rozdílu od teoretického biasu.

[23] Tento klíč je v [1] zvolen jako demonstrační.

- Pořadí, na kterém se nachází skutečný klíč.

  Pozn.: Funkce pro lineární kryptoanalýzu samozřejmě tuto hodnotu nezná, protože nezná skutečný klíč. Volající program ovšem tento skutečný klíč zná, protože pro něj generoval množinu vzorků, a následně, po provedení lineární kryptoanalýzy, s ním zkonfrontuje seřazený seznam kandidátních klíčů. Očekávali bychom, že správný klíč se bude nacházet na první nebo jedné z prvních pozic.

- Detailnější analýza jednotlivých bitů klíče. V praxi se totiž ukazuje, že s rostoucím počtem rund klesá úspěšnost nalezení správného klíče[24]. Navrhl jsem proto několik metod, jak i u nesprávně umístěného klíče nalézt aspoň některé jeho bity. Podrobný popis těchto metod je nad rámec této práce, zde se omezím na konstatování, že výstupem metod je pro každý bit určení jednoho ze čtyř možných stavů:

  - Bit vůbec nebyl zjišťován (leží mimo masku danou použitou lineární aproximací). Tento bit je ve výstupu značen pomocí tečky.
  - Bit se podařilo určit správně, tzn. vypočítaný bit odpovídá bitu skutečně použitého klíče. Tento bit je na výstupu značen nulou nebo jedničkou, podle toho, jaká je jeho skutečná hodnota.
  - Bit se podařilo určit, ale špatně, tzn. metoda určila hodnotu bitu právě opačnou, než jaká je ve skutečném klíči. Tento bit je značen písmenem `x`.
  - Bit se nepodařilo určit, jeho hodnota je příliš nejistá. Tento bit je značen otazníkem.

  Pro snadnější zpracování tohoto výstupu jsou dále vypsány počty správně určených bitů a počty špatně určených bitů a u některých metod ještě další statistiky výpočtu (např. počet zohledňovaných kandidátních klíčů).

5.5.2.2 Příkaz `ALLKEYS`

Příkaz `ALLKEYS` navazuje na funkčnost příkazu `ONEKEY` a rozšiřuje ji na všechny klíče: Zatímco `ONEKEY` prováděl lineární kryptoanalýzu pro jeden určený klíč, `ALLKEYS` postupně zkouší všech 65536 možných klíčů: pro každý vygeneruje množinu vzorků otevřeného a šifrového textu, provede lineární kryptoanalýzu, určí pozici správného klíče a provede detailní bitovou analýzu. Umožňuje tak ověřit, jestli jsou metodami lineární kryptoanalýzy některé klíče snáze odhalitelné než jiné.

[24] tzn. jeho pozice v seřazeném seznamu kandidátních klíčů se v průměru stále zvyšuje

Výstupem je opět CSV soubor, který pro každý ověřovaný klíč zachycuje správný klíč v poslední rundě, pozici tohoto správného klíče v seřazeném seznamu kandidátních klíčů, jeho skutečný bias a absolutní rozdíl od teoretického biasu a také detailní analýzu bitů podle některé z mnou navržených metod.

Poznámka: Výpočet příkazu `ALLKEYS` poměrně dlouho trvá, na mém počítači (Intel Core i5-2400S, 16 GB RAM, Windows 7 x64) pro 65536 vzorků až dva dny. Není to až tak úplně nečekané, protože hlavní smyčka probíhá 2^{40}krát (65536 klíčů krát 65536 vzorků krát 256 kandidátních klíčů) a obsahuje značně složité operace, nemluvě už o dalších pomocných smyčkách (generování vzorků, detailní analýza bitů apod.). Doporučuje se trpělivost a nebo výstup příkazu přesměrovat do souboru, jehož obsah zůstane zachován poté, co výpočet přerušíte.

5.5.3 Dosažené výsledky pro 2 rundy

Pro kryptoanalýzu redukovaného Baby Rijndael se dvěma rundami jsem použil lineární aproximaci podle obrázku 5.4 s celkovým biasem $\frac{1}{4}$. Ověřoval jsem všechny existující klíče (operace `ALLKEYS`), každý s 400 vzorky otevřeného a šifrového textu.

Zhodnocení výsledků pro všechny klíče ukazuje, že dvourundový Baby Rijndael je lineární kryptoanalýzou snadno prolomitelný.

Průměrná pozice správného klíče	1,01
Směrodatná odchylka správného klíče	0,16
Medián pozice správného klíče	1
Modus pozice správného klíče	1
Nejlepší pozice správného klíče	1
Nejhorší pozice správného klíče	16

Tabulka 5.2: LK pro 2 rundy Baby Rijndael

Podrobné výsledky naleznete na přiloženém CD v souboru `Vysledky/2rundy/AllKeys.csv`.

