VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

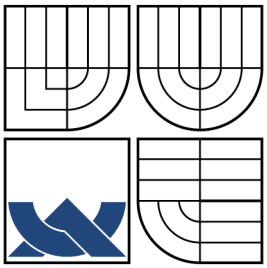
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

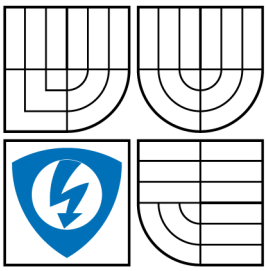
KRYPTOANALÝZA POSTRANNÍMI KANÁLY

DIZERTAČNÍ PRÁCE
DOCTORAL THESIS

AUTOR PRÁCE Ing. ZDENĚK MARTINÁSEK
AUTHOR



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

KRYPTOANALÝZA POSTRANNÍMI KANÁLY
SIDE CHANNEL CRYPTANALYSIS

DIZERTAČNÍ PRÁCE
DOCTORAL THESIS

AUTOR PRÁCE — Ing. ZDENĚK MARTINÁSEK
AUTHOR

VEDOUCÍ PRÁCE — doc. Ing. VÁCLAV ZEMAN, Ph.D.
SUPERVISOR

BRNO 2013

ABSTRAKT

Postranní kanály v oblasti kryptografie zásadním způsobem mění pohled na bezpečnost celého kryptografického systému. Již nestačí analyzovat bezpečnost algoritmu pouze z matematického hlediska pomocí abstraktních modelů, ale stejný důraz musí být kladen na implementaci algoritmů. Disertační práce v úvodu vysvětluje základní pojmy, princip útoku postranními kanály a jejich základní dělení. V následující části jsou určeny cíle dizertační práce. Hlavním cílem disertační práce je navrhnout a experimentálně ověřit novou metodu analýzy proudovým postranním kanálem, která bude využívat neuronové sítě. Tento hlavní cíl vznikl z rozboru používaných analýz proudovým postranním kanálem uvedených v následujících kapitolách. Tyto kapitoly obsahují podrobný rozbor současně používaných analýz proudovým postranním kanálem a rozbor šifrovacího algoritmu AES. Algoritmus AES byl vybrán, z důvodu odolnosti proti konvenčnímu způsobu analýz. Následující kapitola popisuje získané dílčí experimentální výsledky optimalizace stávajících metod, vliv parametrů ovlivňující proudovou spotřebu a výsledky navržené analýzy pomocí neuronových sítí včetně diskuze získaných výsledků. Tento typ útoku proudovým postranním kanálem nebyl dosud publikován, jedná se tedy o zcela novou myšlenku. