VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

KLASIFIKÁTORY PROUDOVÝCH OTISKŮ

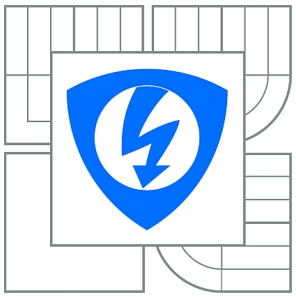
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. ONDŘEJ ZAPLETAL
AUTHOR

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

KLASIFIKÁTORY PROUDOVÝCH OTISKŮ

CLASSIFIERS OF POWER PATTERNS

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

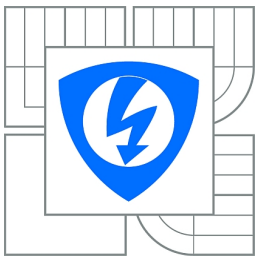
AUTOR PRÁCE — Bc. ONDŘEJ ZAPLETAL

AUTHOR

VEDOUCÍ PRÁCE — Ing. ZDENĚK MARTINÁSEK, Ph.D.

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Diplomová práce

magisterský navazující studijní obor
Telekomunikační a informační technika

Student:	Bc. Ondřej Zapletal	***ID:***	125707
Ročník:	2	***Akademický rok:***	2013/2014

NÁZEV TÉMATU:

Klasifikátory proudových otisků

POKYNY PRO VYPRACOVÁNÍ:

V rámci diplomové práce nastudujte problematiku využití neuronových sítí při analýze proudovým postranním kanálem. Vytvořte přehledný rozbor současného stavu problematiky a dnes používaných metod při proudové analýze založené na strojovém učení. Seznamte se se soutěží DPA contest 4 http://www.dpacontest.org a nastudujte a zprovozněte všechny potřebné nástroje pro realizaci a vyhodnocení útoku. Podrobně nastudujte použitou metodu maskování algoritmu AES. Navrhněte útok využívající MLP k útoku na toto maskované schéma šifrovacího algoritmu. S proudovými průběhy z DPA contest realizujte kompletní útok využívající MLP a odešlete útok do soutěže na zhodnocení. Dosažené výsledky přehledně zpracujte.

DOPORUČENÁ LITERATURA:

[1] Heuser, A., Zohner, M.: Intelligent machine homicide - breaking cryptographic devices using supportvector machines. In: COSADE. (2012) 249-264
[2] Bartkewitz, T., Lemke-Rust, K.: Ecient template attacks based on probabilistic multi-class support vector machines. In: Proceedings of the 11th international conference on Smart Card Research and Advanced Applications. CARDIS'12, Springer-Verlag (2013) 263-276
[3] Hospodar, G., Gierlichs, B., Mulder, E.D., Verbauwhede, I., Vandewalle, J.: Machine learning in side-channel analysis: a first study. J. Cryptographic Engineering 1(4) (2011) 293-302

Termín zadání:	10.2.2014	***Termín odevzdání:***	30.5.2014

Vedoucí práce: Ing. Zdeněk Martinásek, Ph.D.
Konzultanti diplomové práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

ABSTRAKT

V průběhu posledních několika let se z útoků postranními kanály stala významná hrozba pro bezpečnost kryptografických modulů. Existuje několik typů útoků postranními kanály, které lze použít pro prolomení většiny šifrovacích algoritmů (např. AES, DES, RSA). Tato diplomová práce se věnuje problematice proudových postranních kanálů, pro které existují různé metody proudové analýzy, např. jednoduchá proudová analýza (SPA), diferenciální proudová analýza (DPA), útok pomocí šablon, atd. Výše zmíněné metody jsou v práci podrobně popsány. Také je zde zkoumáno uplatnění technik strojového učení, konkrétně neuronových sítí a algoritmu SVM, v oblasti proudové analýzy. Praktická část práce se zaměřuje na prolomení maskovaného šifrovacího algoritmu AES. Jehož implementace je použita v soutěži DPA Contest.

KLÍČOVÁ SLOVA

Útoky postranními kanály, proudová analýza, SPA, DPA, útoky pomocí šablon, neuronové sítě, SVM, DPA Contest

ABSTRACT

Over the last several years side-channel analysis has emerged as a major threat to securing sensitive information in cryptographic devices. Several side-channels have been discovered and used to break implementations of all major cryptographic algorithms (AES, DES, RSA). This thesis is focused on power analysis attacks. A variety of power analysis methods has been developed to perform these attacks. These methods include simple power analysis (SPA), differential power analysis (DPA), template attacks, etc. This work provides comprehensive survey of mentioned methods and also investigates the application of a machine learning techniques in power analysis. The considered learning techniques are neural networks and support vector machines. The final part of this thesis is dedicated to implemenation of the attack against protected software AES implementation which is used in the DPA Contest.

KEYWORDS

Side channel attacks, power analysis, SPA, DPA, template attacks, neural networks, SVM, DPA Contest

ZAPLETAL, Ondřej *Klasifikátory proudových otisků*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2014. 59 s. Vedoucí práce byl Ing. Zdeněk Martinásek, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Klasifikátory proudových otisků“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce panu Ing. Zdeňku Martináskovi, Ph.D. za odborné vedení, konzultace a podnětné návrhy k práci.

Brno ....

(podpis autora)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

PODĚKOVÁNÍ

Výzkum popsaný v této diplomové práci byl realizován v laboratořích podpořených z projektu SIX; registrační číslo CZ.1.05/2.1.00/03.0072, operační program Výzkum a vývoj pro inovace.

EVROPSKÁ UNIE
EVROPSKÝ FOND PRO REGIONÁLNÍ ROZVOJ
INVESTICE DO VAŠÍ BUDOUCNOSTI

OBSAH

Úvod **10**

1 Kryptologie **11**
1.1 Útoky na kryptografické moduly 12

2 Proudová spotřeba modulu **16**
2.1 Proudová spotřeba obvodů CMOS 16
2.2 Modely proudové spotřeby 17
2.2.1 Model Hammingovy vzdálenosti 17
2.2.2 Model Hammingovy váhy 18
2.2.3 Další modely proudové spotřeby 19

3 Metody proudové analýzy **20**
3.1 Jednoduchá proudová analýza 20
3.2 Diferenciální proudová analýza 21
3.2.1 DPA útok založený na korelačním koeficientu 23
3.2.2 DPA útok založený na rozdílu středních hodnot 25
3.2.3 DPA útok založený na vzdálenosti středních hodnot 26
3.2.4 Příklad DPA útoku pomocí korelačního koeficientu 26
3.3 Útoky pomocí šablon 29
3.3.1 Základní popis 29
3.3.2 Fáze vytváření šablon 30
3.3.3 Fáze klasifikace 30
3.4 Proudová analýza pomocí strojového učení 31
3.4.1 Neuronové sítě 31
3.4.2 Support Vector Machines 33

4 Proudová analýza s využitím ANN **36**
4.1 DPA Contest 36
4.2 Implementace kryptografického modulu 36
4.3 Proudové průběhy pro DPA Contest V4 37
4.4 Obecný návrh útoku s využitím ANN 40
4.5 Realizace útoku a vyhodnocení 41
4.5.1 Nástroj AttackWrapper 41
4.5.2 Realizace útoku 43
4.5.3 Vyhodnocení 44

5 Závěr **46**

Literatura **47**

Seznam symbolů, veličin a zkratek **50**

Seznam příloh **51**

A Advanced Encryption Standard **52**

A.1 Proces šifrování pomocí AES 52

A.1.1 Key Expansion 53

A.1.2 Inicializační fáze 53

A.1.3 `SubBytes` transformace 53

A.1.4 `ShiftRows` transformace 54

A.1.5 `MixColumns` transformace 54

A.1.6 `AddRoundKey` transformace 55

A.1.7 Pruběh finální rundy 56

B Šablona pro implementaci útoku **57**

C Obsah přiloženého DVD **59**

SEZNAM OBRÁZKŮ

1.1 Obory kryptologie. 11
1.2 Typy útoků na kryptografické moduly. 13
1.3 Útoky postranními kanály na kryptografický modul. 15
1.4 Konvenční kryptoanalýza. 15
2.1 Struktura tranzistoru MOSFET s indukovaným vodivým kanálem. . . 16
2.2 Model CMOS invertoru a jeho princip činnosti. 17
2.3 Příklad modelu Hammingovy vzdálenosti. 18
2.4 Příklad modelu Hammingovy váhy. 18
3.1 Jednoduchá proudová analýza šifrovacího algoritmu AES. 20
3.2 Blokový diagram znázorňující kroky 3 až 5 DPA útoku [17]. 24
3.3 Řádky matice **R** pro hypotézy klíče 40 až 43. 28
3.4 Všechny řádky matice **R** se zvýrazněnou hypotézou klíče 43. 28
3.5 Struktura neuronové sitě. 32
3.6 Aktivní uzel neuronové sítě ve skryté a výstupní vrstvě. 32
3.7 Princip lineárního oddělení dvou tříd s nelineárními hranicemi. 34
3.8 Pohled na optimální oddělovací nadrovinu. 35
4.1 Proudový průběh z DPA Contest V4. 39
4.2 Adresářová struktura po rozbalení jednotlivých archivů. 39
4.3 Útok na maskovaný šifrovací algoritmus s využitím ANN. 40
4.4 Blokový diagram popisující funkci nástroje AttackWrapper. 41
4.5 Korelace mezi offsetem a proudovým průběhem maskovaného AES. . 43
4.6 Výpis terminálu při spuštěném útoku. 44
A.1 Dvojrozměrné pole bajtů (*Stav*). 53
A.2 Transformace `SubBytes`. 54
A.3 Transformace `ShiftRows`. 54
A.4 Transformace `MixColumns`. 55
A.5 Transformace `AddRoundKey`. 55
A.6 Princip šifrování (dešifrování) algoritmem AES-128 [3]. 56

ÚVOD

Diplomová práce se věnuje problematice kryptoanalýzy postranními kanály se zaměřením na proudovou analýzu a následnou klasifikací naměřených proudových průběhů kryptografického modulu.

První část diplomové práce obsahuje stručný souhrn důležitých pojmů z oblasti kryptologie včetně popisu základních typů útoků na kryptografické moduly.

Následující část se zabývá analýzou proudové spotřeby kryptografických modulů. Definuje základní modely proudové spotřeby, které jsou následně použity při proudové analýze modulu.

Čtvrtá část podrobně popisuje metody proudové analýzy. Kromě jednoduché a diferenciální proudové analýzy je věnována pozornost útokům pomocí šablon, které představují nejúčinnější formu útoku postranním kanálem. Dále jsou zde uvedeny metody proudové analýzy využívající techniky strojového učení, mezi které patří neuronové sítě a algoritmy SVM.

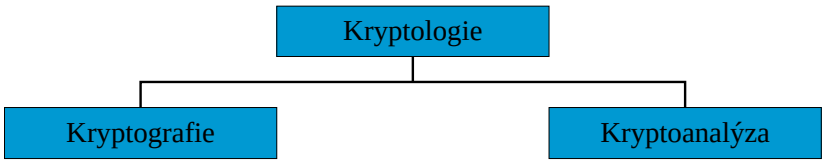
Poslední kapitolu tvoří praktická část zabývající se návrhem a implementací útoku na maskovaný šifrovací algoritmus AES. Nejprve je podrobně popsán obecný návrh útoku, který je následně implementován v prostředí Matlab. Realizovaný útok využívá kombinaci proudové analýzy založené na technice strojového učení a diferenciální proudové analýzy s korelačním koeficientem. Úspěšnost útoku je vyhodnocena v soutěži DPA Contest.

1 KRYPTOLOGIE

Kryptologie je vědní obor, který v sobě zahrnuje dva podobory, a to kryptografii a kryptoanalýzu (viz obr. 1.1). Kryptografie se zabývá problematikou návrhu a implementace kryptografických primitiv. Kryptografická primitiva tvoří základní stavební bloky např. šifer, hash funkcí nebo podepisovacích schémat pro dosažení:

- **důvěrnosti**, tedy utajení přenášených zpráv před neoprávněnou osobou.
- **Autentičnosti**, pro možnost ověření si původu zprávy.
- **Integrity**, kdy zpráva nesmí být během přenosu modifikována.
- **Nepopiratelnosti**, tedy schopnosti prokázat totožnost autora, který zprávu vytvořil (např. digitálním podpisem).

Cílem kryptoanalýzy je získat utajované informace z kryptografických systémů. Pomocí různých metod kryptoanalýzy je možné určit jejich zabezpečení. Proces kryptoanalýzy umožňuje návrhářům vytvářet kryptografické systémy s co možná největším stupněm zabezpečení.



Obr. 1.1: Obory kryptologie.

Moderní kryptografické systémy využívají k zajištění důvěrnosti, autentičnosti a integrity dat šifrovací algoritmy. Jsou to matematické funkce nejčastěji se dvěma vstupními parametry. Prvním z nich je zpráva, zpravidla označována jako otevřený text, a druhým je tajný klíč. Šifrovací algoritmus na svém výstupu vytváří zašifrovaný text. Tomuto procesu se říká šifrování, opačnému pak dešifrování.

Kryptografie se dělí na symetrickou a asymetrickou. Komunikující strany v symetrické kryptografii sdílí společný tajný klíč. Typickým představitelem symetrického šifrovacího algoritmu je Advanced Encryption Standard (AES) [7]. Je to bloková šifra, která pracuje s bloky dat pevně dané velikosti. V případě algoritmu AES nabývá hodnota bloku 128 bitů. Velikost klíče může být 128, 192 nebo 256 bitů. Algoritmus AES je podle toho označován: AES-128, AES-192 a AES-256. Popis činnosti algoritmu AES je uveden v příloze A.

V asymetrické kryptografii se používají dva typy klíčů. První typ se využívá při šifrování, tzv. veřejný klíč, a druhý při dešifrování, tzv. soukromý klíč. V současnosti

existuje celá řada asymetrických šifrovacích algoritmů. Nejznámějším je pravděpodobně Rivest-Shamir-Adleman algoritmus (RSA) [24], který používá klíče o délce až 4096 bitů.

Prolomit šifrovací algoritmus znamená nalézt tajný klíč na základě veřejně dostupné informace. Touto informací mohou být otevřené a zašifrované texty. Šifrovací algoritmus je považován za bezpečný, nemůže-li být prolomen v dostatečně krátkém čase a k jeho prolomení je potřeba nadměrný výpočetní výkon, který není v praxi dosažitelný. Náročnost prolomení u mnohých algoritmů roste exponenciálně s velikostí klíče. Z toho vyplývá, že délka klíče významně ovlivňuje bezpečnost šifrovacího algoritmu.

Zařízení, které má v sobě implementován konkrétní šifrovací algoritmus se nazývá kryptografický modul. Samotný modul s šifrovacím algoritmem může být realizován hardwarově, softwarově nebo kombinací obou principů. Při vykonávání kryptografických operací pracují moduly s citlivými daty (např. tajný klíč), které uchovávají ve své paměti. Také komunikují s okolním prostředím.

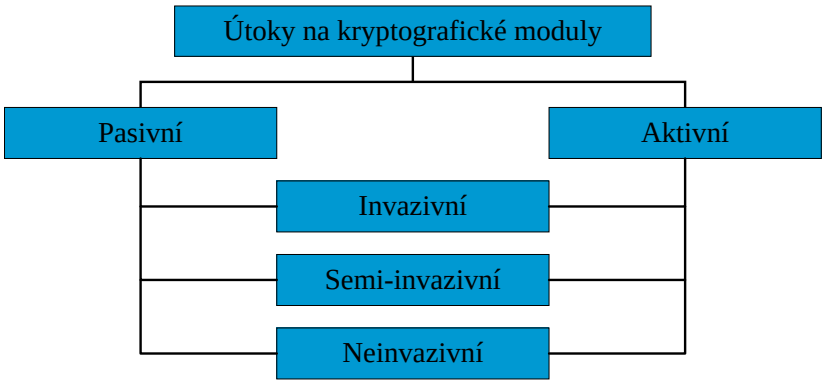
Použití kryptografických modulů vnáší do bezpečnosti šifrovacích algoritmů nové problémy. V praxi je důležitá bezpečnost celého kryptografického systému. Tedy bezpečnost nejen šifrovacího algoritmu, ale i kryptografického modulu.

Prolomit kryptografický modul znamená získat aktuálně používaný tajný klíč. Osoba, jejíž snahou je získat tento klíč, se nazývá útočník. Samotný pokus o získání klíče se nazývá útok na kryptografický modul. K vyhodnocení úrovně zabezpečení kryptografického modulu je nezbytné brát do úvahy odbornost útočníka a jeho znalosti o daném modulu.

1.1 Útoky na kryptografické moduly

V průběhu posledních let bylo nalezeno mnoho typů útoků na kryptografické moduly s cílem získat tajný klíč. Techniky využívané pro dosažení tohoto cíle lze rozdělit podle několika kritérií.

Pro útok na kryptografický modul a možné získání tajné informace jsou zejména potřebné finanční prostředky. Důležitou úlohu hraje časová náročnost útoku, přístup k potřebnému vybavení a také odborné znalosti útočníka. Ve většině odborných publikací zabývajících se útoky na kryptografické moduly jsou zmiňována dvě základní kritéria pro dělení těchto útoků (viz obr. 1.2). První kritérium je, zda je útok pasivní či aktivní.



Obr. 1.2: Typy útoků na kryptografické moduly.

Pasivní útoky

Při pasivním útoku pracuje kryptografický modul podle daných specifikací a norem. Útok je zaměřen na pozorování fyzikálních projevů daného modulu (tzn. čas potřebný k šifrování či dešifrování, proudový odběr z napájecího zdroje atd.)