Téměř každý klíč nalezneme hned na první pozici, jak bychom od lineární kryptoanalýzy očekávali, pouze 257 klíčů (tzn. 0,39 procenta) se nachází na pozici jiné, a i potom jde o některou z předních pozic — nejhorší pozice 16 značí, že mezi prvními šestnácti klíči určitě nalezneme i ten správný, tzn. v případě potřeby stačí těchto 16 klíčů vyzkoušet.

Odhlížíme zde ovšem od praktické realizace útoku: při 256 kandidátních klíčích a 400 vzorcích na kandidátní klíč musíme v rámci lineární kryptoanalýzy provést 102400 výpočtů, což je o 56 procent více, než bychom

potřebovali na útok hrubou silou, kterému by navíc stačil jediný vzorek otevřeného a šifrového textu. Snižováním počtu vzorků úspěšnost lineární kryptoanalýzy klesá, průměrná pozice je cca 1,05 pro 256 vzorků (s náročností obdobnou hrubé síle přes všechny klíče) a 1,57 pro 128 vzorků (odpovídá útoku hrubou silou v průměrném případě). Použitím alternativní lineární aproximace s jen jedním aktivním S-boxem v poslední rundě, např. s maskou otevřeného textu `0x3006`, a přepisem `0x36` na `0x10` s biasem $\frac{1}{8}$ v S-boxu, sice klesá výpočetní náročnost, bohužel ale také přesnost určení klíče — v tomto případě na průměrnou pozici 2,64.

Nicméně z hlediska schopnosti šifry odolávat kryptoanalýze není až tak podstatné, jak náročná je analýza ve srovnáním s prostým útokem hrubou silou. Podstatná je zde skutečnost, že pouhé dvě rundy nezajišťují dostatečnou difúzi klíče na to, aby šifra odolala, a to přesto, že její S-box je dobře navržen.

5.5.4 Dosažené výsledky pro 3 rundy

Pro kryptoanalýzu třírundové verze Baby Rijndael jsem použil aproximaci podle obrázku 5.5 se sloučeným `SubBytes` a `MixColumns` a biasem $\frac{1}{16}$ i aproximaci podle obrázku 5.3 s biasem $\frac{1}{256}$, která vyjadřovala první bit `MixColumns` pomocí součtu vstupů. Velikost množiny vzorků jsem na počátku testů nastavil na 10000 a protože dosažené výsledky nebyly úplně dobré, postupně jsem ji zvětšoval až na 65535 vzorků. I s takto velkým rozsahem vzorků byly výsledky lineární kryptoanalýzy poměrně špatné:

	aprox. obr. 5.5 (2 aktivní S-boxy)	aprox. obr. 5.3 (1 aktivní S-box)
Průměrná pozice správného klíče	51,14	9,13
Směrodatná odchylka správného klíče	74,29	5,29
Medián pozice správného klíče	21	8
Modus pozice správného klíče	1	16
Nejlepší pozice správného klíče	1	1
Nejhorší pozice správného klíče	256	16
Očekávaný bias	0,0625	0,00390625
Průměrný skutečný bias	0,062501	0,002931
Směrodatná odchylka skutečného biasu	0,036643	0,002535

Tabulka 5.3: LK pro 3 rundy Baby Rijndael

Podrobné výsledky naleznete na přiloženém CD v souboru `Vysledky/3rundy/AllKeys-A.csv` pro první aproximaci a `Vysledky/3rundy/AllKeys-B.csv` pro druhou.

Na první pohled je zřejmé, že původní pěkné pozice správného klíče u dvourundového Baby Rijndael se podstatně zhoršily — sice se stále dařilo dosáhnout i první pozice, průměrná i střední hodnota je ale podstatně horší. Také ostatní statistické veličiny vykazují značné zhoršení proti dvěma rundám: z toho vidíme, že ve třech rundách už dochází k podstatně lepší difúzi.

Pokud chceme hodnotit úspěšnost lineárních aproximací navzájem, tabulka by mohla svádět k závěru, že přestože má druhá aproximace mnohem horší bias, dává lepší výsledky. Musíme si však uvědomit, že zatímco první aproximace má dva aktivní S-boxy a hledá tedy nejlepší z 256ti možných klíčů, druhá aproximace má aktivní S-box jen jeden a hledá mezi 16ti klíči. Ve výsledku tedy druhá aproximace nachází správný klíč v průměru až v druhé polovině seznamu a nejčastěji (modus) dokonce na poslední pozici, zatímco první aproximace nachází klíč zhruba na přelomu první a druhé pětiny seznamu a nejčastěji stále na první pozici. Navíc v polovině případů nacházíme správný klíč v prvních 8 procentech seznamu, zatímco u druhé aproximace až ve více než 31 procentech. Z toho jednoznačně vyplývá, že metoda náhrady operace `MixColumns` vyjádřením jediného výstupního bitu pomocí součtu určitých bitů vstupujících je méně vhodná než aproximace pomocí `SubBytes+MixColumns`.

