ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ZADÁNÍ DIPLOMOVÉ PRÁCE

Název: Redukované modely šifry Rijndael
Student: Bc. Lukáš Solil
Vedoucí: Ing. Josef Kokeš
Studijní program: Informatika
Studijní obor: Počítačová bezpečnost
Katedra: Katedra počítačových systémů
Platnost zadání: Do konce letního semestru 2016/17

Pokyny pro vypracování

Seznamte se s šifrou Rijndael a s jejími existujícími redukovanými modely.
Formulujte vlastnosti šifry klíčové pro její bezpečnost a její realizaci.
Navrhněte systém pro podporu tvorby dalších redukovaných modelů.
Implementujte základní verzi takového systému.
Demonstrujte využití těchto modelů pro kryptoanalýzu.
Odhadněte vztah dosažených výsledků k AES/Rijndael.

Seznam odborné literatury

* Bergman, C.: A Description of Baby Rijndael. Iowa State University, 2005.
* Daemen, J., Rijmen, V.: The Design of Rijndael. Berlin: Springer-Verlag, 2002. ISBN 978-3-540-42580-9.
* Kokeš, J.: Kryptoanalýza šifry Baby RIjndael. ČVUT FIT, 2013.

L.S.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 11. prosince 2015

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra ... Počítačová bezpečnost

Diplomová práce

Redukované modely šifry Rijndael

Bc. Lukáš Solil

Vedoucí práce: Ing. Josef Kokeš

27. dubna 2016

Poděkování

Rád bych poděkoval vedoucímu této diplomové práce, panu Ing. Josefu Kokešovi, za cenné rady a připomínky, které mi při jejím zpracovávání poskytoval, za velmi rychlé odpovědi na moje otázky pokládané v nejrůznějších denních i nočních hodinách a za všechen čas, který mi byl ochoten věnovat při odborných konzultacích.

Dále bych chtěl poděkovat své snoubence, slečně Bc. Dominice Kobrlové, za pomoc s anglickým textem, který je součástí aplikace Průvodce pro vytváření modelů.

Na závěr bych rád poděkoval svému kolegovi, panu Mgr. Pavlu Vondruškovi, za odborné konzultace na poli matematiky a kryptologie.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) 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. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 27. dubna 2016 .....

Odkaz na tuto práci

Solil, Lukáš. *Redukované modely šifry Rijndael.* Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Práce se zabývá redukcí velikosti šifry Rijndael. Zkoumá vlastnosti šifry a kritéria, podle kterých byla navržena. Získané poznatky využívá k formulaci postupu pro návrh redukovaných modelů Rijndaelu. Součástí práce jsou nástroje, které uživateli umožní zmíněný návrh modelu provést a následně využít jeho implementaci. Využitelnost modelů pak demonstruje na kryptoanalýze opakovaným šifrováním.

Klíčová slova AES, Rijndael, Baby Rijndael, model Rijndaelu, redukce, šifra, kryptoanalýza opakovaným šifrováním

Abstract

The thesis deals with the reduction of the cypher Rijndael. It examines the characteristics of the cypher and its design criteria. The obtained knowledge is used to create a Rijndael reduced model design process. The thesis provides tools enabling the user to follow the steps of the process and after that, to use the new model implementation. The practical use of the models is demonstrated on repetitive encryption cryptanalysis.

Keywords AES, Rijndael, Baby Rijndael, Rijndael model, reduction, cypher, repetitive encryption cryptanalysis

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Šifra Rijndael je, zejména ve své implementaci dle standardu AES, jednou z nejpoužívanějších a nejzkoumanějších šifer. Její kryptoanalýzou se zabývají odborníci z celého světa, přesto se v ní zatím nepodařilo objevit zásadní bezpečnostní slabinu. Jedním z velkých problémů při jejím zkoumání je velikost prostoru možných klíčů a stavů šifry, kvůli kterému probíhají experimenty dlouhou dobu nebo vůbec nejsou v rozumném čase proveditelné.

Kokeš ve své práci [1] ukázal, že tento problém lze za určitých podmínek řešit pomocí redukované verze šifry nazvané Baby Rijndael [2]. Tu lze využít ke studiu struktury šifry a k experimentálnímu ověřování dopadů útoků, které jsou potenciálně rychlejší než hrubá síla. Ačkoli je Baby Rijndael dobrým modelem šifry, může být velmi náročné ověřit, zda lze výsledky provedených experimentů přenést na standardní verzi Rijndaelu.

Tato práce přináší možnost vytvoření dalších redukovaných modelů šifry Rijndael. S jejich pomocí může kryptoanalytik rychle ověřit výsledky svých pokusů, může škálovat velikost šifry, na které pokusy provádí, a za pomoci statistiky může odhadnout dopady vybraného útoku na standardní verzi Rijndaelu. Také má možnost vytvářet modely, které na rozdíl od Baby Rijndaelu pracují s větším klíčem než je délka jednoho bloku šifry.

Cíl práce

Cílem práce je umožnit výzkumníkovi na poli kryptoanalýzy Rijndaelu vytvářet redukované modely, které svou strukturou i vlastnostmi co nejlépe odpovídají verzi šifry definované ve standardu AES.

Za tímto účelem by měl vzniknout nástroj, který kryptoanalytikovi pomůže vhodný model sestavit. Měl by zjednodušit návrh redukovaných modelů natolik, aby se výzkumník nebyl nucen zabývat detailní analýzou těch částí šifry, které se přímo netýkají jeho práce. Dále by měla být vytvořena parametrizovatelná implementace, jež umožní navržené modely rovnou realizovat.

Dalším cílem této práce je demonstrovat využitelnost redukovaných modelů Rijndaelu na praktické ukázce kryptoanalýzy.

Kapitola 1

Rijndael

Tato kapitola připomíná strukturu šifry Rijndael a uvádí kritéria návrhu jednotlivých operací tak, jak jsou popsána v publikaci [3]. Zmíněná kritéria jsou pro tuto práci zásadní, neboť jsou vodítkem při rozhodování o omezeních redukovaných modelů. V některých případech je v práci využita také motivace autorů z textu [4], ta je ale citována až ve chvíli, kdy je to třeba a není součástí shrnutí v této kapitole.

Pro účely této práce je kniha [3] hlavním zdrojem informací o šifře a pokud se v textu vyskytne označení „specifikace Rijndaelu“, je tím myšlena právě tato publikace.

Technické a vědecké práce by se podle autorů Rijndaelu [4, str. 2] neměly odkazovat na dokument [4]. V případě této práce je text využíván jako zdroj informací o motivacích autorů, protože některé z nich nejsou v ostatních publikacích zmíněny.

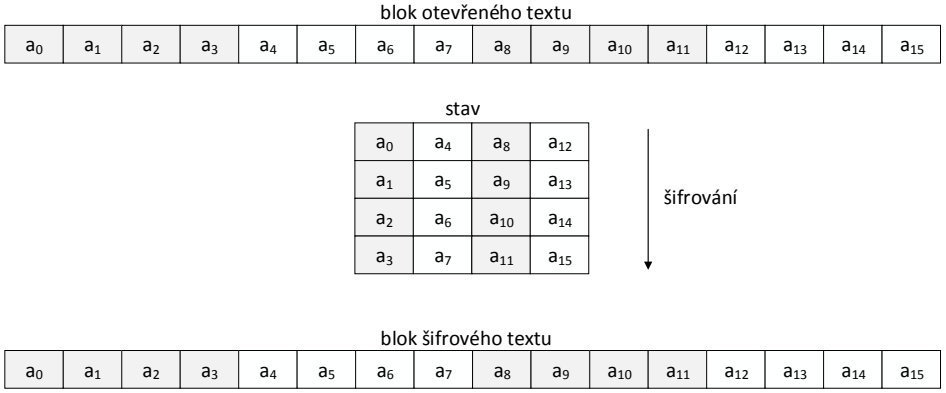
1.1 AES

Specifikace Rijndaelu umožňuje vytváření různě velkých instancí tím, že se změní velikost bloku a vstupního klíče. Jediným rozdílem mezi obecným Rijndaelem a standardem AES (viz [5]) je velikost množiny přípustných hodnot těchto parametrů.

Zatímco Rijndael připouští specifikovat nezávisle na sobě velikost bloku i klíče jako násobek 32 bitů s minimální hodnotou 128 bitů a maximální 256 bitů, AES podporuje pouze bloky délky 128 bitů a klíče velikostí 128, 192 nebo 256 bitů.

1.2 Struktura Rijndaelu

Rijndael je symetrická bloková iterativní šifra. Vstupní blok otevřeného textu (dále jen OT) je uložen do matice $n \times m$ nazvané stav šifry (dále jen stav).



Obrázek 1.1: Transformace OT na stav a zpět na ŠT

Šifrování i dešifrování se skládá ze série operací (tzv. rundy), které jsou opakovaně aplikovány na aktuální stav, což způsobuje změnu hodnot v jednotlivých buňkách matice stavu. Výstup ze šifry, tedy šifrový text (dále jen ŠT), vznikne jednoduchou transformací stavu po poslední rundě šifry. Způsob naplnění stavu pomocí OT a získání ŠT je znázorněn na obrázku 1.1

Před začátkem šifrování nebo dešifrování je nejprve vstupní klíč rozšířen expanzí (viz 1.6) na $N_r + 1$ rundovních klíčů, kde N_r je počet rund šifry (viz 1.5). Tyto rundovní klíče jsou poté používány při operaci `AddRoundKey` (viz 1.4.4).

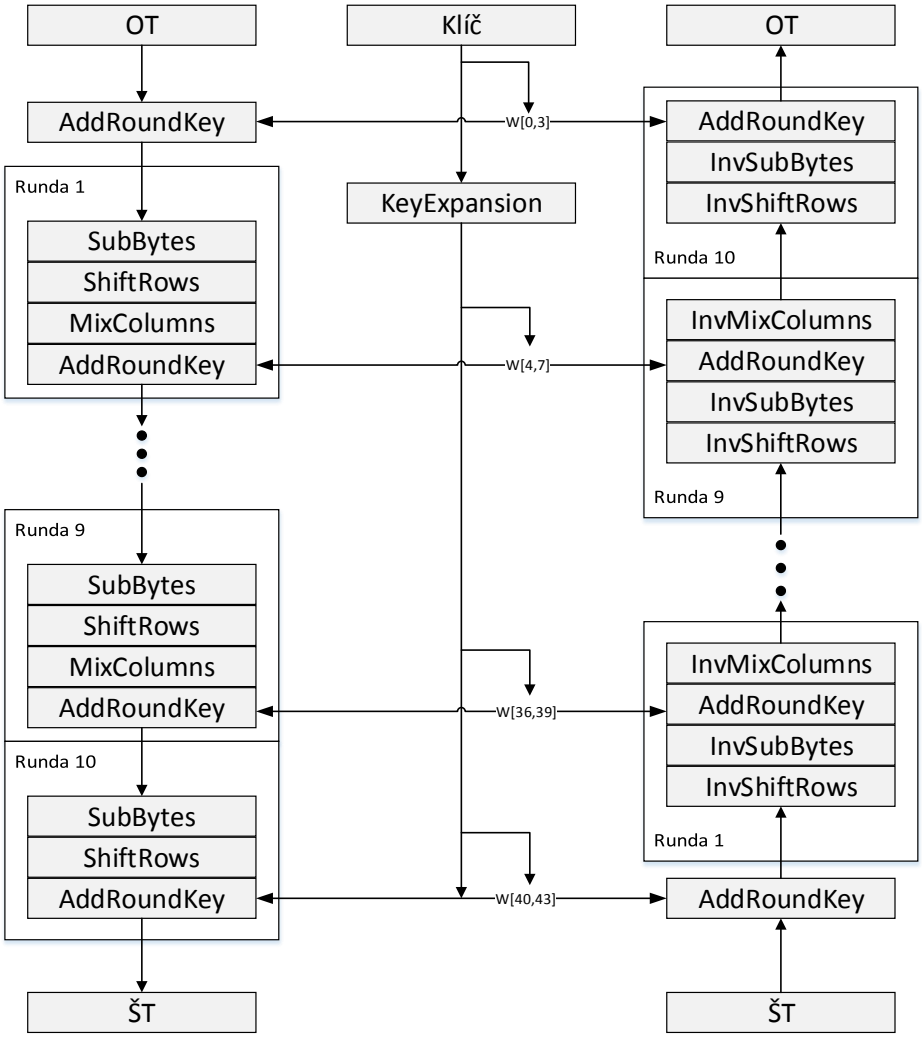
Jednotlivé rundy se skládají z operací `SubBytes`, `ShiftRows`, `MixColumns` a `AddRoundKey` (viz sekce 1.4). Pravidla jejich použití při šifrování a dešifrování jsou patrná z obrázku 1.2. Poslední runda při šifrování a první runda při dešifrování neobsahuje operaci `MixColumns`, ostatní rundy probíhají pokaždé se stejnými operacemi ve stejném pořadí.

Redukované modely, které tato práce umožňuje vytvářet, mají být co nejpodobnější standardní verzi šifry. Proto jejich struktura přesně odpovídá specifikaci Rijndaelu.

1.3 Reprezentace hodnot

Jak již bylo řečeno, šifra pracuje s maticí nazvanou stav. Velikost buňky stavu je jeden byte, tedy osm bitů. Pro operace šifry představuje každý bit jeden koeficient polynomu z tělesa polynomů GF(2^8), ve kterém se počítá modulo zvolený ireducibilní polynom označovaný $m(x)$. Autoři šifry zvolili pro Rijndael konkrétně polynom $x^8 + x^4 + x^3 + x + 1$.

Podle [4, str. 27], nebyl k volbě tohoto polynomu žádný zvláštní důvod. Byl vybrán jako první ireducibilní polynom 8. stupně podle použité literatury [7, str. 378].



Obrázek 1.2: Struktura šifry Rijndael (překresleno z [6, str. 28]).

Pro účely této práce, stejně jako tomu bývá u většiny literatury zabývající se Rijndaelem, jsou polynomy někdy označovány binárním řetězcem, kde každý bit představuje jeden koeficient polynomu, nejvýznamnější bit odpovídá koeficientu nejvyšší mocniny a nejméně významný bit označuje poslední člen polynomu. Ekvivalentně je občas také používán hexadecimální zápis tohoto binárního řetězce. Polynomy jsou takto označovány zejména v aplikacích, které v rámci této práce vznikly.

1.4 Rundovní operace

Na rundovní operace jsou podle [3, str. 34] kladeny následující podmínky:

1. Invertabilita. Každá z operací musí mít inverzi.
2. Jednoduchost. Jsou preferovány jednodušší komponenty před komplexními.

Splnění prvního bodu je důležitým kritériem při sestavování redukovaných modelů Rijndaelu. Druhá podmínka je zaručena výběrem specifikovaných operací.

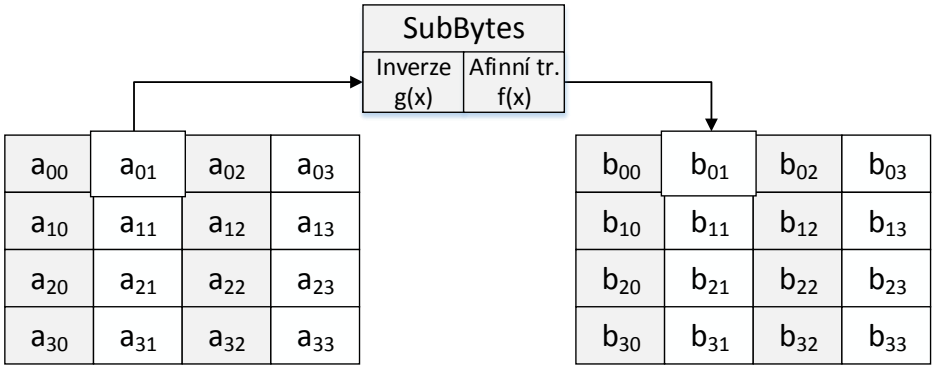
1.4.1 SubBytes

Operace `SubBytes` je nelineární komponenta šifry, která každé buňce stavu přiřadí novou hodnotu na základě hodnoty původní. Využívá složení dvou transformací označovaných $g(x)$ a $f(x)$. První zmíněná transformace je nelineární. Druhá transformace je afinní a slouží k zesložitění algebraického vyjádření operace `SubBytes`. Výběr těchto dvou transformací musí splňovat kritéria uvedená v [3, str. 35,36]:

1. Nelinearita
   a) Korelace. Maximální korelace mezi vstupem a výstupem musí být co nejmenší.
   b) Pravděpodobnost propagace rozdílů. Maximální pravděpodobnost propagace rozdílů musí být co nejmenší.
2. Složitost algebraického vyjádření. Algebraické vyjádření operace `SubBytes` musí být co nejsložitější.

První podmínku splňuje operace inverze ve zvoleném tělese $\mathrm{GF}(2^8)$. Druhou podmínku naplňuje mnoho afinních transformací, proto byla přidána ještě dvě omezení:

3. Nulové body. Operace `SubBytes` musí vždy mít na výstupu jiný polynom, než jaký dostane na vstup.



Obrázek 1.3: Operace `SubBytes` (překresleno z [5, str. 16] a upraveno).

4. Opačné body. Operace `SubBytes` nesmí mít na výstupu inverzní polynom k polynomu, který dostane na vstup.

Stále zůstává mnoho afinních transformací, z nichž byla vybrána následující:

$$\begin{pmatrix} b_7 \\ b_6 \\ b_5 \\ b_4 \\ b_3 \\ b_2 \\ b_1 \\ b_0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} \times \begin{pmatrix} a_7 \\ a_6 \\ a_5 \\ a_4 \\ a_3 \\ a_2 \\ a_1 \\ a_0 \end{pmatrix} \bigoplus \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} \tag{1.1}$$

Kde $a_7 \cdots a_0$ jsou bity vstupu a $b_7 \cdots b_0$ bity výstupu představující binární zápis koeficientů vstupního polynomu $a(x)$ respektive výstupního polynomu $b(x)$. Operace $\times$ představuje standardní maticové násobení v tělese GF(2) a operace $\bigoplus$ je sčítání v tělese polynomů $GF(2^8)$.