Aktivní útoky

Při aktivním útoku je snahou útočníka ovlivnit vstupy kryptografického modulu nebo prostředí, ve kterém se nachází. Tato manipulace způsobí netypické chování modulu a vyvolá jeho chybový stav. Tajný klíč je poté možné odhalit využitím tohoto abnormálního chování.

Druhé kritérium se zaměřuje na dostupná rozhraní kryptografických modulů. Každý kryptografický modul má několik fyzických a logických rozhraní. Podle přístupu k jednotlivým rozhraním, která jsou využívána k útoku, je možné rozlišit invazivní, semi-invazivní a neinvazivní útoky. Každý z těchto útoků může být buď pasivní, nebo aktivní.

Invazivní útoky

Invazivní útok patří mezi nejefektivnější druh útoku na kryptografický modul. K získání tajného klíče je možno využít veškeré dostupné prostředky.

Při invazivním útoku se nejprve odstraní pouzdro chránící modul za účelem získat přímý přístup k jeho vnitřním komponentům. Pomocí měřicí sondy lze poté zachytit např. informace přenášené po datové sběrnici kryptografického modulu. Tato část invazivního útoku je pasivní, pokud je sonda použita pouze k měření signálů na

sběrnici. Lze však také přerušit vodivé cesty sběrnice a ovlivnit tak proces probíhající uvnitř modulu. Invazivní útok se poté stane aktivním. K tomuto záměru je možno využít laserový nůž či fokusovaný paprsek iontů.

Kryptografické moduly s vyšší úrovní zabezpečení mohou obsahovat technologie, které se snaží invazivním útokům zabránit. Nejčastější technikou je detekce pokusu o narušení fyzické ochrany. Při zjištění pokusu o průnik k vnitřním částem modulu se jeho paměť vynuluje, dále může přerušit svou činnost a informovat obsluhu.

Invazivní útoky jsou velice účinné. Pro jejich vykonání je však potřeba drahé vybavení. Často dochází také k úplnému zničení modulu.

Semi-invazivní útoky

Při semi-invazivním útoku je potřebný přístup k vnitřním částem kryptografického modulu, není ale nutné vytvořit přímý kontakt s vodivými cestami uvnitř modulu. Ochranná vrstva polovodičů uvnitř modulu zůstává neporušená.

Semi-invazivní útok může být též pasivní či aktivní. Aktivním se stává pokud je útočníkem do kryptografického modulu úmyslně zavedena chyba. To může být provedeno např. rentgenovými paprsky, světelnými paprsky nebo elektromagnetickým polem.

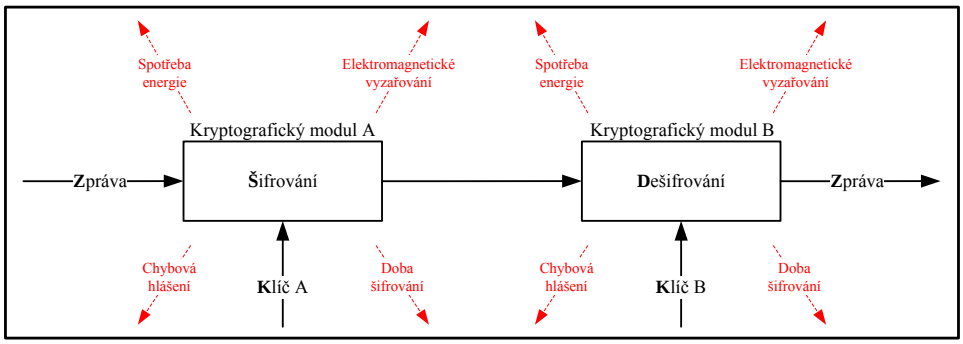
Tento typ útoku nevyžaduje tak drahé vybavení jako invazivní útok. Stále je však na provedení velice náročný a vyžaduje dostatek času a potřebné znalosti.

Neinvazivní útoky

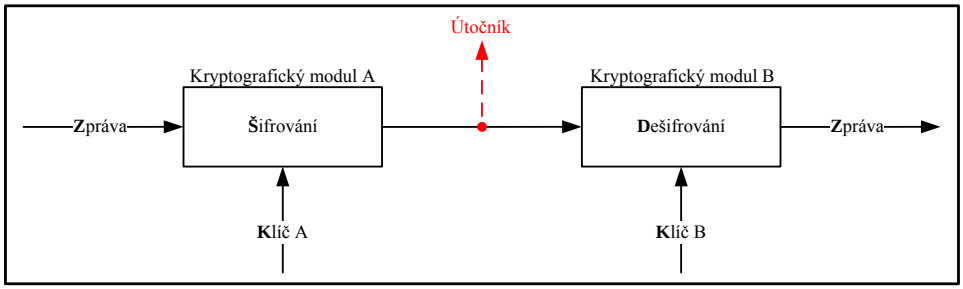
Neinvazivní útok na kryptografický modul využívá pouze jeho přímo přístupných rozhraní. Není ovlivněna správná funkce modulu a útočník nezanechá za sebou žádné stopy. Útok je tedy nedetekovatelný. Protože pro většinu neinvazivních útoků stačí relativně levné vybavení, představují tyto útoky obrovské riziko z hlediska bezpečnosti kryptografických modulů.

V současné době se obrací pozornost zejména na pasivní neinvazivní útoky, tedy útoky postranními kanály (viz obr. 1.3), které nahrazují útoky hrubou silou (viz obr. 1.4). Mezi nejčastěji se vyskytující typy patří časový postranní kanál [12], proudový postranní kanál [14] a útok elektromagnetickým postranním kanálem [20]. Tajný klíč z kryptografického modulu může být získán např. měřením času potřebného pro vykonání daných operací, jeho následnou analýzou a porovnáním s procesy probíhajícími uvnitř modulu. Je také možné měřit proudový odběr modulu z napájecího zdroje nebo jeho elektromagnetické pole.

Kromě útoků postranními kanály existují i aktivní neinvazivní útoky. Mohou využívat změn napájecího napětí nebo teploty prostředí, ve kterém se modul nachází.



Obr. 1.3: Útoky postranními kanály na kryptografický modul.



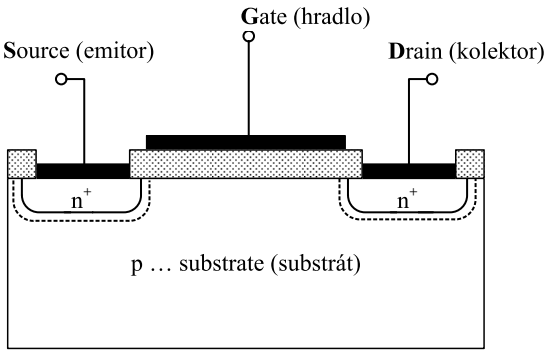
Obr. 1.4: Konvenční kryptoanalýza.

2 PROUDOVÁ SPOTŘEBA MODULU

Proudový odběr kryptografického modulu z napájecího zdroje není konstantní v čase, ale mění se v závislosti na aktuálně zpracovávaných datech a probíhajících operacích. Útok proudovým postranním kanálem je založen na analýze proudové spotřeby modulu PA (Power Analysis) a jejím vztahem k právě vykonávaným operacím uvnitř modulu.

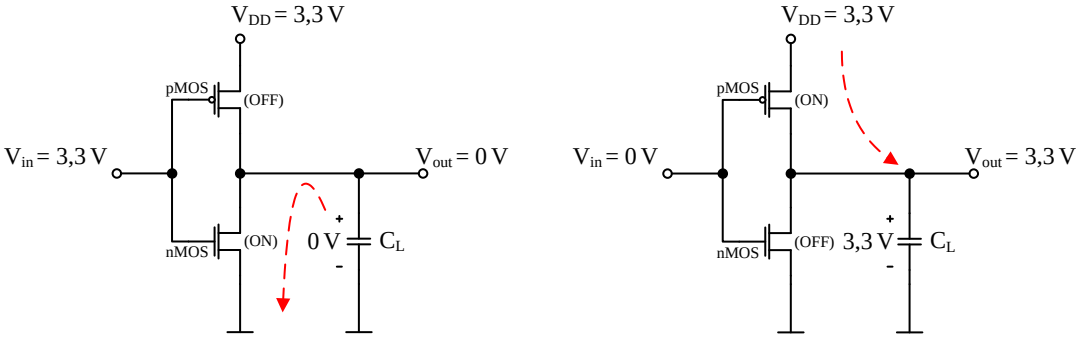
2.1 Proudová spotřeba obvodů CMOS

Digitální elektronické obvody uvnitř kryptografických modulů jsou nejčastěji tvořeny z napěťově řízených tranzistorů (obr. 2.1) vyráběné technologií CMOS (Complementary Metal-Oxide Semiconductor).



Obr. 2.1: Struktura tranzistoru MOSFET s indukovaným vodivým kanálem.

Primárním stavebním prvkem logiky založené na technologii CMOS je invertující člen (obr. 2.2). Skládá se ze dvou tranzistorů pMOS a nMOS zapojených jako spínače řízené napětím. Pokud je na vstup prvního invertoru přivedena log. 1, pMOS tranzistor je uzavřen (OFF) a nMOS tranzistor otevřen (ON). Na výstupu prvního invertoru bude log. 0. Druhý invertor má na svém vstupu log. 0. V této situaci je pMOS tranzistor otevřen (ON) a nMOS tranzistor uzavřen (OFF). Na výstupu druhého invertoru bude log. 1. Celková proudová spotřeba invertoru může být rozdělena na dvě základní části [25]. První část tvoří statická proudová spotřeba, kdy invertor setrvává v ustáleném stavu. Její hodnota je velice malá. Druhá část je tvořena dynamickou proudovou spotřebou nastávající při změně stavů z log. 0 na log. 1 a obráceně. Dochází ke vzniku proudových špiček, které jsou způsobeny nabíjením a vybíjením parazitní kapacitní zátěže připojené na výstupy invertorů.



Obr. 2.2: Model CMOS invertoru a jeho princip činnosti.

2.2 Modely proudové spotřeby

Při útoku proudovým postranním kanálem je často nezbytné přiřadit hodnotám dat zpracovávaných kryptografickým modulem hodnoty proudové spotřeby. Tento proces bývá označován jako simulace proudové spotřeby modulu. Je důležité podotknout, že při útoku proudovým postranním kanálem nejsou důležité absolutní hodnoty proudové spotřeby, ale pouze relativní rozdíly mezi simulovanými hodnotami proudové spotřeby.

Vzhledem k omezeným znalostem útočníka o napadeném kryptografickém modulu bývají simulace proudové spotřeby často velice jednoduché. V následující části budou představeny dva základní modely proudové spotřeby. Model Hammingovy vzdálenosti (HD) a model Hammingovy váhy (HW). Ve stručnosti budou zmíněny jejich možné varianty.

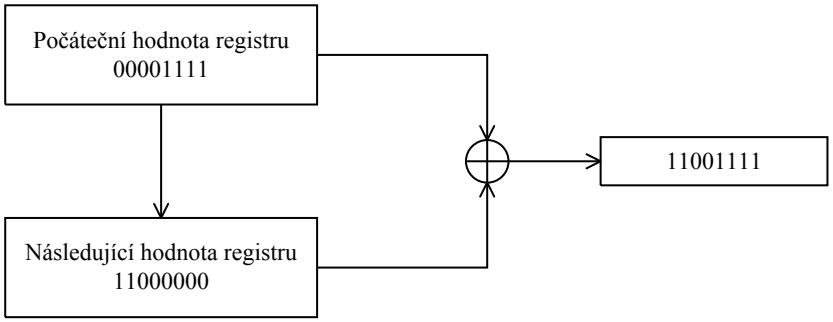
2.2.1 Model Hammingovy vzdálenosti

Hlavní myšlenkou tohoto modelu je určit počet přechodů mezi logickými hodnotami $0 \rightarrow 1$ a $1 \rightarrow 0$ v digitálním obvodě za určitý časový interval. Výsledný počet přechodů v daném časovém intervalu je poté použit k popisu proudové spotřeby obvodu. V případě velmi krátkých časových intervalů lze získat proudový průběh, který bude obsahovat počet přechodů vyskytujících se v příslušném časovém intervalu.

Model Hammingovy vzdálenosti předpokládá, že logické členy uvnitř digitálního obvodu přispívají k jeho proudové spotřebě rovnoměrně a nejsou tedy žádné rozdíly mezi přechody $0 \rightarrow 1$ a $1 \rightarrow 0$. Hammingova vzdálenost mezi dvěma hodnotami v_0 a v_1 může být určena jako $HD(v_0, v_1) = HW(v_0 \oplus v_1)$.

Často bývá model Hammingovy vzdálenosti používán pro popis proudové spotřeby datových či adresových sběrnic, registrů nebo jiných částí digitálního obvodu.

Obr. 2.3 znázorňuje změnu počáteční hodnoty v_0 uvnitř registru na hodnotu v_1 po příchodu hodinového signálu. Změna hodnoty uvnitř registru si vyžádá určité množství energie. Výsledná Hammingova vzdálenost ukazuje, že množství odebrané energie je úměrné číslu 6.



Obr. 2.3: Příklad modelu Hammingovy vzdálenosti.

2.2.2 Model Hammingovy váhy

Model Hammingovy váhy je mnohem jednodušší než výše uvedený model. V případě modelu Hammingovy váhy se předpokládá, že proudová spotřeba např. datové sběrnice je úměrná počtu datových bitů s logickou hodnotou 1. Jestliže žádný datový bit nebude nastaven do logické 1, proudová spotřeba sběrnice bude výrazně nižší než se všemi bity nastavenými do logické hodnoty 1. Příklad je zobrazen na obr. 2.4.

Obr. 2.4: Příklad modelu Hammingovy váhy.

Tento model bývá aplikován v případech, kdy útočník nemá dostatek informací o napadeném kryptografickém modulu nebo v případech, kdy nezná po sobě jdoucí hodnoty dat přenášených po sběrnici. V praxi poskytuje pouze nedokonalý popis proudové spotřeby modulu. Proto by měl být používán pouze v omezených podmínkách.

2.2.3 Další modely proudové spotřeby

Výše popsané modely Hammingovy vzdálenosti a Hammingovy váhy jsou velice jednoduché, a proto jsou útočníky využívány pro popis proudové spotřeby kryptografického modulu nebo jeho části nejčastěji. Další modely proudové spotřeby rozšiřují předchozí dva základní. Zpravidla bývají vytvářeny speciálně pro daný typ kryptografického modulu.

Např. původní model Hammingovy vzdálenosti může být rozšířen o možnost přiřazovat odlišné váhy pro každý typ přechodu. Přechodu z logické hodnoty 0 na hodnotu 1 může být přiřazena dvakrát vyšší váha než z hodnoty 1 na hodnotu 0.

3 METODY PROUDOVÉ ANALÝZY

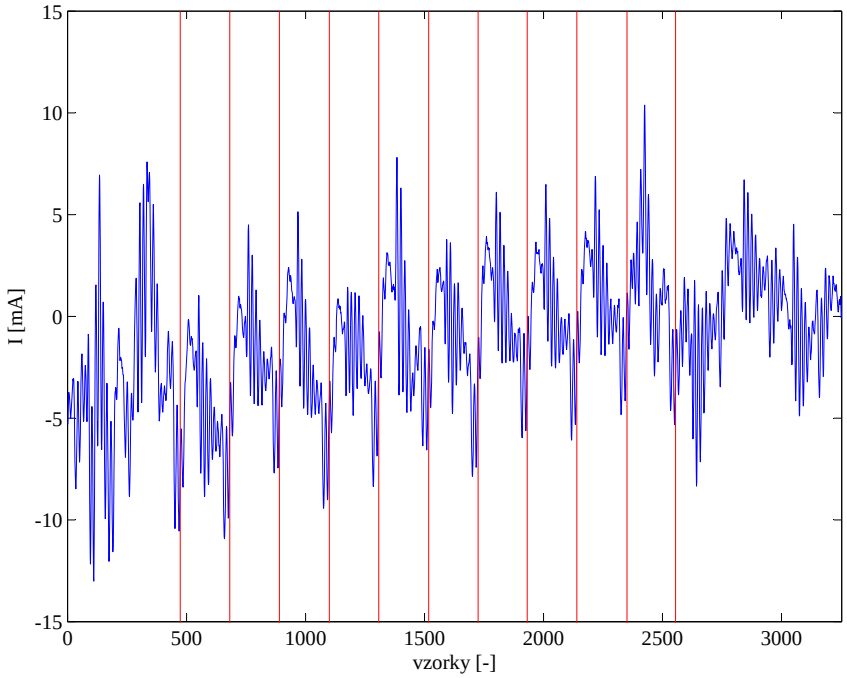
Pro získání tajného klíče z kryptografického modulu, za pomoci proudové analýzy, byly navrženy a prakticky realizovány různé metody. Obecně tyto metody zahrnují určitou formu digitálního zpracování signálů DSP (Digital Signal Processing) a výpočetní statistiky. Tato kapitola shrnuje nejpoužívanější metody proudové analýzy od nejjednodušších až po metody využívající mechanismy strojového učení.

3.1 Jednoduchá proudová analýza

Jednoduchá proudová analýza SPA (Simple Power Analysis) je postavena na přímém pozorování proudového odběru kryptografického modulu z napájecího zdroje. Účelem tohoto pozorování je odhadnout, kterou kryptografickou operaci v daném čase modul provádí a s jakými hodnotami dat pracuje. Útočník musí k provedení tohoto útoku znát přesnou implementaci šifrovacího algoritmu uvnitř modulu.

Útočník může při SPA útoku disponovat pouze jedním proudovým průběhem. Obvykle však má k dispozici několik průběhů. Tyto průběhy mohou být změřeny pro jeden otevřený text, nebo pro několik různých textů. Výhoda většího množství průběhů pro stejný otevřený text spočívá v možnosti redukovat šum.