Zkoumání detailních výsledků první aproximace ukazuje zajímavou skutečnost: Úspěšnost nalezení správného klíče v ní velmi výrazně závisí na hodnotě druhé a třetí hexadecimální číslice[25] hlavního klíče šifry: Seřadíme-li podrobné výsledky pro všechny možné klíče podle dvou prostředních číslic hlavního klíče, zjistíme, že zatímco průměrná pozice správného klíče se pro klíče jako `0x?00?` nebo `0x?FC?` velmi blíží jedné (tzn. správný klíč má nejlepší bias), klíče `0x?01?` nebo `0x?FE?` nacházejí správný klíč na pozici zhruba kolem dvaceti až třiceti a například pro klíče `0x?06?` nebo `0x?FF?` nácházíme správný klíč typicky až úplně na konci tabulky na 220–256. pozici. První pozice se podařilo dosáhnout pro celkem 24010 klíčů (36,6 procenta všech klíčů), nejvýše šestnácté pozice pro 29203 klíčů (44,6 procenta všech klíčů).[26]

Důvody pro toto chování se mi nepodařilo odhalit. Závislost je příliš přesná na to, aby mohla být náhodná, na druhou stranu ale nenacházím žádný mechanismus, jak by hodnota *hlavního klíče* mohla tak dramaticky ovlivňovat schopnost lineární kryptoanalýzy odhalit hodnotu *posledního rundovního klíče*, který by z principu expanze klíče v šifře Baby Rijndael ne-

[25] To odpovídá skutečnosti, že použitá aproximace má v poslední rundě aktivní druhý a třetí S-box.

[26] viz soubor `Vysledky/3rundy/AllKeys-A-0xx0.csv`

měl na hlavním klíči nijak lineárně záviset. Prověření možnosti, že expanze klíče přesto vytváří přímou závislost mezi hlavním klíčem šifry a úspěšností hledání posledního rundovního klíče v třírundové verzi Baby Rijndael, by mělo být předmětem dalšího zkoumání.

5.5.5 Dosažené výsledky pro 4 rundy

Plná, čtyřrundová verze Baby Rijndael pokračuje při lineární kryptoanalýze v trendu viditelném z kratších verzí: kvalita difúze se stále zvyšuje a schopnost nalezení klíče pomocí lineární kryptoanalýzy je stále menší, což se projevuje v tom, že pozice správného klíče v seznamu kandidátních klíčů seřazených podle biasu je stále proměnlivější. Zmizela i pravidelnost ve vztahu mezi hlavním klíčem a pozicí správného posledního pozorovaná v třírundové verzi — ve čtyřrundové verzi už jsem nenašel žádnou závislost mezi klíčem a schopností lineární kryptoanalýzy ho odhalit. To značí, že Baby Rijndael stačí čtyři rundy k dosažení odolnosti proti lineární kryptoanalýze.

Kryptoanalýzu plné šifry jsem prováděl se dvěma aproximacemi, A (obrázek 5.6) a B (obrázek 5.7), vždy na množině všech 65536 vzorků otevřeného a šifrového textu. Jedním ze zajímavých výsledků byl velmi výrazný rozdíl v pořadí nejlepšího klíče:

	aprox. A (obr. 5.6)	aprox. B (obr. 5.7)
Průměrná pozice správného klíče	106,54	63,91
Směrodatná odchylka správného klíče	79,99	59,86
Medián pozice správného klíče	84	45
Modus pozice správného klíče	256	1
Nejlepší pozice správného klíče	1	1
Nejhorší pozice správného klíče	256	256
Očekávaný bias	0,00390625	0,00390625
Průměrný skutečný bias	0,004047	0,003921
Směrodatná odchylka skutečného biasu	0,002407	0,001586

Tabulka 5.4: LK pro 4 rundy Baby Rijndael

Podrobné výsledky naleznete na přiloženém CD v souboru `Vysledky/4rundy/AllKeys-A.csv` pro aproximaci A a `Vysledky/4rundy/AllKeys-B.csv` pro aproximaci B.

Ze statistik vidíme, že průměrná pozice správného klíče se při použití aproximační funkce B zlepšila o 42,63 pozic (40,0 procenta), o 39 pozic (46,4 procenta) se zlepšila střední hodnota pozice a nejčastější hodnota pozice se z 256 (nejhorší možná pozice) změnila na 1 (nejlepší možná pozice). Také o 25,2 procenta klesla variabilita pozice vyjádřená směrodatnou odchylkou.

Rozdíl mezi aproximacemi A a B spočívá v jejich konstrukci. Aproximaci A jsem sestavil v rámci hledání algoritmu pro vytváření dobrých aproximací a soustředil jsem se v ní pouze na dosažení dobrého biasu a dvou aktivních S-boxů v poslední rundě. Aproximaci B jsem sestavoval dodatečně tak, aby navíc používala co největší počet bitů otevřeného textu a co největší počet bitů vstupujících do poslední sady `SubBytes` — zatímco A vyjadřuje vztah mezi pěti bity celkem, B používá bitů osm (tři v otevřeném textu a pět před `SubBytes`). Ostatní charakteristiky jsou totožné. A i B samozřejmě používají každá jiné aproximace jednotlivých S-boxů, ale jejich struktura — např. aktivní S-boxy v jednotlivých rundách — a také jejich bias jsou totožné. Zdá se tedy, že počet zúčastněných bitů v aproximaci šifry má vliv na schopnost nalézt správný klíč na dobré pozici.

