Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE

Michal Botka

Kryptoanalýza AES

Katedra algebry

Vedoucí diplomové práce: Doc. RNDr. Jiří Tůma, DrSc.
Studijní program: matematické metody informační bezpečnosti

2008

Děkuji svému vedoucímu práce Doc. RNDr. Jiřímu Tůmovi, DrSc. za jeho cenné rady a pomoc při tvorbě této diplomové práce.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

V Praze dne 9.12. 2008 Michal Botka

Obsah

Název práce: Kryptoanalýza AES
Autor: Michal Botka
Katedra (ústav): Katedra algebry
Vedoucí diplomové práce: Doc. RNDr. Jiří Tůma, DrSc.
e-mail vedoucího: tuma@karlin.mff.cuni.cz

Abstrakt: V předložené práci studujeme bezpečnost šifry AES. Zabýváme se možnostmi, jak blokovou šifru a její části matematicky reprezentovat a jak tyto reprezentace využít k algebraickým útokům. Uvádíme přehled známých algoritmů, jež lze k útokům použít. Pozornost věnujeme též možnosti převedení problému řešení soustavy polynomiálních rovnic na SAT problém a vysvětlujeme princip fungování SAT solverů.

Klíčová slova: AES, algebraické útoky, popis S-boxu

Title: Cryptanalysis of AES
Author: Michal Botka
Department: Department of Algebra
Supervisor: Doc. RNDr. Jiří Tůma, DrSc.
Supervisor's e-mail address: tuma@karlin.mff.cuni.cz

Abstract: In the present work we study a security of the AES cipher. We concern in a mathematical representation of a block cipher and how to use it to algebraic attacks. We show a summary of known algorithms which are useful for these attacks. We show how to convert problem of solving the system of polynomial equations to SAT problem and we describe how SAT solvers work.

Keywords: AES, algebraic attacks, S-box description

Kapitola 1

Úvod

Cílem této práce je uvést metody a algoritmy, jež jsou v současnosti zkoumány v souvislosti s možností prolomení šifry AES, a dále se pokusit zhodnotit dopad těchto metod na její bezpečnost.

Předchůdcem AES (Advanced Encryption Standard) je standard DES (Data Encryption Standard), jež byl vyvinut v 70. letech 20. století. Standardizace přispěla k rychlému rozšíření šifry a současně se šifra stala předmětem výzkumu mnoha odborníků po celém světě. V 90. letech byly objeveny nové metody kryptoanalýzy - lineární kryptoanalýza [12] a diferenciální kryptoanalýza [5]. Díky novým objevům v kryptografii a také s přispěním prudkého rozvoje výpočetní techniky se ukázal být DES nakonec nedostatečný a byla proto vyhlášena soutěž o jeho nástupce. V říjnu roku 2000 zvítězila šifra Rijndael, z které vznikl standard AES. Šifra byla navržena tak, aby byla odolná proti všem do té doby známým útokům, zejména již zmiňované lineární a diferenciální kryptoanalýze.

Poměrně novou skupinu útoků představují tzv. algebraické útoky, na které je zaměřena tato práce. Algebraický útok lze charakterizovat jako převedení kryptografického problému na matematický problém, jímž může být typicky problém řešení soustavy polynomiálních rovnic více proměnných nad konečným tělesem, a vyřešení tohoto problému pomocí matematických metod. Matematické výsledky jsou nakonec interpretovány zpět do kryptografie.

Dalšími oblastmi kryptoanalýzy jsou útoky pomocí postranních kanálů a chybové útoky. Princip útoků pomocí postranních kanálů spočívá ve využití fyzikálních veličin, které lze nějakým způsobem získat ze zařízení, jež vykonává danou kryptografickou operaci. Nejedná se tedy o útok na algorit-

mus samotný, nýbrž o útok na jeho implementaci. Měřenými fyzikálními veličinami mohou být například čas, množství elektrické energie nebo intenzita elektromagnetického pole.

Předpokladem chybových útoků je schopnost útočníka zasáhnout v daném kroku do výpočtu kryptografického zařízení a změnit bit nebo množinu bitů v jeho paměti. Po nashromáždění dostatečného množství otevřených a šifrových textů se provede porovnání s texty, u kterých nebyla chyba vyvolána, čímž lze získat informace o klíči. Konkrétní modely chybových útoků na AES lze nalézt například v článku [11].

Útoky pomocí postranních kanálů ani chybovými útoky se nebudeme v této práci zabývat.

Kapitola 2

Advanced Encryption Standard

Advanced Encryption Standard (AES) je 128-bitová bloková šifra definovaná standardem FIPS-197 [14] organizace the National Institute of Standards and Technology (NIST). Podle délky klíče, jež smí být 128, 192 nebo 256 bitů, se šifra označuje zkratkami AES-128, AES-192 nebo AES-256. Šifra se vyznačuje bohatou algebraickou strukturou, díky čemuž je možné ji poměrně snadno implementovat a rovněž i matematicky popsat.

2.1 Základní těleso

Nejprve uvedeme definice a značení základních pojmů, jež budeme používat v následujících kapitolách.

Definice 2.1. *Označme* $m(x) = x^8 + x^4 + x^3 + x + 1$ *irreducibilní polynom nad tělesem* GF(2). *Pomocí tohoto polynomu sestrojíme konečné těleso* $\mathbb{F}$ *jako faktorový okruh* $\mathrm{GF}(2)[x]/m(x)\mathrm{GF}(2)[x]$. *Těleso* $\mathbb{F}$ *nazveme základním tělesem.*

Poznámka 2.2. *Reprezentace základního tělesa je definována specifikací standardu AES. V každé třídě* B *faktorového okruhu* $\mathrm{GF}(2)[x]/m(x)\mathrm{GF}(2)[x]$ *leží právě jeden polynom* $\sum b_k x_k$ *stupně nejvýše* 7. *V dalším textu budeme prvek* $B \in F$ *v případě potřeby reprezentovat tímto jednoznačně určeným polynomem* $\sum b_k x_k$, *případně vektorem jeho koeficientů* $(b_7, b_6, ..., b_0) \in \mathrm{GF}(2)^8$. *Prvky základního tělesa jednoznačně odpovídají osmicím bitů tvořících jeden byte.*

Definice 2.3. *Definujme zobrazení* $\alpha : \mathrm{GF}(2) \to \mathbb{Z}$ *tak, že* $\alpha(0) = 0$ *a* $\alpha(1) = 1$*, a dále definujme*

$$|B| := \sum_{k=0}^{7} \alpha(b_k) 2^k.$$

Nezáporné celé číslo $|B|$ *nazveme norma* B*.*

Poznámka 2.4. *Pomocí normy* $N = |B|$ *budeme stručněji zapisovat* B *jako* $\{\mathtt{N}\}$*, kde užijeme buď binární nebo hexadecimální zápis. Například* $x^6 + x^5 + x + 1 \in \mathbb{F}$ *zapíšeme jako* $\{\mathtt{01100011}\}$ *resp.* $\{\mathtt{63}\}$*. Takto se zapisují hodnoty bytů, kterým odpovídají prvky ze základního tělesa* $\mathbb{F}$*.*

Definice 2.5. *Zobrazení* $\phi : \mathbb{F} \to \mathbb{F}$ *nazveme bit-lineární, jestliže pro každé* $a, b \in \mathbb{F}$ *platí*

$$\phi(a + b) = \phi(a) + \phi(b).$$

Poznámka 2.6. *Aditivní grupa tělesa* $\mathbb{F}$ *spolu s jednoznačně definovaným násobením prvky tělesa* $\mathrm{GF}(2)$ *tvoří vektorový prostor dimenze osm nad* $\mathrm{GF}(2)$*. Zobrazení* ϕ *je bit-lineární, právě když je lineární jako zobrazení na vektorovém prostoru* $\mathrm{GF}(2)^8$*.*

Lemma 2.7. *Nechť* $\phi : \mathbb{F} \to \mathbb{F}$ *je bit-lineární zobrazení. Pak existuje právě jeden polynom* $f(x)$ *tvaru* $\sum_{k=0}^{7} a_k x^{2^k}$*, kde* $a_0, \ldots, a_7 \in \mathbb{F}$*, pro který platí, že pro všechna* $a \in \mathbb{F}$ *je* $\phi(a) = f(a)$*.*

Důkaz. Označme $\mathcal{L}$ množinu všech bit-lineárních zobrazení a $\mathcal{P}$ množinu všech polynomů požadovaného tvaru. Nechť $f(x) = \sum_{k=0}^{7} a_k x^{2^k} \in \mathcal{P}$. Pak pro každé $x, y \in \mathbb{F}$ platí $f(x+y) = \sum_{k=0}^{7} a_k (x+y)^{2^k} = \sum_{k=0}^{7} a_k (x^{2^k} + y^{2^k}) = f(x) + f(y)$ a tedy $\mathcal{P} \subseteq \mathcal{L}$. $\mathbb{F}$ s operací sčítání tvoří abelovskou grupu generovanou prvky $1, x, x^2, \ldots, x^7$. Každé bit-lineární zobrazení je jednoznačně určeno hodnotami na generátorech. Z rovnosti $|\mathcal{L}| = |\mathcal{P}| = |\mathbb{F}|^8$ plyne dokazované. □

2.2 Struktura AES

V následující sekci popíšeme základní komponenty šifry a nakonec šifru samotnou.

Definice 2.8. *Nechť* $\mathbb{F}$ *je základní těleso. Matici* $S = (s_{i,j})$ *typu (4,4) nad* $\mathbb{F}$ *nazveme stavová matice, prostor všech těchto matic budeme značit* Ω*.*

Poznámka 2.9. *Šifra AES pracuje s bloky délky 128 bitů. Každý blok rozdělíme na 16 po sobě následujících bytů a ty pak postupně po řádcích napíšeme do matice o čtyřech řádcích a čtyřech sloupcích, čímž dostaneme reprezentaci bloku šifry pomocí stavové matice.*

Definice 2.10. *Písmenem σ označíme prosté zobrazení na $\mathbb{F}$, které je definováno jako $\sigma(a) = \phi(\psi(a))$ pro $a \in \mathbb{F}$, kde*

1. *$\psi(0) = 0$ a $\psi(a) = a^{-1}$ pro $a \in \mathbb{F}^*$*
2. *$\phi(a) = b$, kde $a = a_0 + a_1x + \ldots + a_7x^7$, $b = b_0 + b_1x + \ldots + b_7x^7$, $a_i, b_i \in \mathrm{GF}(2)$ pro $i = 0, 1, \ldots, 7$ a platí:*