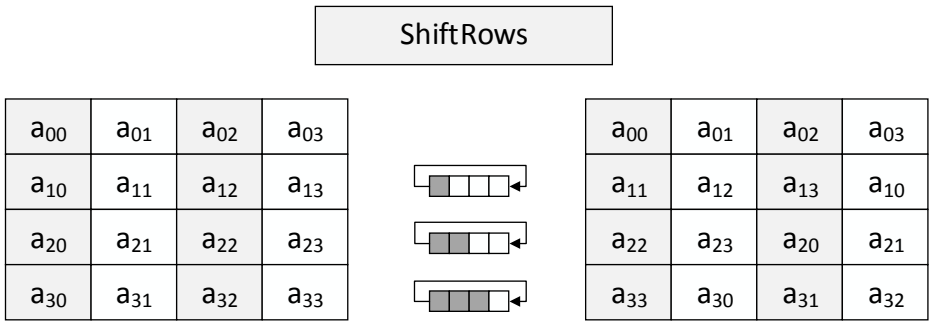
Diagram operace `SubBytes` je zachycen na obrázku 1.3.

1.4.2 ShiftRows

Operace `ShiftRows` je difuzní prvek šifry. Jde o provedení transpozice celých bytů cyklickým posunem řádků stavu. Byty v řádku i se posunou o C_i pozic vlevo.

Kritéria pro výběr vhodných konstant posunu jsou podle [3, 37] následující:

1. Optimální difuze. Konstanty musí být všechny navzájem různé.



Obrázek 1.4: Operace `ShiftRows` (překresleno z [5, str. 17] a upraveno).

2. Jiné dopady difuze. Musí být maximalizována odolnost proti útokům volných diferenciálů (truncated differencial attacks) a proti saturačním útokům (saturation attacks).

Při využití čtvercového stavu se 4 sloupci je zřejmě pro splnění první podmínky nutné využít posun o 0, 1, 2 a 3 pozice. V tomto případě není druhá podmínka relevantní. Při vyšším počtu sloupců stavu jsou C_i volena právě na základě odolnosti vůči zmíněným útokům.

Nezávisle na velikost stavu je operace ve specifikaci Rijndaelu zavedena tak, že platí

$$C_i < C_{i+1}; i \in \{0, 1, 2\}, \tag{1.2}$$

$$C_0 = 0. \tag{1.3}$$

Operace je graficky znázorněna na diagramu 1.4.

1.4.3 MixColumns

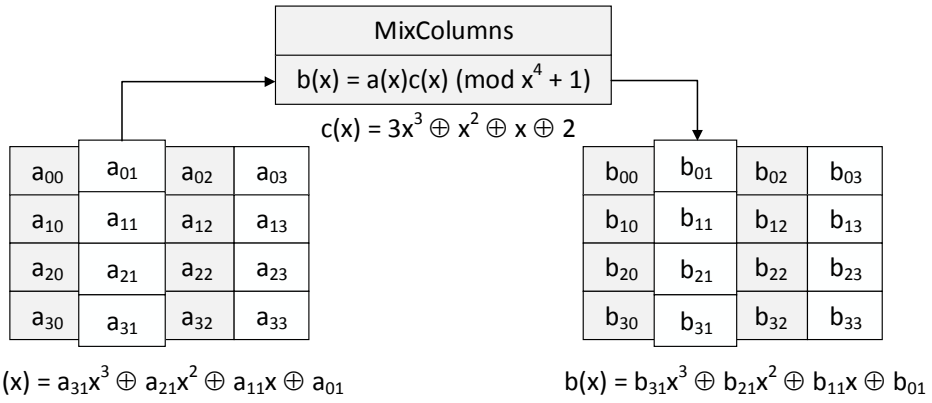
Operace `MixColumns` je další difuzní komponenta šifry. Jejím cílem je distribuce informace z jednoho do všech bytů sloupce. Kritéria na ni kladená, jsou v [3, str. 39] shrnuta do následujících bodů:

1. Dimenze. Operace pracuje se 4bytovými sloupci

2. Linearita. Operace by měla být lineární nad GF(2)

3. Difuze. Operace musí dosahovat horní meze hodnoty branch number (viz 2.5).

4. Výkon na 8bitových procesorech. Výkon 8bitových procesorů při provádění operace musí být vysoký.



Obrázek 1.5: Operace `MixColumns` (překresleno z [5, str. 18] a upraveno).

První a poslední požadavek jsou přidány z důvodu dobrého výkonu šifry a pro redukované modely mají pouze informativní význam. Důležité jsou body 2 a 3. Ty jsou zajištěny výběrem vhodné transformace `MixColumns` a jejích parametrů.

Autoři Rijndaelu zvolili operaci

$$b(x) = c(x) \cdot a(x) \mod x^4 + 1, \tag{1.4}$$

kde $a(x)$, $b(x)$ a $c(x)$ jsou polynomy nejvýše 3. stupně s koeficienty z vybraného tělesa GF(2^8), které v případě $a(x)$ a $b(x)$ reprezentují jednotlivé buňky sloupce stavu. Polynom $a(x)$ je vstupem operace a $b(x)$ jejím výstupem. Zbývající polynom je zvolen konstantně jako $c(x) = 03x^3 + 01x^2 + 01x + 02$. Konstantní polynom operace `MixColumns` je dále v této práci vždy označován $c(x)$.

Operace `MixColumns` je zachycena na obrázku 1.5.

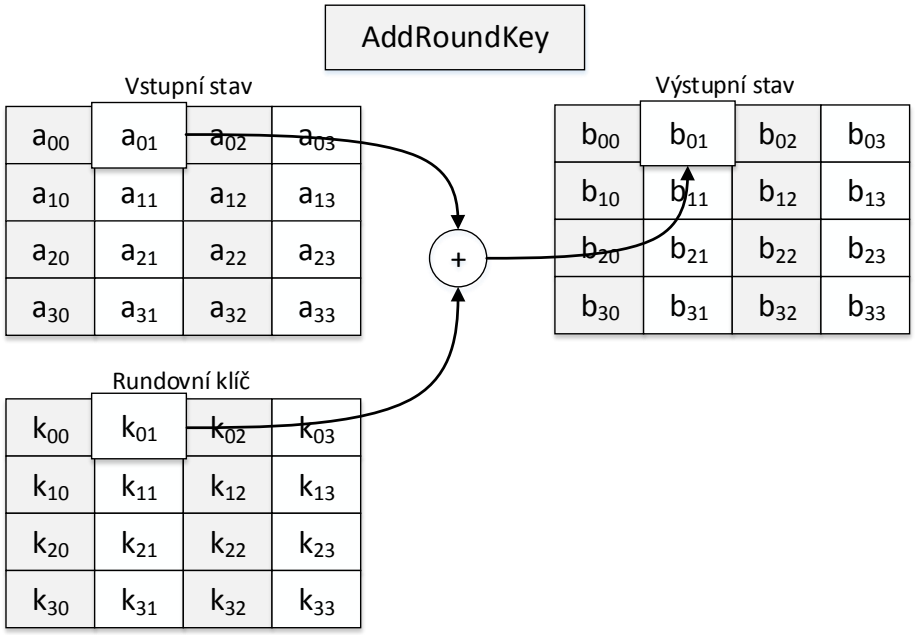
1.4.4 AddRoundKey

Operace `AddRoundKey` modifikuje stav tím, že k němu přičte rundovní klíč (viz 1.6). Sčítání probíhá v tělese GF(2) nad jednotlivými bity stavu a rundovního klíče. K přičítání klíče dochází v každé rundě jak při šifrování, tak při dešifrování. Klíč je přičítán navíc ještě před začátkem šifrování respektive na konci dešifrování.

Operaci `AddRoundKey` znázorňuje obrázek 1.6.

1.5 Počet rund

Počet rund byl zvolen pro Rijndael s velikostí bloku i klíče 128 bitů přidáním bezpečnostního rámce k nejvyššímu počtu rund napadnutelnému známými



Obrázek 1.6: Operace `AddRoundKey` (překresleno z [6, str. 27]).

útoky schopnými nalézt klíč v průměrně lepším čase než hrubá síla. Nižší odolnost proti útokům zjistili autoři nejvýše u 6 rund. Bezpečnostní rámec je zvolen na hodnotu 4 rund.

Pro větší velikost bloku nebo klíče bylo tabulkou (viz [3, str. 42]) stanoveno pravidlo na počet rund odpovídající vztahu

$$N_r := 6 + \max(N_b, N_k). \tag{1.5}$$

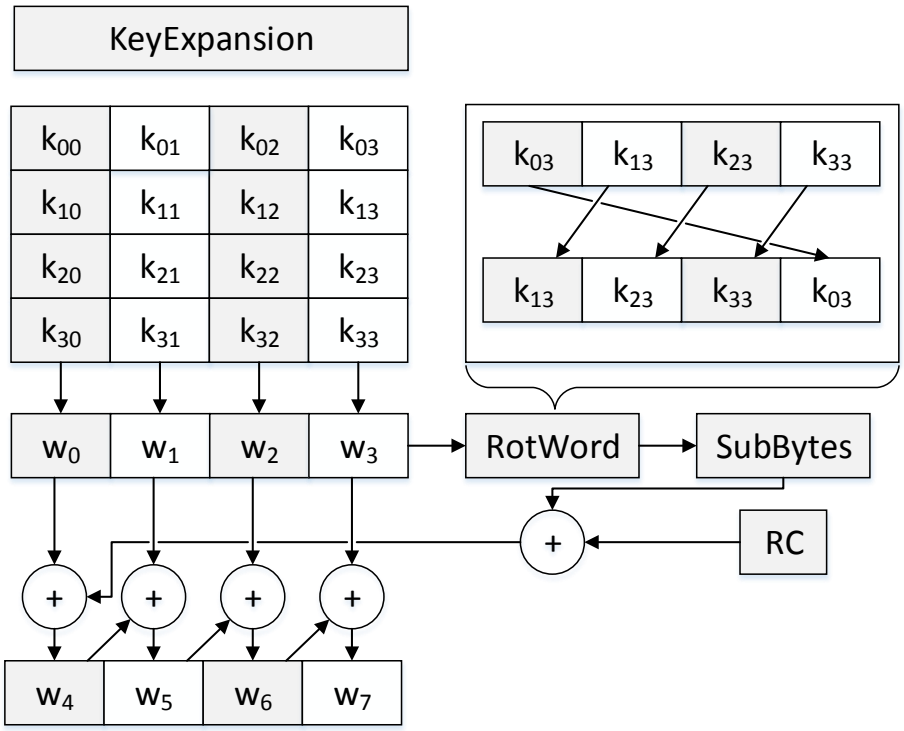
N_r označuje počet rund, N_b značí velikost bloku v bitech dělenou 32 a N_k je počet bitů klíče dělený 32. Toto značení je převzato ze specifikace Rijndaelu a je používáno i v dalších částech této práce.

1.6 Plánování klíče

Vstupní klíč je expandován na $N_r + 1$ rundovních klíčů o velikosti stavu. Jednotlivé rundovní klíče jsou následně využívané operací `AddRoundKey`.

Kritéria pro plánování klíče byla zvolena následovně:

1. Efektivita.
   a) Pracovní paměť. Operace by neměla mít vysoké nároky na paměť.
   b) Výkon. Operace by měla dosahovat dobrého výkonu na široké škále procesorů.



Obrázek 1.7: Operace `KeyExpansion` (překresleno z [6, str. 30]).

2. Eliminace symetrie. Díky konstantám odlišným pro každou rundu by měla operace eliminovat symetrii.

3. Difuze. Rozdíly ve vstupním klíči by se měly pomocí operace efektivně promítnout v expandovaném klíči.

4. Nelinearita. Operace by měla poskytnout dostatek nelinearity, aby nebylo možné odhadnout rozdíly v expandovaném klíči pouze na základě znalosti rozdílů ve vstupním klíči.

Autoři Rijndaelu navrhli algoritmus (viz [3, str. 45, 46]), který je použitelný i pro redukované modely, takže pro potřeby této práce není nutné rozebírat kritéria detailně. Je zde uveden alespoň krátký popis. Více detailů lze nalézt v [3]. Expanze klíče podle zmíněného algoritmu je zachycena na diagramu 1.7.

Efektivity je dosaženo tím, že se stále pracuje s celými byty. Rundovní klíč si lze představit jako matici bytů stejného formátu jako má stav. Při tvorbě sloupce i rundovního klíče jsou potřeba pouze sloupce $i-1$ a $i-N_k$. Tím pádem lze u zařízení s malou pracovní pamětí expandovat klíč průběžně a udržovat vždy jen $N_k + 1$ sloupců rundovních klíčů.

Rundovní konstanty, které ruší symetrii mezi jednotlivými rundami, jsou definovány jako mocniny polynomu x ve zvoleném tělese GF(2^8).

Difuze je dosahováno způsobem výběru jednotlivých sloupců rundovních klíčů a nelinearitu zajišťuje operace `SubBytes`.

1.7 Dešifrování

Dešifrování probíhá podobně jako šifrování, jen je při něm využito jiné pořadí operací (viz 1.2). Použité rundovní operace využívají inverzní polynomy, které jsou k dispozici díky tomu, že byly vybírány operace s inverzí.

Kapitola 2

Redukované modely

Rijndael specifikuje matematický koncept, na základě kterého je možné vytvářet blokové šifry s různou délkou vstupu a klíče. Pro standard AES ([5, str. 9]) byly vybrány konkrétní hodnoty těchto dvou parametrů. Jejich změnou lze vytvořit novou šifru, která sice nebude podle standardu AES, ale bude splňovat specifikaci Rijndaelu.

Mohlo by se zdát, že pro vytvoření redukovaného modelu stačí změnit výše zmíněné parametry. Problém ale spočívá v tom, že Rijndael stanovuje minima obou dvou na 128 bitů. Některé hodnoty jsou vybírány pomocí matematických operací s konstantami, které specifikace neumožňuje měnit.

Definice 1. Redukovaný model Rijndaelu je taková verze šifry Rijndael, která má velikost bloku menší nebo rovnu 128 bitům, velikost klíče menší nebo rovnu 256 bitům a z hlediska struktury, vlastností a bezpečnosti je podobná šifře Rijndael dle standardu AES.

Dále v této práci jsou v některých případech redukované modely Rijndaelu označovány pouze jako redukované modely či jen modely, pokud nehrozí zmatení čtenáře.

Aby mohly být vytvářeny redukované modely, je nutné provádět zásahy do pevně daných částí Rijndaelu. V této kapitole jsou identifikovány konstanty a parametry, jejichž změnou je možné snižovat velikost vytvořených modelů pod stanovené minimum 128 bitů bloku a klíče. U každé konstanty je rozebrán dopad její změny na jednotlivé operace šifry, zejména na jejich základní požadavky specifikované při návrhu Rijndaelu (viz 1). Shrnutí této kapitoly je zahrnuto v průvodci pro vytváření modelů (viz 3), jehož implementace je součástí této práce a nachází se na přiloženém CD.

Cílem této práce je umožnit navrhovat modely Rijndaelu, na kterých bude možné provádět experimenty na struktuře šifry bez nutnosti použití velkého výpočetního výkonu. Práce se zaměřuje na prakticky využitelné modely a stanovuje omezení na horní hranice jednotlivých parametrů tak, aby nepřekročily

hodnoty dané specifikací Rijndaelu. Pro zvětšování Rijndaelu není nutné zasahovat do pevných částí jeho specifikace.

2.1 Bundle

Základním elementem šifry Rijndael je byte. Každá z operací šifry pracuje s byty. Podle specifikace je v Rijndaelu používán standardní byte, který má velikost 8 bitů.

Pro vytváření redukovaných modelů je velikost bytu jedním z klíčových parametrů, který je možné změnit tak, aby došlo k zásadnímu snížení výpočetní náročnosti při práci s modelem. Například zkrácením bytu ve standardním AES-128 o 1 bit, dojde ke snížení počtu operací nutných pro vyzkoušení všech klíčů hrubou silou z 2^{128} na 2^{112}, tedy přibližně na $0,0015\%$.

2.1.1 Značení

Tato práce používá místo termínu byte, který je obvykle spojován s velikostí 8 bitů, označení bundle. Bundle může mít proměnlivou velikost v závislosti na nastavení daného redukovaného modelu Rijndaelu. Toto označení bylo převzato z [3, str. 72 – 126], kde se hovoří o struktuře Rijndaelu obecněji. Označení v této práci není přeloženo a je skloňováno stejně jako termín byte. Pro účely této práce je počet bitů v jednom bundlu označován N_{bundle}.

2.1.2 Dopady změny hodnoty N_{bundle}

2.1.2.1 Těleso

Rijndael považuje hodnotu bytu za reprezentaci polynomu v tělese $\mathrm{GF}(2^8)$ s ireducibilním polynomem stupně 8. Při změně délky bytu je nutné zvolit těleso s odpovídajícím počtem prvků, v němž budou operace probíhat. K tomu je třeba vybrat ireducibilní polynom stupně N_{bundle}. V tělese se změní pouze velikost množiny, operace zůstanou definovány shodně s původní specifikací Rijndaelu.

Jak již bylo zmíněno, ireducibilní polynom byl zvolen jako první ze seznamu uvedeném v použité literatuře. Pro redukované modely není podmínkou, aby byl vybrán stejným způsobem. Kvůli zjednodušení návrhu modelů je ale toto kritérium doporučené. Vybraný polynom je v této práci ve shodě s [3] označován $m(x)$.

Těleso polynomů $\mathrm{GF}(2^{N_{bundle}})$, kde se počítá modulo zvolený polynom $m(x)$, bude v dalších částech této práce značeno T.

2.1.2.2 Stav

Dle specifikace Rijndaelu si lze stav představit jako obdélníkové pole bytů se 4 řádky a N_b sloupci. Do stavu se zapíše celý jeden blok šifry. V závislosti

na tom, jak je tento blok veliký, je dán počet sloupců stavu, jež je pro jeho zapsání potřeba. Ten lze odvodit jako počet bitů bloku dělený počtem bitů v řádku. Je zřejmé, že velikost bundlu ovlivní počet sloupců, jelikož z ní plyne množství bitů, které se vyskytují v jednom řádku.

2.1.2.3 SubBytes

Kvůli změně tělesa, se kterým `SubBytes` pracuje, je nutné upravit transformace, pomocí nichž je konstruována tabulka S-boxu. Na základě kritérií, která jsou uvedena ve specifikaci Rijndaelu (viz 1.4), navrhuje tato práce následující pravidla pro výběr obou transformací.

Nelineární transformace $g : T \to T$ může být definována podobně jako ve specifikaci Rijndaelu:

$$g(a(x)) := \begin{cases} a(x)^{-1} \mod m(x), & \text{pokud } a(x) \neq 0 \\ 0, & \text{jinak} \end{cases} \tag{2.1}$$

Znakem 0 je v tomto případě označen neutrální prvek aditivní grupy tělesa T.