Obr. 3.1 zachycuje průběh proudové spotřeby kryptografického modulu při šifrování algoritmem AES. V proudovém průběhu je jasně patrný vzor, který se desetkrát opakuje. Odpovídá deseti rundám algoritmu AES v jeho 128 bitové verzi.



Obr. 3.1: Jednoduchá proudová analýza šifrovacího algoritmu AES.

Informace o celkovém počtu rund algoritmu AES je veřejně dostupná [7], a proto útočník nezískal z modulu žádnou citlivou informaci. Nicméně, pouhé vizuální prozkoumání proudového průběhu může být přípravným krokem k účinnějšímu útoku, např. rozhodnutím, jaké části průběhu jsou pro útočníka důležité.

Tento typ útoku může být použit k prolomení implementace šifrovacího algoritmu RSA, u kterého lze pozorovat rozdíl ve spotřebě proudu mezi operacemi násobení a umocnění. Podobně pro většinu implementací šifrovacího algoritmu DES je možné pozorovat rozdíl ve spotřebě proudu pro operace permutace a posunu [13].

3.2 Diferenciální proudová analýza

Diferenciální proudová analýza DPA (Differential Power Analysis) je účinnější a efektivnější než výše zmíněná SPA. Při útoku prostřednictvím DPA nemusíme znát důkladně napadený kryptografický modul. Obvykle postačuje znalost o tom, jaký šifrovací algoritmus daný modul obsahuje.

DPA vyžaduje velké množství naměřených průběhů proudové spotřeby kryptografického modulu při šifrování či dešifrování vstupních dat. Ze zjištěných průběhů je možné získat tajný klíč. K tomu DPA využívá matematický aparát, zejména statistické analýzy a technik pro korekci chyb umožňujících odhalit tajný klíč i z průběhů obsahujících velký šum.

Při provádění DPA útoku se postupuje podle následujících pěti kroků [17].

Krok 1: Volba mezivýsledku vykonávaného šifrovacího algoritmu

V prvním kroku DPA útoku se zvolí mezivýsledek šifrovacího algoritmu, který je vykonáván kryptografickým modulem. Tento mezivýsledek musí být funkcí $f(d, k)$, kde d jsou známá vstupní data (zpravidla otevřený či zašifrovaný text) a k představuje malou část použitého klíče, jenž lze odhadnout (např. první bajt).

Krok 2: Změření průběhů proudové spotřeby při šifrování

Druhým krokem DPA útoku je změření proudové spotřeby kryptografického modulu při šifrování nebo dešifrování různých bloků dat D. Pro všechny operace šifrování či dešifrovaní potřebuje útočník znát hodnoty zpracovávaných dat d, které se podílí na výpočtu mezivýsledku zvoleného v kroku 1. Hodnoty známých dat tvoří vektor $\mathbf{d} = (d_1, \ldots, d_D)'$, kde d_i označuje hodnotu i-tého zpracovaného bloku vstupních dat.

Při vykonávání těchto operací je útočníkem zaznamenávána proudová spotřeba kryptografického modulu. Každému průběhu spotřeby $\mathbf{t}'_i = (t_{i,1}, \ldots, t_{i,T})$, kde T označuje dobu trvání průběhu, odpovídá jedna hodnota zpracovávaných dat d_i. Útočník měří proudovou spotřebu pro každý zpracovávaný blok dat D, a proto mohou být průběhy zapsány jako matice $\mathbf{T}$ o velikosti $D \times T$. Pro DPA útok je klíčové, aby měřené průběhy proudové spotřeby byly správně zarovnané. To znamená, že hodnoty proudové spotřeby v libovolném sloupci $\mathbf{t}_j$ matice $\mathbf{T}$ musí odpovídat stejné operaci. Toho lze dosáhnout správnou synchronizací použitého měřicího zařízení.

Krok 3: Sestavení matice hypotéz mezivýsledků

Dalším krokem útoku je pro všechny možné hodnoty klíče k určit hypotetické mezivýsledky. Možné hodnoty klíče lze zapsat jako vektor $\mathbf{k} = (k_1, \ldots, k_K)$, kde K označuje celkový počet možných klíčů. Jednotlivým prvkům vektoru se říká hypotézy neboli odhady klíče. Z vektoru známých dat $\mathbf{d}$ a vektoru hypotéz všech klíčů je útočník schopen jednoduše vypočítat hypotetické mezivýsledky $f = (d, k)$ pro všechny šifrovací operace D a pro všechny hypotézy klíče K. Výsledkem bude matice $\mathbf{V}$ o velikosti $D \times K$ s prvky

$$v_{i,j} = f(d_i, k_j)\ i = 1, \ldots, D\ j = 1, \ldots, K. \tag{3.1}$$

Sloupec j matice $\mathbf{V}$ obsahuje mezivýsledky, které byly vypočítány na základě příslušné hypotézy klíče k_j. Jeden sloupec matice $\mathbf{V}$ tedy obsahuje ty mezivýsledky, které byly kryptografickým modulem vypočítány během D operací šifrování a dešifrování. Hodnota klíče uvnitř modulu je prvkem vektoru $\mathbf{k}$. Index tohoto prvku je označen ck. Klíč používaný modulem poté odpovídá prvku k_{ck}. Cílem DPA útoku je nalézt, jaký sloupec matice $\mathbf{V}$ byl zpracováván během D operací šifrování a dešifrování a získat tak k_{ck}.

Krok 4: Určení závislosti proudové spotřeby na mezivýsledku

Čtvrtým krokem DPA útoku je namapování matice $\mathbf{V}$ obsahující hypotézy mezivýsledků do matice $\mathbf{H}$ reprezentující hypotézy proudové spotřeby. V tomto kroku se využívá simulace proudové spotřeby kryptografického modulu (viz kapitola 2.2). Vytvořený model spotřeby přiřadí každému hypotetickému mezivýsledku $v_{i,j}$ hypotetickou hodnotu proudové spotřeby $h_{i,j}$.

Čím větší má útočník znalosti o analyzovaném modulu, tím lepší simulaci spotřeby je schopen vytvořit a tím zefektivnit DPA útok. Mezi často využívané modely spotřeby patří model Hammingovy vzdálenosti a Hammingovy váhy.

Krok 5: Porovnání hypotetických hodnot proudové spotřeby se změřenými průběhy

V posledním kroku DPA útoku se porovnávají hypotetické hodnoty proudové spotřeby závislé na odhadu klíče (hodnoty ve sloupci $\mathbf{h}_i$ matice $\mathbf{H}$) se změřenými průběhy (hodnoty ve sloupci $\mathbf{t}_j$ matice $\mathbf{T}$). Výsledkem je matice $\mathbf{R}$ o velikosti $K \times T$. Každý prvek $r_{i,j}$ matice $\mathbf{R}$ je výsledkem porovnání mezi sloupci $\mathbf{h}_i$ a $\mathbf{t}_j$. Čím větší je hodnota prvku $r_{i,j}$, tím je míra lineární závislosti (korelace) mezi sloupci $\mathbf{h}_i$ a $\mathbf{t}_j$ větší. Porovnávání může být provedeno prostřednictvím různých metod, které budou popsány v následující části. Na obr. 3.2 jsou graficky znázorněny kroky 3 až 5.

Naměřené průběhy vyjadřují proudovou spotřebu kryptografického modulu při vykonávání šifrovacího algoritmu pro různá vstupní data. Zvolený mezivýsledek z kroku 1 je součástí tohoto algoritmu. Modul tedy během každého procesu šifrování či dešifrování různých vstupních dat pracuje s mezivýsledky $\mathbf{v}_{ck}$. Tzn., že naměřené průběhy jsou v určitých pozicích na těchto mezivýsledcích závislé. Tuto pozici lze označit jako ct a platí, že hodnoty proudové spotřeby ve sloupci $\mathbf{t}_{ct}$ závisí na mezivýsledcích $\mathbf{v}_{ck}$.

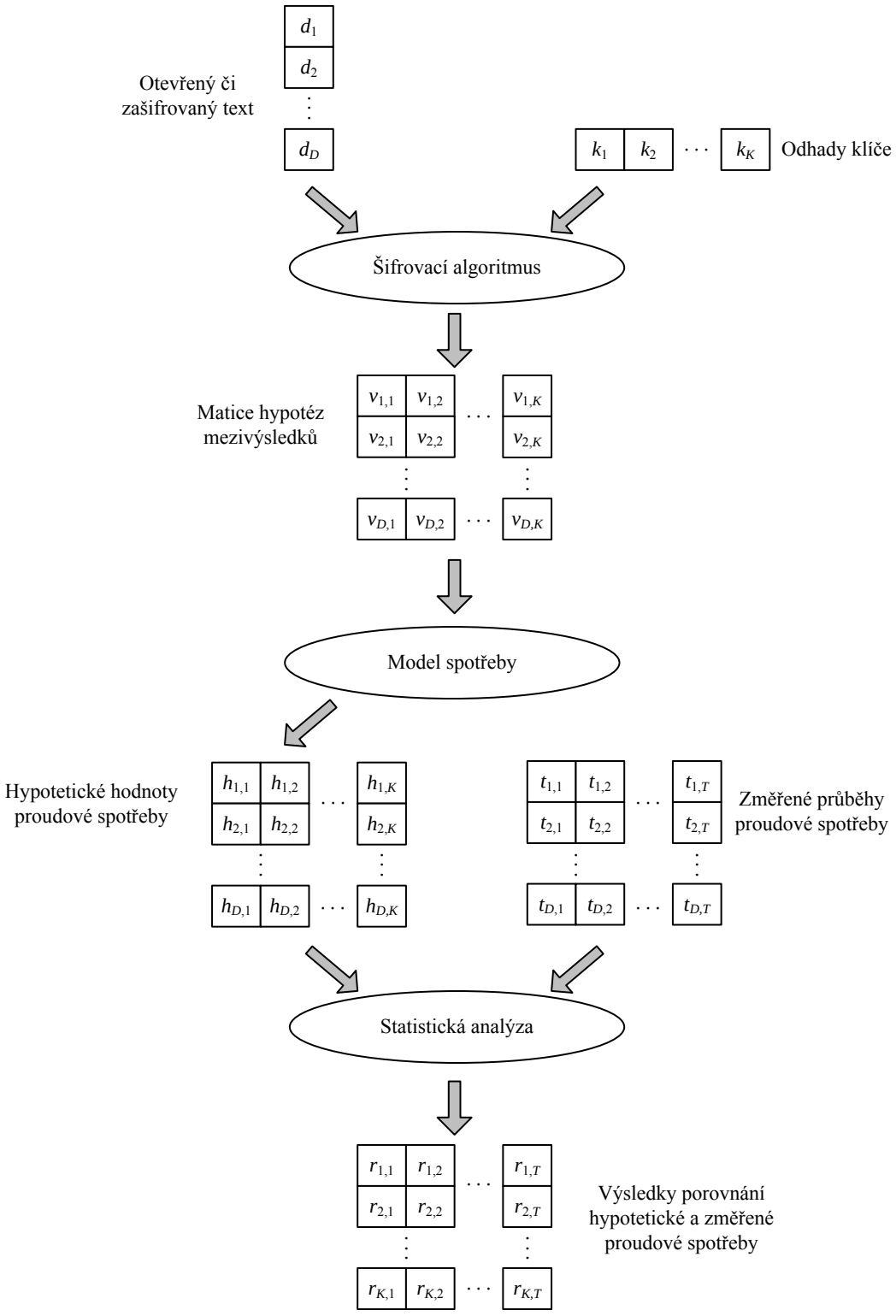
Na základě mezivýsledků $\mathbf{v}_{ck}$ byly útočníkem nasimulovány hypotetické hodnoty proudové spotřeby $\mathbf{h}_{ck}$. Proto platí, že sloupce $\mathbf{h}_{ck}$ a $\mathbf{t}_{ct}$ jsou na sobě silně závislé. Jejich korelací vznikne hodnota $r_{ck,ct}$ v matici $\mathbf{R}$, která bude nejvyšší v celé této matici. Protože ostatní sloupce matic $\mathbf{H}$ a $\mathbf{T}$ neprokazují takovou závislost, budou všechny ostatní hodnoty v matici $\mathbf{R}$ menší. Útočník je tedy schopen získat správný klíč k_{ck} pouhým nalezením nejvyšší hodnoty v matici $\mathbf{R}$.

Při praktickém provedení DPA útoku se může stát, že hodnoty v matici $\mathbf{R}$ nabývají přibližně stejných hodnot. To je obvykle způsobeno nezměřením dostatečného množství průběhů proudové spotřeby ke stanovení závislosti mezi sloupci matic $\mathbf{H}$ a $\mathbf{T}$. Čím více průběhů bude naměřeno, tím budou sloupce matic $\mathbf{H}$ a $\mathbf{T}$ obsahovat více prvků a tím lze lépe charakterizovat vztah mezi sloupci.

3.2.1 DPA útok založený na korelačním koeficientu

Ve statistice představuje korelace vzájemný lineární vztah mezi dvěma náhodnými proměnnými X a Y. Míru korelace vyjadřuje korelační koeficient $\rho(X, Y)$, který nabývá hodnot v intervalu $\langle -1, 1 \rangle$. Korelační koeficient [17] je definován pomocí kovariance vztahem

$$\rho(X, Y) = \frac{\mathrm{cov}(X, Y)}{\sqrt{\sigma^2(X) \cdot \sigma^2(Y)}}, \tag{3.2}$$



Obr. 3.2: Blokový diagram znázorňující kroky 3 až 5 DPA útoku [17].

kde kovariance $\mathrm{cov}(X, Y)$ představuje střední hodnotu součinu odchylek obou proměnných X a Y od jejich středních hodnot a $\sigma^2(X)$ a $\sigma^2(Y)$ představují rozptyl proměnných X a Y.

Pokud X a Y jsou kvantitativní náhodné veličiny se společným dvourozměrným normálním rozdělením, tak pro konkrétní hodnoty (x_i, y_i), kde $i = 1, 2, \ldots, n$, je korelační koeficient [17] určen vztahem

$$r = \frac{\sum_{i=1}^{n} (x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n} (x_i - \bar{x})^2 \cdot \sum_{i=1}^{n} (y_i - \bar{y})^2}}. \tag{3.3}$$

Při DPA útoku se korelační koeficient používá pro stanovení lineární závislosti mezi sloupci $\mathbf{h}_i$ a $\mathbf{t}_j$, kde $i = 1, \ldots, K$ a $j = 1, \ldots, T$. Výsledkem je matice korelačních koeficientů $\mathbf{R}$. Každý prvek $r_{i,j}$ matice $\mathbf{R}$ je určen D prvky sloupců $\mathbf{h}_i$ a $\mathbf{t}_j$. Využitím předchozího vztahu 3.3 lze získat vztah pro výpočet hodnoty prvku $r_{i,j}$

$$r_{i,j} = \frac{\sum_{d=1}^{D} (h_{d,i} - \bar{h}_i) \cdot (t_{d,j} - \bar{t}_j)}{\sqrt{\sum_{d=1}^{D} (h_{d,i} - \bar{h}_i)^2 \cdot \sum_{d=1}^{D} (t_{d,j} - \bar{t}_j)^2}}, \tag{3.4}$$

kde $\bar{h}_i$ a $\bar{t}_j$ značí střední hodnoty sloupců $\mathbf{h}_i$ a $\mathbf{t}_j$.

3.2.2 DPA útok založený na rozdílu středních hodnot

V prvním článku o DPA útoku [14] byla pro porovnání sloupců matic $\mathbf{H}$ a $\mathbf{T}$ použita tzv. metoda rozdílu středních hodnot.

Útočník vytvoří binární maticí $\mathbf{H}$, kdy předpokládá, že proudová spotřeba pro určité hodnoty mezivýsledků se liší od proudové spotřeby všech ostatních hodnot. Posloupnost logických nul a jedniček v každém sloupci matice $\mathbf{H}$ je funkcí vstupních dat $\mathbf{d}$ a možné hodnoty klíče k_i. K ověření, zda daný odhad hodnoty klíče k_i je správný, může útočník podle logických hodnot ve sloupci $\mathbf{h}_i$ rozdělit matici $\mathbf{T}$ na dvě skupiny řádků (proudových průběhů). První skupina obsahuje jen ty řádky matice $\mathbf{T}$, jejichž indexy odpovídají indexům vektoru $\mathbf{h}_i$ s hodnotami logické nuly. Druhá skupina obsahuje všechny zbývající řádky matice $\mathbf{T}$. Útočník poté může určit střední hodnoty řádků. Vektor $\mathbf{m}'_{0i}$ představuje střední hodnotu řádků v první skupině a $\mathbf{m}'_{1i}$ ve druhé skupině. Odhad hodnoty klíče k_i je správný, jestliže v určitém čase nastane mezi vektory $\mathbf{m}'_{0i}$ a $\mathbf{m}'_{1i}$ významný rozdíl.

Rozdíl mezi vektory $\mathbf{m}'_{0i}$ a $\mathbf{m}'_{1i}$ značí, že mezi $\mathbf{h}_{ck}$ a některými sloupci matice $\mathbf{T}$ je vzájemný vztah. Podobně jako v předchozím útoku pomocí korelačního koeficientu nastane tento rozdíl v momentě, kdy šifrovací algoritmus pracuje s hodnotou mezivýsledku odpovídající $\mathbf{h}_{ck}$. V případě, kdy odhad klíče není správný, bude rozdíl mezi vektory $\mathbf{m}'_{0i}$ a $\mathbf{m}'_{1i}$ víceméně nulový. Výsledkem DPA útoku založeném na rozdílu středních hodnot je matice $\mathbf{R}$, ve které každý řádek této matice představuje rozdíl

mezi vektory $\mathbf{m}'_{0i}$ a $\mathbf{m}'_{1i}$ pro jeden odhad klíče. Výpočet matice $\mathbf{R}$ [17] je proveden pomocí níže uvedených vztahů

$$m_{1i,j} = \frac{1}{n_{1i}} \cdot \sum_{l=1}^{n} h_{l,i} \cdot t_{l,j}, \tag{3.5}$$

$$m_{0i,j} = \frac{1}{n_{0i}} \cdot \sum_{l=1}^{n} (1 - h_{l,i}) \cdot t_{l,j}, \tag{3.6}$$

$$n_{1i} = \sum_{l=1}^{n} h_{l,i}, \tag{3.7}$$

$$n_{0i} = \sum_{l=1}^{n} (1 - h_{l,i}) \text{ a} \tag{3.8}$$

$$\mathbf{R} = \mathbf{M}_1 - \mathbf{M}_0, \tag{3.9}$$

kde n označuje počet řádků matice $\mathbf{H}$, tedy počet proudových průběhu použitých při útoku.

3.2.3 DPA útok založený na vzdálenosti středních hodnot

Metoda vzdálenosti středních hodnot zdokonaluje metodu rozdílu středních hodnot z kapitoly 3.2.2, protože bere do úvahy i rozptyl. Je založena na statistickém testu, který určuje, zda střední hodnoty dvou rozdělení pravděpodobnosti se rovnají či nikoli.

Matice $\mathbf{T}$ se opět rozdělí na dvě skupiny řádků pro všechny možné hodnoty klíče. Střední hodnoty nejsou nyní od sebe pouze odečteny, ale jsou porovnány pomocí testu vzdálenosti středních hodnot. Prvky matice $\mathbf{R}$ [17] lze vypočítat následujícím vztahem

$$r_{i,j} = \frac{m_{1i,j} - m_{0i,j}}{s_{i,j}}, \tag{3.10}$$

kde $s_{i,j}$ je směrodatná odchylka rozdělení pravděpodobnosti daných dvou skupin.

3.2.4 Příklad DPA útoku pomocí korelačního koeficientu

V následujícím příkladě DPA útoku[1] je do mikrokontroléru kryptografického modulu implementována jen část algoritmu AES. AES vykonává pouze operaci `AddRoundKey` a operaci `SubBytes`.

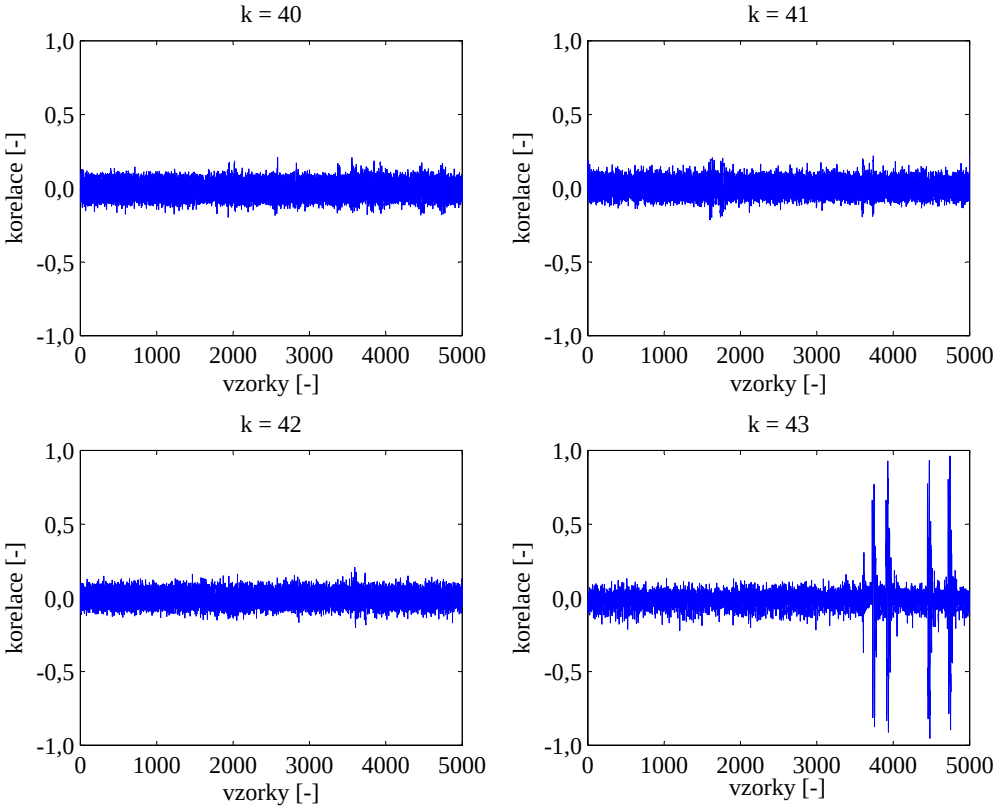
[1]Data potřebná pro demonstraci DPA útoku získána z http://www.cs.bris.ac.uk/home/eoswald/opensca.html

V prvním kroku DPA útoku je nejprve zvolen mezivýsledek algoritmu AES. Tento mezivýsledek závisí na známých datech a části klíče, jenž lze odhadnout. Pro konkrétní útok byl zvolen první výstupní bajt operace `SubBytes` v první rundě AES. V dalším kroku jsou změřeny průběhy proudové spotřeby mikrokontroléru pro první rundu AES při šifrování 200 náhodných bloků otevřeného textu. Tím vznikne matice proudových průběhů $\mathbf{T}$. Třetí krok spočívá ve stanovení hypotéz mezivýsledků algoritmu AES. Tyto mezivýsledky jsou určeny pro 200 bloků známého otevřeného textu. Výsledkem bude matice $\mathbf{V}$. Pro její prvky platí $v_{i,j} = S(d_i \oplus k_j)$, kde $d_1, \ldots, d_{200}$ jsou první bajty z každého bloku otevřeného textu a $k_j = j - 1$ pro $j = 1, \ldots, 256$. Matice $\mathbf{V}$ obsahuje celkově 200×256 hypotéz mezivýsledků.

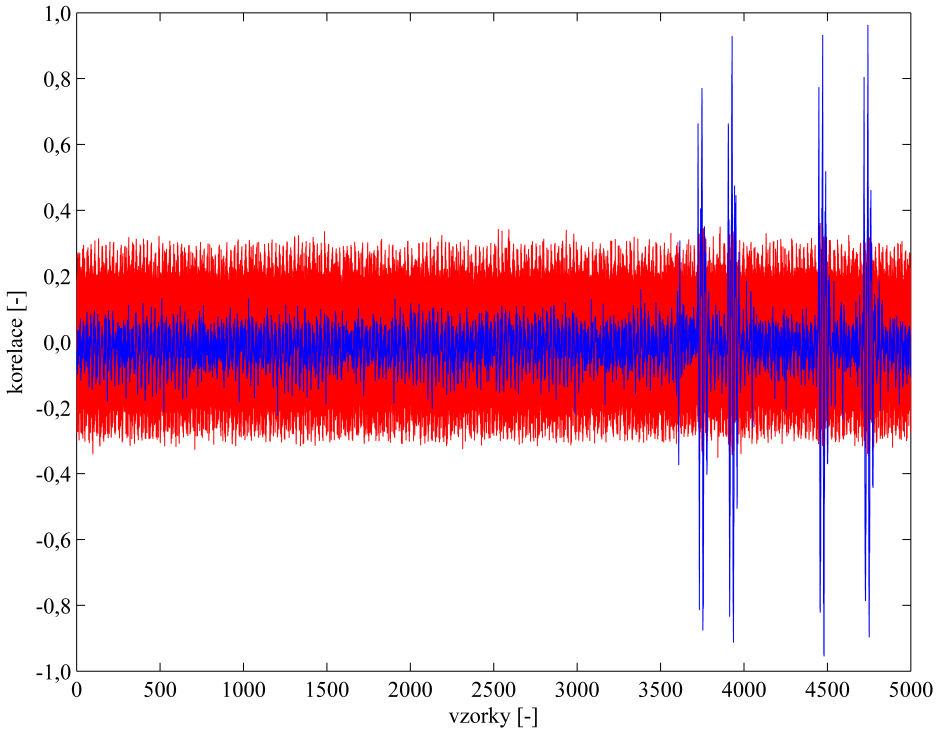
Ve čtvrtém kroku je každému hypotetickému mezivýsledku $v_{i,j}$ z matice $\mathbf{V}$ přiřazena hypotetická hodnota proudové spotřeby $h_{i,j}$. K tomu je nutné vytvořit simulaci proudové spotřeby kryptografického modulu, tzv. model spotřeby. V uvedeném příkladě je použit model Hammingovy váhy. Jeho aplikací na matici $\mathbf{V}$ hypotetických mezivýsledků vznikne matice $\mathbf{H}$ pro hypotézy proudové spotřeby. Poslední krok DPA útoku slouží k vyhodnocení míry lineární závislosti (korelace) hypotéz proudové spotřeby pro všechny odhady klíče (hodnoty ve sloupci $\mathbf{h}_i$ matice $\mathbf{H}$) a změřených průběhů (hodnoty ve sloupci $\mathbf{t}_j$ matice $\mathbf{T}$). Výsledkem je matice korelačních koeficientů $\mathbf{R}$.

Matici $\mathbf{R}$ lze zobrazit jako množinu grafů. Každý graf znázorňuje jeden řádek matice $\mathbf{R}$, tedy jednu hypotézu (odhad) klíče. Na obr. 3.3 jsou zachyceny grafické průběhy pro hypotézy klíče 40 až 43 právě provedeného útoku. V průběhu pro hypotézu klíče 43 lze pozorovat velké špičky. Tyto špičky jsou ve skutečnosti nejvyšší v celé matici $\mathbf{R}$. Všechny ostatní hodnoty matice $\mathbf{R}$ jsou výrazně menší. Tato skutečnost poskytuje útočníkovi množství důležitých informací. Jedna z nejdůležitějších je, že první bajt tajného klíče má hodnotu 43. Z počtu špiček daného průběhu dále vyplývá, že mikrokontrolér s tímto mezivýsledkem pracuje v několika instrukcích. To je zpravidla obvyklé pro softwarovou implementaci šifrovacího algoritmu, kdy při práci s hledaným mezivýsledkem dochází k jeho přesunu mezi pamětí a registrem mikrokontroléru nebo obráceně.

Sloupce v matici $\mathbf{H}$ nejsou vzájemně všechny nezávislé. Pokud sloupec matice $\mathbf{H}$ vede k velké hodnotě korelačního koeficientu, mohou i některé další sloupce vést k určité korelaci. Popsanou situaci lze pozorovat na obr. 3.3, ve kterém průběhy pro další hypotézy klíče také obsahují špičky. Ty jsou však výrazně menší a nezabrání odhalení klíče. Obr. 3.4 obsahuje průběhy pro všechny klíče. Průběh pro klíč 43 je modrý, zatímco pro ostatní klíče jsou průběhy červené. Významné špičky se nacházejí pouze v modrém průběhu.



Obr. 3.3: Řádky matice **R** pro hypotézy klíče 40 až 43.



Obr. 3.4: Všechny řádky matice **R** se zvýrazněnou hypotézou klíče 43.

3.3 Útoky pomocí šablon

Útoky pomocí šablon TA (Template Attacks) představují nejúčinnější formu útoku postranním kanálem. Poprvé byly představeny v článku [10]. Tento typ útoků může prolomit implementaci šifrovacího algoritmu, jejíž odolnost závisí na předpokladu, že útočník má přístup pouze k jednomu či omezenému počtu proudových průběhů. Základním požadavkem je, aby útočník měl přístup k identickému kryptografickému modulu, který může libovolně programovat.

Na rozdíl od jiných typů útoků, se útoky pomocí šablon skládají ze dvou fází. V první fázi je nejprve nutné vytvořit šablony. Tyto šablony budou následně využity ve druhé části útoku ke klasifikaci. V následující části bude stručně popsána teorie útoku pomocí šablon a jednotlivé fáze.

3.3.1 Základní popis

Proudové průběhy mohou být charakterizovány vícerozměrným normálním rozdělením, které je popsáno vektorem středních hodnot $\mathbf{m}$ a kovarianční maticí $\mathbf{C}$. Tato dvojice, označená $(\mathbf{m}, \mathbf{C})$, představuje šablonu. Snahou útočníka je vytvořit popis kryptografického modulu pomocí šablon. Vytvořit např. šablony pro určité instrukce šifrovacího algoritmu implementovaného uvnitř modulu. K tomu slouží útočníkovi identický typ experimentálního modulu, který má plně pod svojí kontrolou. Na experimentálním modulu jsou postupně zaznamenány průběhy proudové spotřeby pro různé hodnoty dat d_i a klíčů k_j. Po seskupení proudových průběhů, které odpovídají danému páru (d_i, k_j), lze vypočítat vektor středních hodnot a kovarianční matici vícerozměrného normálního rozdělení. Útočník získá šablony pro každý pár $(d_i, k_j) : h_{d_i,k_j} = (\mathbf{m}, \mathbf{C})_{d_i,k_j}$.

Pomocí vytvořených šablon je ve druhé fázi útoku vytvořena klasifikace proudového průběhu z napadeného kryptografického modulu. Tedy pro daný proudový průběh $\mathbf{t}$ a šablonu $h_{d_i,k_j} = (\mathbf{m}, \mathbf{C})_{d_i,k_j}$ je vypočítána pravděpodobnost [17]

$$p(\mathbf{t}; (\mathbf{m}, \mathbf{C})_{d_i,k_j}) = \frac{\exp\left(-\frac{1}{2} \cdot (\mathbf{t} - \mathbf{m})' \cdot \mathbf{C}^{-1} \cdot (\mathbf{t} - \mathbf{m})\right)}{\sqrt{(2 \cdot \pi)^T \cdot \det \mathbf{C}}}. \tag{3.11}$$

Výsledná pravděpodobnost vyjadřuje míru shody mezi šablonou a proudovým průběhem. Tato pravděpodobnost je vypočítána pro každou šablonu. Nejvyšší pravděpodobnost pak určuje správnou šablonu. Tento přístup se nazývá metoda maximální věrohodnosti. Protože každá šablona je přiřazena ke konkrétnímu klíči, může být určen i správný klíč.

3.3.2 Fáze vytváření šablon

V předchozí části bylo vysvětleno, jakým způsobem jsou vytvářeny šablony pro popis kryptografického modulu, který vykonává určité instrukce šifrovacího algoritmu. Důležitým faktorem při vytváření šablon je velikost kovarianční matice $\mathbf{C}$. Její velikost ovlivňuje nejen nároky na výpočetní paměť pro uchování šablon, ale i výpočetní náročnost při klasifikaci ve druhé fázi útoku, neboť při výpočtu pravděpodobností dochází k inverzi kovarianční matice.

Velikost kovarianční matice závisí na délce proudového průběhu, tedy na počtu vzorků samotného průběhu. Např. požadavek na množství paměti roste kvadraticky se zvyšujícím se počtem vzorků. Proto je vhodné z každého proudového průběhu vybrat jen ty vzorky, které obsahují největší množství informace o použitém klíči. Tyto vzorky jsou nazývány body zájmu POI (Point of Interest) a jejich počet je dále značen N_{IP}. Efektivní způsob, jak tyto body vybrat, popisuje práce [22].

3.3.3 Fáze klasifikace

Při klasifikaci proudového průběhu je možné narazit na různé problémy spojené s kovarianční maticí. Prvním z nich je, jak bylo napsáno výše, závislost velikosti matice na počtu zájmových bodů. Druhým problémem může být špatná podmíněnost kovarianční matice, která bude působit problémy při výpočtu inverzní matice ve vztahu 3.11. Další výpočetní problémy mohou vzniknout z důvodu malých hodnot v exponentu.

Za účelem vyvarovat se umocňování ve vztahu 3.11 je možné logaritmovat obě strany rovnice. Správný klíč poté bude odpovídat šabloně s nejmenší absolutní hodnotou logaritmu pravděpodobnosti

$$\ln p(\mathbf{t}; (\mathbf{m}, \mathbf{C})) = -\frac{1}{2}(\ln\left((2 \cdot \pi)^{N_{\mathrm{IP}}} \cdot \det \mathbf{C}\right) + (\mathbf{t} - \mathbf{m})' \cdot \mathbf{C}^{-1} \cdot (\mathbf{t} - \mathbf{m})). \tag{3.12}$$

Dále je možné vytvořit redukovanou verzi šablony, která bude obsahovat pouze vektor středních hodnot. Kovarianční matici lze nahradit jednotkovou maticí. To znamená, že kovariance mezi zájmovými body není uvažována. Nahrazením kovarianční matice se zjednoduší výpočet pravděpodobnosti vícerozměrného normálního rozdělení na

$$p(\mathbf{t}; \mathbf{m}) = \frac{1}{\sqrt{(2 \cdot \pi)^{N_{\mathrm{IP}}}}} \cdot \exp\left(-\frac{1}{2} \cdot (\mathbf{t} - \mathbf{m})' \cdot (\mathbf{t} - \mathbf{m})\right). \tag{3.13}$$

Opět může být aplikován logaritmus, čímž se odstraní umocňování a výsledkem bude

$$\ln p(\mathbf{t};\mathbf{m}) = -\frac{1}{2}(\ln(2\cdot\pi)^{N_{\text{IP}}} + (\mathbf{t}-\mathbf{m})'\cdot(\mathbf{t}-\mathbf{m})). \tag{3.14}$$

3.4 Proudová analýza pomocí strojového učení

Techniky strojového učení jsou často využívaný v klasifikačních[2] a regresních[3] úlohách. Strojové učení se zabývá počítačovými algoritmy, které umožňují počítačovému systému učit se na základě zkušenosti. Vzhledem k tomu, že lze útoky postranními kanály formulovat jako klasifikační úlohy, představuje strojové učení užitečný nástroj v kryptoanalýze.

Techniky strojového učení byly již v kryptoanalýze aplikovány [23]. V práci [1] byly techniky strojového učení využity pro útok akustickým postranním kanálem na tiskárny. Práce [18] se zabývá využitím neuronových sítí při proudové analýze a práce [15] porovnává různé metody strojového učení při útoku na šifrovací algoritmus 3DES. Následující kapitoly tvoří stručný úvod do problematiky strojového učení se zaměřením na neuronové sítě a algoritmy SVM (Support Vector Machines)

3.4.1 Neuronové sítě

Lidé a zvířata zpracovávají informace pomocí neuronových sítí ve svém mozku. Tyto sítě tvoří bilióny neuronů (nervových buněk), které si vyměňují krátké elektrické signály nazývané akční potenciály. Počítačové algoritmy napodobující tyto biologické struktury se formálně nazývají umělé neuronové sítě ANN (Artificial Neural Networks), běžně však nazývané zkráceně jako neuronové sítě.

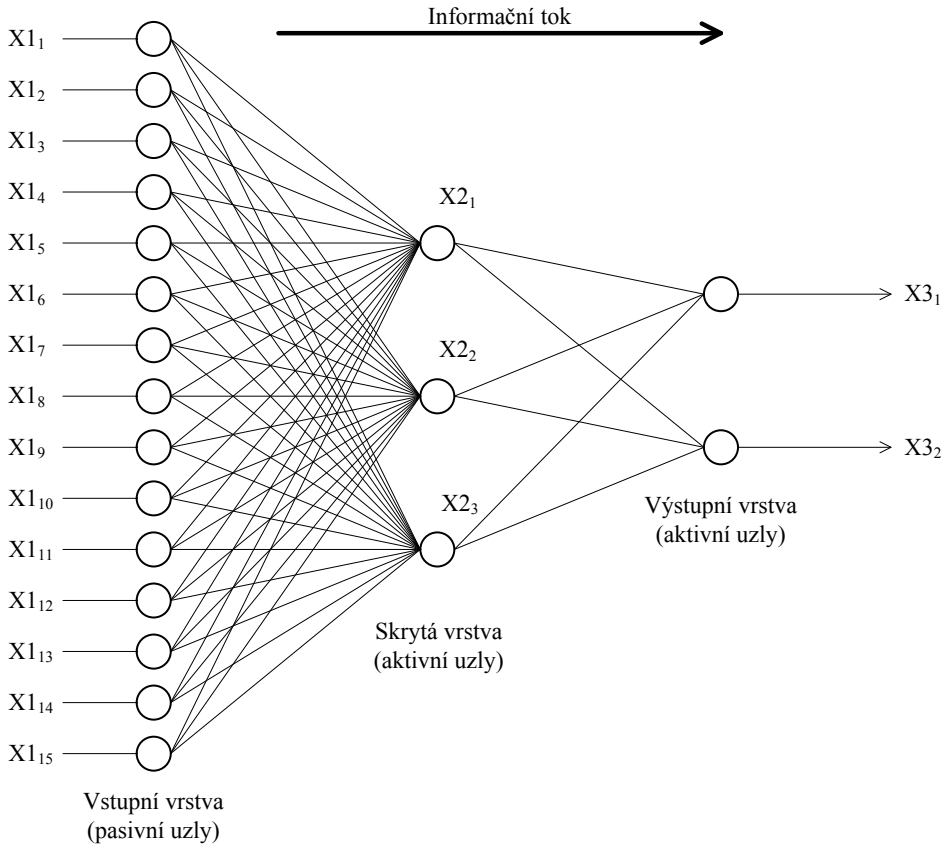
Existuje celá řada struktur neuronových sítí. Běžně používaná struktura je zachycena na obr. 3.5. Tato neuronová síť se skládá ze tří vrstev, a to vstupní vrstvy, skryté vrstvy a výstupní vrstvy. Každá vrstva je tvořena jedním nebo více uzly. Spojení mezi uzly udává tok informací z jednoho uzlu k dalšímu. Pro tento konkrétní typ neuronové sítě platí, že tok informací směřuje ze vstupu na výstup. Jiné typy neuronových sítí mohou obsahovat složitější zapojení (např. zpětnovazební cesty).

Uzly ve vstupní vrstvě jsou pasivní. Nemodifikují data. Na vstupu přijmou jednu hodnotu, kterou duplikují na všechny svoje výstupy. Naproti tomu, uzly ve skryté a výstupní vrstvě jsou aktivní. Tzn., že modifikují data, jak je zachycuje obr. 3.6. Proměnné $X1_i$, kde $i = 1, \ldots, 15$, uchovávají hodnoty pro ohodnocení.

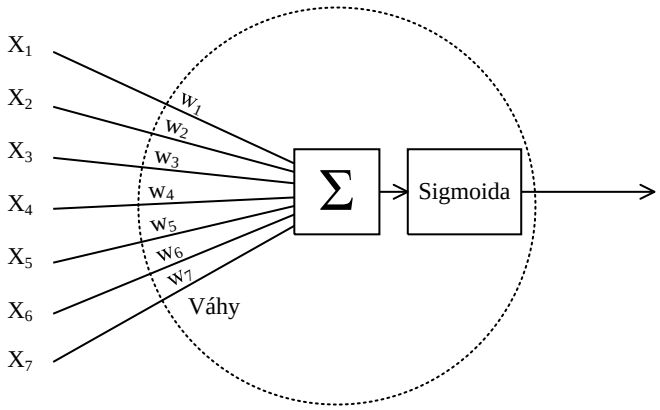
Každá hodnota je ze vstupní vrstvy duplikována a odeslána všem skrytým uzlům. Hodnoty jsou při vstupu do skryté vrstvy násobeny dopředu stanovenými vahami. Vážené vstupy se poté sčítají a výsledkem je jediné číslo. Před opuštěním

[2]Dělení vstupních dat do dvou nebo více tříd

[3]Odhad hodnoty výstupu na základě vstupu



Obr. 3.5: Struktura neuronové sitě.



Obr. 3.6: Aktivní uzel neuronové sítě ve skryté a výstupní vrstvě.

uzlu prochází toto číslo skrz nelineární matematickou funkci nazývanou sigmoida, která omezuje výstup daného uzlu. Tzn., že vstupem funkce je hodnota v intervalu $(-\infty, +\infty)$ a na výstupu funkce může být pouze hodnota mezi 0 a 1.

Výstupy ze skryté vrstvy reprezentují proměnné $X2_i$, kde $i = 1, \ldots, 3$. Podobně jako předtím dojde k duplikování a odeslání další vrstvě. Aktivní uzly výstupní vrstvy zkombinují a modifikují data. Výsledkem této neuronové sítě jsou dvě hodnoty $X3_1$ a $X3_2$.

Neuronové sítě mohou mít různý počet vrstev a různý počet uzlů uvnitř vrstvy. Většina aplikací využívá třívrstvou strukturu s maximálně několika stovkami vstupních uzlů. Skrytá vrstva má zpravidla desetiprocentní velikost vstupní vrstvy. Pro učení neuronových sítí se nejčastěji využívá algoritmus Backpropagation. Pracuje tak, že vyhodnocené řešení je porovnáno s očekávaným. Podle zjištěné odchylky se vypočítá, o kolik se změní váhy uzlů, aby se tato odchylka od očekávaného řešení co nejvíc minimalizovala.

3.4.2 Support Vector Machines

Klasifikační metoda SVM patří do kategorie tzv. jádrových algoritmů. Výhodou těchto metod je schopnost nalézt lineární hranice oddělující určité třídy ve vstupním prostoru a reprezentovat složité nelineární funkce. Základním principem je převod původního vstupního prostoru do vícedimenzionálního, ve kterém je již možné oddělit třídy lineárně. Princip je zobrazen na obr. 3.7.

Klasifikace lineárně separovatelných dat pomocí SVM probíhá následujícím způsobem. Na počátku je k dispozici trénovací množina s N_1 třídami[4] a testovací množina s N_2 třídami. Každá třída z trénovací množiny obsahuje několik atributů (prvků[5]) x_i pro $i = 1, \ldots N_1$ a má přiřazenu hodnotu y_i. Úkolem SVM je klasifikovat každou třídu x_i pro $i = 1, \ldots N_2$ z testovací množiny podle příslušných atributů. Další popis SVM bude pro zjednodušení omezen na problém binární klasifikace, tedy na problém lineárního oddělení dvou tříd.

Lineární SVM hledají optimální oddělovací nadrovinu[6] $\mathcal{H}$ popsanou následující rovnicí

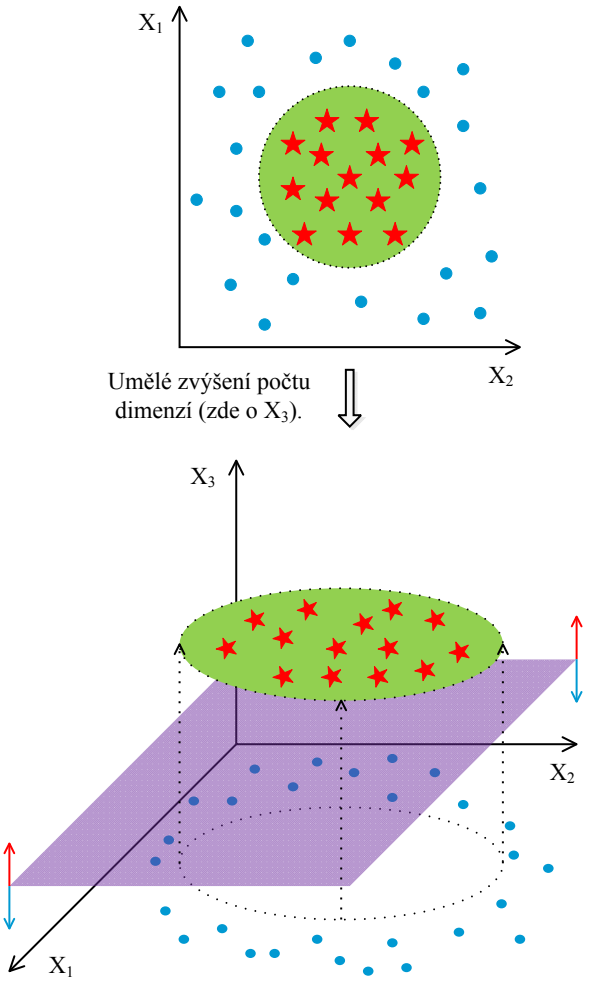
$$\mathbf{w} \cdot \mathbf{x} + b = 0, \tag{3.15}$$

kde $\mathbf{w} \in \mathbb{R}^n$ je normálový vektor nadroviny, $\frac{b}{||\mathbf{w}||}$ kolmá vzdálenost od počátku pro $b \in \mathbb{R}$ a $\mathbf{w} \cdot \mathbf{x}$ skalární součin, která maximalizuje vzdálenost od bodů z lineárně sepa-

[4] V kontextu proudové analýzy představují třídy změřené proudové průběhy.

[5] V kontextu proudové analýzy představují prvky důležité vzorky proudového průběhu.

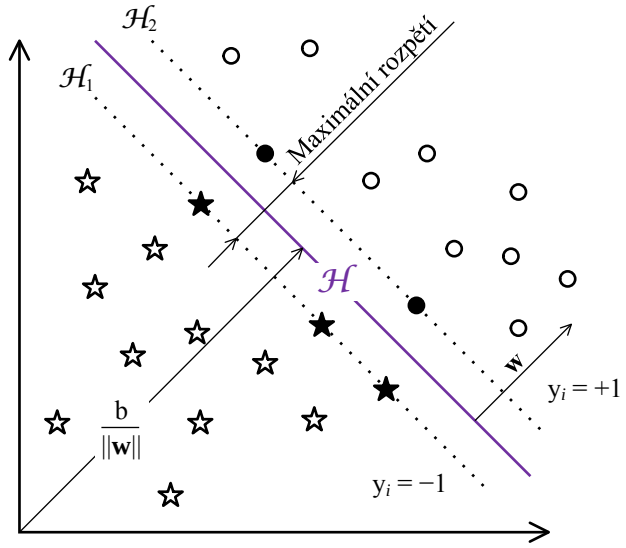
[6] Nadrovinou se v geometrii rozumí pro daný prostor dimenze n jakýkoliv jeho podprostor dimenze $n - 1$.



Obr. 3.7: Princip lineárního oddělení dvou tříd s nelineárními hranicemi.

rovatelné trénovací množiny $\mathcal{D} = \left\{(\mathbf{x}_i, y_i) |\ \mathbf{x}_i \in \mathbb{R}^n, y_i \in \{-1, 1\}_{i=1}^{l}\right\}$. Na obr. 3.8 je zobrazen optimální lineární oddělovač (nadrovina) pro klasifikaci dvou tříd pomocí SVM. Body, které leží nejblíže vlastnímu oddělovači, se nazývají support vectors a na obr. 3.8 jsou označeny černou barvou. Jejich funkcí je „podpora“ oddělovací nadroviny. Ostatní body nejsou pro oddělovač důležité. Metoda SVM tedy hledá takové trénovací body, které jsou důležité pro nalezení optimálního oddělovače tříd. Nalezení optimální nadroviny je podrobněji popsáno v článku [4].

Porovnáním SVM útoku s útokem pomocí šablon se zabývá práce [9]. Srovnává zejména počet potřebných proudových průběhů pro úspěšnou klasifikaci. Další práce [2] využívá modifikovanou metodu SVM pro klasifikaci proudových průběhů.



Obr. 3.8: Pohled na optimální oddělovací nadrovinu.

4 PROUDOVÁ ANALÝZA S VYUŽITÍM ANN

Cílem této kapitoly je na základě získaných teoretických znalostí realizovat útok na kryptografický modul, který využívá neuronové sítě. K praktické realizaci útoku jsou využity nástroje a naměřené proudové průběhy ze soutěže DPA Contest [6], jejíž stručný popis následuje dále. Další část kapitoly popisuje implementovaný šifrovací algoritmus uvnitř modulu, návrh řešení, praktickou realizaci útoku a jeho vyhodnocení.

4.1 DPA Contest

V roce 2008 výzkumná skupina[1] z francouzské technické univerzity Telecom ParisTech vytvořila první verzi soutěže DPA Contest. Jejich cílem bylo poskytnout odborné veřejnosti objektivní způsob pro porovnání různých metod proudové analýzy, především DPA útoku. Nezávislé porovnání těchto metod je velice problematické z několika důvodů. Např. kvůli použití jiných měřicích zařízení, odlišné implementace šifrovacího algoritmu, různým podmínkám okolního prostředí, atd. DPA Contest se snaží tyto faktory, ovlivňující porovnání, odstranit. K tomu veřejně poskytuje proudové průběhy z kryptografického modulu, které jsou měřeny za předem stanovených podmínek. Pro všechny průběhy jsou také poskytnuty použité otevřené texty, tajné klíče a odpovídající šifrované texty. Dále poskytuje testovací framework pro detailní vyhodnocení vytvořeného útoku. Tím umožňuje nezávislým výzkumníkům možnost vyvíjet vlastní útoky a porovnávat je s ostatními.

V současné době probíhá již čtvrtá verze této soutěže, která tvoří základ následující praktické části.

4.2 Implementace kryptografického modulu

Použitý kryptografický modul (čipová karta Atmel ATMega-163) pro DPA Contest V4 má v sobě implementován maskovaný algoritmus AES-256. Pro maskování AES je použita metoda RSM (Rotating Sbox Masking). Podle [19] se tato metoda maskování vyznačuje jednoduchou implementací a nízkou výpočetní náročností, avšak odolnou vůči některým útokům postranními kanály.

Cílem útoku je získat prvních 128 bitů tajného klíče. Vzhledem k tomu, že první runda AES-128 je stejná i pro variantu AES-256, lze se dále zaměřit pouze na AES-128.