Tuto hypotézu jsem následně prakticky ověřil tím, že jsem obě aproximace mírně upravil na straně otevřeného textu: Využil jsem toho, že podle bodu 1c z vlastností `SubBytes+MixColumns` (str. 46) existují pro každou aproximaci `SubBytes+MixColumns` s biasem $\frac{1}{4}$ dva různé vstupy, které dají stejný výstup. Vstupem pro první S-box je otevřený text, tudíž jsem pro něj zvolil druhou z hodnot, která se v S-boxu transformuje na původní výstup. Tak vznikly aproximace A' a B', které se od A a B liší *pouze* v počtu aktivních bitů v plaintextu: Zatímco A mělo jeden aktivní bit (`0x0001`), A' má aktivní bity 3 (`0x000D`), a obdobně k B s třemi aktivními bity (`0x000D`) jsem vytvořil B' se dvěma aktivními bity (`0x000C`). Na tyto alternativní aproximace jsem použil operaci `ALLKEYS` a měřil, jak se změní průměrná pozice správného klíče. Skutečně se ukázalo, že aproximace A' svoji pozici proti A poněkud zlepšila, v souladu s tím, že A' má víc aktivních bitů než A, a aproximace B' pozici naopak poněkud zhoršilo, což odpovídá sníženému počtu jejích aktivních bitů proti B. Navíc se ukázalo, že B' stále má lepší průměrnou pozici než A', stejně jako má větší počet aktivních bitů.

Jde však nanejvýš o pozorování, rozhodně ne o důkaz postavený na solidních teoretických základech. Už proto, že teoretické zdůvodnění tohoto jevu se mi nepodařilo najít ani v literatuře, ani úvahou nad tím, jak lineární kryptoanalýza funguje. Podrobnější rozbor příčin a důsledků by měl být předmětem dalšího výzkumu.

Hledal jsem také další způsoby, jak zlepšit efekt lineární kryptoanalýzy na Baby Rijndael. Ověřoval jsem zejména tyto tři hlavní přístupy:

- Aproximace pro čtyři rundy vykazuje s rostoucím počtem vzorků neustálé zlepšování průměrné pozice správného klíče, bohužel při velikosti bloku 16 bitů je po 65536 vzorcích veškerý prostor vyčerpán. Šlo by ho nějakým způsobem uměle zvětšit?

- Když už nelze získat dostatečně určité výsledky na úrovni celého kandidátního klíče, šlo by alespoň určit některé jeho bity? V této otázce jsem dosáhl jistých dílčích výsledků, jde však o natolik rozsáhlou problematiku, že jsem se rozhodl nezahrnovat ji do této práce, dokud ji nebudu mít zpracovanou komplexně.

- Lineárních aproximací se stejným biasem lze pro šifru sestavit více, použitím různých aproximací jednotlivých S-boxů i volbou jiných aktivních bitů v otevřeném textu nebo v poslední sadě S-boxů. Zkoumal jsem, jestli bych zkombinováním výsledků z různých aproximací nedokázal zlepšit celkovou úspěšnost, nepodařilo se mi však žádnou spolehlivou metodu odhalit. Z časových důvodů jsem ovšem testoval jen malou množinu aproximací a jen základními postupy; do další práce by bylo vhodné se na podrobný průzkum interakce co největšího množství různých aproximací zaměřit, zejména v kombinaci s určováním jednotlivých bitů z předchozího bodu.

5.5.6 Zvětšení prostoru vzorků

Z pokusů, které jsem dělal pro 4 rundy, jasně vyplývá, že 65536 vzorků je málo. Otázka zní, šlo by prostor vzorků zvětšit, když máme šifru s 16bitovým blokem?

Do jisté míry ano, ale bohužel ne tak, aby to pomohlo v lineární kryptoanalýze. Pro diferenciální analýzu existuje možnost zvětšit prostor vzorků tím, že bloky šifrujeme ne jen jedním pevným klíčem, ale i dalšími klíči, které jsou z tohoto pevného klíče odvozené a liší se od něj pouze v bitech, které leží mimo aktivní S-boxy v poslední rundě. Za těchto okolností totiž vzniknou další unikátní dvojice otevřeného a šifrového textu, které se neliší ve své (diferenciální) aproximaci. Je to způsobeno tím, že díky konstrukci tzv. diferenční stopy z vnitřních stavů šifry úplně vypadnou rundovní klíče.