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

Definice 2.11. *Nechť $X = (x_{i,j}) \in \Omega$, $Y = (y_{i,j}) \in \Omega$ a $K = (k_{i,j}) \in \Omega$. Transformace SubBytes substituuje každý byte pomocí zobrazení σ.*

$$SubBytes : \begin{pmatrix} x_{0,0} & x_{0,1} & x_{0,2} & x_{0,3} \\ x_{1,0} & x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,0} & x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,0} & x_{3,1} & x_{3,2} & x_{3,3} \end{pmatrix} \mapsto \begin{pmatrix} \sigma(x_{0,0}) & \sigma(x_{0,1}) & \sigma(x_{0,2}) & \sigma(x_{0,3}) \\ \sigma(x_{1,0}) & \sigma(x_{1,1}) & \sigma(x_{1,2}) & \sigma(x_{1,3}) \\ \sigma(x_{2,0}) & \sigma(x_{2,1}) & \sigma(x_{2,2}) & \sigma(x_{2,3}) \\ \sigma(x_{3,0}) & \sigma(x_{3,1}) & \sigma(x_{3,2}) & \sigma(x_{3,3}) \end{pmatrix}$$

Transformace $ShiftRows$ cyklicky posune řádky.

$$ShiftRows : \begin{pmatrix} x_{0,0} & x_{0,1} & x_{0,2} & x_{0,3} \\ x_{1,0} & x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,0} & x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,0} & x_{3,1} & x_{3,2} & x_{3,3} \end{pmatrix} \mapsto \begin{pmatrix} x_{0,0} & x_{0,1} & x_{0,2} & x_{0,3} \\ x_{1,1} & x_{1,2} & x_{1,3} & x_{1,0} \\ x_{2,2} & x_{2,3} & x_{2,0} & x_{2,1} \\ x_{3,3} & x_{3,0} & x_{3,1} & x_{3,2} \end{pmatrix}$$

Transformace MixColumns vynásobí každý sloupec MDS maticí.

$$MixColumns : \begin{pmatrix} x_{0,0} & x_{0,1} & x_{0,2} & x_{0,3} \\ x_{1,0} & x_{1,1} & x_{1,2} & x_{1,3} \\ x_{2,0} & x_{2,1} & x_{2,2} & x_{2,3} \\ x_{3,0} & x_{3,1} & x_{3,2} & x_{3,3} \end{pmatrix} \mapsto \begin{pmatrix} y_{0,0} & y_{0,1} & y_{0,2} & y_{0,3} \\ y_{1,0} & y_{1,1} & y_{1,2} & y_{1,3} \\ y_{2,0} & y_{2,1} & y_{2,2} & y_{2,3} \\ y_{3,0} & y_{3,1} & y_{3,2} & y_{3,3} \end{pmatrix}, \text{ kde}$$

$$\begin{pmatrix} y_{0,j} \\ y_{1,j} \\ y_{2,j} \\ y_{3,j} \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \begin{pmatrix} x_{0,j} \\ x_{1,j} \\ x_{2,j} \\ x_{3,j} \end{pmatrix}, \text{ pro } j = 1, 2, 3, 4.$$

Transformace AddRoundKey přičte ke stavové matici rundovní klíč K.

$AddRoundKey : (X, K) \mapsto X + K$

Složením několika transformací získáme operace, jež se nazývají rundy. Počet rund je určen standardem dle délky klíče. AES-128 má 10 rund, AES-192 12 rund a AES-256 14 rund. Do každé rundy vstupuje kromě stavové matice rundovní klíč, jež je odvozen z šifrového klíče pomocí algoritmu nazývaného *expanze klíče*.

Definice 2.12. *Nechť n je počet rund šifry. Označme $K_0, K_1, \ldots, K_n \in \Omega$ rundovní klíče, $R_0, R_1, \ldots, R_n$ rundovní funkce, $P \in \Omega$ blok otevřeného a $C \in \Omega$ blok šifrového textu. Šifrování bloku zprávy je pak definována takto:*

1. $M_0 = P$,
2. $M_{i+1} = R_i(M_i, K_i)$ *pro* $i = 0, \ldots, n$,
3. $C = M_{n+1}$.

Rundovní funkce se skládají z transformací $SubBytes$, $ShiftRows$, $MixColumns$ a $AddRoundKey$:

1. $R_0(M, K) = AddRoundKey(M, K)$,
2. $R_i(M, K) = AddRoundKey(MixColumns(ShiftRows(SubBytes(M))), K)$ *pro* $i = 1, \ldots, n-1$,
3. $R_n(M, K) = AddRoundKey(ShiftRows(SubBytes(M)), K)$,

pro každé $M, K \in \Omega$.

Na začátku této kapitoly jsme uvedli, že definované délky klíče u AES jsou 128, 192 a 256 bitů. Pro následující účely nebudeme uvádět délku klíče v bitech, nýbrž v 32-bitových slovech, což budeme značit N_k. Přípustné délky klíče vyjádřené tímto parametrem jsou tedy $4, 6$ a 8.

Definice 2.13. *Nechť* $N_k = 4, 6$ *nebo* 8. *Vektoru* $E = (e_0, e_1, \ldots, e_{4N_k-1}) \in \mathbb{F}^{4N_k}$ *budeme říkat šifrový klíč. Nechť* r *je počet rund a* $K_n = (k_{i,j}^{(n)}) \in \Omega$, *kde* $n = 0, 1, \ldots, r$, *jsou rundovní klíče. Expanze klíče, tj. algoritmus, jež z šifrového klíče vytvoří rundovní klíče, je definován následovně. Označme sloupce matic* $K_0, K_1, \ldots, K_r$ *po řadě* $w_0, w_1, \ldots, w_{4(r+1)}$. *Pro* $i < N_k$ *je* w_i *definováno takto:*

$$w_i = (e_{4i}, e_{4i+1}, e_{4i+2}, e_{4i+3})$$

Pro $N_k \leq i < 4(r+1)$ *je definice rekurentní:*

$$w_i = w_{i-N_k} + t_{i-1},$$

kde

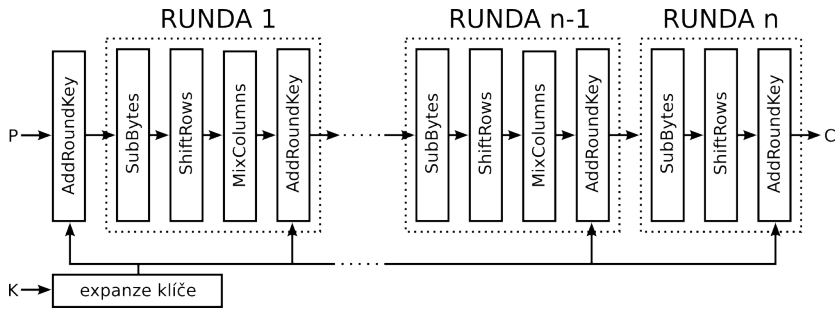
1. $t_{i-1} = SubWord(RotWord(w_{i-1})) + Rcon_(i/N_k)$, *pro* $i \equiv 0(modN_k)$,
2. $t_{i-1} = SubWord(w_{i-1})$, *pro* $N_k > 6$ *a* $i \equiv 4(modN_k)$,
3. $t_{i-1} = w_{i-1}$ *jinak.*

$SubWord$ *a* $RotWord$ *jsou zobrazení na* $\mathbb{F}^4$ *definované pro* $(a_0, a_1, a_2, a_3) \in \mathbb{F}^4$ *takto:*

$$\begin{aligned} SubWord(a_0, a_1, a_2, a_3) &= (\sigma(a_0), \sigma(a_1), \sigma(a_2), \sigma(a_3)) \\ RotWord(a_0, a_1, a_2, a_3) &= (a_1, a_2, a_3, a_4). \end{aligned}$$

$Rcon(k)$ *je definováno jako* $(x^{[k]-1}, 00, 00, 00)$, *kde* $[.]$ *je dolní celá část.*

Na obr. 2.1 je znázorněna struktura AES.



Obr. 2.1

2.3 CTC Cipher

Šifru CTC (Courtois Toy Cipher) navrhl Nicolas T. Courtois v článku [7] z roku 2006 jako blokovou šifru vhodnou pro výzkum a testování algebraických útoků. Šifra má volitelný počet rund i velikost bloku, který je ale vždy násobkem 3 bitů. S-box je definován jako náhodná nelineární permutace na množině $\{0,1\}^3$. Šifra je odolná proti lineární a diferenciální kryptoanalýze a obdobně jako v případě AES se každá runda skládá ze tří částí, kterými jsou substituční a difúzní vrstva a přičtení rundovního klíče.

Pro následující popis budeme používat značení z výše uvedeného článku. Označíme-li B počet S-boxů, které se v jedné rundě vyskytují, a s jejich velikost, pak velikost bloku je Bs. Počet rund označme N_r. Pro $i = 1, 2, \ldots, N_r$ a $j = 0, 1, \ldots, Bs - 1$ dále označme $X_{i,j}$ j-tý bit, který vstupuje do i-té rundy, $Y_{i,j}$ j-tý bit, který vystupuje v i-té rundě ze substituční vrstvy, a $Z_{i,j}$ j-tý bit, který vystupuje z i-té rundy před přičtením rundovního klíče. Otevřený text budeme značit Z_0 a šifrový text X_{N_r+1}.

Označíme-li K_0 šifrový klíč, rundovní klíče $K_0, K_1, \ldots, K_{N_r}$ jsou z něho odvozeny vztahem:

$$K_{i,j} = K_{0,(i+1) \bmod Bs},$$

kde druhý index značí příslušný bit daného klíče.

Substituční vrstva rozdělí vstup X_i po trojicích bitů a na každou tuto trojici aplikuje S-box S.

$$(Y_{i,3k}, Y_{i,3k+1}, Y_{i,3k+2}) = S(X_{i,3k}, X_{i,3k+1}, X_{i,3k+1}),$$

pro $i = 1, 2, \ldots N_r$ a $k = 0, 1, \ldots, B - 1$.

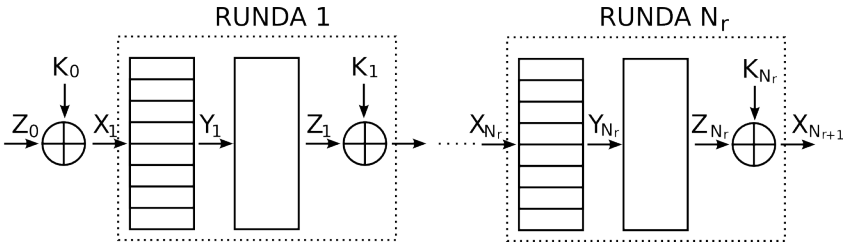
Difúzní vrstva je definována vztahy:

$$\begin{aligned} Z_{i,257\ mod\ Bs} &= Y_{i,0} \text{ pro } i = 1, 2, \ldots, N_r, \\ Z_{i,1987j+257\ mod\ Bs} &= Y_{i,0} + Y_{i,j+137 \bmod Bs} \text{ pro } j \neq 0 \end{aligned}$$

Poslední operací v rundě je přičtení klíče. Přičtení klíče navíc předchází první rundu.

$$X_{i+1,j} = Z_{i,j} + K_{i,j} \text{ pro } i = 0, 1, \ldots, N_r \text{ a } j = 0, 1, \ldots, Bs - 1,$$

Struktura CTC je shrnuta na obr. 2.2.