[1] Webová stránka výzkumné skupiny http://www.comelec.telecom-paristech.fr/en.html

Maskování RSM algoritmu AES používané v DPA Contest V4 využívá 16 masek $v = \{v_0, v_1, \ldots, v_{15}\}$, které jsou veřejně dostupné. Na začátku každého šifrování se vygeneruje náhodná hodnota od 0 do 15, tzv. offset, která je tajná. Poté na každý i-tý bajt otevřeného textu a masku $v_{i+\text{offset}}$ je aplikována operace XOR

$$\begin{Bmatrix} m_0 \ldots m_{12} \\ m_1 \ldots m_{13} \\ m_2 \ldots m_{14} \\ m_3 \ldots m_{15} \end{Bmatrix} \oplus \begin{Bmatrix} v_{0+\text{offset}} \ldots v_{12+\text{offset}} \\ v_{1+\text{offset}} \ldots v_{13+\text{offset}} \\ v_{2+\text{offset}} \ldots v_{14+\text{offset}} \\ v_{3+\text{offset}} \ldots v_{15+\text{offset}} \end{Bmatrix} = \begin{Bmatrix} s_0 \ldots s_{12} \\ s_1 \ldots s_{13} \\ s_2 \ldots s_{14} \\ s_3 \ldots s_{15} \end{Bmatrix} = Stav, \tag{4.1}$$

kde hodnota $i + \text{offset}$ je spočítána operací modulo 16. Původní substituce `SubBytes` se nahradí šestnácti maskovanými $\texttt{MaskedSubBytes}_i(s_j) = \texttt{SubBytes}(s_j \oplus v_i) \oplus v_{i+1}$, kde $i \in \{1, 2, \ldots, 16\}$. Po lineární části AES následuje dodatečná operace, která zajistí, aby *Stav* vstupující do další rundy byl roven

$$\left(\texttt{MC} \circ \texttt{SR} \circ \texttt{SBox} \circ \begin{Bmatrix} s_0 \oplus v_{0+\text{offset}+r} \oplus O_0^r \ldots \\ s_1 \oplus v_{0+\text{offset}+r} \oplus O_1^r \ldots \\ s_2 \oplus v_{0+\text{offset}+r} \oplus O_2^r \ldots \\ s_3 \oplus v_{0+\text{offset}+r} \oplus O_3^r \ldots \end{Bmatrix} \right) \oplus \begin{Bmatrix} v_{0+\text{offset}+r+1} \ldots v_{12+\text{offset}+r+1} \\ v_{1+\text{offset}+r+1} \ldots v_{13+\text{offset}+r+1} \\ v_{2+\text{offset}+r+1} \ldots v_{14+\text{offset}+r+1} \\ v_{3+\text{offset}+r+1} \ldots v_{15+\text{offset}+r+1} \end{Bmatrix}, \tag{4.2}$$

kde $r \in \{1, \ldots, 9\}$ (pro AES-128) je číslo rundy a O_i^r je $(i+1)$ bajt r-tého subklíče. `SR` a `MC` reprezentují popořadě transformace `ShiftRow` a `MixColumns`. Maskování pro poslední rundu ($r = 10$ pro AES-128) zajistí, že na jejím výstupu bude

$$\left(\texttt{SR} \circ \texttt{SBox} \circ \begin{Bmatrix} s_0 \oplus v_{0+\text{offset}+r} \oplus O_0^r \ldots \\ s_1 \oplus v_{0+\text{offset}+r} \oplus O_1^r \ldots \\ s_2 \oplus v_{0+\text{offset}+r} \oplus O_2^r \ldots \\ s_3 \oplus v_{0+\text{offset}+r} \oplus O_3^r \ldots \end{Bmatrix} \right) \oplus \begin{Bmatrix} v_{0+\text{offset}+r+1} \ldots v_{12+\text{offset}+r+1} \\ v_{1+\text{offset}+r+1} \ldots v_{13+\text{offset}+r+1} \\ v_{2+\text{offset}+r+1} \ldots v_{14+\text{offset}+r+1} \\ v_{3+\text{offset}+r+1} \ldots v_{15+\text{offset}+r+1} \end{Bmatrix}. \tag{4.3}$$

Poslední maska je odstraněna operací XOR. Další informace o použitém způsobu maskování v článku [19] a na webové stránce [6].

Princip fungování maskovaného algoritmu AES lze popsat pomocí algoritmu 4.1. Přičemž jeho původní nechráněnou verzi lze získat odstraněním řádků zvýrazněných modrou barvou a záměnou operace `MaskedSubBytes` za operaci `Subbytes`.

4.3 Proudové průběhy pro DPA Contest V4

Proudové průběhy pro použitý kryptografický modul byly měřeny v laboratořích Digital Electronic Systems výzkumné skupiny Telecom ParisTech za stanovených podmínek následujícími přístroji:

- elektromagnetickou sondou Langer RF-U 5-2,
- předzesilovačem Langer PA303,
- digitálním osciloskopem Lecroy Waverunner 6100A,
- regulovatelným zdrojem Agilent E3631A.

Algoritmus 4.1 Maskování algoritmu AES-256 v DPA Contest V4 [6].

Vstup: Otevřený text X; 16 bajtů X_i, kde $i \in \langle 0, 15 \rangle$,
Expanze klíče; 15 128bitových konstant $\mathsf{RoundKey}[r]$, kde $r \in \langle 0, 14 \rangle$

Výstup: Šifrovaný text X; 16 bajtů X_i, kde $i \in \langle 0, 15 \rangle$,

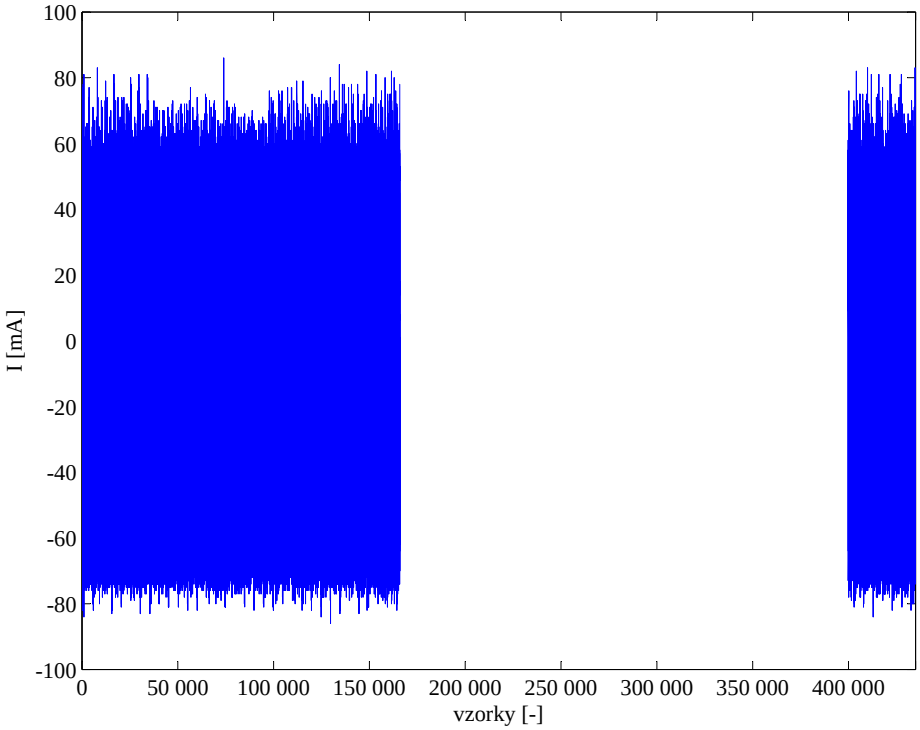
Náhodně generovaný $\mathsf{offset} \in \langle 0, 15 \rangle$ s rovnoměrným rozdělením.
$X = X \oplus \mathsf{Mask}_{\mathsf{offset}}$
// Maskování otevřeného textu
// Všechny rundy kromě poslední
for $r \in \langle 0, 12 \rangle$ **do**
    $X = X \oplus \mathsf{RoundKey}[r]$
// AddRoundKey
    for $i \in \langle 0, 15 \rangle$ **do**
        $X_i = \mathsf{MaskedSubBytes}_{\mathsf{offset}+i+r}(X_i)$
    end for
    $X = \mathsf{ShiftRows}(X)$
    $X = \mathsf{MixColumns}(X)$
    $X = X \oplus \mathsf{MaskCompensation}_{\mathsf{offset}+1+r}$
end for
// Poslední runda
$X = X \oplus \mathsf{RoundKey}[13]$
for $i \in \langle 0, 15 \rangle$ **do**
    $X_i = \mathsf{MaskedSubBytes}_{\mathsf{offset}+13+r}(X_i)$
end for
$X = \mathsf{ShiftRows}(X)$
$X = X \oplus \mathsf{RoundKey}[14]$
// Odmaskování šifrovaného textu
$X = X \oplus \mathsf{MaskCompensationLastRound}_{\mathsf{offset}+14}$

Pro soutěž DPA Contest V4 je veřejně zpřístupněno 100 000 průběhů. Všechny průběhy odpovídají operaci šifrování pro stejný tajný klíč a zobrazují pouze první rundu algoritmu AES (viz obr. 4.1).

K testování vytvořeného útoku je nutné, kromě změřených průběhů, stáhnout z webových stránek soutěže indexový soubor obsahující potřebné informace pro každý průběh. Jeden řádek souboru odpovídá jednomu průběhu a informace v něm jsou odděleny mezerou takto:

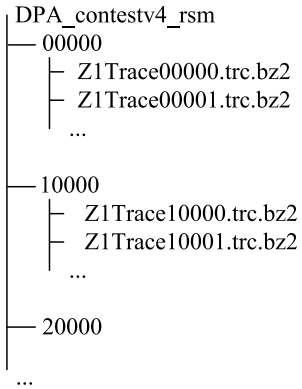
- **1. sloupec:** klíč AES-256 (256 bitů reprezentováno 64 hexadecimálními čísly),
- **2. sloupec:** otevřený text (128 bitů reprezentováno 32 hexadecimálními čísly),
- **3. sloupec:** šifrovaný text (128 bitů reprezentováno 32 hexadecimálními čísly),

- **4. sloupec:** offset (dekadická hodnota 0–15 reprezentována 1 hexadecimálním číslem),
- **5. sloupec:** název adresáře s uloženými průběhy,
- **6. sloupec:** název průběhu.



Obr. 4.1: Proudový průběh z DPA Contest V4.

Pro zjednodušení práce jsou proudové průběhy rozděleny do 10 samostatných komprimovaných archivů. Každý archiv obsahuje 10 000 průběhů, které po rozbalení vytvoří adresářovou strukturu jako na obr. 4.2.



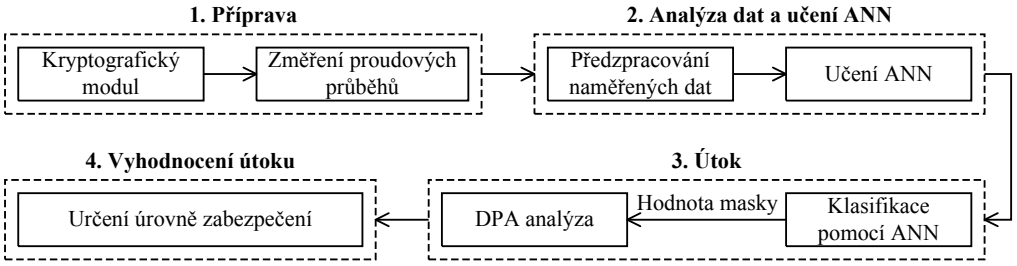
Obr. 4.2: Adresářová struktura po rozbalení jednotlivých archivů.

4.4 Obecný návrh útoku s využitím ANN

Útok na maskovaný šifrovací algoritmus AES, podrobněji popsaný v předchozí kapitole 4.2, tvoří dvě části. V první části je extrahována hodnota masky. K tomuto účelu je aplikována naučená neuronová síť. Po odmaskování následuje druhá část útoku, která využívá diferenciální proudovou analýzu DPA s korelačním koeficientem pro odhalení tajného klíče.

K naučení neuronové sítě je nutné pro kontrolovaný kryptografický modul naměřit sadu proudových průběhů $\mathbf{T}$. Následně vybrat p bodů t_1, t_2, t_p, které jsou výrazně korelovány se známými hodnotami masky. Tyto body slouží pro učení neuronové sítě, která pro daný proudový průběh T v čase t_1, t_2, t_p vrací d hodnot masky.

Obr. 4.3 rekapituluje celý postup. Nejprve je provedena implementace maskovaného šifrovacího algoritmu do kryptografického modulu. Po změření proudových průběhů následuje další fáze, a to redukce počtu bodů každého průběhu na ty, které poskytují největší množství informace o použité masce a jsou vhodné pro učení neuronové sítě. Třetí fáze tvoří samotný útok s cílem získat hodnotu tajného klíče. Tato fáze poté umožňuje při vyhodnocení určit odolnost kryptografického modulu tedy jeho úroveň zabezpečení, např. na základě počtu průběhů potřebných k odhalení klíče.

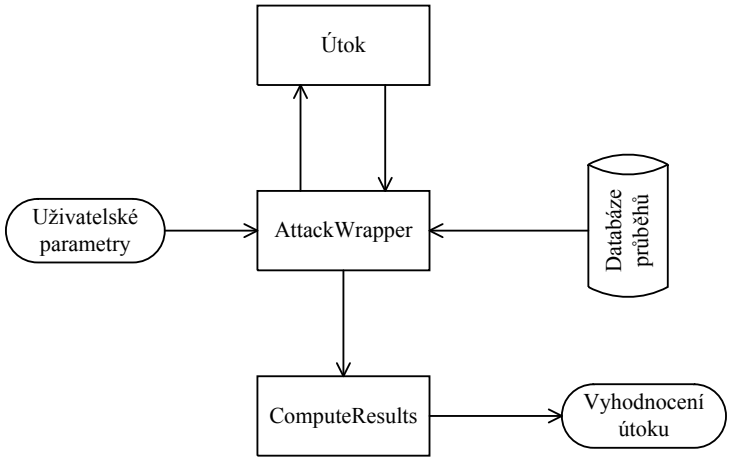


Obr. 4.3: Útok na maskovaný šifrovací algoritmus s využitím ANN.

Úspěšnost navrženého útoku odhalit korektní hodnotu klíče přímo souvisí se schopností neuronové sítě určit správnou hodnotu masky. Tedy čím menší chyba mezi správnými a vyhodnocenými hodnotami masky, tím větší bude korelace mezi změřenými průběhy a hypotetickou proudovou spotřebou pro správný klíč. V případě, kdy neuronová síť dokáže určit hodnoty masky s pravděpodobností 100 %, tak DPA analýza odhalí klíč se stejným počtem průběhů jako v případě nemaskovaného šifrovacího algoritmu.

4.5 Realizace útoku a vyhodnocení

Autoři soutěže DPA Contest nabízí ke stažení softwarové nástroje pro implementaci vlastního útoku. Jedním z nich je AttackWrapper představující softwarové jádro, které umožňuje přistupovat k databázi průběhů a k souborovému systému, spouštět vytvořený útok a na závěr předat výsledky k vyhodnocení nástroji ComputeResults. ComputeResults z výsledků vyhodnotí úspěšnost vytvořeného útoku. Blokový diagram na obr. 4.4 znázorňuje operace prováděné nástrojem AttackWrapper.



Obr. 4.4: Blokový diagram popisující funkci nástroje AttackWrapper.

4.5.1 Nástroj AttackWrapper

AttackWrapper je nástroj určený ke spouštění útoku a k jeho vyhodnocení. Může být nainstalován na jakýkoli operační systém unixového typu (Linux, Mac OS X, atd.) nebo Windows. K implementaci útoku lze použít libovolný programovací jazyk (C, C++, Python, Perl) nebo Matlab.

Společně s tímto nástrojem jsou také dodávány nástroje ComputeResults a Traces2Text. První zmíněný slouží k analýze uložených výsledků z AttackWrapperu a druhý umožňuje převést proudový průběh do čitelného textového souboru.

Protože k implementaci útoku bylo zvoleno prostředí Matlab 7. 12. 0 běžící na operačním systému Ubuntu 13. 10, bude další popis užívaní nástroje AttackWrapper omezen na tento operační systém. K instalaci nástroje je zapotřebí mít k dispozici:

- počítačovou platformu x86 nebo x86-64,
- kompilátor C++(např. g++),
- knihovnu libbz2.

Instalace nástroje se provádí v terminálu operačního systému pomocí následujících příkazů.

```
$ tar xzf attack_wrapper-2.0.tar.gz
$ cd attack_wrapper-2.0
$ ./configure
$ make
```

Pokud instalace proběhne v pořádku, tak se v adresáři `src` objeví tři spustitelné soubory (`attack_wrapper`, `compute_results` a `traces2text`).

Ovládání nástroje AttackWrapper probíhá přes příkazovou řádku. Následující příkaz slouží k zobrazení dostupných parametrů.

```
$ attack_wrapper --help
```

Tyto parametry jsou povinné:

- -e edice: verze soutěže (v4_RSM),
- -d adresář: umístění složky s proudovými průběhy,
- -x soubor: umístění indexového souboru.

Ostatní jsou volitelné:

- -i iterace: počet použitých průběhů k analýze (ve výchozím stavu použity všechny průběhy),
- -o soubor: název souboru s výsledky (ve výchozím stavu results),
- -t: přepsání souboru s výsledky (ve výchozím stavu nepřepisuje),
- -f: režim FIFO (bude popsán později).

Po spuštění útoku a jeho vykonání lze výsledný soubor předat nástroji ComputeResults, viz následující příkaz.

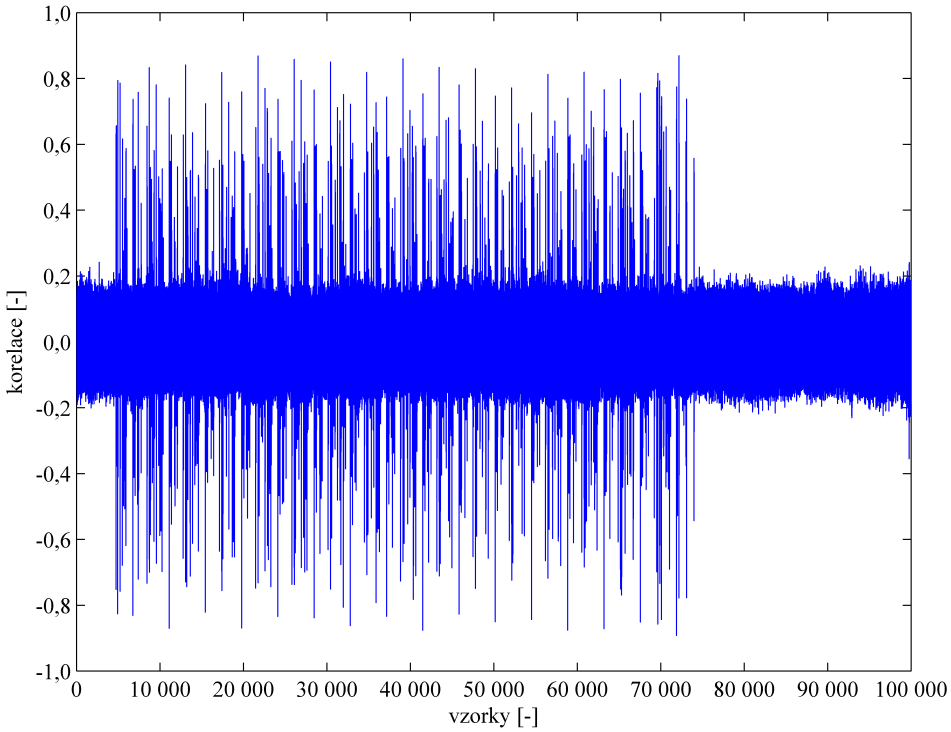
```
$ attack_wrapper -f -d DPA_contestv4_rsm -x ...
    dpav4_rsm_index -e v4_RSM fifo
$ compute_results results
```

Útok vytvořený v prostředí Matlab komunikuje s nástrojem AttackWrapper prostřednictvím režimu FIFO (First In First Out). Tento režim využívá dva speciální soubory (tzv. roury). Nejprve je spuštěn AttackWrapper, který vytvoří soubory `xxx_from_wrapper` v režimu pouze pro čtení a `xxx_to_wrapper` v režimu pouze pro zápis. Poté se manuálně spustí útok v Matlabu, který tyto soubory otevře a využívá je pro čtení nebo zápis dat. Písmena `xxx` jsou nahrazena posledním argumentem příkazu pro spuštění AttackWrapperu.