V lineární kryptoanalýze bohužel tuto techniku obecně použít nelze, protože vnitřní stavy šifry některé bity klíče stále obsahují. Značíme je $\sum K$ a používáme je při vysvětlování, jak lineární kryptoanalýza funguje, ve vlastním výpočtu je ale nepoužíváme, protože jejich hodnota je pro daný vzorek textů fixní a slouží jenom k tomu, že směruje pravděpodobnost buď k $\frac{1}{2} + \epsilon_{1,...,n}$ nebo k $\frac{1}{2} - \epsilon_{1,...,n}$. To ovšem platí pouze za podmínky, že $\sum K$ je konstantní, což platí tehdy, když použitý klíč je konstantní. Jakmile začneme používat více různých klíčů $K_1, K_2, ..., K_m$, stane se $\sum K$ proměnnou. Použitá lineární aproximace pak pro různé klíče bude obsahovat různé konstanty $\sum K_i, i \in \{1, 2, ..., m\}$. Pokud je expanze klíče dobře nadefinována, měly by se jednotlivé hodnoty rovnoměrně rozdělit mezi 0 a 1. V tu chvíli

ovšem je pravděpodobnost splnění lineární aproximace u části bloků směrována k $\frac{1}{2} + \epsilon_{1,\ldots,n}$ a u části bloků k $\frac{1}{2} - \epsilon_{1,\ldots,n}$, přičemž my *nevíme, která část bloků je směřována ke které hodnotě* a nemůžeme je tedy rozlišit. Můžeme je pouze zprůměrovat, což bude s rostoucím m stále pravděpodobněji směrovat pravděpodobnost splnění aproximace k $\frac{1}{2}((\frac{1}{2} + \epsilon_{1,\ldots,n}) + (\frac{1}{2} - \epsilon_{1,\ldots,n})) = \frac{1}{2}$, tedy do stejné situace, jako kdybychom bity volili zcela náhodně.

Přesto může tato technika do jisté míry fungovat v případě, že zvolíme při generování vzorků takové klíče K_i, aby *všechny* měly konstantní $\sum K_i$. To však lze zaručit pouze v případě, že klíče vytváří útočník. Využití je tedy možné pouze pro teoretickou analýzu, pro praxi technika použitelná není.

V rámci kryptoanalýzy Baby Rijndael jsem tuto techniku vyzkoušel na aproximaci A (obr. 5.6) a získal tak 131072 vzorků, se kterými se mi následně podařilo zlepšit průměrnou pozici správného klíče o 22,2 procent (tabulka 5.5).

	65536 vzorků	131072 vzorků
Průměrná pozice správného klíče	106,54	82,92
Směrodatná odchylka správného klíče	79,99	74,46
Medián pozice správného klíče	84	58
Modus pozice správného klíče	256	1
Nejlepší pozice správného klíče	1	1
Nejhorší pozice správného klíče	256	256
Očekávaný bias	0,00390625	0,00390625
Průměrný skutečný bias	0,004047	0,002191
Směrodatná odchylka skutečného biasu	0,002407	0,001452

Tabulka 5.5: LK pro 4 rundy Baby Rijndael a zvětšený počet vzorků

Podrobné výsledky naleznete na přiloženém CD v souboru `Vysledky/4rundy/AllKeys-A-131072.csv`.

5.6 Vyhodnocení výsledků

Z předchozích sekcí kapitoly je zřejmé, že na Baby Rijndael lze uplatnit techniky lineární kryptoanalýzy. V první sekci jsem ukázal, že šifru lze vyjádřit ve tvaru, který je vhodný pro použití lineární kryptoanalýzy. Druhá a třetí sekce ukazuje, že operace `SubBytes`, jediná nelineární komponenta Baby Rijndael, vykazuje i přes svoji definici pomocí modulární inverze určité lineární charakteristiky, které jsem ve čtvrté sekci využil pro sestavení lineární aproximací s vysokým biasem pro varianty šifry s různým počtem rund.

V páté sekci jsem ovšem ukázal, že ani tyto příznivé podmínky nejsou dostačující k tomu, aby se Baby Rijndael podařilo prolomit technikami lineární kryptoanalýzy. Dokud je počet rund nízký, může sice LK vykazovat jistou míru úspěšnosti — v případě pouhých dvou rund značnou, v případě tří rund už méně výraznou —, ale úspěšnost se s rostoucím počtem rund stále zhoršuje a v případě čtyřrundové verze už lineární kryptoanalýza dává jen dvoubitovou výhodu proti pouhému náhodnému tipování klíčů, ovšem za cenu značně náročných výpočtů.