Nelineární transformace bude po změně velikosti bundlu stále splňovat podmínky nelinearity na ni kladené, protože se nezmění operace, pouze velikost množiny multiplikativní grupy. Operace inverze vykazuje dobré vlastnosti pro zvýšení odolnosti proti lineární a diferenciální kryptoanalýze, i přes to ale může při zmenšujícím se počtu prvků tělesa docházet ke vzniku linearit či k propagaci rozdílů hodnot. Kryptoanalýzy, které využívají tohoto oslabení, je možné testovat na malých modelech, ale výsledky je vhodné ověřit na větších verzích.

Afinní transformace $f : T \to T$ ze specifikace Rijndaelu, může být přepsána do polynomiálního tvaru, který lze využít v redukovaných modelech:

$$f(a(x)) := p(x) + a(x)q(x) \mod n(x) \tag{2.2}$$

Jednotlivé modely se liší výběrem polynomů $p(x)$, $q(x)$ a $n(x)$. Ty mají tvar

$$n(x) := x^{N_{bundle}} + 1, \tag{2.3}$$

$$p(x) := \sum_{i=0}^{N_{bundle}-1} \alpha_i x^i, \tag{2.4}$$

$$q(x) := \sum_{i=0}^{N_{bundle}-1} \beta_i x^i, \tag{2.5}$$

kde $\alpha, \beta \in \mathrm{GF}(2)$. Není sice nezbytně nutné, aby byl modul definován přesně tímto způsobem a může být použit i jiný polynom shodného stupně, ale protože je takto zaveden v Rijndaelu, připouští tato práce pouze redukované modely

s takto definovaným modulem afinní transformace. V souladu se specifikací Rijndaelu musí být polynomy $p(x)$ a $q(x)$ vybrány tak, aby platilo

$$gcd(p(x), n(x)) = 1, \tag{2.6}$$

$$\forall a(x) \in T : f \circ g(a(x)) \neq a(x), \tag{2.7}$$

$$\forall a(x) \in T : f \circ g(a(x)) \neq a(x)^{-1}. \tag{2.8}$$

Protože tento výběr ponechává stále velmi mnoho možností, jak afinní transformaci zvolit, navrhuje tato práce ještě další omezení, jejichž cílem je výběr zúžit na vhodné kandidáty. Omezení jsou založena především na motivacích autorů uvedených v [4, str. 26] a na charakteristikách polynomů vybraných pro standardní Rijndael. Koeficienty α a β by měly splňovat

$$\frac{\sum \alpha_i}{N_{bundle}} \approx \frac{1}{2}, \ \frac{\sum \beta_i}{N_{bundle}} \approx \frac{1}{2}, \tag{2.9}$$

$$\sum_{i=0}^{N_{bundle}-1} i\alpha_i \approx min, \quad \sum_{i=0}^{N_{bundle}-1} i\beta_i \approx min, \tag{2.10}$$

$$\sum_{i=\lceil \frac{N_{bundle}}{2} \rceil}^{N_{bundle}-1} \beta_i \approx \sum_{i=0}^{\lfloor \frac{N_{bundle}}{2} \rfloor} \beta_i. \tag{2.11}$$

Pro vytváření S-boxu ve standardním Rijndaelu byla podle [4, str. 26] zvolena afinní transformace

$$f(a(x)) := (x^7+x^6+x^2+x) \oplus a(x)(x^7+x^6+x^5+x^4+1) \mod x^8+1. \tag{2.12}$$

To ale neodpovídá transformaci uvedené ve specifikaci Rijndaelu (viz 1.4.1), takže se zdá, že v [4] je chyba a správné vyjádření je

$$b(x) = (x^6 + x^5 + x + 1) \oplus a(x)(x^4 + x^3 + x^2 + x + 1) \mod x^8 + 1, \tag{2.13}$$

což potvrzuje například i článek [8, str. 2292], který tento tvar afinní transformace uvádí.

Afinní transformace slouží k tomu, aby zvýšila odolnost proti interpolačním a jiným algebraickým útokům. Jejím cílem je, aby algebraické vyjádření S-boxu bylo co nejsložitější.

Některé práce, například [9],[10] nebo [8], ukazují, že výběr afinní transformace nebyl zvolen nejlepším možným způsobem, protože polynom, který vznikne při algebraickém vyjádření S-boxu, je sice vysokého stupně, ale obsahuje pouze 9 členů, což je poměrně málo a může to být využito ke kryptoanalýze. Výše zmíněné články se zabývají vylepšeními, která tento nedostatek odstraňují.

Kritérium pro komplexitu algebraického výrazu není úplně jednoznačné a dobře měřitelné. V Rijndaelu je navíc zvolena taková transformace, že jsou

u ní známé nedostatky. Z toho důvodu nelze stanovit způsob výběru afinní transformace pro redukované modely tak, aby výsledek co nejpřesněji odpovídal svými algebraickými vlastnostmi standardnímu Rijndaelu.

Autoři Rijndaelu bohužel neuvádějí detailnější kritéria volby aditivního polynomu, takže není možné provést analogický výběr. Podmínka 2.11 byla přidána proto, aby se zvýšila podobnost aditivního polynomu modelu se standardním Rijndaelem. Ani s touto podmínkou však nelze zajistit dostatečnou podobnost redukovaného modelu pro algebraickou kryptoanalýzu S-boxu.

Při používání modelů pro tento druh kryptoanalýzy je třeba nejprve ověřit, zda jsou vlastnosti využívané touto kryptoanalýzou dostatečně podobné standardnímu Rijndaelu. Pokud ne, může být třeba zvolit jiný model. V případě, že není dosažitelný matematický popis vlastností potřebných pro danou algebraickou kryptoanalýzu, je vhodné ji alespoň aplikovat na více různých modelů, čímž se sníží riziko, že je využívána slabina daného modelu.

2.1.2.4 ShiftRows

Operace `ShiftRows` pracuje s jednotlivými bundly jako s fixními elementy stavu, jejichž obsah nemění, pouze je přesouvá. Z toho důvodu nemá na ni změna velikosti bundlu zásadní vliv.

2.1.2.5 MixColumns

Operace `MixColumns` považuje sloupce stavu za polynomy s koeficienty z tělesa T, které násobí modulo $x^4 + 1$ s konstantním polynomem $c(x)$ (viz 1.4.3). Změna velikosti bytu má vliv na jednotlivé koeficienty polynomů. Samotnou operaci poznamená spíše změna počtu řádků stavu (pro potřeby této práce označován jako N_{rows}), která je popsaná v 2.2.

Operace může být pro potřeby redukovaných modelů definovaná podobně jako v Rijndaelu

$$b(x) = a(x) \cdot c(x) \mod x^{N_{rows}} + 1, \tag{2.14}$$

kde $a(x)$, $b(x)$ a $c(x)$ jsou polynomy stupně nejvýše $N_{rows} - 1$ s koeficienty z tělesa T, které v případě $a(x)$ a $b(x)$ reprezentují jednotlivé buňky sloupce stavu. Polynom $a(x)$ je vstupem operace a $b(x)$ jejím výstupem. Při vytváření redukovaného modelu se sníženou velikostí bytu je třeba vybrat takový konstantní polynom $c(x)$, jehož koeficienty

1. budou voleny tak, aby měl celý polynom inverzi modulo $x^{N_{rows}} + 1$,
2. budou voleny tak, aby hodnota branch number (viz 2.5) dosáhla svojí horní meze,
3. volitelně budou mít co nejnižší Hammingovu váhu.

První výše zmíněný bod zajistí, že operace bude invertibilní. Druhý bod je zásadní pro zajištění relevantní difuzní síly a je blíže popsán v oddělené sekci tohoto textu 2.5. Třetí bod je pouze volitelný, protože není cílem redukovaných modelů, aby dosahovaly vysoké rychlosti, ovšem v případě, že jsou splněny ostatní body, je vhodné volit koeficienty s nižší Hammingovou vahou, protože tak byly vybrány v Rijndaelu.

2.1.2.6 Expanze klíče

Změna velikosti bundlu nemá na expanzi zásadní vliv. Celý proces expanze samozřejmě probíhá v novém tělese, tak jako všechny operace v redukovaném modelu. Je třeba upravit rundovní konstanty (v Rijndaelu označované `RC`), jejich definice ale může zůstat ve tvaru

$$\mathtt{RC}_i := x^{i-1} \mod m(x), \tag{2.15}$$

kde $i \geq 1$ je číslo kroku expanze klíče, který rozšíří vytvářený klíč na $i + 1$ násobek velikosti vstupního klíče. V případě, že se rovnají velikosti vstupního klíče a bloku šifry, odpovídá i číslu rundovního klíče.

2.1.2.7 Výběr a přičtení klíče

Protože se výběr rundovního klíče z expandovaného klíče provádí na základě počtu elementů stavu, může jej změna velikosti bytu ovlivnit pouze nepřímo tím, že změní počet sloupců stavu (viz 2.1.2.2).

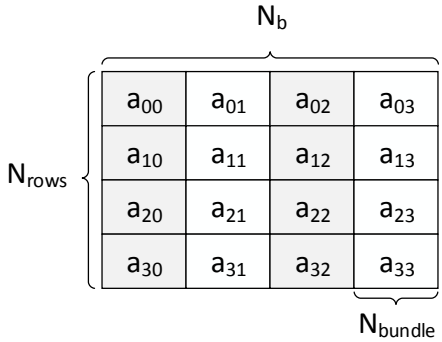
Přičtení vybraného klíče probíhá beze změny dle pravidel Rijndaelu, pouze operace sčítání pracuje s menší množinou prvků tělesa.

2.1.3 Omezení kladená na hodnoty N_{bundle}

Ačkoli by bylo teoreticky možné velikost bytu zvětšovat i nad hodnotu danou Rijndaelem, stanovuje tato práce horní mez hodnoty N_{bundle} na 8 bitů. Minimální hodnota velikosti bundlu nemůže být 1, protože by nemohly být splněny podmínky na vytvoření afinní transformace: nelze vytvořit transformaci, pro kterou bude platit 2.7 a 2.8. Dolní mezí velikosti bundlu je proto hodnota 2. Pro tu lze již vytvořit požadovanou afinní transformaci, například

$$f(a(x)) := x \oplus a(x)x \mod x^2 + 1. \tag{2.16}$$

Ostatní operace také mohou při $N_{bundle} = 2$ pracovat podle požadavků. Model s touto velikostí bundlu není vhodný pro experimenty, protože bude vykazovat velké množství umělé linearity a přenosů rozdílů (dané malým počtem bitů) i jednoduchou algebraickou strukturu. Může být ale vhodný pro grafické znázornění práce Rijndaelu.



Obrázek 2.1: Proměnné stavu

2.2 Velikost stavu

Počet řádků stavu (v této práci značen N_{rows}) je ve specifikaci Rijndaelu stanoven fixně na hodnotu 4. Jeho snížením lze vytvářet redukované modely s menším stavem a šifrovým klíčem, čímž se zjednoduší některé operace a sníží se počet bundlů, na které musí být aplikované. Obrázek 2.1 zachycuje proměnné, které lze při návrhu redukovaných modelů upravovat.

Množství sloupců stavu (značené dle specifikace Rijndaelu N_b) lze spočítat dle vzorce

$$N_b = \frac{N_{block}}{N_{rows} N_{bundle}}. \tag{2.17}$$

Označení N_{block} je v této práci používáno pro počet bitů bloku. V konfiguraci Rijndaelu, která byla zvolena pro standard AES, je vždy počet sloupců roven počtu řádků. Toto omezení sice není nutné aplikovat na redukované modely, protože specifikace Rijndaelu umožňuje vytvářet obdélníkové stavy, ale z hlediska praktického využití navrhovaných modelů je možné jej ponechat. Modely, které tato práce umožňuje vytvářet jsou omezené na čtvercové stavy s N_{rows}^2 bundly.

2.2.1 Blok šifry

Hodnota N_{rows} je jednou z hlavních příčin, proč je velikost bloku vždy násobkem 32 s minimem stanoveným na 128 bitů. Pokud je totiž stav reprezentován jako obdélníkové pole se 4 řádky, skládá se jeden jeho sloupec ze 4 bytů. Když má jeden byte 8 bitů, obsahuje jeden sloupec právě 32 bitů. Velikost bloku je tedy omezena tak, aby bylo možné vždy přidat celé sloupce pro zapsání bloku do stavu. Minimum stanovené na 128 bitů je dané tím, že počet sloupců stavu musí být minimálně stejný jako počet řádků, aby byly splněny podmínky na operaci `ShiftRows` (viz 2.2.3).

Z výše uvedeného vztahu pro počet sloupců stavu vzniká jednoduchou úpravou vzorec

$$N_{block} = N_b N_{rows} N_{bundle} \tag{2.18}$$

pro výpočet velikosti bloku z jednotlivých parametrů stavu. Vzhledem k omezení na čtvercové stavy, platí pro velikost bloku rovnost

$$N_{block} = N_{rows}^2 N_{bundle}. \tag{2.19}$$

2.2.2 SubBytes

Jelikož se operace `SubBytes` aplikuje na jednotlivé bundly stavu či šifrového klíče, má změna velikosti stavu dopad pouze na to, kolikrát je operace `SubBytes` spuštěna, ale ne na její samotný průběh.

2.2.3 ShiftRows

Operace `ShiftRows` posune každý řádek i cyklicky o C_i bundlů vlevo. Velikost stavu musí být volena tak, aby platilo

$$\forall i < N_{rows}, \forall j < N_{rows} : C_i = C_j \Rightarrow i = j, \tag{2.20}$$
$$\forall i < N_{rows}, \forall j < N_{rows} : C_i > C_j \Rightarrow i > j. \tag{2.21}$$

Podmínka 2.20 plyne přímo z požadavků specifikace (1). Bod 2.21 byl zaveden na základě charakteristiky operace `ShiftRows`, kdy je vždy dodrženo, že řádek s nižším číslem je posunut o méně bundlů než všechny následující řádky. Tato podmínka vznikla také na základě motivací popsaných v [4, str. 27], kde je přímo uvedena podmínka $C_0 = 0$ a dále je, jako u všech operací Rijndaelu, požadována maximální jednoduchost.

Vzhledem k podmínkám 2.20 a 2.21 a omezení na čtvercové stavy, je operace `ShiftRows` vždy dána jednoznačně pro všechny povolené velikosti stavu. Jak je uvedeno v [3, str. 37], podmínku 2 na odolnost vůči vybraným útokům není v tomto případě třeba ověřovat.

2.2.4 MixColumns

Snížení počtu řádků stavu vynucuje změnu konstantního polynomu $c(x)$ a modulu. V Rijndaelu byl pro tuto operaci zvolen modul $x^4 + 1$. Redukované modely, které tato práce dovoluje vytvořit, využívají polynom $x^{N_{rows}} + 1$. Bylo by sice možné vybrat i jiný polynom odpovídajícího stupně, ale, podobně jako při výběru modulu v afinní transformaci, je pro maximální podobnost se standardním Rijndaelem vhodné vybrat co nejpodobnější polynom.

Kritéria k výběru konstantního polynomu $c(x)$ jsou popsána v části 2.1.2.5. Velikost stavu zásadně ovlivní podmínky, které musí $c(x)$ splňovat, aby operace `MixColumns` dosáhla horní meze hodnoty branch number. Tato problematika je podrobněji popsána v sekci 2.5.

2.2.5 Šifrový klíč

Jediná podmínka spojená s velikostí stavu je, že šifrový klíč bude mít po expanzi právě $N_{rows}N_b(N_r + 1)$ bundlů. V Rijndaelu je zadefinován shodný počet řádků stavu a počet bundlů ve slově při expanzi klíče na hodnotu 4. Díky tomu vzniká shodné omezení na šifrový klíč jako na blok šifry. Obě hodnoty musí být násobkem 32. Tuto podmínku ale není obecně nutné v redukovaných modelech Rijndaelu dodržet (viz 2.3).

2.2.6 Omezení kladená na hodnotu N_{rows}

Protože tato práce neumožňuje vytvářet modely s většími parametry, než má standardní Rijndael, musí platit $N_{rows} \leq 4$.

Hodnota nemůže být rovna 1, protože kdyby tomu tak bylo, měl by výsledný stav jediný řádek a operace `ShiftRows` by byla ze šifry úplně vynechána, což by ji připravilo o difuzi ve stavu. Podobně by byla znehodnocena i operace `MixColumns`. Proto je velikost stavu omezena podmínkou $N_{rows} \geq 2$.

V případě, že by byly povoleny obdélníkové stavy, je ještě třeba vzít v úvahu omezení počtu sloupců $N_b \geq N_{rows}$. V případě, že by byl počet sloupců stavu nižší než počet řádků, nemohla by být splněna podmínka difuze (viz 1.4.2 požadavek 1).

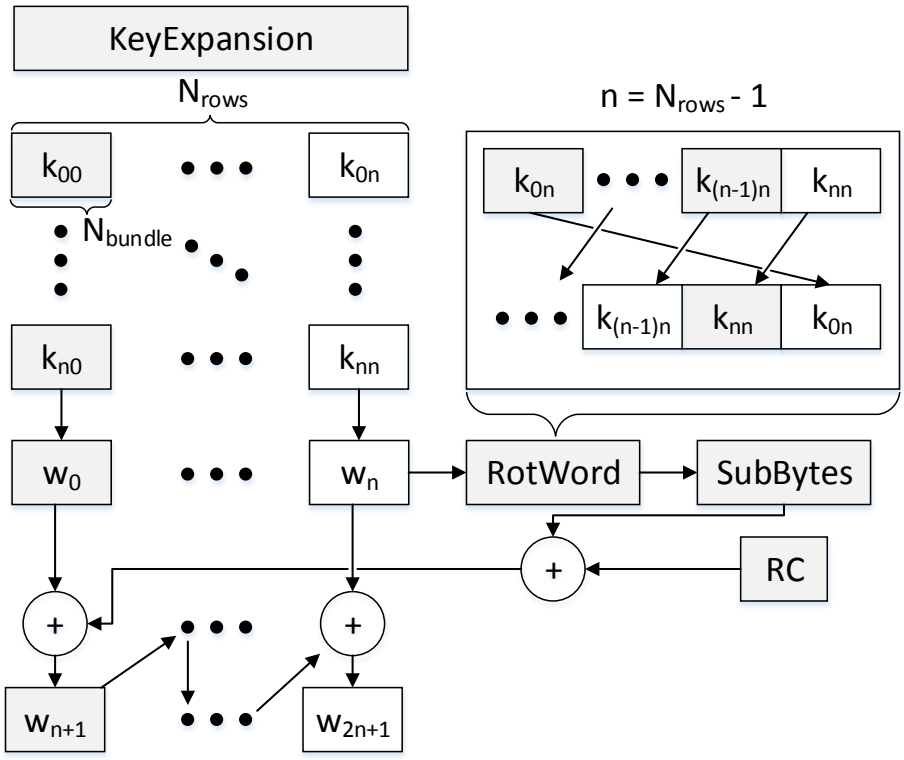
2.3 Velikost vstupního klíče

Plánování klíče se v Rijndaelu skládá z operací expanze a selekce. Expanze vytvoří ze šifrového klíče pole bytů, ze kterého následně selekce vybírá různé rundovní klíče, jež jsou na začátku šifry a v každé její rundě aplikovány operací `AddRoundKey` na aktuální stav.