4.5.2 Realizace útoku

Po nastudování funkce potřebných nástrojů a obecném návrhu byl útok naprogramován ve vývojovém prostředí Matlab. K implementaci neuronové sítě je použit volně dostupný Netlab Neural Network toolbox[2].

Před vytvářením a učením neuronové sítě bylo třeba nalézt v proudových průbězích body, které jsou výrazně korelovány s tajnou hodnotou offsetu. Na obr. 4.5 lze pozorovat, že zejména první tři čtvrtiny zachyceného proudového průběhu obsahují velké množství informace o hodnotě offsetu. Je tedy možné předpokládat, že neuronová síť určí správný offset s vysokou pravděpodobností.



Obr. 4.5: Korelace mezi offsetem a proudovým průběhem maskovaného AES.

Vytvořená neuronová síť má klasickou třívrstvou strukturu, jak popisuje kapitola 3.4.1 a pro její učení byl zvolen algoritmus Backpropagation. Učící množinu tvořilo 1 500 proudových průběhů. Během předzpracování se z každého průběhu vybralo 48 bodů, které vykazovaly nejvyšší korelaci s hodnotou offsetu.

Testovací množina se skládala také z 1 500 proudových průběhů. Na testovací množině byla ověřena úspěšnost klasifikace offsetu. Procentuální úspěšnost dosahovala 99,7 %.

[2]Dostupný z http://www.aston.ac.uk/eas/research/groups/ncrg/resources/netlab/downloads/

Pro fázi předzpracování byla vyzkoušena i technika analýzy hlavních komponent PCA (Principal Component Analysis), která slouží ke snížení dimenze dat s co nejmenší ztrátou informace. Protože učící množina obsahuje velké množství dat, přibližně $1\,500 \cdot 435\,002 \approx 2^{29}$ bajtů[3], je tato technika paměťově i časově náročné. Proto nebyla pro realizaci útoku použita.

Po vytvoření funkční neuronové sítě byl implementován DPA útok založený na korelačním koeficientu, který útočí na výstup operace `SubBytes`.

Pro tvorbu útoku v prostředí Matlab lze využít šablonu v příloze B. Zdrojový kód realizovaného útoku využívajícího ANN je možné nalézt na přiloženém DVD.

4.5.3 Vyhodnocení

Vytvořený útok na maskovaný šifrovací algoritmus AES byl nejprve otestován na veřejně dostupných proudových průbězích. Na obr. 4.6 je zobrazen výpis terminálového okna při spuštěném útoku. Řádek zvýrazněný červenou barvou udává, že již při analýze 20 proudových průběhů byl nalezen šifrovací klíč maskovaného algoritmu AES. Průměrná doba zpracování jednoho průběhu činila 585 ms.

```
I - Trace #000000: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.7 s) [                ]
I - Trace #000001: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [                ]
I - Trace #000002: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [    .           ]
I - Trace #000003: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [        .       ]
I - Trace #000004: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [ *    * . .    . ]
I - Trace #000005: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [..*     .  ** .. ]
I - Trace #000006: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [***   ..  ** .*.]
I - Trace #000007: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [***. .** .* ***]
I - Trace #000008: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.5 s) [**** .** .* ***]
I - Trace #000009: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [**** *** .* ***]
I - Trace #000010: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****.*** .** ***]
I - Trace #000011: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****.*** *** ***]
I - Trace #000012: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****.*** .** ***]
I - Trace #000013: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****.*** *** ***]
I - Trace #000014: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [******** *** ***]
I - Trace #000015: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [******** *** ***]
I - Trace #000016: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [******** ***.***]
I - Trace #000017: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [******** ***.***]
I - Trace #000018: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [******** ***.***]
I - Trace #000019: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [******** *******]
I - Trace #000020: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****************]
I - Trace #000021: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****************]
I - Trace #000022: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****************]
I - Trace #000023: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****************]
I - Trace #000024: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****************]
I - Trace #000025: Reading trace / Sending trace / Waiting for results / Saving results / Done (0.6 s) [****************]
```

Obr. 4.6: Výpis terminálu při spuštěném útoku.

[3]Každý vzorek proudového průběhu je představován 8bitovou hodnotou.

Po úspěšném otestování byl útok zaslán do soutěže DPA Contest V4. Podle uveřejněných výsledků na webových stránkách soutěže v sekci Hall of Fame bylo pro odhalení hodnoty klíče maskovaného algoritmu AES potřeba 23 proudových průběhů. Přičemž průměrná doba zpracování jednoho průběhu činila 1 100 ms.

Oba výsledky ukazují, že maskovací schéma implementované v DPA Contest V4 může být v praxi prolomeno využitím neuronových sítí v kombinaci s diferenciální proudovou analýzou. Různé časy zpracování proudových průběhů jsou způsobeny použitím počítačů s rozdílnou výpočetní silou.

5 ZÁVĚR

Diplomová práce se zabývá problematikou proudové analýzy a klasifikací proudových otisků získaných při útoku proudovým postranním kanálem na kryptografický modul. Součástí práce je přehledný rozbor v současnosti používaných metod proudové analýzy včetně jejich popisu. Dále se práce věnuje iniciativě DPA Contest, která se snaží pro výzkumníky z oblasti kryptoanalýzy postranních kanálů vytvořit možnost objektivního porovnání různých typů útoků na kryptografické moduly.

Úvodní část práce tvoří definice základních pojmů, které se týkají oblasti kryptografie a kryptoanalýzy.

Dále popisuje současné metody proudové analýzy, které lze rozdělit do dvou kategorií. První kategorii tvoří klasické statistické metody. Mezi tyto metody patří jednoduchá proudová analýza založená na přímé interpretaci proudového odběru modulu. Dále pak diferenciální proudová analýza využívající závislosti mezi hypotetickými hodnotami proudové spotřeby a naměřenými průběhy. A na závěr také útok pomocí šablon, který představuje nejsilnější formu útoku z výše zmíněných statistických metod.

Druhou kategorii tvoří metody proudové analýzy využívající techniky strojového učení. Do této kategorie lze zařadit neuronové sítě. Vhodnost jejich použití v kryptoanalýze proudovým postranním kanálem prokázala práce [18]. Dále lze do této kategorie zařadit algoritmy SVM, které jsou často srovnávány s výše uvedeným útokem pomocí šablon [2], [9].

Praktickou část práce tvoří návrh a realizace útoku na maskovaný šifrovací algoritmus AES. Navržený útok se skládá ze dvou částí. Nejprve je získána hodnota masky pomocí algoritmu strojového učení. Za tímto účelem je aplikována neuronová síť, která dokáže s vysokou přesností určit její hodnoty. Po odmaskování šifrovacího algoritmu je použita diferenciální proudová analýza využívající korelační koeficient, jejíž cílem je výstup operace `SubBytes`. Ta umožňuje velice efektivně určit hodnotu tajného klíče na základě korelace mezi změřeným proudovým průběhem a hypotetickou proudovou spotřebou pro správný klíč.

Implementovaný útok byl po otestovaní odeslán do soutěže DPA Contest V4. Její výsledky ukázaly schopnost realizovaného útoku efektivně odhalit hodnotu tajného klíče maskovaného šifrovacího algoritmu AES. Hodnota klíče byla odhalena již po analýze 23 proudových průběhů. Čas zpracování jednoho průběhu se pohyboval okolo 1 100 ms. Porovnání s ostatními útoky přihlášenými do soutěže je možné nalézt na http://www.dpacontest.org/v4/hall_of_fame.php.

LITERATURA

[1] BACKES, Michael, Markus DÜRMUTH, Sebastian GERLING, Manfred PINKAL a Caroline SPORLEDER. Acoustic Side-channel Attacks on Printers. In: *Proceedings of the 19th USENIX Conference on Security* [online]. 2010 [cit. 2014-05-26]. Dostupné z WWW: <https://www.usenix.org/legacy/events/sec10/tech/full_papers/Backes.pdf>.

[2] BARTKEWITZ, Timo a Kerstin LEMKE-RUST. Efficient Template Attacks Based on Probabilistic Multi-class Support Vector Machines. In: *Smart card research and advanced applications 11th International Conference, CARDIS 2012, Graz, Austria, November 28-30, 2012, Revised selected papers* [online]. 2013 [cit. 2014-05-26]. Dostupné z WWW: <http://link.springer.com/10.1007/978-3-642-37288-9_18>.

[3] BUČEK, Jiří. *Útok postrannímí kanály*. Praha: CryptoFest, 11. 6. 2011.

[4] BEN-HUR, Asa a Jason WESTON. A User's Guide to Support Vector Machines. In: *Data mining techniques for the life sciences* [online]. 2010 [cit. 2014-05-26]. Dostupné z WWW: <http://pyml.sourceforge.net/doc/howto.pdf>.

[5] CORTES, Corinna a Vladimir VAPNIK. Support-Vector Networks. In: *Machine Learning* [online]. 1995, vol. 20, issue 3 [cit. 2014-05-26]. Dostupné z WWW: <http://link.springer.com/10.1023/A:1022627411411>.

[6] *DPA Contest* [online]. 2013 [cit. 2014-05-26]. Dostupné z WWW: <http://www.dpacontest.org/home/>.

[7] FIPS PUB 197. *Advanced Encryption Standard (AES)* [online]. National Institute of Standards and Technology, 26. 11. 2001 [cit. 2014-05-26]. 51 s. Dostupné z WWW: <http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf>.

[8] FIPS PUB 46-3. *Data Encryption Standard (DES)* [online]. National Institute of Standards and Technology, 25. 10. 1999 [cit. 2014-05-26]. 26 s. Dostupné z WWW: <http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf>.

[9] HEUSER, Annelie a Michael ZOHNER. Intelligent Machine Homicide. In: *Constructive side-channel analysis and secure design Third International Workshop, COSADE 2012, Darmstadt, Germany, May 3-4, 2012. Proceedings* [online]. Heidelberg: Springer, 2012, s. 249 [cit. 2014-05-26]. Dostupné z WWW: <http://www.springerlink.com/index/10.1007/978-3-642-29912-4_18>.

[10] CHARI, Suresh, Josyula R. RAO a Pankaj ROHATGI. Template Attacks. In: *Cryptographic Hardware and Embedded Systems - CHES 2002* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003 [cit. 2014-05-26]. Dostupné z WWW: <http://www.springerlink.com/index/10.1007/3-540-36400-5_3>.

[11] JOLLIFFE, I.T. *Principal Component Analysis*. 2nd ed. New York: Springer, 2002, 487 s. ISBN 03-879-5442-2.

[12] KOCHER, Paul C. Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: *Advances in Cryptology — CRYPTO '96* [online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996 [cit. 2014-05-26]. Dostupné z WWW: <http://www.springerlink.com/index/10.1007/3-540-68697-5_9>.

[13] KOCHER, Paul C., Joshua JAFFE a Benjamin JUN. *Introduction to Differential Power Analysis and Related Attacks* [online]. 1998 [cit. 2014-05-26]. Dostupné z WWW: <http://www.cryptography.com/public/pdf/DPATechInfo.pdf>.

[14] KOCHER, Paul C., Joshua JAFFE a Benjamin JUN. *Differential Power Analysis* [online]. 1998 [cit. 2014-05-26]. Dostupné z WWW: <http://www.cryptography.com/public/pdf/DPA.pdf>.

[15] LERMAN, Liran, Gianluca BONTEMPI a Olivier MARKOWITCH. Side Channel Attack: an Approach based on Machine Learning. In: *Proceedings of 2nd International Workshop on Constructive Side-Channel Analysis and Security Design, COSADE 2011* [online]. 2011 [cit. 2014-05-26]. Dostupné z WWW: <http://cosade2011.cased.de/files/2011/cosade2011_talk3_paper.pdf>.

[16] LERMAN, Liran, Stephane F. MEDEIROS, Gianluca BONTEMPI a Olivier MARKOWITCH. *A machine learning approach against a masked AES* [online]. 2014 [cit. 2014-05-26]. Dostupné z WWW: <http://student.ulb.ac.be/l̃lerman/papers/CARDIS2014.pdf>.

[17] MANGARD, Stefan, Elisabeth OSWALD a Thomas POPP. *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. New York: Springer, 2007. 337 s. ISBN 978-0-387-30857-9.

[18] MARTINÁSEK, Zdeněk. *Kryptoanalýza postranními kanály*. Brno, 2013. Dizertační práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací. Vedoucí práce doc. Ing. Václav Zeman, Ph.D.

[19] NASSAR, M., Y. SOUISSI, S. GUILLEY a J-L DANGER. RSM: A small and fast countermeasure for AES, secure against 1st and 2nd-order zero-offset SCAs. In: *Design, Automation and Test in Europe* [online]. 2012 [cit. 2014-05-26]. ISBN 978-1-4577-2145-8. Dostupné z WWW: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6176671>.

[20] NEČAS, Ondrej. *Útok elektromagnetickým postranním kanálem*. Brno, 2011. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, Vedoucí práce Ing. Peter Stančík.

[21] PEETERS, Eric. *Advanced DPA theory and practice: towards the security limits of secure embedded circuits*. New York: Springer, 2013, 139 s. ISBN 978-1-4614-6782-3.

[22] RECHBERGER, Christian a Elisabeth OSWALD. Practical Template Attacks. In: *Information Security Applications* [online]. Berlin Heidelberg: Springer, 2005 [cit. 2014-05-26]. Dostupné z WWW: <http://www.springerlink.com/index/10.1007/978-3-540-31815-6_35>.

[23] RIVEST, Ronald L. Cryptography and machine learning. In: *Advances in Cryptology – ASIACRYPT '91 International Conference on the Theory and Application of Cryptology Fujiyosida, Japan, November 1991 Proceedings* [online]. Berlin, Heidelberg: Springer-Verlag, 1993, s. 427 [cit. 2014-05-26]. Dostupné z WWW: <http://www.springerlink.com/index/10.1007/3-540-57332-1_36>.

[24] RSA Laboratories. *PKCS #1 v2.2: RSA Cryptography Standard* [online]. EMC Corporation, 2011 [cit. 2014-05-26]. 63 s. Dostupné z WWW: <http://www.emc.com/emc-plus/rsa-labs/pkcs/files/h11300-wp-pkcs-1v2-2-rsa-cryptography-standard.pdf>.

[25] WESTE, Neil H a David Money HARRIS. *CMOS VLSI Design: A Circuits and System Perspective*. Boston: Addison Wesley, 2005. 838 s. ISBN 0-321-26977-2.

[26] Zapletal, Ondrej. *Proudový postranní kanál*. Brno, 2012. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, Vedoucí práce Ing. Zdeněk Martinásek.

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

3DES	Triple DES
ANN	Artificial Neural Networks
AES	Advanced Encryption Standard
CMOS	Complementary Metal-Oxide Semiconductor
CHES	Cryptographic Hardware and Embedded Systems
DES	Data Encryption Standard
DPA	Differential Power Analysis
DSP	Digital Signal Processing
FIFO	First In First Out
FPGA	Field Programmable Gate Array
HD	Hamming Distance
HW	Hamming Weight
NIST	National Institute of Standards and Technology
PA	Power Analysis
PCA	Principal Component Analysis
POI	Point of Interest
RSA	Rivest-Shamir-Adleman algoritmus
RSM	Rotating Sbox Masking
SPA	Simple Power Analysis
SQL	Structured Query Language
SVM	Support Vector Machines
TA	Template Attacks
XOR	Exkluzivní disjunkce

SEZNAM PŘÍLOH

A Advanced Encryption Standard **52**
A.1 Proces šifrování pomocí AES 52
A.1.1 Key Expansion 53
A.1.2 Inicializační fáze 53
A.1.3 `SubBytes` transformace 53
A.1.4 `ShiftRows` transformace 54
A.1.5 `MixColumns` transformace 54
A.1.6 `AddRoundKey` transformace 55
A.1.7 Pruběh finální rundy 56

B Šablona pro implementaci útoku **57**

C Obsah přiloženého DVD **59**

A ADVANCED ENCRYPTION STANDARD

V roce 1997 americký úřad National Institute of Standards and Technology (NIST) zahájil proces vytváření nového standardu pro šifrování Advanced Encryption Standard, který by byl schopný poskytnout dostatečné zabezpečení pro citlivé vládní informace a nahradil zastaralý Data Encryption Standard (DES) [8]. Po zdlouhavém procesu byl na konci roku 2001 zvolen algoritmus Rijndael, pojmenovaný podle tvůrců Joana Daemena a Vincenta Rijmena, jako nový AES standard. Celé znění standardu AES je možné nalézt v literatuře [7].