Hlavním limitem, který se bude u delších variant Baby Rijndael ještě výrazněji uplatňovat, ovšem je velikost bloku a v důsledku toho počet vzorků otevřeného a šifrového textu: Zlepšení pozice správného klíče po umělém navýšení vzorků ve čtyřrundové verzi ukazují, že úspěšnosti lineární kryptoanalýzy by prospělo, kdybychom mohli použít více než 65536 vzorků. Je ovšem obtížné najít spolehlivý způsob, jak to udělat, a ještě obtížnější by pak bylo tento způsob uplatnit i v praxi.[27]

5.7 Aplikace na Rijndael

V kapitole 3 jsem ukázal, že Baby Rijndael je velmi přesná varianta plného Rijndael.[28]. Tak ostatně byla tato šifra i navrhována, s úmyslem, aby šlo závěry získané z Baby Rijndael uplatnit i na plný Rijndael, u kterého není výpočetně zvládnutelné je prakticky ověřit. Co tedy můžeme odvodit z výsledků lineární kryptoanalýzy?

- Rijndael lze převést do podoby SPN vhodné pro lineární kryptoanalýzu stejnými technikami, jaké jsem použil pro Baby Rijndael. Postup ovšem bude náročnější vzhledem k tomu, že stav má nejméně 4×4 prvky a ne jen 2×2 prvky; to bude mít vliv zejména na složitost reprezentace `ShiftRows`.

- Obdobně bude složitější analýza S-boxu. Samotný `SubBytes` je výpočetně zvládnutelný poměrně dobře, protože má v Rijndael stejnou velikost 256×256 jako v Baby Rijndael `SubBytes+MixColumns`, u kterého jsem analýzu udělal. Otázkou ovšem je schopnost analy-

[27] Jak ukazuji v šesté sekci, momentálně to dokážu pouze za situace, kdy si mohu klíč sám volit.

[28] S možnou výjimkou pro diferenciální kryptoanalýzu a Square attack, protože zatímco Rijndael volí `ShiftRows` tak, aby těmto útokům co nejlépe odolával, v Baby Rijndael je volba `ShiftRows` vynucena malými rozměry stavu šifry a odolnost vůči těmto útokům tak nemohla být zohledněna.

zovat `MixColumns` nebo `SubBytes+MixColumns`, jejichž aproximační tabulka u Rijndael nabývá rozměru 65536×65536.

- Otázkou je, co analýza S-boxu ukáže v případě, že ji dokážeme provést. S-box Baby Rijndael vykazoval poměrně výrazné linearity (bias $\frac{1}{4}$), to jsou ovšem, vzhledem ke konstrukci `SubBytes` pomocí modulární inverze, linearity umělé, vzniklé v důsledku příliš malého prostoru tělesa $GF(2^4)$. Můžeme očekávat, že v Rijndael sice také obdobné umělé linearity vniknou, jejich bias však skoro jistě bude mnohem menší a v důsledku toho bude i bias lineárních aproximací celé šifry menší.

- Na to bude mít zcela jistě vliv i to, že větší stav Rijndael znamená v každé rundě mnohem více aktivních S-boxů; každý aktivní S-box s biasem jiným než $\pm\frac{1}{2}$ pak v důsledku piling-up lemma (str. 31) snižuje celkový bias aproximace šifry.

- Další negativní vliv na bias celkové aproximace má počet rund, který je u Rijndael 10 oproti čtyřem u Baby Rijndael.

Jeden každý z těchto bodů vede k tomu, že lineární kryptoanalýza má u Rijndael mnohem horší výchozí podmínky než u Baby Rijndael. S ohledem na to, že ani u Baby Rijndael s optimální aproximací se mi nepodařilo dosáhnout výrazného úspěchu v hledání klíče, lze očekávat, že úspěšnost u plného Rijndael bude ještě nižší. Jedinou naději pro LK spatřuji v zvláštní závislosti úspěšnosti hledání klíče na hlavním klíči šifry; pokud by v mechanismu expanze klíče byl problém, který tuto závislost vytváří i u Rijndael, mohl by větší stav vést k tomu, že se tato závislost projeví i ve větším počtu rund než jsou jen tři. Dokud ovšem tuto závislost u Rijndael neprokážeme, lze z výše uvedeného soudit na to, že Rijndael není lineární kryptoanalýzou ohrožen.

Závěr

Cílem této diplomové práce bylo prověřit odolnost šifry Rijndael (AES) vůči technikám lineární kryptoanalýzy, a to na zjednodušeném modelu šifry Baby Rijndael. Nutnou podmínkou pro to bylo dokázat, že Baby Rijndael skutečně je vhodným modelem pro tento typ analýz a že závěry učiněné pro Baby Rijndael bude možné aplikovat i na Rijndael.