Obr. 2.2

Kapitola 3

Algebraický popis šifry

V této kapitole ukážeme, jak algebraicky reprezentovat blokovou šifru.

3.1 Algebraický popis S-boxu

Definice 3.1. *Nechť $n \in \mathbb{N}$, $x = (x_1, x_2, \ldots, x_n)$, $y = (y_1, y_2, \ldots, y_n) \in \{0,1\}^n$. Na množině $\{0,1\}^n$ definujme uspořádání $\ll$ takto:*

$$x \ll y \iff 1 + \sum_{k=1}^{n} x_k 2^{k-1} < 1 + \sum_{k=1}^{n} y_k 2^{k-1}$$

Definice 3.2. *Nechť $n \in \mathbb{N}$. Bijektivní zobrazení $\sigma : \{0,1\}^n \to \{0,1\}^n$ nazveme S-box, číslu n budeme říkat velikost S-boxu. Pro S-box σ dále definujme množinu $S(\sigma) \subset \{0,1\}^{2n}$ takto:*

$$S(\sigma) = \{(x_1, \ldots, x_n, y_1, \ldots, y_n) \in \{0,1\}^{2n} |\ \sigma(x_1, \ldots, x_n) = (y_1, \ldots, y_n)\}$$

Množina $S(\sigma)$ má právě $N = 2^n$ prvků, které uspořádáme do posloupnosti $(s_k)_{k=1}^{N}$ tak, aby platilo $s_i \ll s_j$ pro každé $i < j$.

Definice 3.3. *Nechť T je konečná množina termů v proměnných $x_1, \ldots, x_n$, $y_1, \ldots, y_n$ pro nějaké $n \in \mathbb{N}$, jež obsahuje termy $1, x_1, \ldots, x_n, y_1, \ldots, y_n$. Vektorový prostor nad termy T budeme značit $V(T)$ a definujeme ho jako prostor všech polynomů tvaru $\sum_{z \in T} a_z z$ pro nějaká $a_z \in \mathrm{GF}(2)$. Pro libovolný polynom f z $V(T)$ definujme množinu $N(f) \subset \{0,1\}^{2n}$ takto:*

$$N(f) = \{(x_1, \ldots, x_n, y_1, \ldots, y_n) \in \{0,1\}^{2n} |\ f(x_1, \ldots, x_n, y_1, \ldots, y_n) = 0\}$$

Pro libovolnou množinu $W \subset V(T)$ definujme $N(W) = \cap_{f \in W} N(f)$.

Poznámka 3.4. *Dimenze vektorového prostoru $V(T)$ je rovna počtu termů $t = |T|$. Bázi, jež tvoří monomy s termy z T a jednotkovými koeficienty, budeme říkat kanonická báze.*

Lemma 3.5. *Nechť $V(T)$ je vektorový prostor v termech T, W je podprostor $V(T)$ dimenze m a $g_1, g_2, \ldots, g_m$ je báze W. Potom $N(W) = \cap_{k=1,\ldots,m} N(g_k)$.*

Důkaz. Zřejmě platí $N(W) \subset \cap_{k=1,\ldots,m} N(g_k)$. Je-li $a \in \cap_{k=1,\ldots,m} N(g_k)$, pak $g_k(a) = 0$ pro všechna $k \leq m$. Zvolme $f = \sum_{k=1}^{m} c_k g_k \in W$, $c_k \in \mathrm{GF}(2)$. Platí $f(a) = \sum_{k=1}^{m} c_k g_k(a) = 0$ a tedy $a \in N(W)$. □

Lemma 3.6. *Nechť $V(T)$ je vektorový prostor v termech T, $R \subset \{0,1\}^{2n}$ a $W \subset V(T)$ je maximální množina taková, že $N(W) = R$. Potom W je vektorový prostor.*

Důkaz. Buď $f, g \in W$. Z rovnosti $N(W) = R$ plyne, že pro každé $r \in R$ je $f(r) = g(r) = 0$ a tedy $(f + g)(r) = 0$. Protože W je maximální, je $f + g \in W$. □

Lemma 3.7. *Nechť $V(T)$ je vektorový prostor v termech T, $f, g \in V(T)$. Pak platí $N(f + g) = (N(f) \cap N(g)) \cup (V(T) - (N(f) \cup N(g)))$.*

Důkaz. Polynom $f + g$ je roven nule právě v těch bodech prostoru $\{0,1\}^{2n}$, kde jsou oba polynomy f i g současně nulové nebo nenulové. □

Naším cílem je nalézt pro S-box σ a množinu termů T v proměnných x_1, x_2, …, x_n, y_1, y_2, … y_n takový podprostor $W \subset V(T)$, aby $S(\sigma) \subset N(W)$ resp. $S(\sigma) = N(W)$. Podmínku $S(\sigma) \subset N(W)$ splňuje například triviální prostor W, neboť $N(0) = \{0,1\}^{2n}$. V následujím textu dokážeme, že prostor W, který zkonstruujeme, má maximální dimenzi.

Nechť $t = |T|$, n je velikost S-boxu σ, $N = 2^n$ a $(s_k)_{k=1}^{N}$ jsou uspořádané prvky množiny $S(\sigma)$. Nechť polynomy $f_1, f_2, \ldots, f_t$ tvoří kanonickou bázi vektorového prostoru $V(T)$. Definujme matici $A = (a_{i,j})$ typu $(t, N + t)$ nad GF(2) takto:

$$a_{i,j} = \begin{cases} f_i(s_j) & \text{je-li } j \leq N \\ \delta_{i,j-N} & \text{je-li } j > N \end{cases},$$

kde δ je Kroneckerovo delta. Dříve než přistoupíme k algoritmu, uvedeme následující lemma.

Lemma 3.8. *Nechť $B = (b_{i,j})$ a $C = (c_{i,j})$ jsou matice typu $(t, N + t)$ nad* GF(2) *takové, že C lze dostat z B pomocí elementární řádkové úpravy. Nechť $g_1, g_2, \ldots, g_t$ jsou polynomy takové, že pro každé $i \leq t$ platí:*

$$\begin{aligned} g_i(s_j) &= b_{i,j} \text{ pro } \ j \leq N, \\ g_i &= \sum_{k=1}^{t} b_{i,k+N} f_k. \end{aligned}$$

Pro $i \leq t$ dále definujme polynomy h_i takto:

$$h_i = \sum_{k=1}^{t} c_{i,k+N} f_k.$$

Pak pro každé $i \leq t$ a $j \leq N$ platí:

$$h_i(s_j) = c_{i,j}.$$

Důkaz. Nechť pro nějaké $i \leq t$ je

$$c_{i,j} = \sum_{m=1, m\neq i}^{t} r_m b_{m,j},$$

kde $j = 1, 2, \ldots, N + t$ a $r_m \in$ GF(2). Potom platí:

$$\begin{aligned} h_i(s_j) &= \left(\sum_{k=1}^{t} c_{i,k+N} f_k \right)(s_j) \\ &= \left(\sum_{k=1}^{t} \left(\sum_{m=1, m\neq 1}^{t} r_m b_{m,k+N} \right) f_k \right)(s_j) \\ &= \left(\sum_{m=1, m\neq 1}^{t} r_m \left(\sum_{k=1}^{t} b_{m,k+N} f_k \right) \right)(s_j) \\ &= \left(\sum_{m=1, m\neq 1}^{t} r_m g_m \right)(s_j) \\ &= \sum_{m=1, m\neq 1}^{t} r_m g_m(s_j) \\ &= \sum_{m=1, m\neq 1}^{t} r_m b_{m,j} = c_{i,j} \end{aligned}$$

□

Algoritmus 3.9. *Vstupem algoritmu je matice A. Pomocí posloupnosti elementárních řádkových úprav ji upravíme tak, aby ve výsledné matici B byla podmatice tvořena prvními N sloupci v řádkově odstupňovaném tvaru.*

Pro $1 \leq i \leq t$ definujme polynomy $g_i = \sum_{k=1}^{t} b_{i,k+N} f_k$. Podle lemmatu 3.8 platí, že $b_{i,j} = g_i(s_j)$ pro $i \leq t$ a $j \leq N$. Definujme W jako prostor generovaný všemi g_i, pro něž je pro každé $j \leq N$ $g_i(s_j) = 0$.

Lemma 3.10. *Pro podprostor W zkonstruovaný výše platí: $S(\sigma) \subset N(W)$.*

Důkaz. Buď $g_1, \ldots, g_b$ báze W z algoritmu 3.9 a $a \in S(\sigma)$. Podle lemmatu 3.5 stačí ukázat, že $g_i(a) = 0$ pro všechna $i \leq b$. Rovnost platí z definice g_i a skutečnosti, že $a = s_k$ pro nějaké k. □

Lemma 3.11. *Nechť pro $f \in V(T)$ platí, že $S(\sigma) \subset N(f)$, potom $f \in W$.*

Důkaz. Buď $g_1, \ldots, g_b$ báze W z algoritmu 3.9 a $g_{b+1}, \ldots, g_t$ takové prvky, že $g_1, \ldots, g_t$ je báze $V(T)$. Vezměme prvních N sloupců matice B, jež je výsledkem algoritmu 3.9, a vyberme z ní všechny nenulové řádky. Tato matice má právě $t-b$ řádků a její řádková hodnost je rovněž $t-b$ díky řádkově odstupňovaném tvaru. Pomocí elementárních řádkových úprav lze tuto matici převést na matici C, kterou definujeme takto:

$$C = \begin{pmatrix} g_{b+1}(s_1) & g_{b+1}(s_2) & \ldots & g_{b+1}(s_N) \\ g_{b+2}(s_1) & g_{b+2}(s_2) & \ldots & g_{b+2}(s_N) \\ \vdots & \vdots & \ddots & \vdots \\ g_t(s_1) & g_t(s_2) & \ldots & g_t(s_N) \end{pmatrix}.$$

Nechť $f = \sum_{k=1}^{t} c_k g_k$. Kdyby pro nějaké $k > b$ bylo $c_k \neq 0$, byl by vektor $(f(s_1), f(s_2), \ldots, f(s_N))$ nenulový, což je ve sporu s $S(\sigma) \subset N(f)$. □

Lemma 3.12. *Nechť σ je S-box velikosti n a $V(T)$ prostor polynomů, dimenze t. Pak existuje podprostor $W \subseteq V(T)$ dimenze $\geq t - 2^n$, pro který platí $S(\sigma) \subset N(W)$.*

Důkaz. Konstrukci prostoru W dává algoritmus 3.9. Dimenze je rovna počtu řádků matice B, pro něž je prvních 2^n sloupců nulových. □

Poznámka 3.13. *Ukazuje se, že odhad dimenze je velmi pesimistický. V případě S-boxu AES máme $n = 8$. Uvažujeme-li pouze termy stupně nejvýše 2, je $t = 81$. Předchozí lemma nám tedy negarantuje, že vůbec nějaká rovnice existuje. Podle článku* [4] *je těchto rovnic* 23.