Velikost šifrového klíče je omezená ve specifikaci Rijndaelu na násobky 32 s minimální hodnotou 128. Důvod vzniku těchto omezení ale nelze hledat ve velikosti stavu, nýbrž v definici expanze klíče. Za předpokladu, že by tato operace žádná omezení na šifrový klíč nekladla, mohl by mít libovolnou délku od 1 bitu dále. Operace expanze by z něj poté vygenerovala pole správné délky, ze kterého by následně operace selekce vybírala dle standardního algoritmu daného specifikací Rijndaelu.

Při návrhu redukovaných modelů Rijndaelu je tedy třeba se zaměřit především na expanzi klíče. Bylo by možné vytvořit různé algoritmy pro tuto operaci, které by splňovaly podmínky dané autory Rijndaelu. Algoritmus by ale pravděpodobně vykazovat odlišné vlastnosti než původní Rijndael, což by způsobilo špatnou použitelnost modelu. Proto tato práce využívá v redukovaných modelech upravené verze původního algoritmu s hodnotami poměrnými k velikosti modelu.

Dle [3, str. 31] se šifrový klíč expanduje na dvourozměrné pole bytů označované W, které má 4 řádky a $N_b(N_r + 1)$ sloupců. Je zřejmé, že toto pole



Obrázek 2.2: Operace `KeyExpansion` v redukovaných modelech (překresleno z [6, str. 30] a upraveno).

koresponduje počtem řádků se stavem (tedy má v redukovaných modelech N_{rows} řádků) a počtem sloupců s $N_r + 1$ stavy. Při selekci se vždy z tohoto pole odebere část, která svými rozměry odpovídá jednomu stavu. Výstupní pole by mohlo vypadat i jinak, ale mělo by obsahovat stejný počet bytů jako pole W, takže by bylo možné takový výstup vždy mapovat na pole W.

Vstupem operace je šifrový klíč. Jak již bylo řečeno, jeho velikost nemusí být omezená stejně jako v Rijndaelu, ale aby bylo dosaženo co největší podobnosti při expanzi klíče, je požadováno vstupní pole, jehož velikost je násobkem N_{rows} a její minimální hodnota je N_{rows}^2 bundlů. Vstup lze zapsat do obdélníkového pole bundlů s N_{rows} řádky. Počet sloupců tohoto pole je v této práci značen dle [3] N_k a je roven počtu bundlů šifrového klíče děleným počtem jeho řádků.

Algoritmy pro expanzi jsou ve specifikaci Rijndaelu uvedeny 2. Jeden z nich je pro případ, kdy $N_k \leq 6$, druhý pro $N_k > 6$. Důvod výběru právě konstanty 6 není uveden ani v jednom z dokumentů, kde autoři Rijndaelu popisují svoji motivaci k výběru jednotlivých částí šifry.

Algoritmus 2.1: `KeyExpansion` pro $N_k \leq 1,5N_{rows}$

```
KeyExpansion(byte K[N_rows][N_k], byte W[N_rows][N_b(N_r+1)])
{
for(j=0; j<N_k; j++)
  for(i=0; i<N_rows; i++) W[i][j] = K[i][j];
for(j=N_k; j<N_b(N_r + 1); j++)
  {
  if (j mod N_k == 0)
    {
    W[0][j]=W[0][j-N_k] XOR S[W[1][j-1]] XOR RC[j/N_k];
    for(i=1; i<N_rows; i++)
      W[i][j]=W[i][j-N_k] XOR S[W[i+1 mod 4][j-1]];
    }
  else
    {
    for(i=0; i<N_rows; i++)
      W[i][j]=W[i][j-N_k] XOR W[i][j-1];
    }
  }
}
```

Pro redukované modely je třeba určit hraniční hodnotu pro výběr algoritmu v závislosti na velikosti stavu. Zdá se, že je vhodné použít první algoritmus pokud $N_k \leq 1,5N_{rows}$ a v opačném případě použít druhý algoritmus, protože tak dojde k poměrnému zmenšení standardního Rijndaelu a počtu kroků, kdy je použita větev. Je možné, že by bylo vhodnější stanovit hranici vhodnějším způsobem, to se ale v rámci této práce nepodařilo prokázat.

Algoritmy 2.1 a 2.2 představují verzi algoritmů z [3, ste. 45, 46] upravenou pro redukované modely. Pro lepší srozumitelnost je první z nich schématicky zachycen na obrázku 2.2

2.3.1 Omezení kladená na velikost vstupního klíče

Jak již bylo řečeno, ze specifikace Rijndaelu vyplývá, že minimální velikost klíče je rovna velikosti stavu. V [3, str. 31] jsou velikosti stavu i klíče shora omezeny na 256 bitů. Toto omezení aplikované na redukované modely vypadá následovně:

$$N_{rows} \leq N_k \leq 2N_{rows} \tag{2.22}$$

Ačkoli specifikace Rijndaelu nepodporuje klíče délky větší než 256 bitů, byl navržen algoritmus nazvaný AES-512, který počítá s délkou bloku i klíče 512 bitů (viz publikace [11] a [12]). Vzhledem k tomu, že využívá shodnou délku bloku a klíče, je pro jeho simulaci možné využít vhodný redukovaný

Algoritmus 2.2: `KeyExpansion` pro $N_k > 1,5N_{rows}$

```
KeyExpansion(byte K[Nrows][Nk], byte W[Nrows][Nb(Nr+1)])
{
for(j=0; j<Nk; j++)
  for(i=0; i<Nrows; i++) W[i][j] = K[i][j];
for(j=Nk; j<Nb(Nr + 1); j++)
  {
  if (j mod Nk == 0)
    {
    W[0][j]=W[0][j-Nk] XOR S[W[1][j-1]] XOR RC[j/Nk];
    for(i=1; i<Nrows; i++)
      W[i][j]=W[i][j-Nk] XOR S[W[i+1 mod 4][j-1]];
    }
  else if (j mod Nk == Nrows)
    {
    for(i=0+ i<Nrows;i++)
       W[i][j]=W[i][j-Nk] XOR S[W[i][j-1]];
    }
  else
    {
    for(i=0; i<Nrows; i++)
      W[i][j]=W[i][j-Nk] XOR W[i][j-1];
    }
  }
}
```

model, který má tyto hodnoty také shodné. Pro tento algoritmus tedy není třeba, aby redukované modely umožňovaly klíče delší než dvojnásobek stavu. Z toho důvodu tato práce omezení ponechá. Pokud by se ukázalo prakticky využitelné délku klíče zvětšit, je třeba v redukovaných modelech přizpůsobit expanzi klíče podle toho, jak jej bude generovat algoritmus, který budou redukované modely simulovat.

2.4 Počet rund

Základní počet rund pro nejmenší velikost Rijndaelu je stanoven na 10 a zvětšuje se, pokud roste velikost bloku nebo šifrového klíče. Jak ale popisují autoři Rijndaelu, počet nemusí být dán fixně a může se změnit ve chvíli, kdy by se ukázalo, že je stanovený počet nedostatečný při obraně proti nějakému druhu kryptoanalýzy. Samotná hodnota byla stanovena jako počet rund, u kterých je možné získat výhodu proti hrubé síle, navýšená o rezervní rundy (viz [3, str. 41]).

Dle [2, str. 1] je počet rund předmětem možného nastavení, což je prakticky využito v [1, str. 52-56]. U obecných velikostí redukovaných modelů je velmi náročné určit počet rund na základě účinnosti vybraných druhů kryptoanalýzy s přidáním přiměřené rezervy. Tato metoda navíc nezaručuje vyšší podobnost se standardní verzí šifry než poměrné snížení vzhledem k velikosti modelu. Je ale třeba určit, dle jakých parametrů bude velikost poměřována.

Tato práce navrhuje základní počet rund podle původního algoritmu, kterým se určuje zvýšení počtu rund, tedy

$$N_r = \max(N_b, N_k) + 6. \tag{2.23}$$

Prakticky je tedy navrhován počet rund z rozsahu od 8 do 14. Omezení tuto hodnotu ale není striktní a může být předmětem změny. Je zřejmé, že takto pro model odpovídající Baby Rijndaelu není navržena hodnota 4, která je normálně pro Baby Rijndael používána. Počet rund stanovený autorem Baby Rijndaelu ale není nijak podložen, takže není nutné stanovit metodu, která by jej dosáhla (viz [2, str. 1]).

2.5 Branch number

Při návrhu operace `MixColumns` je třeba dosáhnout pomocí zvoleného konstantního polynomu horní hranice hodnoty branch number.

Branch number je pro zadanou lineární transformaci F definováno jako

$$B(F) := \min_{a \neq 0}(w(a) + w(F(a)). \tag{2.24}$$

$w(a)$ je bundlová váha, tedy počet nenulových bundlů v poli bundlů (tedy ve sloupci stavu modifikovaném operací `MixColumns`) a.

Věta 1. *Buď RM redukovaný model s N_{rows} řádky stavu a operací `MixColumns` MC, potom horní mez hodnoty $B(MC)$ je rovna $N_{rows} + 1$.*

Důkaz. Důkaz je založen na [4, str. 27]. Buď a sloupec stavu RM před operací MC. Protože z definice branch number vyplývá, že a nemůže mít všechny bundly rovné 0 ($a \neq 0$), je nutně alespoň jeden bundle nastaven na nenulovou hodnotu, tedy

$$w(a) \geq 1. \tag{2.25}$$

Operace MC může nastavit na nenulové hodnoty maximálně tolik elementů, kolik jich je v poli a, a těch je N_{rows}. Z toho vyplývá, že

$$w(MC(a)) \leq \ N_{rows}, \tag{2.26}$$

proto

$$\max(w(MC(a))) = N_{rows}. \tag{2.27}$$

Za předpokladu, že $w(MC(a))$ je maximální možný, platí

$$\begin{aligned} B(MC) &= \min_{a \neq 0}(w(a) + \max(w(MC(a)))) \\ &= \min_{a \neq 0}(w(a) + N_{rows}) \\ &= \min_{a \neq 0}(W(a)) + N_{rows}, \end{aligned} \tag{2.28}$$

a protože $w(a) \geq 1$, zřejmě

$$B(MC) = 1 + N_{rows}, \tag{2.29}$$

což je horní mez hodnoty $B(MC)$, jelikož právě pro tu je splněn předpoklad. □

V literatuře (např. [13]) jsou někdy definovány zvlášť diferenciální a lineární branch number. Jak je ale ukázáno v [3, str. 132], pro lineární transformaci jsou tyto hodnoty shodné, a proto je ve vztahu k `MixColumns` možné se zabývat pouze výše zmíněnou definicí.

2.5.1 Výpočet branch number

Zásadní otázkou pro vytváření redukovaných modelů Rijndaelu je, jakým způsobem lze branch number vypočítat. Protože jsou hodnoty ve sloupci stavu z konečného tělesa a počet řádků stavu je také konečný, bylo by možné vyzkoušet všechny varianty sloupců, spočítat pro ně $w(a) + w(F(a))$, kde $F(a)$ je operace `MixColumns` na sloupec a, a následně zjistit branch number jako minimum z vypočítaných hodnot. Tento postup je ale výpočetně velmi náročný. Z toho důvodu je třeba najít způsob, jiného ověření horní meze branch number.

V [13, str. 309] jsou stanoveny podmínky, které musí splňovat jednotlivé byty konstantního polynomu `MixColumns`, aby bylo dosaženo horní meze branch number u standardního Rijndaelu. Za pomoci těchto podmínek lze horní mez ověřit dostatečně rychlou cestou. Následující část se zabývá stanovením podobných podmínek i pro jiné velikosti stavu než 4×4

2.5.1.1 Koeficienty konstantního polynomu `MixColumns`

Stanovení podmínek na koeficienty konstantního polynomu `MixColumns` je rozděleno do 3 částí podle možných velikostí stavu redukovaného modelu. V prvních dvou částech jsou prováděny výpočty v tělese T. Operace sčítání značená $\oplus$ je standardním součtem polynomů. Inverze prvku a označená a^{-1} značí inverzní prvek v multiplikativní grupě tělesa. To, že invertovaný prvek není 0, je vždy zajištěno podmínkami zjištěnými v předchozích výpočtech. Číslicí 0 je označován neutrální prvek aditivní grupy tělesa T. Operace sčítání i násobení tělesa T jsou komutativní, což je v následující části práce využíváno.

2.5.1.2 Stav 2×2

Věta 2. *Buď RM redukovaný model se stavem 2×2 bundly a operací* `MixColumns` *MC, která využívá násobení konstantním polynomem*

$$c(x) := c_1 x + c_0. \tag{2.30}$$

Horní meze hodnoty $B(MC)$ je dosaženo při dodržení podmínek

$$c_0 \neq 0; c_1 \neq 0, \tag{2.31}$$
$$c_0 \neq c_1. \tag{2.32}$$

Důkaz. Pro stav o dvou řádcích a dvou sloupcích je operace MC následující:

$$\begin{pmatrix} c_0 & c_1 \\ c_1 & c_0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \tag{2.33}$$

Kde c_i jsou koeficienty konstantního polynomu MC, jednotlivé složky proměnného vektoru x mohou nabývat hodnot z T a vektor y je odpovídající výstup operace. Jsou povoleny veškeré možné vstupy s výjimkou vektoru, jež má všechny složky nulové.

Mohou nastat následující situace:

- $w(x) = 1$

  V tomto případě je pouze jedna složka vektoru x nenulová. BÚNO lze předpokládat, že nenulovou hodnotu má x_1. Platí

  $$\begin{aligned} c_1 x_1 &= y_0, \\ c_0 x_1 &= y_1. \end{aligned} \tag{2.34}$$

  Aby byla dosaženo horní meze $B(MC)$, musí být oba výstupy nenulové. Protože x_1 je nenulové a při násobení v tělese polynomů T nemůže být výsledkem 0, pokud není alespoň jeden polynom roven 0, stačí, aby bylo splněno

  $$\begin{aligned} c_0 &\neq 0, \\ c_1 &\neq 0. \end{aligned} \tag{2.35}$$

- $w(x) = 2$

  V tomto případě jsou obě složky vektoru x nenulové. Aby byla hodnota $B(MC)$ nižší než její horní mez, musí být složky výstupního vektoru nulové. Je tedy třeba řešit rovnice

  $$\begin{aligned} c_0 x_0 \oplus c_1 x_1 &= 0, \\ c_1 x_0 \oplus c_0 x_1 &= 0. \end{aligned} \tag{2.36}$$

Po vyjádření x_0 z první rovnice ($x_0 = c_0^{-1}c_1x_1$) lze dosadit do druhé a výsledek upravit

$$\begin{aligned} c_1^2x_1c_0^{-1} \oplus c_0x_1 = 0, \\ (c_1^2c_0^{-1} \oplus c_0)x_1 = 0. \end{aligned} \tag{2.37}$$

Protože x_1 je nenulové

$$\begin{aligned} c_1^2c_0^{-1} \oplus c_0 = 0, \\ c_1^2 = c_0^2. \end{aligned} \tag{2.38}$$

Vzhledem k podmínce 2.35 je c_1 nenulové a jeho inverze existuje. V T, je tedy další podmínka pro dosažení horní meze $B(MC)$

$$c_0 \neq c_1. \tag{2.39}$$

Shodná podmínka je nalezena i při vyjádření druhé složky vektoru x.

□

2.5.1.3 Stav 3×3

Věta 3. *Buď R redukovaný model se stavem 3×3 bundly a operací `MixColumns` MC, která využívá násobení konstantním polynomem*

$$c(x) := c_2x^2c_1x + c_0. \tag{2.40}$$

Horní meze hodnoty $B(MC)$ je dosaženo při splnění všech následujících podmínek

$$\begin{aligned} c_0 \neq 0,\ c_1 \neq 0,\ c_2 \neq 0, \\ c_2^2 \neq c_0c_1,\ c_1^2 \neq c_0c_2,\ c_0^2 \neq c_1c_2, \\ (c_0^2 \oplus c_1c_2)^2 \neq (c_1^2 \oplus c_0c_2)(c_2^2 \oplus c_0c_1), \\ (c_1^2 \oplus c_0c_2)^2 \neq (c_2^2 \oplus c_0c_1)(c_0^2 \oplus c_1c_2), \\ (c_2^2 \oplus c_0c_1)^2 \neq (c_1^2 \oplus c_0c_2)(c_0^2 \oplus c_1c_2). \end{aligned} \tag{2.41}$$

Důkaz. Pro tuto velikost stavu má operace MC tvar

$$\begin{pmatrix} c_0 & c_2 & c_1 \\ c_1 & c_0 & c_2 \\ c_2 & c_1 & c_0 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix} \tag{2.42}$$

Při výpočtu $B(MC)$ mohou nastat následující situace:

- $w(x) = 1$

  Pouze jedna složka vektoru x nenulová. BÚNO lze předpokládat, že nenulovou hodnotu má x_1 a platí

$$\begin{aligned} c_2x_1 = y_0, \\ c_0x_1 = y_1, \\ c_1x_1 = y_2. \end{aligned} \tag{2.43}$$

Obdobně jako u stavu velikosti 2×2 stačí splnit, že

$$\begin{aligned} c_0 &\neq 0, \\ c_1 &\neq 0, \\ c_2 &\neq 0. \end{aligned} \tag{2.44}$$

Stejných podmínek je dosaženo, i pokud je nenulová jiná složka vektoru x.