Toto se podařilo v plné míře splnit. V druhé a třetí kapitole jsem porovnal celkovou strukturu i jednotlivé komponenty obou šifer a ukázal, že Baby Rijndael splňuje všechny požadavky, principy a designová rozhodnutí, kterými byl motivován návrh šifry Rijndael, s možnou výjimkou odolnosti operace `ShiftRows` vůči diferenciální analýze a Square útoku (viz str. 18). To je už sám o sobě nesmírně důležitý výsledek, protože *umožňuje analyzovat odolnost Rijndael pomocí redukované šifry Baby Rijndael a tím prakticky ověřovat teoretické útoky, které by kvůli výpočetní náročnosti na plném Rijndael otestovat nešly.*

V další kapitole jsem stručně popsal principy lineární kryptoanalýzy a v páté kapitole je aplikoval na šifru Baby Rijndael. Navrhl jsem několik postupů, jak Baby Rijndael převést na strukturu substituční a permutační sítě, kterou používá lineární kryptoanalýza, a analyzoval S-boxy šifry. Dále jsem sestavil lineární aproximaci šifry pro dvě, tři a čtyři rundy a navrhl algoritmus, který umožňuje *sestavit lineární aproximaci s optimálním biasem pro libovolný počet rund šifry*; výpočtem všech možných lineárních aproximací jsem hrubou silou ověřil, že tento algoritmus funguje a má popsané vlastnosti (viz str. 50). Vzhledem k podobnosti Rijndael a Baby Rijndael lze očekávat, že tento algoritmus půjde obdobně použít i pro Rijndael.

Sestavil jsem několik lineárních aproximací šifry a pokusil se pomocí nich nalézt poslední rundovní klíč, a to postupně pro všechny možné klíče. Výsledky se výrazně lišily podle počtu rund šifry: Zatímco dvourundovou

verzi jsem pomocí lineární kryptoanalýzy plně polomil a našel pro ni správný klíč, ve třírundové verzi se správný klíč podařilo nalézt jen v 36,6 procentech případů a ve čtyřrundové verzi v 4,4 procentech případů. Pro čtyřrundovou verzi se mi navíc nepodařilo najít žádný postup, jak bez apriorní znalosti klíče tento klíč nalézt. To je velmi užitečné pro Rijndael jako takový: Pokud ani na mnohem jednodušší verzi šifry s řadou výhod pro kryptoanalýzu nejsme schopni nalézt správný klíč, lze očekávat, že plný Rijndael bude vůči lineární kryptoanalýze odolávat ještě lépe a tedy jí nebude napadnutelný.

Nalezl jsem ale dvě skutečnosti, které mohou mít význam pro další kryptoanalýzu:

1. Třírundová verze šifry vykazuje vysokou závislost mezi hodnotou *hlavního klíče* šifry a schopností lineární kryptoanalýzy nalézt správnou hodnotu *posledního rundovního klíče* šifry. To signalizuje možné slabé místo v mechanismu expanze klíče (str. 63).

2. Ve čtyřrundové verzi šifry se podařilo experimentálně odhalit možný vztah mezi počtem aktivních bitů lineární aproximace a průměrné schopností lineární kryptoanalýze pomocí této aproximace nalézt správný klíč. To umožňuje zlepšit výběr lineární aproximace přidáním dalšího optimalizačního kriteria ke standardnímu kriteriu biasu (str. 64).

Pro budoucí výzkum doporučuji zejména následující oblasti:

- Prověřit důvody vedoucí k závislosti mezi hlavním klíčem šifry a schopností lineární kryptoanalýzy nalézt klíč v třírundové verzi Baby Rijndael. Předběžně očekávám, že pokud v tomto ohledu v šifře existuje nějaké slabé místo, nalezneme ho nejspíše v mechanismu expanze klíče.

- Vyzkoušet na výsledky lineární kryptoanalýzy čtyřrundové verze Baby Rijndael pokročilejší techniky hledání správného klíče, zejména možnost rozdělení klíče na jednotlivé bity a jejich analýzu.

- Využít modelu v podobě Baby Rijndael k prověření možností realizace dalších kryptoanalytických útoků, zejména těch založených na algebraických vlastnostech šifry ([10], [3]).

Literatura

[1] Bergman, C.: *A Description of Baby Rijndael*. Iowa State University, 2005, [cit. 2013-04-17]. Dostupné z: http://www.math.iastate.edu/cbergman/crypto/homework/babyr/babyr.pdf

[2] Daemen, J.; Rijmen, V.: *AES Proposal: Rijndael*. 1999, [cit. 2013-04-17]. Dostupné z: http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf

[3] Ferguson, N.; Schroeppel, R.; Whiting, D.: *A simple algebraic representation of Rijndael*. 2001, [cit. 2013-04-17]. Dostupné z: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.9.6635&rep=rep1&type=pdf

[4] Heys, H. M.: A Tutorial on Linear and Differential Cryptanalysis. *Technical Report CORR*, , č. 17, 2001, [cit. 2013-04-20]. Dostupné z: http://www.engr.mun.ca/~howard/PAPERS/ldc_tutorial.pdf

[5] Kalvoda, T.; Petr, I.: *Matematika pro kryptologii: Okruh 1*. ČVUT, 2013, [cit. 2013-04-17]. Dostupné z: https://edux.fit.cvut.cz/courses/MI-MKY/

[6] Lohrmann, B.: *Linear and differential cryptanalysis in the context of AES*. University of Calgary, 2006, [cit. 2013-04-17]. Dostupné z: http://www.blohrmann.net/tub/aescryptanalysis.pdf

[7] Lórencz, R.: *Pokročilá kryptologie: DES a AES*. ČVUT, 2011, [cit. 2013-04-17]. Dostupné z: https://edux.fit.cvut.cz/courses/MI-KRY/

[8] Lórencz, R.: *Pokročilá kryptologie: Lineární kryptoanalýza*. ČVUT, 2011, [cit. 2013-04-20]. Dostupné z: https://edux.fit.cvut.cz/courses/MI-KRY/

[9] Matsui, M.: Linear Cryptanalysis Method for DES Cipher. *Lecture Notes in Computer Science*, ročník 1994, č. 765: s. 386–397.