Definice 3.14. *Nechť $n \in \mathbb{N}$. Množinu všech S-boxů velikosti n budeme značit $\mathcal{S}$. Nechť $\alpha_i : \mathrm{GF}(2)^n \to \mathrm{GF}(2)$ jsou booleovské funkce, kde $i = 1, 2, \ldots n$. Řekneme, že S-box $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathcal{S}$ je afinní, jestliže existuje regulární matice $A = (a_{i,j})$ nad $\mathrm{GF}(2)$ řádu n a vektor $B = (b_1, b_2, \ldots, b_n) \in \mathrm{GF}(2)^n$ takový, že:*

$$\begin{pmatrix} \alpha_1(x_1, x_2, \ldots, x_n) \\ \alpha_2(x_1, x_2, \ldots, x_n) \\ \vdots \\ \alpha_n(x_1, x_2, \ldots, x_n) \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \ldots & a_{1,n} \\ a_{2,1} & a_{2,2} & \ldots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \ldots & a_{n,n} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Množinu všech afinních S-boxů budeme značit $\mathcal{A}$.

Definice 3.15. *Na množině $\mathcal{S}$ definujme relaci $\sim$ předpisem:*

$$\sigma \sim \tau \iff \exists \alpha \in \mathcal{A} \;\; \tau = \alpha \circ \sigma$$

Lemma 3.16. *Relace $\sim$ na množině $\mathcal{S}$ je ekvivalence.*

Důkaz. Nechť $\sigma, \tau, \rho \in \mathcal{S}$. Zřejmě $\sigma \sim \sigma$, neboť identický S-box je afinní. Je-li $\sigma \sim \tau$, pak existuje $\alpha \in \mathcal{A}$ takové, že $\tau = \alpha \circ \sigma$, odkud plyne $\sigma = \alpha^{-1} \circ \tau$ a $\tau \sim \sigma$. Je-li $\sigma \sim \tau$ a $\tau \sim \rho$, pak existují $\alpha, \beta \in \mathcal{A}$ takové, že $\tau = \alpha \circ \sigma$ a $\rho = \beta \circ \tau$. Dosazením první rovnice do druhé získáme $\rho = \beta \circ \alpha \circ \sigma$, z čehož plyne $\sigma \sim \rho$, neboť $\beta \circ \alpha$ je afinní. □

Definice 3.17. *Nechť $\sigma \in \mathcal{S}$, T je množina termů a $f_1 = 1, f_2 = x_1, \ldots, f_{n+1} = x_n, f_{n+1} = y_1, \ldots, f_{2n+1} = y_n, f_{2n+2}, \ldots, f_t$ je kanonická báze vektorového prostoru $V(T)$. Pro $k \leq t$ definujme:*

$$u_k = (f_k(s_1), f_k(s_2), \ldots, f_k(s_N)),$$

kde $(s_k)_{k=1}^N$ jsou uspořádané prvky množiny $S(\sigma)$. Definujme čísla $d(\sigma)$ a $D(\sigma)$ takto:

$$\begin{aligned} d(\sigma) &= dim < u_1, u_2, \ldots, u_{2n+1} > \\ D(\sigma) &= dim < u_1, u_2, \ldots, u_t > \end{aligned}$$

Věta 3.18. *Nechť $\sigma, \tau \in \mathcal{S}$. Jestliže $\sigma \sim \tau$, potom*

(a) $d(\sigma) = d(\tau)$

(b) $D(\sigma) = D(\tau)$

Důkaz. Z $\sigma \sim \tau$ plyne, že existuje $\alpha \in \mathcal{A}$ takové, že $\tau(x_1, x_2, \ldots, x_n) = \alpha(\sigma(x_1, x_2, \ldots, x_n))$ pro každé $(x_1, x_2, \ldots, x_n) \in \{0,1\}^n$. Označme $A = (a_{i,j})$ matici příslušnou afinnímu zobrazení α a B vektor z definice 3.13 a nechť u_k a v_k jsou vektory z definice 3.16 příslušné po řadě S-boxům σ a τ. Platí:

$$\begin{aligned}
v_1 &= u_1 \\
v_2 &= u_2 \\
&\vdots \\
v_{n+1} &= u_{n+1} \\
v_{n+2} &= b_1 u_1 + a_{1,1} u_{n+2} + a_{1,2} u_{n+3} + \ldots + a_{1,n} u_{2n+1} \\
v_{n+3} &= b_2 u_1 + a_{2,1} u_{n+2} + a_{2,2} u_{n+3} + \ldots + a_{2,n} u_{2n+1} \\
&\vdots \\
v_{2n+1} &= b_n u_1 + a_{n,1} u_{n+2} + a_{n,2} u_{n+3} + \ldots + a_{n,n} u_{2n+1}
\end{aligned}$$

Odsud vidíme, že elementárními řádkovými úpravami lze převést posloupnost vektorů $u_1, u_2, \ldots, u_{2n+1}$ na $v_1, v_2, \ldots, v_{2n+1}$. Vektory generují týž vektorový prostor a dimenze $d(\sigma)$ a $d(\tau)$ se rovnají.

Analogický argument použijeme pro dimenze $D(\sigma)$ a $D(\tau)$. □

3.2 Algebraický popis S-boxu velikosti 3

V následujím textu budeme předpokládat, že kanonická báze prostoru polynomů obsahuje právě tyto polynomy: $f_1 = 1$, $f_2 = x_1$, $f_3 = x_2$, $f_4 = x_3$, $f_5 = y_1$, $f_6 = y_2$, $f_7 = y_3$, $f_8 = x_1y_1$, $f_9 = x_1y_2$, $f_{10} = x_1y_3$, $f_{11} = x_2y_1$, $f_{12} = x_2y_2$, $f_{13} = x_2y_3$, $f_{14} = x_3y_1$, $f_{15} = x_3y_2$, $f_{16} = x_3y_3$.

Tvrzení 3.19. *Nechť σ je afinní S-box velikosti* 3. *Potom $d(\alpha) = 4$ a $D(\alpha) = 7$.*

Důkaz. Volme $\alpha = \sigma^{-1}$ a označme I identitu. Zřejmě platí $I = \alpha \circ \sigma$ a tedy $I \sim \sigma$. Přímým výpočtem ověříme, že $d(I) = 4$ a $D(I) = 7$. Podle předchozí věty platí z $I \sim \sigma$ rovnost $d(\sigma) = d(I)$ a $D(\sigma) = D(I)$. □

Tvrzení 3.20. *Nechť σ je S-box velikosti* 3, *jež není afinní. Potom $D(\sigma) = 8$.*

Důkaz. Označme prvky kanonické báze $f_1, f_2, \ldots, f_{16}$ a pro $k \leq t$ definujme:

$$u_k = (f_k(s_1), f_k(s_2), \ldots, f_k(s_N)),$$

kde $(s_k)_{k=1}^N$ jsou uspořádané prvky množiny $S(\sigma)$.

Protože σ není afinní, je alespoň jeden z vektorů u_5, u_6, u_7 lineárně nezávislý na u_1, u_2, u_3, u_4. Bez újmy na obecnosti nechť je to u_5, tj. vektor hodnot polynomu y_1. Podívejme se nyní podrobněji na matici

$$\begin{pmatrix} u_5 \\ u_6 \\ u_7 \end{pmatrix}.$$

Matice má 8 navzájem různých sloupců, které odpovídají hodnotám S-boxu na vstupech z $\{0,1\}^3$. Složením S-boxu σ s vhodným afinním zobrazením lze převést tuto matici na následující tvar a podle věty 3.18 se dimenze $D(\sigma)$ zachová.

$$\begin{pmatrix} 0 & 1 & 0 & 0 & \ldots \\ 0 & 0 & 1 & 0 & \ldots \\ 0 & 0 & 0 & 1 & \ldots \end{pmatrix}.$$

Fixací prvních čtyř sloupců vektoru u_5 zbývají pouze čtyři možnosti, jak může vektor u_5 vypadat. Přidáme-li pro každý tento případ k vektoru u_5 vektory u_1, u_2, u_3, u_4 a vektory u_8, u_{11}, u_{14}, lze experimentálně ověřit, že dimenze vektorového prostoru jimi generovaného je 8. □

3.3 S-box AES

S-box šifry AES, který vystupuje v operaci *SubBytes*, je bijektivní zobrazení na množině $\{0,2)^8$. Pro algoritmus 3.7 zvolme tyto polynomy: $f_1 = 1$, $f_2 = x_1$, $\ldots$, $f_9 = x_8$, $f_{10} = y_1$, $\ldots$, $f_{17} = y_8$, $f_{18} = x_1y_1$, $\ldots$, $f_{25} = x_1y_8$, $f_{26} = x_2y_1$, $\ldots$, $f_{81} = x_8y_8$.

S využitím počítače bylo zjištěno, že algoritmus vrátí 23 rovnic. Konkrétní podoba těchto rovnic je k dispozici v příloze na konci této práce. Označíme-li S-box σ, pak $d(\sigma) = 17$ a $D(\sigma) = 58$.

Kapitola 4

Algoritmy pro řešení soustav polynomiálních rovnic

Předpokládejme, že známe algoritmus blokové šifry, na kterou chceme provést algebraický útok. Tento algoritmus algebraicky popíšeme, a tak obdržíme soustavu polynomiálních rovnic. Cílem této kapitoly je uvést algoritmy pro řešení takovýchto soustav rovnic.

4.1 Buchbergerův algoritmus

Buchbergerův algoritmus je univerzální algoritmus pro řešení soustav polynomiálních rovnic více proměnných, který využívá teorii Gröbnerových bází. Pro účely algebraických útoků se ukazuje být tento algoritmus neefektivní, a proto se jím v této práci nebudeme zabývat a uvedeme pouze odkaz na literaturu. Využití algoritmu pro algebraické útoky je popsáno například v práci [1].

V článcích [9] a [10] z let 1999 a 2002 představuje autor algoritmy F4 a F5, které jsou modifikací Buchbergerova algoritmu.

4.2 Relinearizace

Následující text vychází z článku [15]. Předpokladem pro úspěšné vyřešení soustavy tímto algoritmem je vyšší počet rovnic než je počet neznámých. Každý monom je nahrazen novou proměnnou, čímž se soustava polynomiálních rovnic převede na soustavu rovnic lineárních. Ve smyslu stejné množiny

kořenů obou soustav není tato úprava pochopitelně ekvivaletní a může se stát, že kořeny přibydou.