- $w(x) = 2$

  Jedna složka vektoru x je 0. Pro stanovení podmínek není podstatné, která to bude, protože v rovnicích vždy zbude stejná kombinace koeficientů, jen budou násobené jinými složkami vstupu a výsledek bude odpovídat jiné složce výstupu (lze snadno ověřit postupným dosazením 0 za každou složku vektoru x v rovnici 2.42). Vzhledem k tomu, že se testují všechny možné vstupy a u výstupu se měří jen počet nenulových bundlů, výsledkem budou stejné podmínky, ať bude nulová kterákoliv složka vektoru x. BÚNO lze tedy předpokládat, že nulová složka je $x_0 = 0$. V tom případě jsou podmínky následující:

$$\begin{aligned} c_2 x_1 \oplus c_1 x_2 &= y_0, \\ c_0 x_1 \oplus c_2 x_2 &= y_1, \\ c_1 x_1 \oplus c_0 x_2 &= y_2. \end{aligned} \tag{2.45}$$

  Aby nebylo dosaženo horní meze $B(MC)$, musí být alespoň 2 složky výstupu nulové a třetí (y_k) má libovolnou hodnotu. Existují 3 různé možnosti toho, která složka výstupu má hodnotu y_k. Pro všechny 3 je třeba vybrat rovnice, které mají nulovou hodnotu, z jedné z nich vyjádřit jednu složku x a dosadit do druhé. Nezáleží na tom, která složka vektoru x je vyjádřena, protože výsledné podmínky jsou shodné. Zde je ukázka odvození jedné podmínky, pro ostatní se postupuje analogicky:

  Za předpokladu, že $y_2 = y_k$, je třeba použít první a druhou rovnici. Z první rovnice se vyjádří x_1 a dosadí se do druhé rovnice. Vzhledem k 2.44 existuje inverze c_2.

$$\begin{aligned} x_1 &= (y_0 \oplus c_1 x_2) c_2^{-1}, \\ (c_0 y_0 \oplus c_0 c_1 x_2) c_2^{-1} \oplus c_2 x_2 &= y_1, \\ (c_0 y_0 \oplus c_0 c_1 x_2 \oplus c_2^2 x_2) c_2^{-1} &= y_1. \end{aligned} \tag{2.46}$$

  Protože $y_0 = 0$ a $y_1 = 0$

$$(c_0 c_1 \oplus c_2^2) c_2^{-1} x_2 = 0. \tag{2.47}$$

A jelikož inverze nemůže být nulová a x_2 je také nenulové, musí pro splnění rovnice platit podmínka:

$$\begin{aligned} c_0c_1 \oplus c_2^2 &= 0, \\ c_2^2 &= c_0c_1. \end{aligned} \tag{2.48}$$

Aplikací tohoto postupu na zbývající kombinace vzniknou ještě podmínky

$$\begin{aligned} c_1^2 &= c_0c_2, \\ c_0^2 &= c_1c_2. \end{aligned} \tag{2.49}$$

Pokud je alespoň jedna z podmínek splněna, není dosaženo horní meze $B(MC)$, pro její získání je tedy třeba aplikovat negaci podmínek.

- $w(x) = 3$

  V tomto případě není hodnoty horní meze dosaženo pouze ve chvíli, kdy jsou všechny složky výstupu nulové. Řeší se tedy soustava rovnic

$$\begin{aligned} c_0x_0 \oplus c_2x_1 \oplus c_1x_2 &= 0, \\ c_1x_0 \oplus c_0x_1 \oplus c_2x_2 &= 0, \\ c_2x_0 \oplus c_1x_1 \oplus c_0x_2 &= 0. \end{aligned} \tag{2.50}$$

  Pro zvýšení přehlednosti je dále násobení inverzním prvkem znázorněno jako dělení.

  Z první rovnice se BÚNO vyjádří x_0.

$$x_0 = \frac{c_2x_1 \oplus c_1x_2}{c_0}. \tag{2.51}$$

  Protože platí 2.44, existuje inverze c_0. Dosadí se do druhé rovnice a vyjádří se x_1:

$$\begin{aligned} x_1 = \frac{c_1x_0 + c_2x_2}{c_0} &= \frac{\frac{c_1c_2x_1 \oplus c_1^2x_2}{c_0} \oplus c_2x_2}{c_0} \\ x_1 &= \frac{c_1c_2x_1 \oplus c_1^2x_2 \oplus c_0c_2x_2}{c_0^2} \\ x_1 &= \frac{(c_1^2 \oplus c_0c_2)x_2}{(c_0^2 \oplus c_1c_2)} \end{aligned} \tag{2.52}$$

  Díky podmínce 2.49 je jmenovatel nenulový. Po dosazení x_0 a x_1 do třetí rovnice vznikne podmínka

$$\begin{aligned} x_2 &= \frac{c_2x_0 \oplus c_1x_1}{a_0} \\ x_2 &= \frac{((c_1c_2)(c_0^2 \oplus c_1c_2) \oplus (c_2^2 \oplus c_1)(c_1^2 \oplus c_0c_2))x_2}{c_0(c_0^2 \oplus c_1c_2)} \\ \frac{((c_1c_2)(c_0^2 \oplus c_1c_2) \oplus (c_2^2 \oplus c_1c_0)(c_1^2 \oplus c_0c_2) \oplus c_0^2(c_0^2 \oplus c_1c_2))x_2}{c_0^2(c_0^2 \oplus c_1c_2)} &= 0 \end{aligned} \tag{2.53}$$

Protože x_2 je nenulové, stačí položit roven nule jeho koeficient z čitatele:

$$((c_0^2 \oplus c_1 c_2)^2 \oplus (c_2^2 \oplus c_0 c_1)(c_1^2 \oplus c_0 c_2)) = 0 \tag{2.54}$$

Tato forma je pro počítačové zpracování dostatečná a není ji proto třeba dále upravovat nebo zjišťovat podrobnější podmínky, kdy platí.

Tento postup je potřeba aplikovat ještě několikrát. Rozdílné podmínky vzniknou, pokud jsou rovnice vyjadřovány v jiném pořadí. První x_0 lze vyjádřit ze 3 rovnic a x_1 poté ze dvou. Celkově je tedy 6 možností pořadí a mělo by vzniknout 6 podmínek. Když se ale uplatní výše zmíněný postup na všech 6 pořadí, ukazuje se, že podmínky jsou pouze 3, neboť záleží pouze na výběru rovnice, ze které se vyjádří x_0; oba možné výběry druhé rovnice pak vedou na stejné výsledky.

Nezáleží na tom, která složka vektoru x se vyjádří jako první, respektive jako druhá, protože při změně pořadí složek lze vybrat postupně rovnice tak, že vyjdou shodná vyjádření těchto rovnic, jen s odlišnými složkami vektoru. Ty ale všechny nabývají stejných hodnot, takže je lze přejmenovat a tak opět získat vyjádření shodná s těmi, kdy je jako první použito x_0 a druhé x_1. Samozřejmě se tím změní pořadí rovnic, ale protože jsou k získání podmínek použita všechna možná pořadí, budou ve výsledku vygenerovány shodné výsledky.

□

2.5.1.4 Stav 4x4

Pro tuto velikost stavu není nutné podmínky počítat, protože jsou již stanoveny v [13, str. 309]. Je ale třeba ověřit, že jsou splněny předpoklady, které článek stanovuje:

1. C je matice 4×4
2. Prvky C jsou z GF(2^8)
3. Hodnost C je rovna 4
4. Matice C má tvar: $\begin{pmatrix} c_0 & c_3 & c_2 & c_1 \\ c_1 & c_0 & c_3 & c_2 \\ c_2 & c_1 & c_0 & c_3 \\ c_3 & c_2 & c_1 & c_0 \end{pmatrix}$

Předpoklady 1 a 4 jsou zřejmě u matice operace `MixColumns` splněny. Vzhledem k tomu, že operace musí být invertovatelná, je matice regulární, tedy má hodnost 4 a předpoklad 3 je tak také naplněn. Problematický je pouze bod číslo 2, protože redukované modely mohou změnit velikost tělesa. Jak se dalo pozorovat u menších stavů, podmínky nebyly závislé na velikosti tělesa,

ale na jeho operacích, protože vycházely z toho, kdy může součtem vzniknout nulový prvek a ostatní prvky mohly být z libovolně velkého tělesa. Protože se operace tělesa v redukovaných modelech nemění, lze podmínky z [13] použít i pro prvky z T.

Dosažení horní meze hodnoty branch number pro stav 4×4 lze dosáhnout splněním podmínek:

$$\begin{gathered} c_0 \neq 0,\ c_1 \neq 0,\ c_0 \neq 0,\ c_3 \neq 0, \\ c_0 \neq c_2,\ c_1 \neq c_3, \\ c_0^2 \neq c_1 c_3,\ c_2^2 \neq c_1 c_3,\ c_1^2 \neq c_0 c_2, \\ c_3^2 \neq c_0 c_2,\ c_0 c_1 \neq c_2 c_3,\ c_1 c_2 \neq c_0 c_3. \end{gathered} \tag{2.55}$$

2.6 Omezení redukovaných modelů

Tato část shrnuje omezení, která byla stanovena v předchozí kapitole.

- Velikost bundlu je mezi 2 a 8 bity.
- Možné velikosti stavu jsou 2×2, 3×3 nebo 4×4.
- Velikost vstupního klíče se pohybuje v násobcích N_{row} bundlů od N_{rows}^2 do $2N_{rows}^2$.
- Operace `MixColumns` používá vždy modul $x^{N_{rows}} + 1$.
- Afinní transformace používá vždy modul $x^{N_{bundle}} + 1$
- Kritéria výběr afinní transformace modelu obecně nezajišťují dostatečnou podobnost s Rijndaelem pro algebraickou kryptoanalýzu.

Kapitola 3

Průvodce pro vytváření redukovaných modelů Rijndaelu

3.1 Analýza

Jedním z cílů této práce je vytvořit nástroj, jež umožní kryptoanalytikům navrhnout redukovaný model Rijndaelu, na kterém budou moci experimentálně ověřit svoji teorii, aniž by se museli do hloubky zabývat vlastnostmi, které redukci umožňují.

Tato kapitola se zabývá návrhem tohoto nástroje, specifikuje hlavní požadavky, které by měl nástroj splňovat, a stanovuje způsob, jakým budou tyto požadavky naplněny. Dále se detailněji zabývá logickým modelem tohoto nástroje.

3.1.1 Požadavky

Jak bylo řečeno, nástroj je určen především výzkumníky na poli kryptoanalýzy AESu. Z toho důvodu lze předpokládat, že uživatelé budou poučeni o operacích, transformacích a zavedených pojmech šifry Rijndael. Nelze ale předpokládat znalost parametrů, které umožňují vytváření redukovaných modelů a dopady jejich změny na vlastnosti šifry.

Redukované modely Rijndaelu by měly sloužit k tomu, aby výzkumník mohl ověřit svoji teorii experimentálně a aby mohl škálovat velikost problému podle výsledků předchozích experimentů. Nástroj by měl umožnit snadné navrhování takových modelů bez nutnosti hlubšího zkoumání vlastností modelů.

Aby byly tyto cíle naplněny, shrnuje tato práce požadavky na nástroj do následujících bodů:

- bude umožněno měnit parametry redukovaných modelů popsané v kapitole 2

- k dispozici bude popis dopadů změny těchto parametrů jako shrnutí kapitoly 2
- budou ohlídány meze a závislosti těchto parametrů stanovené v kapitole 2
- při libovolném povoleném uživatelském vstupu vznikne redukovaný model odpovídající svou strukturou specifikaci Rijndaelu.
- výstupem bude model vypsaný v textovém formátu a kód v jazyce C připravený pro knihovnu s implementací modelů (viz kapitola 4).
- defaultně nabízené hodnoty povedou k modelu co nejbližšímu specifikaci Rijndaelu.
- bude upřednostněna jednoduchost a menší výběr parametrů modelu před komplexním a složitým řešením

Za rozumný model je považován takový model, který se co nejvíce blíží specifikaci Rijndaelu. V případě, že uživatel stanoví pevně některé parametry, nástroj by měl navrhovat hodnoty ostatních parametrů tak, aby se výsledný model svými vlastnostmi co nejméně lišil od specifikace Rijndaelu.

Poslední dva body slouží k naplnění požadavku, aby uživatel nemusel do hloubky studovat vlastnosti modelů. V případě, že bude nástroj dávat uživateli na výběr z více hodnot, měly by být takové, aby vznikl rozumný model. Pokud se uživatel nebude chtít nějakým parametrem zabývat, měla by mu být nabídnuta hodnota, která povede k vytvoření rozumného modelu.

3.1.2 Volba způsobu implementace

Aby byly splněny výše stanovené podmínky a aby bylo vytváření modelů co možná nejpohodlnější, bylo rozhodnuto vytvořit nástroj v podobě průvodce s grafickým rozhraním. Průvodce v jednotlivých krocích nabídne uživateli parametry modelu a jejich popis, následně jednotlivé operace modelu a části, které je u nich potřeba upravit vzhledem k vybraným parametrům. Nakonec průvodce umožní vygenerovat požadované výstupy. V každém kroku bude průvodce nabízet primárně hodnoty, které povedou k vytvoření rozumného modelu.

3.1.3 Návrh průvodce

Aplikace je logicky horizontálně rozdělena na 3 vrstvy podle architektury model–view–presenter (MVP), viz [14]. První vrstvou (view) je uživatelské rozhraní v podobě jednotlivých obrazovek průvodce. Třetí vrstva (model) je samotný vytvořený model, který slouží k ukládání zadaných vstupů. Druhá vrstva (presenter) zpracovává vstupy z první vrstvy a předává do ní výstupy,

v případě potřeby ukládá nebo nahrává data ze třetí vrstvy. První dvě horizontální vrstvy jsou navíc rozděleny vertikálně. Každá vertikální část odpovídá jedné obrazovce průvodce.

Tento návrh je vhodný pro případ, kdy by se ukázalo, že je třeba rozšířit funkcionalitu programu nebo přidat novou obrazovku. V obou případech bude ovlivněna jen malá část programu a do zbytku není třeba vůbec zasahovat.

3.1.3.1 Rozdělení stránek

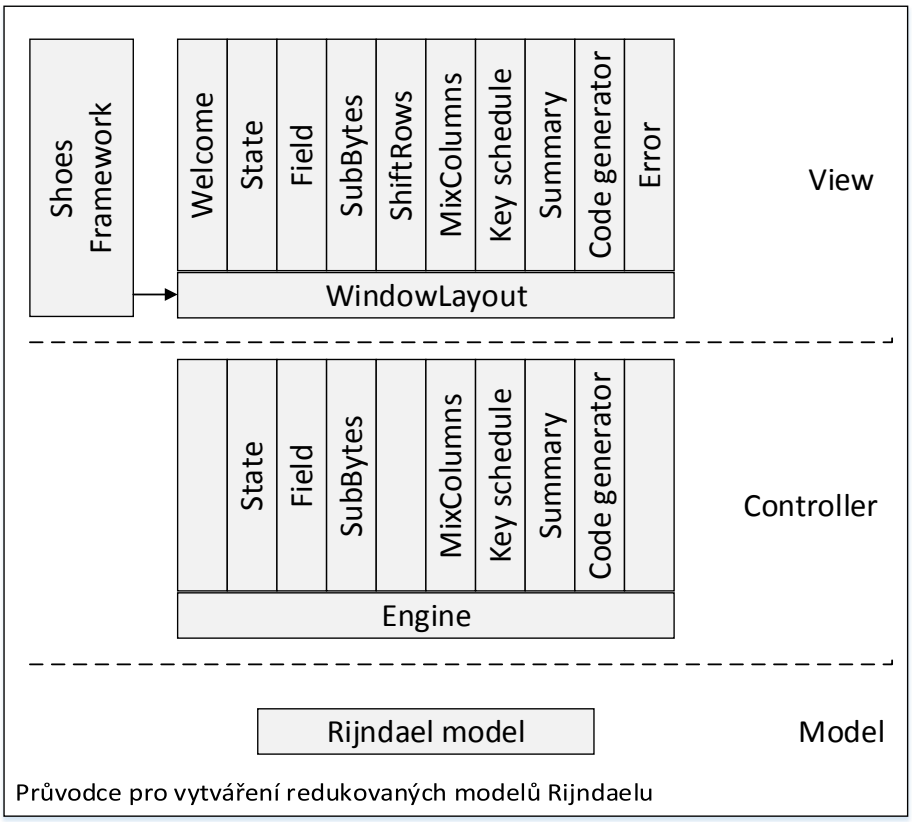
Stránky průvodce jsou rozděleny následovně:

1. Uvítací obrazovka. Vysvětlí uživateli, k čemu aplikace slouží.
2. Stav. Určí velikost stavu a počet bitů v bundlu.
3. Těleso. Pomůže uživateli vybrat ireducibilní polynom $m(x)$.
4. `SubBytes`. Umožní návrh afinní transformace.
5. `ShiftRows`. Je pouze informativní, vzhledem k omezení na čtvercové stavy.
6. `MixColumns`. Slouží k výběru konstantního polynomu $c(x)$.
7. Plánování klíče. Dává možnost zvolit počet rund šifry a velikost vstupního klíče.
8. Shrnutí. Vypíše model v textové formě, aby měl uživatel možnost jej zkontrolovat.
9. Generování kódu. Vypíše model ve formě kódu, který lze vložit do programu v jazyce C.

Návrh vyplývá ze závislostí, které jednotlivé stránky mají (například pro určení afinní transformace je nutné mít k dispozici velikost bundlu a polynom $m(x)$). Rozdělení a pořadí je stanoveno tak, aby při postupném vyplňování jednotlivých stránek byly vždy k dispozici potřebné informace. Grafické znázornění návrhu je zachyceno na obrázku 3.1.

3.1.4 Cílová platforma

Cílovou skupinou uživatelů by měli být především výzkumníci a akademici. U nich lze předpokládat, že v případě potřeby používají nástroje operačního systému Linux. Navíc je obvyklé že užitečné open-source programy bývají k dispozici právě pro tento systém, proto bylo rozhodnuto, že aplikace bude vytvořena pro tuto platformu.



Obrázek 3.1: Návrh průvodce pro vytváření redukovaných modelů.

3.2 Realizace

V této kapitole jsou podchycena důležitější rozhodnutí, která vývoj aplikace průvodce doprovázela. Dále jsou zde řešeny problémy, které při implementaci nastaly.

3.2.1 Výběr implementačních nástrojů

Pro implementaci byl zvolen jazyk Ruby. Důvody byly zejména tyto:

- Jde o interpretovaný jazyk, takže odpadá nutnost kompilace, změny ve zdrojovém kódu se projeví ihned, možnost interaktivního testování.
- Podpora grafického rozhraní.
- Podpora dynamických datových typů.
- Integrovaná podpora bitových operací.

- Dobrá dokumentace.
- Podpora cílového operačního systému Linux.

Nevýhodou je pomalejší zpracování než nabízejí některé jiné programovací jazyky.