Algoritmus AES umožňuje zašifrování vstupní bitové posloupnosti o délce 128 bitů do 128 bitové výstupní posloupnosti. Tyto vstupní a výstupní bitové posloupnosti jsou označovány jako bloky dat otevřeného a šifrovaného textu. Šifrovací klíče algoritmu AES mohou podle standardu nabývat délky 128, 192 či 256 bitů. Podle délky šifrovacího klíče existují různé varianty AES, a to AES-128, AES-192 a AES-256.

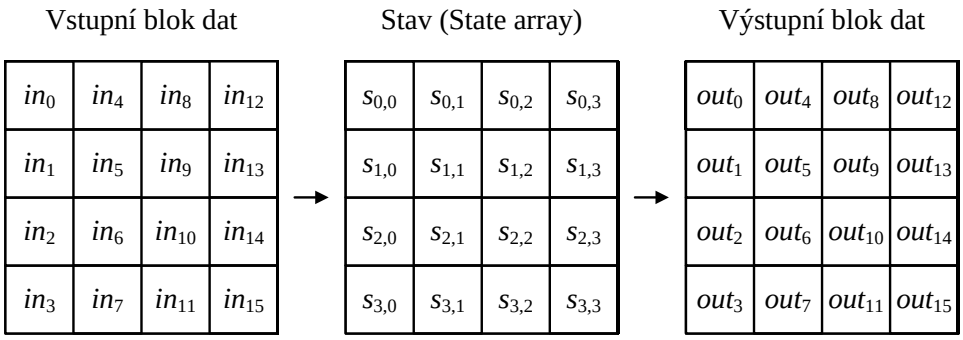
Celý proces šifrování a dešifrování vstupního bloku dat, jak znázorňuje obr. A.6, probíhá v tzv. rundách (Rounds), které se skládají ze čtyř transformací. Pokud je vstupní blok dat šifrován, dochází k jeho modifikaci za pomoci těchto transformací: `SubBytes`, `ShiftRows`, `MixColumns` a `AddRoundKey`. Při dešifrování se pracuje s inverzními transformacemi, tedy `InvSubBytes`, `InvShiftRows`, `InvMixColumns` a `AddRoundKey`. Jak tyto transformace probíhají, je podrobněji popsáno v následující části. Počet rund, kterými vstupní blok dat musí opakovaně projít, závisí na délce použitého klíče (viz tab. A.1).

Tab. A.1: Počet rund pro různé varianty AES.

Varianta AES	Délka vstupního bloku	Délka klíče	Počet rund
AES-128	128 bitů	128 bitů	10 rund
AES-192	128 bitů	192 bitů	12 rund
AES-256	128 bitů	256 bitů	14 rund

A.1 Proces šifrování pomocí AES

Algoritmus AES provádí výše zmíněné transformace na dvojrozměrném poli bajtů velikosti 4×4 označovaném jako *Stav* (State array). Každý bajt v tomto poli je označen dvěma indexy, číslem řádku r a číslem sloupce c. Pozice jednotlivých bajtů $s_{r,c}$ jsou tedy jednoznačně určeny. Na začátku procesu šifrování či dešifrování je vstupní blok dat nejprve nakopírován do dvojrozměrného pole *Stavu* a po finální rundě nakopírován z pole *Stavu* do výstupního bloku dat (viz obr. A.1).



Obr. A.1: Dvojrozměrné pole bajtů (*Stav*).

A.1.1 Key Expansion

Algoritmus AES vytvoří z šifrovacího klíče jednotlivé rundovní klíče, podrobněji popsáno v literatuře [7].

A.1.2 Inicializační fáze

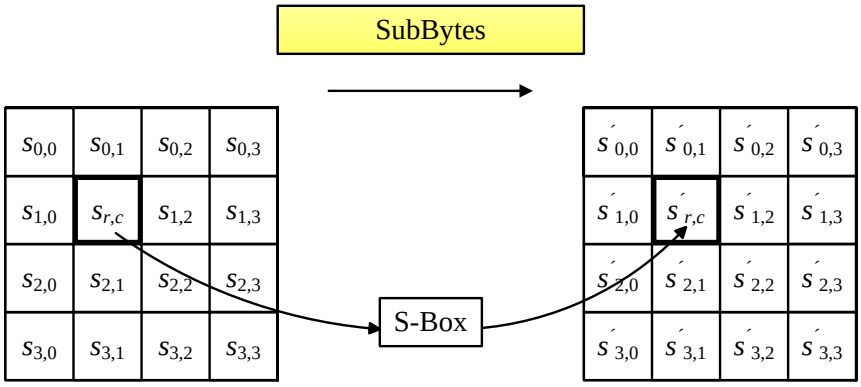
Každý bajt *Stavu* je kombinován s příslušným bajtem šifrovacího klíče prostřednictvím bitové funkce exkluzivní disjunkce (XOR), jejíž pravdivostní hodnoty jsou uvedeny v tab. A.2. Tato transformace se nazývá `AddRoundKey`.

Tab. A.2: Pravdivostní tabulka logické operace XOR.

B	A	A $\oplus$ B
0	0	0
0	1	1
1	0	1
1	1	0

A.1.3 `SubBytes` transformace

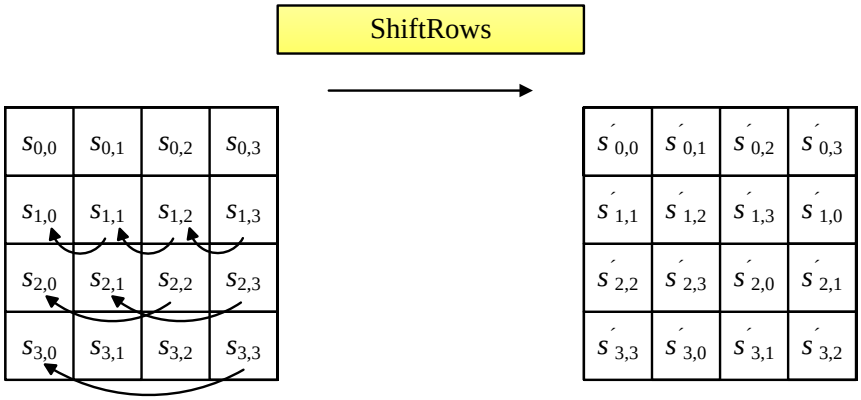
Tato nelineární transformace provede substituci (nahrazení) každého bajtu vytvořeného v inicializační fázi hodnotou v substituční tabulce, tzv. S-Boxu (viz obr. A.2).



Obr. A.2: Transformace `SubBytes`.

A.1.4 `ShiftRows` transformace

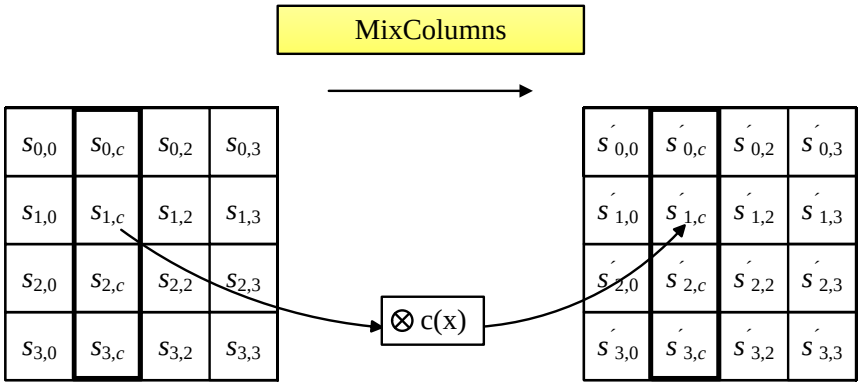
Bajty v posledních třech řádcích *Stavu* jsou cyklicky posunuty doleva o určitý počet kroků. Počet kroků, o které je bajt v řádku matice posunut, udává číslo řádku matice. První řádek se označuje jako nultý (viz obr. A.3).



Obr. A.3: Transformace `ShiftRows`.

A.1.5 `MixColumns` transformace

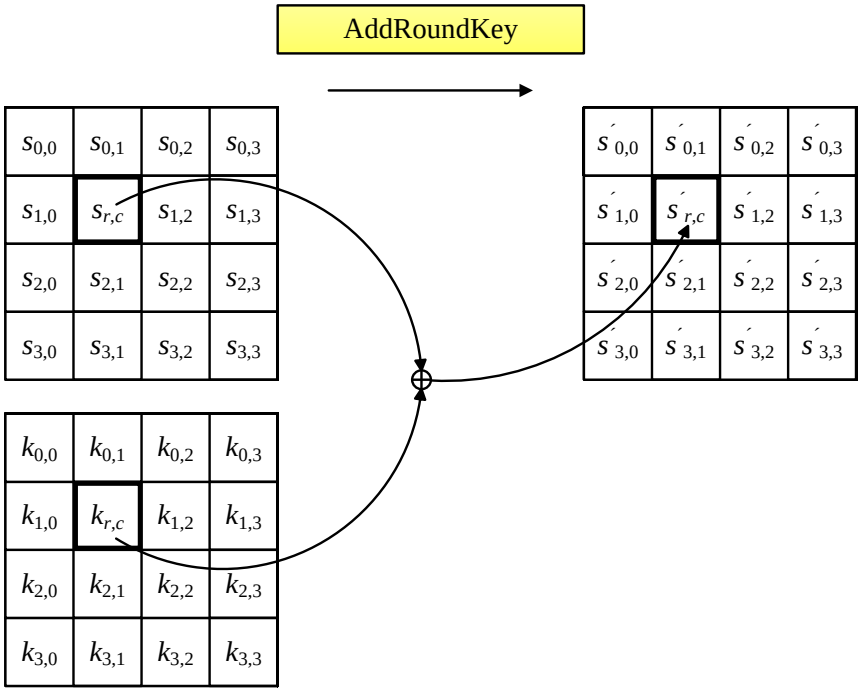
Při této transformaci je každý sloupec *Stavu* vynásoben pevně daným polynomem $c(x)$ (viz obr. A.4).



Obr. A.4: Transformace `MixColumns`.

A.1.6 `AddRoundKey` transformace

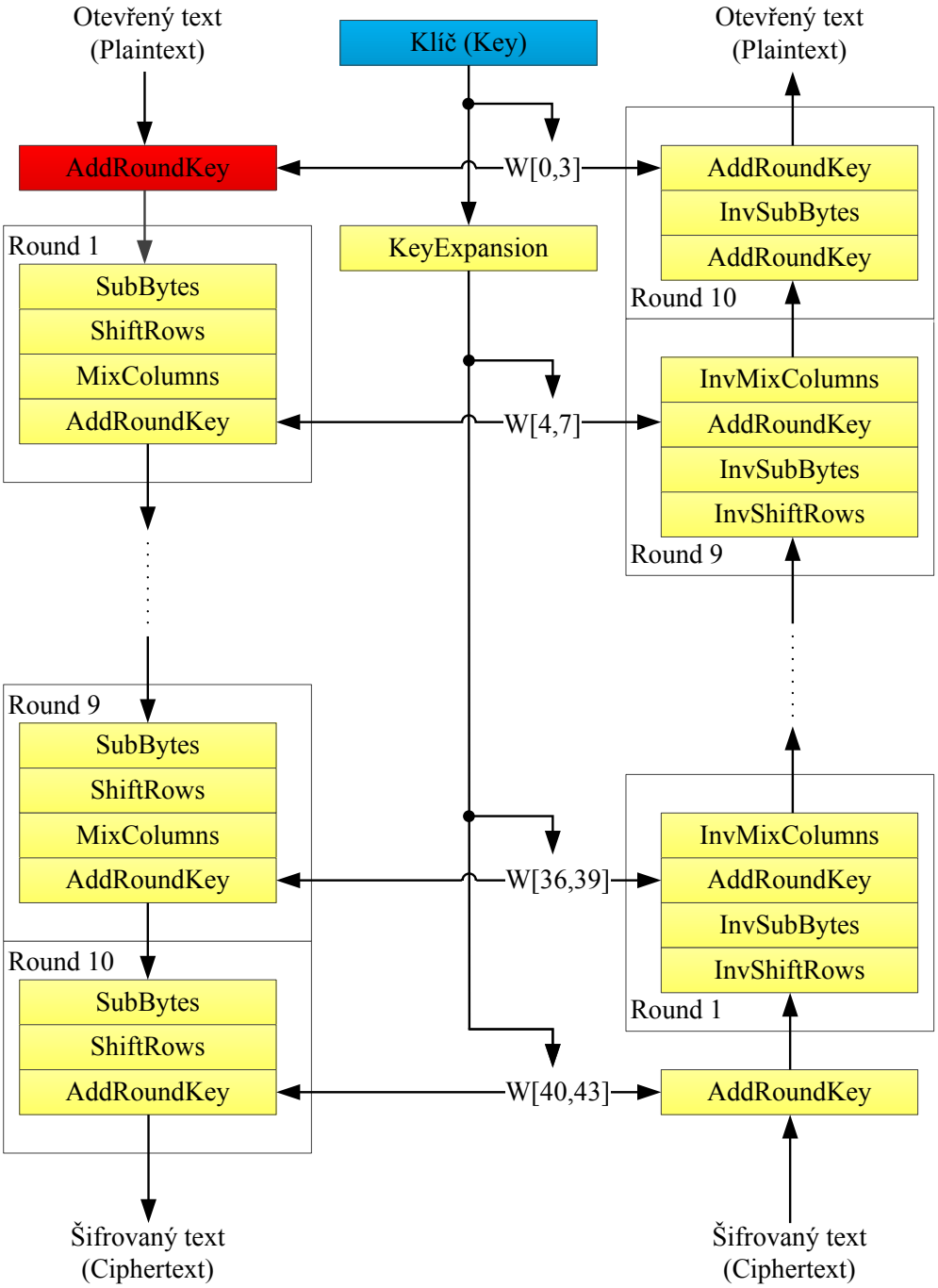
Každý bajt *Stavu* je kombinován s příslušným bajtem rundovního klíče prostřednictvím bitové funkce XOR. Tato transformace odpovídá inicializační fázi. Využívá se však rundovních klíčů odvozených z hlavního šifrovacího klíče (viz obr. A.5).



Obr. A.5: Transformace `AddRoundKey`.

A.1.7 Pruběh finální rundy

Finální runda je identická jako devět předchozích. Obsahuje však pouze transformace `SubBytes`, `ShiftRows`, a `AddRoundKey`.



Obr. A.6: Princip šifrování (dešifrování) algoritmem AES-128 [3].

B ŠABLONA PRO IMPLEMENTACI ÚTOKU

```
%
% Framework for developping attacks in Matlab under Unix
% for the DPA contest V4, AES256 RSM implementation
%
% Version 1, 24/07/2013
%
% Guillaume Duc <guillaume.duc@telecom-paristech.fr>
%

% FIFO filenames (TODO: adapt them)

fifo_in_filename = 'fifo_from_wrapper';
fifo_out_filename = 'fifo_to_wrapper';

% Number of the attacked subkey
% TODO: adapt it
attacked_subkey = 0;


% Open the two communication FIFO

[fifo_in,msg] = fopen(fifo_in_filename, 'r');
if fifo_in < 0
    error('Cannot open FIFO: %s', msg);
end

[fifo_out,msg] = fopen(fifo_out_filename, 'w');
if fifo_out < 0
    error('Cannot open FIFO: %s', msg);
end

% Retrieve the number of traces

num_traces = fread(fifo_in, 1, '*uint32', 0, 'l');

% Send start of attack string

fwrite(fifo_out, [10 46 10], 'uint8');

% Main iteration
for iteration = 1:num_traces
    % Read trace
    plaintext = fread(fifo_in, 16, '*uint8'); % 16x1 uint8
```

```
44      ciphertext = fread(fifo_in, 16, '*uint8'); % 16x1 uint8
45      offset = fread(fifo_in, 1, '*uint8'); % 1x1 uint8
46      samples = fread(fifo_in, 435002, '*int8'); % 435002x1 int8
47
48      % TODO: put your attack code here
49      %
50      % Your attack code can use:
51      % - plaintext (16x1 uint8) : the plaintext
52      % - ciphertext (16x1 uint8) : the ciphertext
53      % - offset (1x1 uint8): the offset (0 unless the wrapper is ...
            launched with --provide_offset_v4_rsm)
54      % - samples (435002x1 int8) : the samples of the trace
55      %
56      % And must produce bytes which is a 256 lines x 16 columns array
57      % (matrix) where for each byte of the attacked subkey (the ...
            columns of
58      % the array), all the 256 possible values of this byte are sorted
59      % according to their probability (first position: most ...
            probable, last:
60      % least probable), i.e. if your attack is successful, the value ...
            of the
61      % key is the first line of the array.
62
63      bytes = repmat((0:255)', 1, 16);
64
65
66      % Send result
67      fwrite(fifo_out, attacked_subkey, 'uint8');
68      fwrite(fifo_out, bytes, 'uint8');
69  end
70
71  % Close the two FIFOs
72  fclose(fifo_in);
73  fclose(fifo_out);
```

C OBSAH PŘILOŽENÉHO DVD

- Elektronická verze diplomové práce.
- Softwarové nástroje pro implementaci útoku od autorů DPA Contest.
- Realizovaný útok.
  - Stručný popis útoku.
  - Zdrojový kód útoku.
  - Knihovna pro práci s ANN.
- Proudové průběhy pro testování útoku.