Mějme soustavu m kvadratických rovnic o n neznámých tvaru:

$$\sum_{1\leq i\leq j\leq n} a_{i,j,k}x_ix_j = b_k, \quad k = 1,\ldots,m.$$

Označíme-li $y_{i,j} = x_ix_j$ pro $1 \leq i \leq j \leq n$, dostaneme soustavu m lineárních rovnic o N neznámých, kde $N = \frac{n(n+1)}{2}$:

$$\sum_{1\leq i\leq j\leq n} a_{i,j,k}y_{i,j} = b_k, \quad k = 1,\ldots,m.$$

Na tuto soustavu aplikujeme Gaussovu eliminaci. Řešení vyjádříme parametricky pomocí nových proměnných $t_1,\ldots,t_l$, $l < N$:

$$y_{i,j} = c_o + \sum_{k=1}^{l} c_kt_k, \quad 1 \leq i \leq j \leq n.$$

Pro každé a,b,c,d,e,f,g, kde $1 \leq a < b < c < d \leq n$ a $1 \leq e < f < g \leq n$ platí:

$$\begin{aligned} y_{a,b}y_{c,d} &= y_{a,c}y_{b,d} \\ y_{a,b}y_{c,d} &= y_{a,d}y_{b,c} \\ y_{a,c}y_{b,d} &= y_{a,d}y_{b,c} \\ y_{e,e}y_{f,g} &= y_{e,f}y_{e,g} \end{aligned}$$

Pro všechny volby $a,b,\ldots,g$ získáme $M = 3\binom{n}{4} + \binom{n}{3} = \frac{3n^4-14n^3+21n^2-8n}{24}$ rovnic, do kterých dosadíme za $y_{i,j}$ parametry t a upravíme do tvaru:

$$\sum_{1\leq i\leq l} p_{i,k}t_i + \sum_{1\leq i\leq j\leq l} p_{i,j,k}t_it_j = r_k, \quad k = 1,\ldots,M.$$

Tím jsme obdrželi opět soustavu kvadratických rovnic, na kterou aplikujeme relinearizaci.

4.3 XL

Popis algoritmu a příklad je převzat z článku [15]. Mějme soustavu m kvadratických rovnic o n neznámých nad tělesem K tvaru:

$$\begin{aligned} f_1(x_1, \ldots, x_n) &= b_1 \\ f_2(x_1, \ldots, x_n) &= b_2 \\ &\vdots \\ f_m(x_1, \ldots, x_n) &= b_m, \end{aligned}$$

kde f_k jsou kvadratické polynomy s nulovým absolutním členem, $k \leq m$. Definujeme-li $l_k = f_k(x_1, \ldots, x_n) - b_k$, můžeme soustavu přepsat jako:

$$\begin{aligned} l_1 &= 0 \\ l_2 &= 0 \\ &\vdots \\ l_m &= 0 \end{aligned}$$

Definice 4.1. *Nechť p je polynom. Řekneme, že p je typu $x^k l$, právě když existuje $i \leq m$ a posloupnost indexů $i_1, i_2, \ldots, i_k$ taková, že:*

$$p = \prod_{j=1}^{k} x_{i_j} * l_i.$$

Množinu všech polynomů typu $x^k l$ budeme značit $x^k \mathcal{L}$.

Poznámka 4.2. *Jinými slovy, polynom $p = 0$ je typu $x^k l$, pokud rovnice $p = 0$ vznikne z některé rovnice soustavy vynásobením monomem $\prod_{j=1}^{k} x_{i_j}$.*

Definice 4.3. *Množinu všech monomů stupně k, tj. výrazů tvaru $\prod_{j=1}^{k} x_{i_j}$ pro nějaké $i_1, i_2, \ldots, i_k$, budeme značit X^k.*

Definice 4.4. *Nechť $D \in \mathbb{N}$. Uvažme všechny polynomy tvaru $\prod_j x_{i_j} * l_i$, jejichž stupeň je menší nebo roven D. Definujme $\mathcal{I}_D$ jako lineární obal těchto polynomů. Dále definujme $\mathcal{I}$ jako ideál v prostoru všech polynomů generovaný polynomy l_i, $i = 1, 2, \ldots, m$.*

Poznámka 4.5. *$\mathcal{I}_D$ je vektorový prostor generovaný všemi polynomy typu $x^k l$, $0 \leq k \leq D - 2$.*

Myšlenkou XL algoritmu je najít v $\mathcal{I}_D$ nějakou rovnici, která je snáze řešitelná než vstupní soustava rovnic $\mathcal{I}_0$.

1. **Násobení:** Spočítat všechny součiny $\prod_{j=1}^{k} x_{i_j} * l_i \in \mathcal{I}_D$, $k \leq D-2$.
2. **Linearizace:** Každý monom stupně nejvýše D nahradíme novou proměnnou a na vzniklou soustavu lineárních rovnic aplikujeme Gaussovu eliminaci. Zvolíme takové uspořádání proměnných, aby proměnné odpovídající monomům, jež obsahují x_1, byly eliminovány naposled.
3. **Řešení:** Předpokládáme, že v kroku 2 vznikne alespoň jedna rovnice, jež obsahuje pouze x_1. Tuto rovnici vyřešíme nad konečným tělesem (např. použijeme Berlekampův algoritmus).
4. **Opakování:** Dosadíme řešení z kroku 3 a opakujeme proces pro zjednodušené rovnice.

XL algoritmus je velmi jednoduchý, ale není jasné, pro které hodnoty n a m uspěje, jaká je jeho časová složitost, jaký je jeho vztah k relinearizaci a algoritmům používající Gröbnerovy báze.

Poznámka 4.6. *Všechny rovnice generované v XL leží v $x^k l$ a náleží $\mathcal{I}$, ideálu generovaném l_i. Není třeba uvažovat obecnější rovnice jako l_1^2, neboť ty leží v $\mathcal{I}_4$ a leží tedy ve vektorovém prostoru generovaném rovnicemi typu $x^2 l \cup xl \cup l$.*

Poznámka 4.7. *Někdy je výhodnější pracovat pouze s jistou podmnožinou všech generovaných monomů. Například pokud jsou všechny rovnice homogenní, stačí použít jen monomy lichého (nebo sudého) stupně.*

Příklad 4.8. *Nechť $\mu \neq 0$. Uvažme následující soustavu rovnic:*

$$x_1^2 + \mu x_1 x_2 = \alpha \tag{4.1}$$
$$x_2^2 + \nu x_1 x_2 = \beta \tag{4.2}$$

Zvolme $D = 4$ a uvažme pouze monomy sudého stupně, tj. monomy $x_1^2, x_2^2, x_1 x_2 \in X^2$. Vynásobením rovnic obdržíme dalších 6 rovnic a přidáme je do soustavy.

$$x_1^4 + \mu x_1^3 x_2 = \alpha x_1^2 \tag{4.3}$$
$$x_1^2 x_2^2 + \nu x_1^3 x_2 = \beta x_1^2 \tag{4.4}$$
$$x_1^2 x_2^2 + \mu x_1 x_2^3 = \alpha x_2^2 \tag{4.5}$$
$$x_2^4 + \nu x_1 x_2^3 = \beta x_2^2 \tag{4.6}$$
$$x_1^3 x_2 + \mu x_1^2 x_2^2 = \alpha x_1 x_2 \tag{4.7}$$
$$x_1 x_2^3 + \nu x_1^2 x_2^2 = \beta x_1 x_2 \tag{4.8}$$

Gaussovou eliminací obdržíme rovnice:

(4.1)*:* $x_1x_2 = \frac{\alpha}{\mu} - \frac{x_1^2}{\mu}$
(4.2)*:* $x_2^2 = (\beta - \frac{\alpha\nu}{\mu}) + \frac{\nu}{\mu}x_1^2$
(4.3)*:* $x_1^3x_2 = \frac{\alpha}{\mu}x_1^2 - \frac{x_1^4}{\mu}$
(4.4)*:* $x_1^2x_2^2 = (\beta - \frac{\alpha\nu}{\mu})x_1^2 + \frac{\nu}{\mu}x_1^4$
(4.8)*:* $x_1x_2^3 = \frac{\alpha\beta}{\mu} + (\frac{\alpha\nu^2}{\mu} - \beta\nu - \frac{\beta}{\mu})x_1^2 - \frac{\nu^2}{\mu}x_1^4$
(4.6)*:* $x_2^4 = (\beta^2 - \frac{2\alpha\beta\nu}{\mu}) + (\frac{2\nu\ beta}{\mu} + \beta\nu^2 - \frac{\alpha\nu^2}{\mu})x_1^2 + \frac{\nu^3}{\mu}x_1^4$

Rovnice (4.5) *obsahuje jednu proměnnou:*

$$\alpha^2 + x_1^2(\alpha\mu\nu - \beta\mu^2 - 2\alpha) + x_1^4(1 - \mu\nu) = 0.$$

Jejím vyřešením a dosazením do ostatních rovnic získáme řešení soustavy.

Kapitola 5

Řešení soustav rovnic pomocí SAT solverů

Problém řešení soustavy kvadratických rovnic nad tělesem GF(2) je NP těžký. Problém nalezení takového pravdivostního ohodnocení booleovské formule, pro něž je tato formule splněna, se nazývá SAT (Boolean Satisfiability Problem) a je rovněž NP těžký, dokonce NP úplný. Z teorie složitosti víme, že na NP úplný problém lze v polynomiálním čase redukovat všechny NP problémy.

SAT problém je již dlouho a intenzivně zkoumán a bylo vyvinuto mnoho algoritmů tzv. SAT solverů, které se ho snaží řešit. V této kapitole si ukážeme, jak lze tyto algoritmy použít k algebraickým útokům.

5.1 Definice a značení matematické logiky

Pro definici výrokové formule vezměme neprázdnou množinu At, která neobsahuje žádný ze symbolů (,), $\wedge$, $\vee$, $\rightarrow$, $\neg$. Prvkům množiny At budeme říkat *výrokové atomy* nebo jen *atomy*.

Definice 5.1. *Množina všech výrokových formulí je nejmenší množina výrazů splňující podmínky:*

- *každý výrokový atom je výroková formule,*
- *je-li ϕ výroková formule, pak $\neg\phi$ je výroková formule,*
- *jsou-li ϕ a ψ výrokové formule, pak $(\phi \wedge \psi)$, $(\phi \vee \psi)$ a $(\phi \rightarrow \psi)$ jsou výrokové formule.*

Definice 5.2. *Pravdivostní ohodnocení je každá funkce v z množiny všech výrokových formulí do množiny $\{0,1\}$, která pro libovolné formule ϕ a ψ splňuje podmínky:*

- $v(\phi \wedge \psi) = 1$, *právě když* $v(\phi) = 1$ *a* $v(\psi) = 1$,
- $v(\phi \vee \psi) = 1$, *právě když* $v(\phi) = 1$ *nebo* $v(\psi) = 1$,
- $v(\phi \rightarrow \psi) = 1$, *právě když* $v(\phi) = 0$ *nebo* $v(\psi) = 1$,
- $v(\neg\phi) = 1$, *právě když* $v(\phi) = 0$.

Definice 5.3. *Řekneme, že výroková formule ϕ je splnitelná, jestliže existuje pravdivostní ohodnocení v takové, že $v(\phi) = 1$. Množinu všech splnitelných výrokových formulí označíme* SAT.

Někdy se pod termínem SAT problém označuje problém rozhodnout, zda je daná výroková formule splnitelná. Nás ale zajímá konkrétní pravdivostní ohodnocení, pro které je daná formule pravdivá, a takto budeme pojem SAT problém chápat. Existenci tohoto pravdivostního ohodnocení máme v našem případě zaručenu z konstrukce formule.