Pro tvorbu GUI byl zvolen framework Shoes, který nabízí velmi jednoduché rozhraní. Aby bylo možné zabalit vzniklou aplikaci jako gem [1], byla zvolena implementace tohoto frameworku nazvaná GreenShoes. Tato volba se v průběhu vytváření aplikace ukázala být jako nevhodná, neboť možnosti tohoto frameworku byly velmi omezené a často neposkytovaly potřebnou funkcionalitu.

3.2.2 Problémy jednotlivých stránek průvodce

3.2.2.1 SubBytes

Aplikace používá pro všechny operace polynomiální zápis. Bylo nutné stanovit polynom odpovídající afinní transformaci Rijndaelu a následně se snažit uživateli nabízet takové polynomy, které jsou co nejpodobnější Rijndaelu.

V [4, str. 26] je uvedený polynomiální zápis afinní transformace operace `SubBytes` chybný. Správně je zápis odvozený z maticového zápisu podle [5, str. 16], jak je uvedeno v kapitole 2.13.

Výběr aditivního polynomu

Problém spočívá v tom, že při volbě velikosti bundlu a modulu $m(x)$ jako ve standardním Rijndaelu, nepreferuje průvodce ani za splnění podmínek z 2.1.2.3 odpovídající aditivní polynom.

Důvod je ten, že aplikace nebere v úvahu výsledný algebraický výraz reprezentující operaci `SubBytes` a jeho odolnost proti kryptoanalýze. Tuto vlastnost lze ale jen velmi těžko měřit, protože je závislá na zvolené algebraické kryptoanalýze a modely mají mimo jiné sloužit také k objevování nových typů kryptoanalýz.

Vzhledem k omezením redukovaných modelů (viz 2.6) bylo rozhodnuto ponechat nastavení tak, že průvodce nenabídne stejný aditivní polynom jako Rijndael. Nabídne ale sadu vhodných možností, takže uživatel může zvolit takový polynom, aby algebraická struktura byla co nejpodobnější Rijndaelu z pohledu ověřované algebraické kryptoanalýzy.

Navrhování multiplikativního polynomu

V některých případech dochází k tomu, že aplikace nabídne multiplikativní polynom, ale nenabídne žádné aditivní polynomy z toho důvodu, že žádný z nich nesplňuje kladené podmínky. V takových případech by bylo lepší vůbec nenabízet zmíněný multiplikativní polynom. Toho lze ale těžko dosáhnout

[1] Gem je standardní balíček s programem nebo knihovnou pro jazyk Ruby. Viz [15].

z hlediska implementace, protože v takovém případě by se u všech navrhovaných multiplikativních polynomů musela ověřovat existence aditivního polynomu ještě před tím, než jsou uživateli zobrazeny, tedy při přechodu na obrazovku `SubBytes`. Protože je toto ověřování výpočetně relativně náročné, docházelo by u větších modelů k delšímu výpočtu a program by při přechodu mezi obrazovkami na chvíli zamrzl.

Z hlediska uživatele se zdá jako přijatelnější řešení, když jsou nabízeny i polynomy, které nejsou relevantní, než aby se program zasekával. Nevhodné multiplikativní polynomy není možné uložit, takže nenastane situace, kdy by z nich byl vytvořen model.

3.2.2.2 MixColumns

Problém při určování konstantního polynomu pro operaci `MixColumns` spočívá v ověřování, že bylo dosaženo horní meze branch number. Ukázalo se, že není možné využít pro ověření prosté vyzkoušení všech možností, protože pro modely od $N_{bundle} >= 5$ byla doba tohoto výpočtu neúnosně dlouhá. Proto bylo třeba hledat jinou metodu ověřování této hodnoty. Nalezený způsob je popsán v části 2.5 tohoto textu.

Kapitola 4

Implementace redukovaných modelů Rijndaelu

4.1 Analýza

Tato kapitola se zabývá nástrojem, který umožní navržené redukované modely prakticky využít k šifrování a dešifrování. Jsou zde popsány požadavky na tento nástroj, dále jeho návrh a uživatelské rozhraní.

4.1.1 Požadavky

Nástroj má sloužit především k testování kryptoanalýzy Rijndaelu na zvoleném redukovaném modelu. Lze jej využít i k dalším experimentům a testům. Ve všech případech je potřeba, aby měl nástroj jednoduché rozhraní a aby pracoval efektivně, protože bude pravděpodobně potřeba šifrovat nebo dešifrovat opakovaně. Vstupy do modelu a výstupy z něj by měly být snadno strojově zpracovatelné. Není naopak potřeba, aby nástroj bezpečně zacházel se svým stavem a klíči, protože má sloužit pouze k výzkumům.

Shrnutí výše podmínek na nástroj je tedy následující:

- efektivní šifrování a dešifrování
- jednoduché rozhraní pro volbu modelu
- vstup a výstup vhodný ke strojovému zpracování
- není nutná bezpečnost implementace

4.1.2 Volba způsobu implementace

Vhodnou cílovou platformou pro implementaci je, ze stejných důvodů jako u průvodce pro vytváření redukovaných modelů, OS Linux. Aby byly splněny výše zmíněné podmínky, bylo rozhodnuto vytvořit nástroj v podobě knihovny

jazyka C. Jazyk C byl zvolen zejména pro svou rychlost a snadnou práci s bitovými operacemi.

4.1.3 Návrh knihovny

Knihovna má rozhraní definované v `rijndael_reduced_models.h` v podobě struktury modelu a několika externích funkcí. Uživatelské vstupy jsou zpracovány hlavními funkcemi s implementací algoritmů Rijndaelu ze souboru `rijndael_reduced_models.c`. Ty využívají podpůrné funkce pro kontroly hodnot a kopírování polí, jež jsou uloženy v souboru `rrm_support.c`. Dále využívají funkce pro práci s tělesem polynomů, které obsahuje soubor nazvaný `rrm_polynomial_arithmetic.c`.

Poslední částí knihovny je rozhraní pro práci s chybami a chybovými hláškami, které je uloženo v souborech `rrm_error.c` a `rrm_error.h`.

Výsledná struktura knihovny je zachycena na obrázku 4.1.

4.2 Realizace

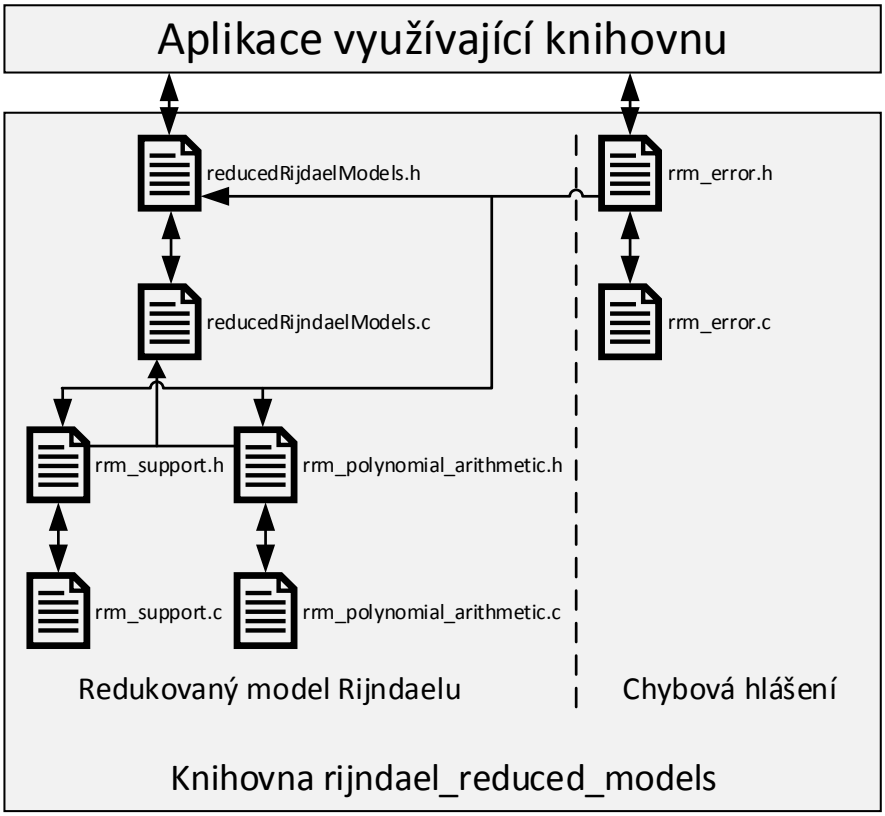
V této části je popsána realizace knihovny v jazyce C. Zejména je zde rozebrána struktura, která slouží jako uživatelské rozhraní. Dále jsou tu zmíněny jednotlivé externí funkce knihovny. Nakonec je zmíněno, jakým způsobem se řeší šifrování a dešifrování vzhledem k požadované efektivitě.

4.2.1 Rozhraní knihovny

4.2.1.1 Struktura `RR_MODEL`

Následující struktura slouží k předávání informací o modelu, které uživatel získá s pomocí průvodce pro návrh redukovaných modelů:

```
typedef struct Model{
  unsigned int numberOfRounds;
  unsigned int numberOfKeyBundles;
  unsigned int bundleSize;
  unsigned int numberOfStateRows;
  unsigned int irreduciblePolynomial;
  BUNDLE * mixColumnsPolynomial;
  BUNDLE * invMixColumnsPolynomial;
  BUNDLE * affineTransformation;
  BUNDLE * sBox;
  BUNDLE * invSBox;
  unsigned int flags;
  void   * internalStructures;
}RR_MODEL;
```



Obrázek 4.1: Struktura knihovny realizující implementaci redukovaných modelů Rijndaelu.

Datový typ `BUNDLE` je alias za `unsigned int`. Významově se liší v tom, že `BUNDLE` používá pouze N_{bundle} nejnižších bitů. Prvních pět proměnných struktury `RR_MODEL` musí být vyplněno v každém případě, ať má model sloužit k šifrování či dešifrování.

Pole `MixColumnsPolynomial` musí být vyplněno pouze v případě, že model slouží k šifrování. Jeho verze s předponou inv naopak v případě, že model slouží k dešifrování.

Pole `AffineTransformation` musí být vyplněno vždy s výjimkou případu, kdy uživatel předá knihovně potřebné S-boxy jako vyplněná pole bundlů a nastaví patřičné příznaky (viz dále).

Pole `sBox` a `invSBox` slouží k vlastní definici S-boxů a standardně není potřeba je vyplňovat, knihovna je doplní sama díky znalosti afinní transformace. Pokud by uživatel z nějakého důvodu chtěl použít vlastní S-box, je možné jej do tohoto pole vyplnit a není nutné udávat afinní transformaci. Totéž platí

i pro inverzní S-box sloužící k dešifrování.

Proměnná `flags` musí být vždy nastavena na hodnotu 0. Poté k ní jsou logickým součtem přidávány příznaky, které knihovna nabízí:

- `DONT_ENCRYPT` - Příznak je nastaven, pokud model neslouží k šifrování. V tom případě není nutné zadávat konstantní polynom operace `MixColumns`.
- `DONT_DECRYPT` - Příznak je nastaven, pokud model neslouží k dešifrování. V tom případě není nutní zadávat konstantní polynom inverzní operace `MixColumns` ani inverzi afinní transformace či inverzní S-box.
- `OWN_SBOX` - Příznak je nastaven, pokud je k šifrování použit uživatelem definovaný S-box a je tedy vyplněno pole `sBox`. V tom případě není potřeba vyplňovat afinní transformaci.
- `OWN_INV_SBOX` - Příznak je nastaven, pokud je k dešifrování použit uživatelem definovaný S-box a je tedy vyplněno pole `invSBox`. V tom případě není potřeba vyplňovat inverzi afinní transformace.

Ukazatel `internalStructures` nemá být uživatelem vyplňován. Ukazuje na struktury používané při práci s modelem. Je zde uložen proto, že struktura `RR_MODEL` je předávána všem externím funkcím knihovny, takže je do ní možné uložit již spočítané informace, které se tak nemusí počítat znovu při každém šifrování a dešifrování, čímž se zvyšuje efektivita implementace.

4.2.1.2 Externí funkce knihovny

Knihovna nabízí uživateli co nejjednodušší rozhraní a tedy i velmi jednoduché funkce:

```
RRM_ERROR initModel      (RR_MODEL * model);
RRM_ERROR rijndaelEncrypt(RR_MODEL * model,
                          BUNDLE * key,
                          BUNDLE * openText,
                          BUNDLE * cypherText);
RRM_ERROR rijndaelDecrypt(RR_MODEL * model,
                          BUNDLE * key,
                          BUNDLE * openText,
                          BUNDLE * cypherText);
RRM_ERROR deleteModel    (RR_MODEL * model);
```

Funkce `initModel` slouží k alokaci paměti a uložení proměnných. Jako vstup požaduje strukturu `RR_MODEL`, do které uživatel vyplnil informace o modelu. Alokované paměti se uloží také do této struktury, takže jde zároveň o parametr výstupní. Pro následné uvolnění paměti slouží `deleteModel`.

Funkce `rijndaelEncrypt` a `rijndaelDecrypt` provedou samotné šifrování, respektive dešifrování. Obě požadují kromě ukazatele na model ještě tři pole bundlů, z nichž všechna musí být předem alokována uživatelem. Pole `key` slouží k uložení klíče a má velikost odpovídající počtu bundlů klíče. Při šifrování je do pole `openText` vyplněn OT a do pole `cypherText` uloží operace výstupní $ŠT$. Při dešifrování je naopak vstupem `cypherText` a výstupem `openText`. Obě pole mají velikost odpovídající počtu bundlů vstupního bloku.

Výstupem ze všech čtyř funkcí je datový typ `RRM_ERROR`, který obsahuje číslo chyby podobně jako je tomu u standardní c knihovny `errno`. Pokud funkce vrátí hodnotu `RRM_SUCCESS`, znamená to, že proběhla v pořádku. V opačném případě nastala chyba, jejíž slovní popis může uživatel získat touto funkcí:

```
const char* rrm_show_error(RRM_ERROR error);
```

Funkce vrací konstantní řetězec s popisem chyby zakončený znakem `\0`.

4.2.2 Efektivita implementace knihovny

Protože lze očekávat, že knihovna bude použita k mnohonásobnému opakování šifrování nebo dešifrování, je třeba se zabývat tím, zda nejsou tyto operace zbytečně pomalé.

V inicializační fázi se předpočítají tabulky S-boxů a rundovní konstanty pro expanzi klíčů, zkontrolují se uživatelské vstupy, alokuje se potřebná paměť a zkopírují se pole zadaná uživatelem.

V rámci šifrování a dešifrování dochází už pouze k nutným operacím, které jsou realizovány bitově tam, kde je to možné. Násobení polynomů probíhá pomocí funkce `xtime`, která je doporučená standardem AES [5, str. 11].

Efektivita je snížena tím, že je knihovna připravena pro širokou škálu modelů, takže ji není možné optimalizovat na takové úrovni jako Rijndael pracující s 8bitovými bundly a 32bitovými sloupci stavu. Navíc jsou přidány podmínky, které ověřují vlastnosti daného modelu a neustále se čtou informace ze struktury `RR_MODEL`, což implementaci také trochu zpomaluje.

Pro zvýšení efektivity byla ještě zvážena možnost vyjmutí expanze klíče před šifrování a dešifrování, což by tyto operace mohlo urychlit v případě, že by stále probíhaly se stejným klíčem. Nakonec ale bylo rozhodnuto v rámci zachování jednoduchého uživatelského rozhraní ponechat expanzi jako součást každého šifrování a dešifrování.

Kapitola 5

Využití redukovaných modelů ke kryptoanalýze

Tato kapitola přináší ukázku využití redukovaných modelů při provádění experimentů. Nejprve je zde představena kryptoanalýza, která by potenciálně mohla ohrozit bezpečnost šifry. Poté je tato kryptoanalýza testována na vybraných redukovaných modelech. Výsledkem kapitoly je vyhodnocení potenciální nebezpečnosti zvolené kryptoanalýzy pro Rijndael.

5.1 Kryptoanalýza opakovaným šifrováním

Tato práce přináší návrh kryptoanalýzy, která využívá opakovaného šifrování k nalezení otevřeného textu bez znalosti klíče. Nebyla nalezena literatura, která by tuto metodu popisovala a uváděla její výsledky. Z toho důvodu byla vybrána jako vhodná ukázka využití modelů.

Pro provedení kryptoanalýzy je třeba mít k dispozici šifrátor, který šifruje stále stejným klíčem, a u kterého je možné volit různé otevřené texty. Cílem je odhalit blok otevřeného textu, jehož šifrovaná podoba je k dispozici. Hledaný otevřený text je značen OT_0 a jemu odpovídající šifrový text $\check{S}T_0$.

Proces šifrování je vyjádřen jako transformace $\check{S}T = E(OT, K)$, kde K je šifrový klíč. Ten zůstává po celou dobu kryptoanalýzy konstantní, proto je zápis zkrácen na $\check{S}T = E_K(OT)$.

Věta 4. *Buď M množina všech možných bloků redukovaného modelu R a $K \in M$ konstantní klíč, potom šifrování redukovaným modelem R klíčem K je bijektivní zobrazení $E_K : M \to M$.*

Důkaz. Je třeba ověřit, že zobrazení E_K je bijektivní, tedy prosté a na.

Zobrazení je prosté. Pokud by nebylo, musel by při šifrování klíčem K vzniknout jeden $\check{S}T$ shodný pro alespoň 2 různé OT. V tom případě by při de-

šifrování $\check{S}T$ nebylo možné jednoznačně určit, který OT byl zašifrován, což z definice blokové šifry není možné.

Zobrazení je na. Protože je E_K prosté, musí mít množina všech $\check{S}T$ (dále $M_{\check{S}T}$) shodný nebo větší počet prvků, než množina všech možných OT (dále M_{OT}). Šifra Rijndael a její redukované modely vždy umožňují šifrování všech možných vstupních bloků odpovídající délky. Tedy

$$\begin{aligned} |M_{OT}| &= |M|, \\ |M_{\check{S}T}| &\geq |M|. \end{aligned} \tag{5.1}$$

Z definice M a z toho, že $\check{S}T$ je výstupním blokem šifry, plyne $M_{\check{S}T} \subseteq M$ a tedy $|M_{\check{S}T}| \leq |M|$. Zřejmě $M_{\check{S}T} = M$. Jelikož jde o zobrazení prosté z celé množiny M do celé množiny M, musí být nutně na.

Z výše uvedeného vyplývá také to, že $E_K \in M \times M$, jde tedy skutečně o zobrazení $M \to M$. □

Dále bude využito značení

$$E_K^n(OT) := \underbrace{E_K(E_K(\cdots E_K(OT)))}_{n}, \tag{5.2}$$

kde $E_K^0(OT) := OT$.