[10] Nover, H.: *Algebraic Cryptanalysis of AES: An Overview*. University of Wisconsin, 2004, [cit. 2013-04-17]. Dostupné z: http://www.math.wisc.edu/~boston/nover.pdf

[11] Selcuk, A. A.: On Bias Estimation in Linear Cryptanalysis. *Lecture Notes in Computer Science*, ročník 2000, č. 1977: s. 52–66, [cit. 2013-04-17]. Dostupné z: http://www.cs.bilkent.edu.tr/~selcuk/publications/Bias_Indo00.pdf

[12] Wikipedia Foundation: *Advanced Encryption Standard process*. [cit. 2013-04-17]. Dostupné z: http://en.wikipedia.org/wiki/Advanced_Encryption_Standard_process

[13] Wikipedia Foundation: *Data Encryption Standard*. [cit. 2013-04-17]. Dostupné z: http://en.wikipedia.org/wiki/Data_Encryption_Standard

[14] Wrolstad, J.: *A differential cryptanalysis of Baby Rijndael*. Iowa State University, 2009, [cit. 2013-04-17]. Dostupné z: http://www.math.iastate.edu/thesisarchive/MST/WrolstadMSTS09.pdf

Příloha A

Seznam použitých zkratek

AES Advanced Encryption Standard

CSV Comma-Separated Values

DES Data Encryption Standard

LK Lineární kryptoanalýza

SPN Substitution-Permutation Network

`SubBytes+MixColumns` Operace, která vyjadřuje spojení `SubBytes` a `MixColumns`.

Příloha B

Obsah přiloženého CD

- `Aplikace` ... Aplikace vytvořené v rámci této práce, včetně zdrojových kódů v Pascalu. Aplikace jsou určeny pro prostředí Delphi, lze je však přeložit i pomocí nástroje FreePascal
  - `BabyRijndael` ... Implementace šifry Baby Rijndael
  - `BranchNumber` ... Pomocný program pro analýzu difúzní síly `MixColumns`
  - `LinearCryptanalysis` ... Aplikace pro provádění jednotlivých kroků lineární kryptoanalýzy Baby Rijndael. Stručný popis syntaxe se zobrazí po spuštění aplikace bez parametrů
- `DiplPrace`
  - `Prace.pdf` ... Text této práce
  - `Zdroj` ... Zdrojová forma práce ve formátu LaTeX
- `Materialy` . Kopie veřejně dostupných materiálů použitých při přípravě této práce, viz použitá literatura
- `Vysledky`
  - `ActualBias.*` . Shrnutí výsledků lineární kryptoanalýzy pro 2, 3 a 4 rundy včetně variant
  - `2rundy` ... Výsledky LK pro 2 rundy Baby Rijndael
    - `AllKeys.*` ... Zdroj dat pro tabulku 5.2
  - `3rundy` ... Výsledky LK pro 3 rundy Baby Rijndael
    - `AllKeys-A.*` Zdroj dat pro tabulku 5.3 — varianta se sloučeným `SubBytes` a `MixColumns`
    - `AllKeys-A-0xx0.*` LK pro 3 rundy, seřazená podle 2. a 3. číslice hlavního klíče šifry
    - `AllKeys-B.*` . Zdroj dat pro tabulku 5.3 — varianta vyjádřením prvního bitu z `MixColumns`
  - `4rundy` ... Výsledky LK pro 4 rundy Baby Rijndael
    - `AllKeys-A.*` ... Zdroj dat pro tabulku 5.4, aproximace A
    - `AllKeys-A-131072.*` ... Zdroj dat pro tabulku 5.5
    - `AllKeys-B.*` ... Zdroj dat pro tabulku 5.4, aproximace B
  - `S-Box`
    - `SubBytes.xlsx` ... Analýza `SubBytes`
    - `SubBytesMixColumns.*` ... Analýza `SubBytes+MixColumns`
    - `SubBytesMixColumnsHighApproximations.txt` ... Seznam aproximací s biasem $\pm\frac{1}{4}$ a $\pm\frac{1}{8}$ seřazený podle vstupů
    - `SubBytesMixColumnsHighApproximations.txt` ... Seznam aproximací s biasem $\pm\frac{1}{4}$ a $\pm\frac{1}{8}$ seřazený podle výstupů