Dále zaveďme některé pojmy, které se běžně používají v matematické logice. *Literál* je každá formule tvaru p nebo $\neg p$, kde p je atom. *Klauzule* je disjunkce libolného počtu literálů. Formule, jež je konjunkcí klazulí, je v *konjunktivním normálním tvaru.*

Věta 5.4. *Každá výroková formule je ekvivalentní s jistou formulí, která je v konjunktivním normálním tvaru.*

Důkaz. Viz. [16]. □

Konjunktivní normální forma je velmi výhodná jako vstup pro většinu SAT solverů.

5.2 Převedení MQ na SAT

Definice 5.5. *Nechť $f_1(x_1,\ldots,x_n), f_2(x_1,\ldots,x_n),\ldots,f_m(x_1,\ldots,x_n)$ jsou polynomy stupně nejvýše 2 nad* GF(2) *v proměnných $x_1, x_2, \ldots, x_n$. Úloze nalézt $(a_1, a_2, \ldots, a_n) \in \{0,1\}^n$ tak, aby pro každé $k \leq m$ platilo:*

$$f_k(a_1, a_2, \ldots, a_n) = 0,$$

se říká MQ problém.

Definice 5.6. *Nechť At je množina výrokových atomů obsahující* $x_1, \ldots, x_n$. *Nechť* $f_1(x_1, \ldots, x_n), f_2(x_1, \ldots, x_n), \ldots, f_m(x_1, \ldots, x_n)$ *jsou polynomy nad* GF(2) *v proměnných* $x_1, x_2, \ldots, x_n$. *Řekneme, že výroková formule* ϕ *je ekvivalentní soustavě rovnic* $f_1 = f_2 = \ldots = f_m = 0$, *právě když:*

$(a_1, a_2, \ldots, a_n) \in \mathrm{GF}(2)^n$ *řeší MQ* $\Longleftrightarrow$ *existuje pravdivostní ohodnocení* v *takové, že* $v(x_k) = a_k$ *pro každé* $k \leq n$ *a* $v(\phi) = 1$.

Nyní ukážeme, jak k soustavě rovnic zkonstruovat ekvivaletní výrokovou formuli v konjunktivní normální formě.

Lemma 5.7. *Výroková formule* ϕ :

$$(x_1 \vee \neg y) \wedge (x_2 \vee \neg y) \wedge \ldots \wedge (x_n \vee \neg y) \wedge (y \vee \neg x_1 \vee \neg x_2 \vee \ldots \vee \neg x_n)$$

je ekvivaletní rovnici $y = \prod_{k=1}^{n} x_n$.

Důkaz. Nechť pro $(b, a_1, \ldots, a_n) \in \mathrm{GF}(2)^{n+1}$ platí $b = \prod_{k=1}^{n} a_n$. Je-li $b = 0$, pak alespoň jedna z neznámých $a_1, \ldots, a_n$ má hodnotu 0, je-li $b = 1$, pak $a_1 = \ldots a_n = 1$. Definujme pravděpodobnostní ohodnocení v tak, že $v(y) = b$ a $v(x_k) = a_k$ pro $k = 1, 2, \ldots, n$. Zřejmě platí $v(\phi) = 1$.

Naopak jestliže v je takové pravděpodobnostní ohodnocení, že $v(\phi) = 1$, je buď $v(y) = 0$ a potom alespoň pro jeden z atomů $x_1, , x_n$ je $v(x_k) = 0$, a nebo $v(y) = 1$ a pak $v(x_1) = \ldots = v(x_n) = 1$ a hodnoty tohoto ohodnocení jsou tedy řešením rovnice. □

Lemma 5.8. *Výroková formule*

$$c_1 \wedge c_2 \wedge \ldots \wedge c_{2^{n-1}},$$

kde c_k *jsou všechny klauzule, v nichž se vyskytují atomy* $x_1, x_2, \ldots, x_n$ *s lichým počtem negací, je ekvivaletní rovnici* $\sum_{k=1}^{n} x_k = 0$.

Důkaz. Nechť $c = l_1 \vee l_2 \vee \ldots \vee l_n$ je klazule obsahující literály $l_1, l_2, \ldots, l_n$, kde $l_k = \neg x_k$ pro lichý počet k, v ostatních případech je $l_k = x_k$. Nechť v je pravdivostní ohodnocení. Platí $v(c) = 0$, právě když $v(l_1) = v(l_2) = \ldots = v(l_n) = 0$ a tedy $v(x_k) = 1$ pro lichý počet k. Formule $c_1 \wedge c_2 \wedge \ldots \wedge c_{2^{n-1}}$ je konjunkcí všech klazulí tohoto typu a není tedy splněna právě pro všechna ohodnocení, jež jsou pro lichý počet atomů rovny 1. Vezměme $(a_1, a_2, \ldots, a_n) \in \mathrm{GF}(2)^n$. $\sum_{k=1}^{n} a_k \neq 0$, právě když počet nenulových a_k je lichý. □

Mějme soustavu rovnic $f_1 = f_2 = \ldots = f_m = 0$, konstrukce ekvivalentní výrokové formule probíhá následovně.

Výroková formule neobsahuje konstanty, proto každý výskyt konstanty v soustavě nahradíme novou proměnnou T, čímž získáme soustavu, kde každá rovnice je součtem monomů stupně alespoň 1.

Dále nahradíme novou proměnnou každý monom a stupně většího než 1. Nově vzniklé rovnice tvaru $a = \prod_{k=1}^{n} x_n$ dokážeme podle lemmatu 5.7 ekvivalentně popsat výrokovou formulí. Zbytek soustavy po nahrazení tvoří již jen lineární rovnice.

Lineární rovnice převedeme pomocí lemmatu 5.8 na výrokovou formuli v konjunktivní normální formě.

Příklad 5.9. Mějme následující soustavu rovnic:

$$\begin{aligned} x_1x_2 + x_3 + 1 &= 0 \\ x_1 + x_2 &= 0 \\ x_2x_3 &= 0 \end{aligned}$$

V prvním kroku nahradíme konstantu 1 novou proměnnou T. Dostáváme:

$$\begin{aligned} x_1x_2 + x_3 + T &= 0 \\ x_1 + x_2 &= 0 \\ x_2x_3 &= 0 \end{aligned}$$

Monomy stupně většího než 1 nahradíme novými proměnnými a jejich vyjádření přidáme do soustavy:

$$\begin{aligned} a + x_3 + T &= 0 \\ x_1 + x_2 &= 0 \\ b &= 0 \\ a &= x_1x_2 \\ b &= x_2x_3 \end{aligned}$$

Nyní použijeme lemmata 5.7 a 5.8 a pro každou rovnici zkonstruujeme ekvivaletní výrokovou formuli:

$$(\neg a \vee x_3 \vee T) \wedge (a \vee \neg x_3 \vee T) \wedge (a \vee x_3 \vee \neg T) \wedge (\neg a \vee \neg x_3 \vee \neg T)$$
$$(\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2)$$
$$\neg b$$
$$(x_1 \vee \neg a) \wedge (x_2 \vee \neg a) \wedge (a \vee \neg x_1 \vee \neg x_2)$$
$$(x_2 \vee \neg b) \wedge (x_3 \vee \neg b) \wedge (b \vee \neg x_2 \vee \neg x_3)$$

Výsledkem je konjunkce všech klauzulí výše a klauzule T.

5.3 Složitost formule

Náročnost vyřešení SAT problému pro konkrétní formuli lze posuzovat podle různých kritérií:

- počet atomů,

- počet klauzulí,

- celková délka všech klauzulí.

Mějme následující kvadratickou rovnici v proměnných $x_1, x_2, \ldots, x_n$ a odhadněme složitost ekvivalentní formule:

$$1 + x_1 + \ldots + x_n + x_1x_2 + \ldots + x_{n-2}x_{n-1} = 0$$

Považujeme-li 1 za konstantní term a označíme-li M počet termů v rovnici, pak $M = 1 + n + \frac{n(n-1)}{2}$. Ekvivalentní formule má právě M atomů, neboť konstantní term je nahrazen atomem T, lineárním termům $x_1, x_2, \ldots, x_n$ odpovídají stejně označené atomy a pro kvadratické termy jsou zavedeny nové proměnné a jim odpovídající atomy. Formule dále obsahuje 1 klauzuli pro konstantu, 3 klauzule pro každý kvadratický monom (lemma 5.7) a 2^{M-1} klazulí (lemma 5.8), celkem tedy právě $1 + 3(M - n - 1) + 2^{M-1}$ klauzulí. Celková délka všech klauzulí je $1 + 7(M - n - 1) + M2^{M-1}$.

Kvadratickou rovnici v n proměnných jsme tedy schopni popsat výrokovou formulí s těmito vlastnostmi:

- obsahuje $1 + n + \frac{n(n-1)}{2}$ atomů,

- obsahuje $1 + 3(M - n - 1) + 2^{M-1}$ klauzulí,

- celková délka všech klauzulí je $1 + 7(M - n - 1) + M2^{M-1}$.

Z výše uvedeného plyne, že počet a délka klauzulí závisí exponenciálně na počtu termů, což je nepoužitelné. Pomocí lemmatu 5.8 dokážeme na výrokovou formuli převést rovnice tvaru $\sum_{k=1}^{M} x_k = 0$. Tato formule je konjunkcí 2^{M-1} klauzulí, každá o M literálech. Nechť $M \geq 4$ je sudé. Položme $h = [M/2] - 2$. Zavedením nových proměnných $y_1, y_2, \ldots, y_h$ lze rovnici

$\sum_{k=1}^{M} x_k = 0$ ekvivalentně přepsat jako následující soustavu $h+1$ rovnic:

$$\begin{aligned} x_1 + x_2 + x_3 + y_1 &= 0 \\ y_1 + x_4 + x_5 + y_2 &= 0 \\ &\vdots \\ y_i + x_{2i+2} + x_{2i+3} + y_{i+1} &= 0 \\ &\vdots \\ y_h + x_{M-2} + x_{M-1} + x_M &= 0 \end{aligned}$$

Pro lichá M lze použít stejný trik, pouze bude mít poslední rovnice soustavy o jeden člen méně. Nyní dokážeme sestrojit k původní kvadratické rovnici ekvivalentní formuli, jež

- obsahuje $1 + n + \frac{n(n-1)}{2} + h$ atomů,

- obsahuje $1 + 3(M - n - 1) + 8h$ klauzulí,

- celková délka všech klauzulí je $1 + 7(M - n - 1) + 32(h + 1)$.

Tuto formuli lze již použít jako vstup SAT solver.

5.4 SAT solver

SAT solver je algoritmus, jež dostane na vstupu výrokovou formuli ϕ a v případě, že je splnitelná, nalezne pro ni takové pravděpodobnostní ohodnocení v, že $v(\phi) = 1$. Algoritmů existuje celá řada a zpravidla nejsou deterministické a používají při výpočtu náhodná čísla. Časovou složitost lze jen velmi obtížné odhadnout, a proto se jí nebudeme v této práci zabývat. SAT solver pro nás bude pouze tzv. black box, tj. nástroj, který dokáže vyřešit některé instance problému, ale bez záruky a jakýchkoliv odhadů náročnosti.