Definice 2. Relace $R \subseteq M \times M$ je nazvána dosažitelnost opakovaným šifrováním s klíčem K, pokud $\forall (x, y) \in R : \exists n \in \mathbb{N}$, tak že $y = E_K^n(x)$.

Věta 5. *Dosažitelnost opakovaným šifrováním s klíčem K je relace ekvivalence.*

Důkaz. Buď $R \subseteq M \times M$ relace dosažitelnost opakovaným šifrováním s klíčem K. Potom jsou splněny podmínky ekvivalence:

- Reflexivita. $\forall x \in M : (x, x) \in R$.

  Operace je reflexivní, protože $x = E_K^0(x)$.

- Symetrie. Pokud $(x, y) \in R$, pak $(y, x) \in R$.

  Operace je symetrická. Díky tomu, že pro libovolné l je E_k^l bijekce a M je konečná množina, musí existovat $n \leq |M|$, takové, že platí

  $$\forall x \in M, \exists n \leq |M| : x = E_K^n(x). \tag{5.3}$$

  Pro důkaz sporem buď použit předpoklad, že $(x, y) \in R$ a zároveň $(y, x) \notin R$. Podle 5.3 musí existovat n_y, takové, že $y = E_K^{n_y}(y)$. Z předpokladu vyplývá $\exists m : y = E_K^m(x)$.

  Pokud by m bylo větší nebo rovno n_y, pak by nutně muselo platit $x = E_K^{m-n_y}$, což by byl spor s tím, že $(y, x) \notin R$. Číslo m tedy musí být menší

Algoritmus 5.1: Kryptoanalýza opakovaným šifrováním

```
i=1
ŠT_1 = E_K(ŠT_0)
opakuj dokud ŠT_i ≠ ŠT_0{
 i = i + 1
 ŠT_i = E_K(ŠT_i-1)
}
```

než n_y. Z toho plyne existence $z \in M : y = E_K^{n_y - m}(z)$. Protože pro libovolné l je E_K^l bijekce, musí $z = x$, což je opět spor s předpokladem.

V důkazu je využito toho, že složení bijektivních zobrazení je také bijektivní zobrazení.

- Tranzitivita. Pokud $(x, y) \in R$ a $(y, z) \in R$, pak $(x, z) \in R$.

  Operace je tranzitivní, protože pokud $y = E_K^n(x)$ a $z = E_K^m(y)$, pak $z = E_K^m \circ E_K^n(x) = E_K^{m+n}(x)$.

□

Definice 3. Kružnicemi v prostoru ŠT jsou nazývány disjunktní třídy ekvivalence, na které rozděluje množinu M relace R dosažitelnost opakovaným šifrováním s klíčem K.

Množina všech prvků ekvivalentních s $x \in M$ je značena

$$[x]_K := y \in M | (x, y) \in R.$$

V případě, že nemůže dojít ke zmatení čtenáře, jsou kružnice v prostoru ŠT dále označovány pouze jako kružnice. Zavedení značení tříd ekvivalence bylo převzato z [16, str. 6].

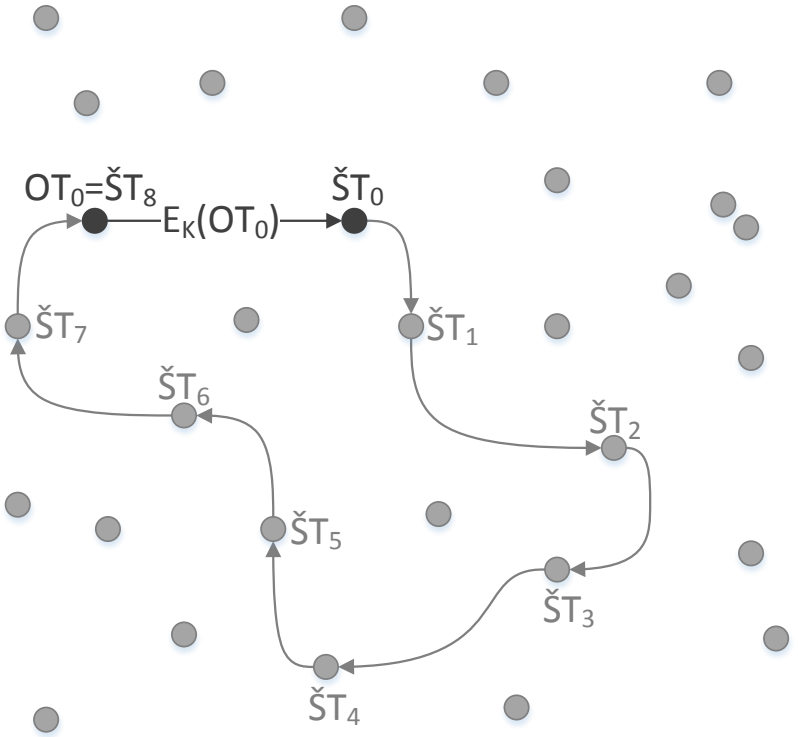
Kryptoanalýza opakovaným šifrováním využívá toho, že pro daný klíč K $OT_0 \in [\text{Š}T_0]_K$ a je tedy dosažitelný opakovaným šifrováním s klíčem K.

Základní verzi metody opakovaného šifrování lze popsat algoritmem 5.1.

V okamžiku, kdy algoritmus skončí, je hledaný OT uložen v proměnné ŠT_i. Ukončovací podmínka cyklu zřejmě musí nastat vždy, protože $OT_0 \in [\text{Š}T_0]_K$.

Graficky je funkce algoritmu znázorněna na obrázku 5.1. Jednotlivé body představují šifrové texty v prostoru ŠT. Šipky vyjadřují proces šifrování klíčem K. Počátek šipky je vstupem a konec výstupem šifrování. Na obrázku je zachycena jedna kružnice v prostoru ŠT.

Otázkou je, jak vypadá prostor šifrových textů. Za předpokladu, že by se skládal z většího množství malých kružnic, znamenalo by to, že kryptoanalýza opakovaným šifrováním může být účinnější při hledání otevřeného textu, než hrubá síla. Kdyby naopak vznikla jediná velká kružnice, znamenalo by to odhalení struktury tam, kde by se šifra měla chovat náhodně. Ke zkoumání toho, jak vypadají jednotlivé třídy šifrových textů lze využít redukované modely.



Obrázek 5.1: Kryptoanalýza opakovaným šifrováním

Tabulka 5.1: Modely využité ke kryptoanalýze opakovaným šifrováním

Parametr	Model 1	Model 2	Model 3
N_{bundle}	4	3	3
N_{rows}	2	2	2
N_k	2	2	2
$m(x)$	x^4+x+1	x^3+x+1	x^3+x+1
aditivní polynom afin. tr.	x^3+x	x	x
multiplikativní polynom afin. tr.	x^3+x^2+x	x^2	x^2
konstantní polynom `MixColumns`	D$x+5$	$3x+1$	$3x+1$
N_r	4	4	5

5.1.1 Průzkum prostoru šifrových textů v redukovaných modelech

K odhadu dopadů výše zmíněné kryptoanalýzy byly využity malé modely, které umožňují vygenerování všech možných bloků $\check{S}T$ ke všem možným šifrovým klíčům K. Byly využity 3 modely udané v [5.1].

Tabulka 5.2: Ukázka naměřených dat

Klíč	Výpis délek cyklů	Unikátních cyklů
0000	1752,208,1594,216,279,12,34,1	8
0001	2526,77,655,758,63,14,2,1	8
0002	946,1529,323,910,342,13,31,2	8
0003	2843,231,756,19,205,37,1,4	8
0004	1914,1385,345,120,251,44,10,25,1,1	10
...	...	...

Z vygenerovaných dat byly pomocí skriptů získány počty a délky kružnic v prostoru šifrových textů pro každý možný šifrový klíč. Tabulka 5.2 uvádí příklad získaných dat. Z těchto údajů již lze statisticky odhadovat dopady použití metody opakovaného šifrování ke kryptoanalýze každého modelu.

Jedním z cílů experimentu je získat střední hodnotu počtu šifrování, jež jsou třeba k nalezení hledaného otevřeného textu, aby bylo tuto metodu možné porovnat s hrubou silou.

Věta 6. *Buď M_K množina všech klíčů a M_{OT} množina všech možných bloků OT v daném redukovaném modelu R. Dále buď k_{iK} i-tá nejdelší kružnice, která vznikne opakovaným šifrováním klíčem $K \in M_K$. Za předpokladu, že klíč $K \in M_K$ a počáteční otevřený text $OT_0 \in M_{OT}$ jsou zvoleny náhodně se stejnou pravděpodobností a nezávisle, je střední hodnota počtu šifrování N_e potřebných pro nalezení OT_0 metodou opakovaného šifrování*

$$E(N_e) = \frac{1}{|M_K|} \sum_{\forall K} E(N_e|X = K), \tag{5.4}$$

kde X je náhodná veličina vyjadřující výsledek výběru klíče z množiny M_K.

Důkaz. Při hledání OT_0 musí algoritmus provést d_{iK} šifrování, kde d_{iK} je délka kružnice k_{iK}, která je algoritmem procházena. Z toho plyne

$$E(N_e) = \sum_{\forall i,K} P(k_{iK}) d_{iK}, \tag{5.5}$$

Kde $P(k_{iK})$ je pravděpodobnost, že je algoritmem procházena právě kružnice k_{iK}. Tato situace nastává v případě, že počáteční otevřený text OT_0 byl zašifrován klíčem K a $OT_0 \in k_{iK}$. Tedy

$$P(k_{iK}) = P(OT_0 \in k_{iK} \cap X = K). \tag{5.6}$$

Což lze zapsat pomocí podmíněné pravděpodobnosti

$$P(k_{iK}) = P(OT_0 \in k_{iK}|X = K)P(X = K). \tag{5.7}$$

Tabulka 5.3: Srovnání počtů šifrování

$E(N_e)$	Model 1	Model 2	Model 3
Opakované šifrování	32761.75	2043.876	2057.914
Hrubá síla	32768	2048	2048

Po dosazení do vzorce pro střední hodnotu

$$E(N_e) = \sum_{\forall i,K} P(OT_0 \in k_{iK} | X = K) P(X = K) d_{iK}. \tag{5.8}$$

Sumu lze rozdělit na dvě sumy a $P(X = K)$ je možné vyjmout před vnitřní sumu, neboť není závislá na i.

$$E(N_e) = \sum_{\forall K} P(X = K) \sum_{\forall i} P(OT_0 \in k_{iK} | X = K) d_{iK}. \tag{5.9}$$

Protože klíče jsou z M_K vybírány se stejnou pravděpodobností, platí

$$E(N_e) = \sum_{\forall K} \frac{1}{|M_K|} \sum_{\forall i} P(OT_0 \in k_{iK} | X = K) d_{iK}. \tag{5.10}$$

Konstantu lze opět vyjmout před sumu a vnitřní sumu je možné napsat jako střední hodnotu

$$E(N_e) = \frac{1}{|M_K|} \sum_{\forall K} E(N_e | X = K). \tag{5.11}$$

□

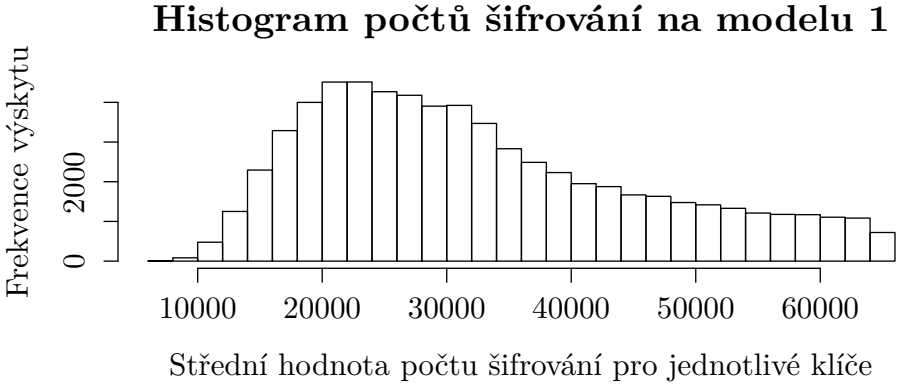
Díky větě 6 je možné spočítat střední hodnoty počtů šifrování pro jednotlivé řádky 5.2 a následně zjistit průměr těchto hodnot. Nyní lze metodu porovnat s hrubou silou, která umožňuje najít OT_0 průměrně v $\frac{|M_{OT}|}{2}$ krocích.

Tabulka 5.3 ukazuje, že průměrně není metoda opakovaného šifrování použitá na redukovaných modelech významně lepší než hrubá síla. Na posledním modelu je dokonce o něco horší. Histogramy 5.2, 5.3 a 5.4 zobrazují frekvenci výskytu různých počtů šifrování pro jednotlivé klíče na každém z modelů.

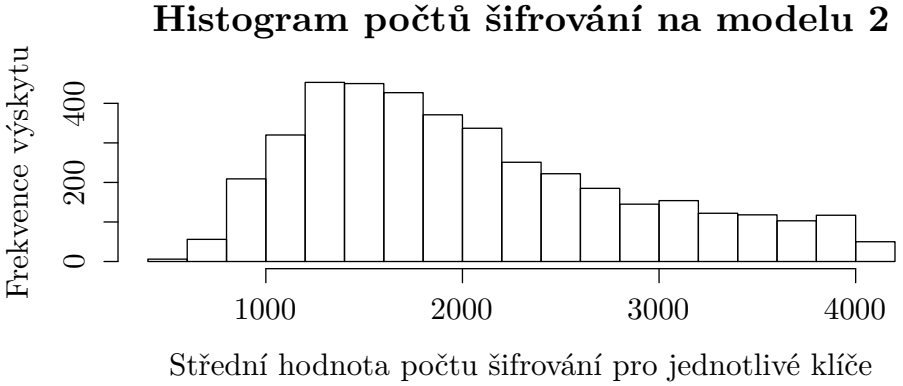
5.2 Návrh úpravy algoritmu

Z naměřených dat vyplynulo, že u velkého množství klíčů zabírá nejdelší kružnice k_0K většinu prostoru šifrových textů. Tím pádem často nastává situace, kdy $d_{0K} > \frac{|M_{OT}|}{2}$, $P(OT_0 \in k_{0K}) > \frac{1}{2}$ a protože $N_e = d_{0K}$, tak $N_e > \frac{|M_{OT}|}{2}$.

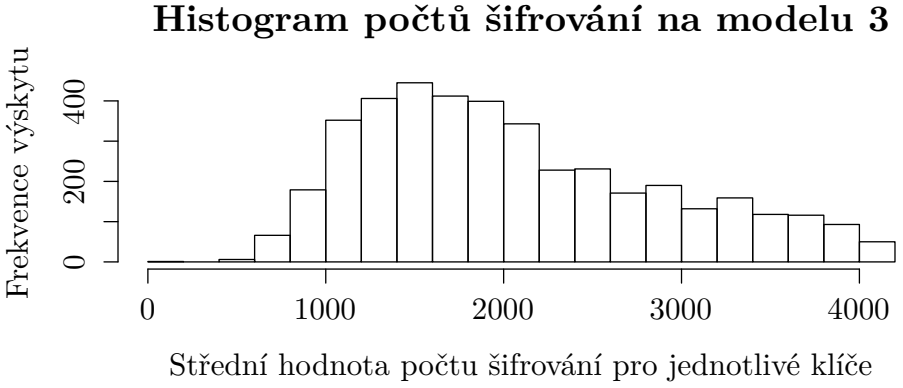
Následující algoritmus se pokouší využít výhody opakovaného šifrování na menších kružnicích, ale zároveň snížit jeho nevýhody na velkých kružnicích tím, že náhodně zvolí několik otevřených textů, pro které provádí opakované šifrování. Cílem je, aby v případě, že OT_0 leží na velké kružnici, byl náhodně zvolený $OT \in M_{OT}$ na této kružnici dále, a tak zkrátil šifrování.



Obrázek 5.2: Histogram středních hodnot počtu šifrování při kryptoanalýze opakovaným šifrování aplikované na model 1.



Obrázek 5.3: Histogram středních hodnot počtu šifrování při kryptoanalýze opakovaným šifrování aplikované na model 2.



Obrázek 5.4: Histogram středních hodnot počtu šifrování při kryptoanalýze opakovaným šifrování aplikované na model 3.

Algoritmus 5.2: Upravená kryptoanalýza opakovaným šifrováním

```
i=0
Napln pole PocatecniOT[pocetNahodnychOT] nahodnymi OT.
PocatecniOT[0]=OT_0
KoncoveŠT[pocetNahodnychOT] =
                  = PocatecniOT[pocetNahodnychOT]
Opakuj:
  Proved step krat, pokud existuje PocatecniOT[i]
    tmp = E(KoncoveŠT[i])
    Pokud tmp = ŠT_0:
      vrat KoncoveŠT[i]
    Pokud tmp = PocatecniOT[i]:
      smaz PocatectniOT[i] a KoncoveŠT[i]
      ukonci cyklus
    Pokud tmp je v PocatecniOT na indexu k:
      smaz Pocatecni[k]
      tmp = KoncoveŠT[k]
      smaz KoncoveŠT[k]
    KoncoveŠT[i] = tmp
  i = (i + 1) mod pocetNahodnychOT
```

Navržený algoritmus je algoritmus 5.2.

Zjevná nevýhoda algoritmu je, že mohou být procházeny i kružnice, na kterých OT_0 neleží. Není snadné matematicky odhadnout, zda v průměrném případě převáží výhoda zkrácení nejdelší kružnice nad nevýhodou procházení nepotřebných kružnic.

Ověření proběhlo experimentálně na různých redukovaných modelech Rijndaelu. Ukázalo se, že nezávisle na zvoleném počtu náhodně volených otevřených textů a velikosti proměnné `step`, není algoritmus významně rychlejší než hrubá síla. Výsledky byly velmi podobné jako u algoritmu 5.1.

5.3 Hledání klíče kryptoanalýzou opakovaným šifrováním

Logická otázka je, zda lze využít opakované šifrování k nalezení šifrového klíče. V případě, že je délka vstupního klíče shodná s velikostí bloku, je možné opakovaně vykonávat $\text{Š}T_{i+1} = E(OT, K = \text{Š}T_i)$. Otevřený text je tentokráte neměnný a naopak se mění klíč šifry.

Lze navrhnout algoritmus podobný jako 5.1.

Problém ale spočívá v tom, že tentokrát není zaručeno, že cyklus skončí, protože může více klíčů vést na jeden šifrový text a mohou se tak tvořit menší

Algoritmus 5.3: Využití opakovaného šifrování pro nalezení klíče

```
i=1
ŠT_1 = E(OT,ŠT_0)
opakuj dokud ŠT_i ≠ ŠT_0{
  i = i + 1
  ŠT_i = E(OT,ŠT_i-1)
}
```

skupiny ŠT, na kterých se bude cyklus neustále opakovat. K algoritmu je třeba přidat mechanizmus, který zaručí ukončení podmínky ve chvíli, kdy se nějaký ŠT zopakuje. Je zřejmé, že algoritmus nemusí vždy hledaný šifrový klíč získat.

Za využití redukovaných modelů je možné zkoumat prostor šifrových textů a tedy i klíčů. Ukazuje se, že obvykle obsahuje několik malých skupin ŠT, do kterých vede mnoho cest (viz obrázek 5.5). Je zřejmé, že ve většině případů algoritmus 5.3 vede na zacyklení a hledaný klíč není nalezen.

Na obrázku 5.5 jsou znázorněny body v prostoru ŠT. Šipkami jsou vyobrazena šifrování se stejným OT a různými klíči. Počátek šipky znázorňuje klíč, který byl na šifrování OT použit a její konec ukazuje na výstupní ŠT. Tučně jsou vyznačena dvě náhodně zvolená šifrování. Z obrázku je patrné, že pokud by pro šifrování byl použit neznámý klíč K_0, nepodařilo by se jej upravenou verzí metody opakovaného šifrování nalézt. Pokud by ale byl použit klíč K_1, byl by metodou rychle nalezen.

5.3.1 Další zjištění

Tato část shrnuje zjištění, která byla učiněna v rámci analýzy získaných dat, ale přesahují zaměření této práce, a proto nebyla dále zkoumána.

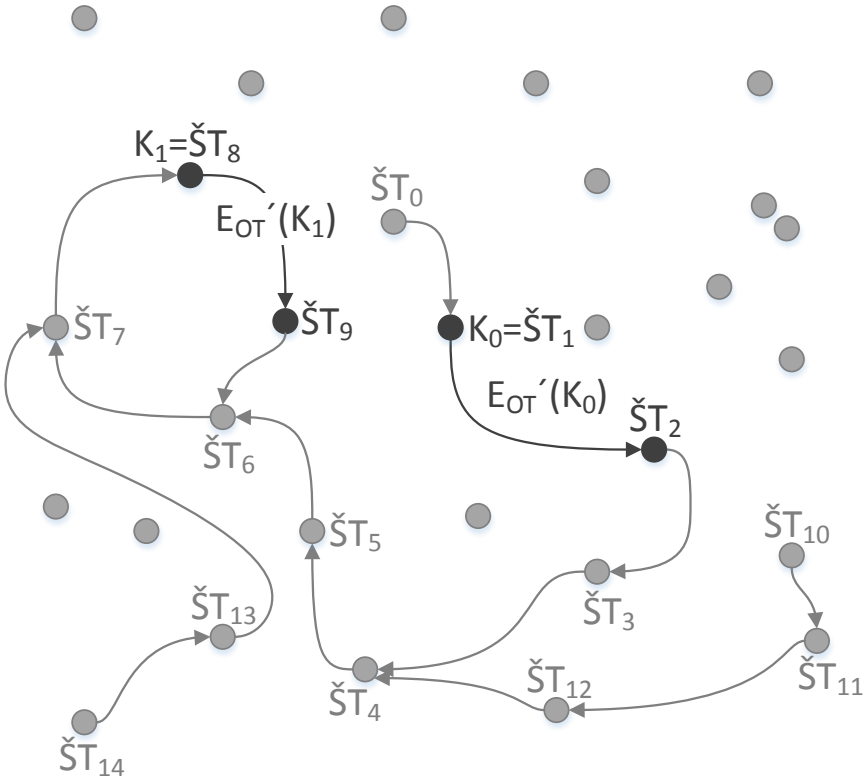
Bylo zjištěno, že počet kružnic v prostoru šifrových textů je pro libovolný zvolený klíč na všech zvolených modelech vždy sudý. Tato vlastnost zřejmě není závislá na velikosti bundlu ani na počtu rund.

Při využití kryptoanalýzy opakovaným šifrováním k nalezení klíče 5.3.1 je vidět, že k otevřenému textu existuje malá množina šifrových klíčů, které jsou vůči této kryptoanalýze velmi náchylné.

5.4 Odhad dopadů výsledků na standardní Rijndael

Protože bylo dosaženo shodných výsledků na několika různě velkých redukovaných modelech Rijndaelu, lze předpokládat, že je možné aplikovat výsledky i na standardní Rijndael.

Tedy při aplikaci algoritmů 5.1 a 5.2 by byl průměrný počet šifrování podobný jako při použití hrubé síly. Při využití algoritmu 5.3 by ve většině



Obrázek 5.5: Kryptoanalýza opakovaným šifrováním upravená pro hledání klíče

případů nebyl klíč nalezen. I ve standardní verzi Rijndaelu existují dvojice otevřeného textu a klíče, kdy lze klíč pomocí opakovaného šifrování najít, ale tato práce nepřináší způsob, jak takové dvojice hledat.

Z výsledků experimentů provedených na redukovaných modelech lze předpokládat, že kryptoanalýzou opakovaného šifrování ani jejími úpravami není významně ohrožena bezpečnost šifry Rijndael.

Kapitola 6

Návrhy na další výzkum

Tato kapitola shrnuje otevřené otázky, které by se mohly stát předmětem navazujících prací.

- Tvorba redukovaných modelů s obdélníkovým stavem. V rámci této práce nebyly navrhovány ani realizovány redukované modely Rijndaelu s jiným, než čtvercovým stavem. V případě, že by se ukázala jako smysluplné tyto modely vytvořit, mohla by vzniknout práce, která by se problematikou zabývala.

- Aplikace kryptoanalýzy opakovaným šifrováním na jiné šifry. Tato práce přináší návrh kryptoanalýzy opakovaným šifrováním. Tato metoda sice není účinná na modely šifry Rijndael, je ale možné, že by mohla být úspěšná při kryptoanalýze jiných šifer.

- Odhalení důvodu pro vznik sudého počtu kružnic. Při aplikaci kryptoanalýzy opakovaným šifrováním vznikají v prostoru šifrových textů pro všechny klíče a všechny testované modely vždy sudé počty kružnic, zjištění důvodu tohoto chování by mohlo být zajímavým výsledkem.

- Zkoumání prostoru klíčů. Při využití metody opakovaného šifrování pro nalezení šifrového klíče bylo zjištěno, že k otevřeným textům existuje malá množina slabých klíčů. Zůstává otázka, zda ke každému klíči existuje otevřený text, vůči kterému je klíč slabý. Dále otázka, zda jsou některé klíče výrazně slabší nebo odolnější vůči této metodě, případně jestli některé otevřené texty umožňují kompromitovat velké množství klíčů. Navazující práce by se mohla zabývat hledáním algoritmu, který by umožnil nalezení otevřených textů, vůči kterým je použitý klíč slabý.

Závěr

V kapitole 1 byly zkoumány vlastnosti šifry Rijndael, kritéria, která doprovázela její vznik a motivace autorů ke zvolení její struktury a jednotlivých operací. Tyto poznatky byly následně využity v kapitole 2 k určení parametrů, kterými je možné měnit velikost modelu Rijndaelu. Tato kapitola také stanovuje pravidla a omezení pro vytváření redukovaných modelů.

Následně byl vytvořen nástroj, který uživateli pomáhá specifikované parametry měnit tak, aby byly dodrženy všechny podmínky kladené na šifru Rijndael a její bezpečnost. Nástroj umožňuje navrhnout širokou škálu redukovaných modelů Rijndaelu, které je poté možné rovnou využít pro šifrování a dešifrování, protože v rámci této práce vznikla také jejich implementace.

V kapitole 5 byla demonstrována využitelnost redukovaných modelů a vytvořených nástrojů. Na nové kryptoanalýze opakovaným šifrováním bylo ukázáno, že redukované modely umožňují provedení experimentů, které by jinak kvůli velikosti standardního Rijndaelu nebyly výpočetně proveditelné. Díky datům získaným z experimentů na několika různých modelech bylo možné odhadnout, že navržená kryptoanalýza opakovaným šifrováním významně neohrožuje bezpečnost šifry Rijndael.

Literatura

[1] Kokeš, J.: *Kryptoanalýza šifry Baby Rijndael*. Diplomová práce, České vysoké učení technické v Praze, Fakulta informačních technologií, Praha, 2013, [cit. 2016-03-24]. Dostupné z: `https://dip.felk.cvut.cz/browse/pdfcache/kokesjo1_2013dipl.pdf`

[2] Bergman, C.: *A Description of Baby Rijndael*. Iowa State University, 21. únor 2005, [cit. 2016-03-24]. Dostupné z: `http://orion.math.iastate.edu/cbergman/crypto/homework/babyr/babyr.pdf`

[3] Daemen, J.; Rijmen, V.: *The Design of Rijndael: AES — the Advanced Encryption Standard*. Information Security and Cryptography, Springer-Verlag Berlin Heidelberg, 2002, ISBN 3-540-42580-2, 238 s.

[4] Daemen, J.; Rijmen, V.: *AES Proposal: Rijndael*. Druhé vydání, 3. září 1999, [cit. 2016-03-24]. Dostupné z: `http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf`

[5] Specification for the Advanced Encryption Standard (AES). Federal Information Processing Standards Publication 197, 2001, [cit. 2016-03-24]. Dostupné z: `http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf`

[6] Lórencz, R.: Symetrická kryptografie. In *Pokročilá kryptologie*, 2013, s. 21–31, [cit. 2016-04-04]. Dostupné z: `https://edux.fit.cvut.cz/archive/B151/MI-KRY/_media/lectures/02/prednaska2.pdf`

[7] Lidl, R.; Niederreiter, H.: *Introduction to finite fields and their applications*. Press Syndicate of the University of Cambridge, 1986, str. 378.

[8] Cui, J.; Huang, L.; Zhong, H.; aj.: An improved AES S-box and its performance analysis. *International Journal of Innovative Computing, Infor-*

mation and Control, ročník 7, č. 5(A), květen 201, ISSN 1349-4198, [cit. 2016-03-25]. Dostupné z: www.ijicic.org/ijicic-10-01041.pdf

[9] Murphy, S.; Robshaw, M. J.: Essential Algebraic Structure Within the AES. In *Advances in Cryptology — CRYPTO*, *Lecture Notes in Computer Science*, ročník 2442, 13. září 2002, s. 1–16, [cit. 2016-03-25]. Dostupné z: http://www.isg.rhul.ac.uk/~sean/crypto.pdf

[10] Jingmei, L.; Baodian, W.; Xinmei, W.: One AES S-box to increase complexity and its cryptanalysis. *Journal of Systems Engineering and Electronics*, ročník 18, 2012: s. 427 – 433.

[11] Jain, R.; Jejurkar, R.; Chopade, S.; aj.: AES Algorithm Using 512 Bit Key Implementation for Secure Communication. *Journal of Systems Engineering and Electronics*, ročník 2, 2014, ISSN 2320-9801, [cit. 2016-03-27]. Dostupné z: http://www.rroij.com/open-access/aes-algorithm-using-512-bit-key-implementationfor-secure-communication.pdf

[12] Moh'd, A.; Jararweh, Y.; Tawalbeh, L.: AES-512: 512-Bit Advanced Encryption Standard algorithm design and evaluation. In *International Conference on Information Assurance and Security*, ročník 7, 5-8. prosinec 2011, [cit. 2016-03-25]. Dostupné z: https://www.researchgate.net/publication/220793242_AES-512_512-Bit_Advanced_Encryption_Standard_algorithm_design_and_evaluation

[13] Grošek, O.; Zajac, P.: Searching for a different AES-class MixColumns operation. In *WSEAS International Conference on Applied Computer Science*, ročník 6, 16-18. prosinec 2006, [cit. 2016-03-25]. Dostupné z: http://www.wseas.us/e-library/conferences/2006tenerife/papers/541-327.pdf

[14] Wikipedia: Model–view–presenter — Wikipedia, The Free Encyclopedia. 2016, [Online; cit. 2016-03-28]. Dostupné z: https://en.wikipedia.org/w/index.php?title=Model%E2%80%93view%E2%80%93presenter&oldid=707149680

[15] Wikipedia: RubyGems — Wikipedia, The Free Encyclopedia. 2015, [Online; cit. 2016-04-18]. Dostupné z: https://en.wikipedia.org/w/index.php?title=RubyGems&oldid=690756633

[16] Kalvoda, T.; Petr, I.: Matematika pro kryptologii. In *Magisterská informatika — Matematika pro kryptologii*, 2014, s. 6–8, [cit. 2016-04-06]. Dostupné z: https://edux.fit.cvut.cz/archive/B142/MI-MKY/_media/lectures/mi-mky-poznamky-v10.pdf

Příloha A

Seznam použitých zkratek a zavedených označení

A.1 Zkratky

AES Advanced Encryption Standard (viz [5])

BÚNO Bez újmy na obecnosti

GF Galois field (viz [3, str. 13])

GUI Graphical user interface

MVP Model-view-presenter (viz [14])

RC Round constant (viz [3, str. 44])

A.2 Zavedená označení

$[x]_K$ Množina prvků $y \in M$, které jsou ekvivalentní s x podle relace dosažitelnost opakovaným šifrováním s klíčem K (viz část 5.1 definice 3).

$c(x)$ Konstantní polynom operace `MixColumns` (viz část 1.4.3).

$f(x)$ Afinní transformace používaná při vytváření S-boxu šifry Rindael nebo redukovaného modelu (viz část 1.4.1).

$g(x)$ Nelineární transformace používaná při vytváření S-boxu šifry Rindael nebo redukovaného modelu (viz část 1.4.1).

$m(x)$ Ireducibilní polynom určující těleso T šifry Rijndael nebo redukovaného modelu (viz část 1.3).

$w(a)$ Počet nenulových bytů (bundlů) v poli bytů (bundlů) a (viz 2.5).

$B(F)$ Hodnota branch number transformace F (viz část 2.5).

$E_K(OT)$ Šifrování $OT \in M$ s klíčem K pomocí šifry Rijndael nebo redukovaného modelu. Synonymum pro $E(OT, K)$ vyjadřující, že K je konstantní (viz část 5.1).

$E(OT, K)$ Šifrování $OT \in M$ s klíčem K pomocí šifry Rijndael nebo redukovaného modelu (viz část 5.1).

$E_K^n(x)$ Složení n šifrování $E_K(x)$, kde $x \in M$ (viz část 5.1 rovnice 5.2).

M Množina všech možných bloků velikosti odpovídající danému redukovanému modelu (viz část 5.1 věta 4).

M_{OT} Množina všech možných bloků otevřených textů daného redukovaného modelu (viz část 5.1 důkaz věty 4).

$M_{ŠT}$ Množina všech možných bloků šifrových textů daného redukovaného modelu (viz část 5.1 důkaz věty 4).

N_b Počet sloupců stavu šifry Rijndael nebo redukovaného modelu (viz část 1.5 rovnice 1.5).

N_{block} Počet bitů bloku šifry Rijndael nebo redukovaného modelu (viz část 2.2 rovnice 2.17).

N_{bundle} Velikost elementu stavu v redukovaném modelu udaná v bitech (viz část 2.1.1).

N_r Počet rund šifry Rijndael nebo redukovaného modelu (viz část 1.2).

N_k Velikost vstupního klíče dělená $N_{rows}N_{bundle}$ (viz část 1.5 rovnice 1.5).

N_{rows} Počet řádků stavu šifry Rijndael nebo redukovaného modelu (viz část 2.1.2.5).

OT Blok otevřeného textu šifry Rijndael nebo redukovaného modelu (viz část 1.2).

OT_0 Tajný OT, který se snaží odhalit kryptoanalýza opakovaným šifrováním (viz část 5.1).

T Těleso polynomů $GF(2^{N_{bundle}})$, kde se násobí modulo $m(x)$ (viz část 2.1.2.1).

ŠT Blok šifrového textu šifry Rijndael nebo redukovaného modelu (viz část 1.2).

ŠT_0 Takový ŠT, který vznikne zašifrováním OT_0 pomocí E_K (viz část 5.1).

W Dvourozměrné pole bytů expandovaného klíče šifry Rijndael (viz část 2.3).

Příloha B

Obsah přiloženého CD

- `README.txt` ..... stručný popis obsahu CD
- `experimenty` ..... adresář s testy kryptoanalýzy opakovaným šifrováním
  - `zprava.pdf` ..... krátká zpráva o provedených experimentech
  - `rijndael_exp` ..... adresář se zdrojovým kódem programu v jazyce C, který využívá knihovnu rijndael_reduced_models. Obsahuje ZIP soubor s exportovaným projektem NetBeans
  - `opakovane_sifrovani` ..... adresář s částí experimentů zabývající se samotnou kryptoanalýzou opakovaným šifrováním
    - `src` ..... adresář se skritpy, které byly využity pro zpracování dat
    - `data.tar.gz` ..... archiv s vygenerovanými daty pro malý model sloužící jako ukázka
- `materialy` . adresář s veřejně dostupnou literatura, z níž tato práce čerpá
- `programy` ..... adresář s Průvodcem pro vytváření redukovaných modelů Rijndaelu a knihovnou rijndael_reduced_models
  - `knihovna` ..... adresář s knihovnou rijndael_reduced_models
    - `src` . adresář se zdrojovým kódem knihovny včetně testů. Obsahuje také ZIP s projektem NetBeans
    - `lib` ..... adresář se statickou knihovnou
    - `include` ..... adresář s externími hlavičkovými soubory knihovny
    - `README.txt` ..... navigace uživatele ke knihovně a hlavičkovým souborům
  - `pruvodce` ... adresář s Průvodcem pro vytváření redukovaných modelů Rijndaelu
    - `src` adresář zdrojových kódů Průvodce ve formátu připraveném pro vytvoření gemu
    - `rrmw-1.0.0.gem` ..... gem s Průvodcem připravený k instalaci
  - `uzivatelska_prirucka` ..... návod na instalaci a použití Průvodce a knihovny ve formátu PDF, který obsahuje vložený soubor ODT, takže jej je možné otevřít v editovatelné podobě
- `text` ..... text diplomové práce
  - `src` ..... adresář se zdrojovou formou práce ve formátu LaTeX
  - `DP_Solil_Lukas.pdf` ..... text práce ve formátu PDF