Základem většiny SAT solverů je Davis-Putnamův algoritmus, vstupem je výroková formule ϕ v konjunktivní normální formě. Pro každý atom uchováváme informaci, zda byl ohodnocen a jakou hodnotou. Na začátku jsou všechny atomy neohodnoceny. Ohodnocení atomů jednoznačně koresponduje s ohodnocením literálů. Dále postupujeme v následujích krocích:

1. **Propagace** - pokud některá klazule obsahuje pouze jeden neohodnocený literál λ a všechny ostatní literály mají přiřazenu hodnotu 0, pak literálu λ přiřadíme hodnotu 1. Tento krok opakujeme, dokud takové klauzule v naší formuli existují.

2. **Přiřazení hodnoty** - vybereme atom a ohodnotíme ho. Výběr a přiřazení hodnoty lze provést podle různých kritérií, lze například zvolit atom s nejvyšším výskytem, vybrat náhodně, případně použít složitější kritérium tzv. heuristiku.

3. **Návrat** - v případě, že vznikne klauzule, jež je ohodnocena hodnotou 0, změníme ohodnocení naposled ohodnoceného atomu. Tomuto stavu budeme říkat konflikt. Pokud byl atom během výpočtu již ohodnocen hodnotami 0 i 1, zrušíme jeho ohodnocení a aplikujeme tento krok na předchozí atom.

Tím jsme nastínili princip SAT solveru. Pro ilustraci uveďme jednoduchý příklad.

Příklad 5.10. Mějme následující formuli:

$$\phi = x_1 \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (x_2 \vee x_4) \wedge (\neg x_3 \vee \neg x_4)$$

Podle kroku 1 ohodnotíme atom x_1 hodnotou 1, čímž je první klauzule formule ϕ splněna. Nyní přistoupíme ke kroku 2. Pro jednoduchost zvolíme atom x_2 a ohodnotíme ho hodnotou 0. Aby byla druhá a třetí klauzule splněna, musí být podle kroku 1 atomy x_3 a x_4 ohodnoceny hodnotou 1. Při tomto ohodnocení ale není splněna poslední klauzule a proto podle kroku 3 zrušíme ohodnocení atomů x_3 a x_4 a ohodnocení atomu x_2 změníme na 1. Znovu vybereme neohodnocený atom, například x_3, a přiřadíme mu ohodnocení 0. Krok 1 nelze nyní použít, musíme opět přiřadit hodnotu neohodnocenému atomu. Přiřadíme-li atomu x_4 hodnotu 0, jsou všechny klauzule splněny a tím je splněna i formule ϕ.

Označíme-li hledané výrokové ohodnocení v, pak $v(x_1) = 1$, $v(x_2) = 1$, $v(x_3) = 0$, $v(x_4) = 0$ a tedy $x(\phi) = 1$.

V praxi je podle článku [13] časově nejnáročnější částí výpočtu propagace. Klíčem k urychlení algoritmu je tedy její efektivní implementace. V okamžiku, kdy dojde k ohodnocení atomu, musíme být schopni detekovat klauzule, které:

- mají všechny literály ohodnocené 0,

- mají právě jeden literál neohodnocený a zbylé jsou ohodnoceny 0.

Je neúnosné procházet po každém přiřazení všechny klauzule a pokaždé testovat, zda nastal jeden z těchto případů. Řešením je technika *sledovaných literálů*, kterou si nyní popíšeme.

V každé klauzuli zvolíme libovolné dva literály, jež nejsou ohodnoceny 0. Těmto literálům budeme říkat *sledované literály*. V okamžiku, kdy dojde k přiřazení hodnoty 0 některému ze sledovaných literálů L, mohou nastat právě dva případy:

1. Klauzule obsahuje alespoň jeden literál, jež není sledovaný a není ohodnocený 0. Tímto literálem nahradíme literál L a tím zůstane vlastnost sledovaných literálů zachována.

2. Všechny literály kromě sledovaných jsou ohodnocené 0. Literál L byl rovněž ohodnocen 0, druhý sledovaný literál musí být tedy nutně ohodnocen 1, čímž je ohodnocena celá klauzule.

Dojde-li během výpočtu k situaci, kdy jsou všechny literály v nějaké klauzuli ohodnoceny 0 a ta je díky tomu nesplněna, musí být proveden návrat a zrušeno ohodnocení naposled ohodnoceného atomu. Při této operaci není třeba jakkoliv měnit sledované literály, neboť jejich vlastnost zůstane zachována.

Nyní se blíže podíváme na druhý krok algoritmu - přiřazení hodnoty. Ve chvíli, kdy nelze použít propagaci a není tedy bezprostředně jasné, který další atom ve formuli ohodnotit, nezbývá než si "hodit mincí", tj. zvolit nějakou cestu bez záruky, že vede k cíli. Existuje celá řada strategií, jak tuto volbu provést. Pokusme se zvážit kritéria, podle nichž lze rozdílné strategie porovnat:

Výpočetní složitost - příliš výpočetně náročná strategie může sice pomoci nalézt řešení s menším počtem kroků, zároveň ale celý algoritmus zatěžuje, takže nemusí být vždy efektivnější než přímé a jednoduché metody.

Počet přiřazení - toto kritérium vychází z myšlenky, že pokud byla přiřazení hodnot volena "chytře", byla větší část atomů ohodnocena při propagaci, čímž byl minimalizován počet "špatných" přiřazení.

Počet konfliktů - vychází z podobné myšlenky jako předchozí kritérium. Pokud byla přiřazení hodnot volena "dobře", byl počet konfliktů minimální a nebylo tedy nutné se příliš často vracet.

Kapitola 6

Závěr

Bezpečnost AES je úzce svázána s S-boxem, neboť ten jediný je nelineární, všechny ostatní komponenty šifry lineární jsou. V práci jsme zkoumali algebraické popisy S-boxů pomocí soustav polynomiálních rovnic nad dvouprvkovým tělesem a ověřili jsme, že S-box AES lze popsat pomocí 23 kvadratických rovnic, což se zdá být překvapivě vysoký počet.

Při testech na náhodně generovaných S-boxech stejné velikosti jako S-box AES nebyl nalezen jediný S-box, pro který by algoritmus vrátil alespoň jednu kvadratickou rovnici. Budeme-li považovat počet rovnic za ukazatel odolnosti S-boxu, pak je S-box AES v tomto smyslu dosti vzdálen od náhodného S-boxu.

Pro AES-128 je podle článku [7] problém nalezení klíče z dvojice otevřeného a šifrového textu ekvivaletní vyřešení soustavy přibližně 8000 kvadratických rovnic o 1600 proměnných, což je v současnosti velmi daleko za hranicemi možností současných výpočetních metod i techniky.

Literatura

[1] Albrecht M.: *Algebraic Attacks on the Courtois Toy Cipher*, 2006.

[2] Bard G. V.: *Algorithms for solving linear and polynomial systems of equations over finite fields with applications to cryptoanalysis*, 2007.

[3] Bard G. V., Courtois N. T., Jefferson C.: *Efficient Methods for Conversion and Solution of Sparse Systems of Low-Degree Multivariate Polynomials over GF(2) via SAT-Solvers*, 1997.

[4] Biryukov A., De Cannière C.: *Block Ciphers and System of Quadratic Equations*, 2003.

[5] Coppersmith D.: *The Data Encryption Standard (DES) and its strength against attacks*, IBM Journal of Research and Development, 1994.

[6] Courtois N. T.: *How fast can be Algebraic Attacks on Block Ciphers?*, 2006.

[7] Courtois N. T., Pieprzyk J.: *Cryptoanalysis of block ciphers with overdefined systems of equations*, 2002.

[8] Daemen J., Rijmen V.: *AES Proposal: The Rijndael*, 1999.

[9] Faugère J. C.: *A new efficient algorithm for computing Gröbner bases (F4)*, 1999.

[10] Faugère J. C.: *A new efficient algorithm for computing Gröbner bases without reduction to zero F5*, 2002.

[11] Giraud C.: *DFA on AES*, 2003.

[12] Matsui M.: *Linear cryptanalysis method for DES cipher*, Advances in Cryptology - EUROCRYPT, 1993.

[13] Moskewicz M. W, Madigan C. F., Zhao Y., Zhang L., Malik S.: *Chaff: Engineering an Efficient SAT Solver*, 2001.

[14] National Institute of Standards and Technology. *FIPS PUB 197: Advanced Encryption Standard*, 2001.

[15] Shamir A., Patarin J., Courtois N., Klimov A.: *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations*, 2000.

[16] Švejdar V.: *Logika, neúplnost, složitost a nutnost*, Academia, Praha, 2002.

Příloha

Výstup algoritmu 3.9 pro S-box šifry AES a bází:
$f_1 = 1$, $f_2 = x_1$, ..., $f_9 = x_8$, $f_{10} = y_1$, ..., $f_{17} = y_8$, $f_{18} = x_1y_1$, ..., $f_{25} = x_1y_8$, $f_{26} = x_2y_1$, ..., $f_{81} = x_8y_8$.

$$1 + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + y_2 + y_3 + y_5 + x_1y_1 + x_1y_4 + x_1y_5 + x_1y_8 + x_2y_1 + x_2y_2 + x_2y_4 + x_2y_6 + x_3y_2 + x_3y_3 + x_3y_4 + x_3y_5 + x_3y_6 + x_3y_7 + x_4y_2 + x_4y_4 + x_4y_6 + x_4y_8 + x_5y_1 + x_5y_2 + x_5y_4 + x_5y_6 + x_5y_7 + x_6y_1 = 0$$

$$x_1 + x_2 + x_7 + x_8 + y_1 + y_5 + y_6 + x_1y_2 + x_1y_5 + x_2y_1 + x_2y_2 + x_2y_4 + x_2y_8 + x_3y_5 + x_3y_8 + x_4y_2 + x_4y_3 + x_4y_5 + x_4y_6 + x_5y_1 + x_5y_3 + x_6y_3 = 0$$

$$x_3 + x_6 + x_8 + y_6 + y_7 + x_1y_3 + x_1y_6 + x_2y_1 + x_2y_2 + x_2y_3 + x_2y_5 + x_3y_1 + x_3y_6 + x_4y_3 + x_4y_4 + x_4y_6 + x_4y_7 + x_5y_2 + x_5y_3 + x_5y_4 + x_5y_6 + x_5y_8 + x_6y_4 = 0$$

$$1 + x_2 + x_4 + x_6 + y_2 + y_3 + y_4 + y_6 + y_7 + x_1y_3 + x_1y_6 + x_1y_7 + x_2y_1 + x_2y_2 + x_2y_5 + x_2y_6 + x_3y_1 + x_3y_3 + x_3y_6 + x_3y_7 + x_4y_3 + x_4y_4 + x_4y_6 + x_4y_8 + x_5y_2 + x_5y_3 + x_5y_6 + x_5y_8 + x_6y_5 = 0$$

$$x_2 + x_4 + x_5 + y_2 + y_3 + y_5 + y_7 + x_1y_3 + x_1y_6 + x_1y_7 + x_1y_8 + x_2y_1 + x_2y_2 + x_2y_4 + x_2y_5 + x_2y_6 + x_2y_7 + x_3y_1 + x_3y_3 + x_3y_4 + x_3y_6 + x_3y_7 + x_3y_8 + x_4y_1 + x_4y_4 + x_4y_8 + x_5y_2 + x_5y_5 + x_6y_6 = 0$$

$$1 + x_2 + x_3 + x_4 + x_8 + y_2 + y_3 + y_4 + y_6 + y_7 + y_8 + x_1y_2 + x_1y_5 + x_1y_6 + x_1y_7 + x_2y_1 + x_2y_3 + x_2y_4 + x_2y_5 + x_2y_6 + x_2y_8 + x_3y_3 + x_3y_6 + x_3y_8 + x_4y_2 + x_4y_3 + x_4y_5 + x_4y_8 + x_5y_1 + x_5y_4 + x_6y_2 + x_6y_7 = 0$$

$$x_5 + x_6 + x_7 + x_8 + y_2 + y_4 + y_5 + y_6 + x_1y_1 + x_1y_2 + x_1y_7 + x_2y_2 + x_2y_6 + x_2y_7 + x_3y_8 + x_4y_1 + x_4y_3 + x_4y_7 + x_4y_8 + x_5y_1 + x_5y_2 + x_5y_6 + x_6y_1 + x_6y_8 = 0$$

$$1 + x_3 + x_6 + x_7 + x_8 + y_1 + y_2 + y_7 + x_1y_6 + x_1y_7 + x_1y_8 + x_2y_1 +$$
$$x_2y_2 + x_3y_4 + x_3y_5 + x_3y_8 + x_4y_3 + x_4y_4 + x_4y_6 + x_5y_2 +$$
$$x_5y_5 + x_5y_6 + x_6y_1 + x_6y_2 + x_7y_1 = 0$$

$$x_2 + x_4 + x_7 + y_4 + y_6 + y_7 + x_1y_1 + x_1y_2 + x_1y_3 + x_1y_5 + x_1y_6 +$$
$$x_2y_4 + x_2y_5 + x_2y_6 + x_2y_7 + x_2y_8 + x_3y_1 + x_3y_2 + x_3y_3 +$$
$$x_3y_5 + x_4y_4 + x_5y_4 + x_5y_5 + x_5y_8 + x_6y_1 + x_7y_2 = 0$$

$$1 + x_1 + x_2 + x_4 + x_6 + x_8 + y_1 + y_2 + y_5 + y_7 + x_1y_2 + x_1y_3 + x_1y_6 +$$
$$x_1y_8 + x_2y_1 + x_2y_4 + x_2y_5 + x_2y_8 + x_3y_1 + x_3y_2 + x_3y_4 +$$
$$x_3y_5 + x_4y_2 + x_4y_3 + x_4y_6 + x_4y_8 + x_5y_1 + x_5y_2 + x_5y_5 +$$
$$x_5y_6 + x_6y_1 + x_7y_3 = 0$$

$$x_1 + x_2 + x_3 + x_6 + x_7 + y_1 + y_2 + y_4 + y_6 + y_7 + y_8 + x_1y_1 + x_1y_6 +$$
$$x_1y_7 + x_2y_2 + x_2y_4 + x_2y_8 + x_3y_1 + x_3y_3 + x_3y_5 + x_3y_6 +$$
$$x_3y_8 + x_4y_1 + x_4y_3 + x_4y_4 + x_4y_5 + x_4y_8 + x_5y_1 + x_5y_3 +$$
$$x_5y_4 + x_5y_5 + x_5y_8 + x_6y_2 + x_7y_4 = 0$$

$$1 + x_1 + x_2 + x_3 + x_6 + x_8 + y_1 + y_8 + x_1y_4 + x_1y_5 + x_2y_2 + x_2y_3 +$$
$$x_2y_5 + x_2y_6 + x_2y_7 + x_2y_8 + x_3y_2 + x_3y_3 + x_3y_4 + x_3y_7 +$$
$$x_3y_8 + x_4y_1 + x_4y_2 + x_4y_3 + x_4y_4 + x_4y_5 + x_5y_4 + x_5y_5 +$$
$$x_5y_8 + x_6y_1 + x_6y_2 + x_7y_5 = 0$$

$$1 + x_1 + x_5 + x_6 + y_1 + y_3 + y_5 + y_6 + y_7 + x_1y_1 + x_1y_2 + x_1y_4 +$$
$$x_1y_5 + x_1y_6 + x_1y_7 + x_1y_8 + x_2y_3 + x_2y_4 + x_2y_5 + x_2y_8 +$$
$$x_3y_1 + x_3y_3 + x_3y_7 + x_3y_8 + x_4y_6 + x_4y_7 + x_5y_2 + x_7y_6 = 0$$

$$1 + x_4 + x_5 + x_8 + y_1 + y_5 + y_6 + y_7 + x_1y_1 + x_1y_2 + x_1y_5 +$$
$$x_1y_7 + x_2y_1 + x_2y_3 + x_2y_4 + x_2y_5 + x_2y_8 + x_3y_1 + x_3y_2 +$$
$$x_3y_4 + x_3y_8 + x_4y_7 + x_4y_8 + x_5y_6 + x_5y_8 + x_7y_7 = 0$$

$$x_2 + x_5 + y_3 + y_6 + y_7 + x_1y_3 + x_1y_8 + x_2y_1 + x_2y_2 + x_2y_4 +$$
$$x_2y_5 + x_2y_7 + x_3y_2 + x_3y_5 + x_3y_6 + x_3y_8 + x_4y_2 + x_4y_3 +$$
$$x_4y_4 + x_4y_6 + x_4y_8 + x_5y_1 + x_5y_2 + x_5y_6 + x_5y_8 + x_6y_1 +$$
$$x_6y_2 + x_7y_8 = 0$$

$$1 + x_1 + x_3 + x_4 + x_5 + x_6 + x_8 + y_1 + y_3 + y_5 + y_8 + x_1y_1 +$$
$$x_1y_2 + x_2y_1 + x_2y_2 + x_2y_5 + x_2y_7 + x_3y_3 + x_4y_1 + x_4y_4 +$$
$$x_4y_5 + x_4y_8 + x_5y_2 + x_5y_5 + x_5y_6 + x_8y_1 = 0$$

$$x_3 + x_5 + x_6 + x_7 + x_8 + y_1 + y_2 + x_1y_1 + x_1y_2 + x_1y_4 + x_1y_7 +$$
$$x_1y_8 + x_2y_2 + x_2y_5 + x_2y_6 + x_2y_7 + x_3y_3 + x_3y_6 + x_3y_8 +$$
$$x_4y_2 + x_4y_3 + x_4y_4 + x_4y_5 + x_4y_6 + x_5y_1 + x_5y_5 + x_6y_2 +$$
$$x_8y_2 = 0$$

$$1 + x_1 + x_2 + x_3 + x_5 + x_7 + x_8 + y_1 + y_4 + y_5 + y_6 + y_7 + x_1y_1 +$$
$$x_1y_2 + x_1y_4 + x_1y_5 + x_1y_6 + x_1y_7 + x_2y_1 + x_2y_3 + x_2y_4 +$$
$$x_2y_6 + x_2y_7 + x_2y_8 + x_3y_1 + x_3y_4 + x_3y_8 + x_4y_2 + x_4y_5 +$$
$$x_5y_6 + x_8y_3 = 0$$

$$x_1 + x_5 + x_6 + x_7 + x_8 + y_2 + y_4 + y_5 + y_6 + y_8 + x_1y_1 + x_1y_2 +$$
$$x_1y_3 + x_1y_4 + x_1y_6 + x_1y_7 + x_2y_3 + x_2y_5 + x_2y_7 + x_3y_1 +$$
$$x_3y_4 + x_3y_7 + x_3y_8 + x_4y_2 + x_4y_3 + x_4y_4 + x_4y_8 + x_5y_1 +$$
$$x_5y_2 + x_5y_4 + x_5y_6 + x_6y_1 + x_8y_4 = 0$$

$$x_2 + x_7 + y_3 + y_4 + y_5 + x_1y_1 + x_1y_3 + x_1y_4 + x_1y_5 + x_2y_1 +$$
$$x_2y_3 + x_2y_6 + x_2y_8 + x_3y_1 + x_3y_3 + x_3y_4 + x_3y_5 + x_3y_6 +$$
$$x_3y_8 + x_4y_1 + x_4y_2 + x_4y_3 + x_4y_4 + x_4y_6 + x_4y_7 + x_4y_8 +$$
$$x_5y_2 + x_5y_3 + x_5y_4 + x_5y_5 + x_5y_8 + x_6y_1 + x_6y_2 + x_8y_5 = 0$$

$$x_1 + x_4 + x_5 + x_7 + y_3 + y_4 + y_5 + y_6 + y_7 + y_8 + x_1y_1 + x_1y_2 +$$
$$x_1y_3 + x_1y_7 + x_1y_8 + x_2y_5 + x_2y_7 + x_2y_8 + x_3y_2 + x_3y_6 +$$
$$x_3y_7 + x_3y_8 + x_4y_1 + x_4y_2 + x_4y_3 + x_4y_4 + x_4y_6 + x_4y_7 +$$
$$x_5y_1 + x_5y_2 + x_5y_4 + x_6y_1 + x_6y_2 + x_8y_6 = 0$$

$$1 + x_1 + x_2 + x_3 + x_5 + x_7 + x_8 + y_1 + y_2 + y_3 + y_4 + y_6 + x_1y_1 +$$
$$x_1y_4 + x_1y_5 + x_1y_6 + x_2y_2 + x_2y_4 + x_2y_6 + x_3y_1 + x_3y_3 +$$
$$x_3y_5 + x_3y_7 + x_4y_3 + x_4y_4 + x_4y_7 + x_5y_1 + x_5y_2 + x_5y_3 +$$
$$x_5y_5 + x_5y_6 + x_5y_8 + x_6y_2 + x_8y_7 = 0$$

$$1 + x_4 + x_5 + x_8 + y_4 + y_5 + y_6 + x_1y_1 + x_1y_2 + x_1y_3 + x_1y_5 +$$
$$x_1y_6 + x_1y_7 + x_1y_8 + x_2y_1 + x_2y_3 + x_2y_4 + x_2y_6 + x_3y_2 +$$
$$x_3y_6 + x_3y_7 + x_4y_1 + x_4y_2 + x_4y_4 + x_4y_5 + x_4y_6 + x_4y_7 +$$
$$x_4y_8 + x_5y_4 + x_5y_5 + x_6y_2 + x_8y_8 = 0$$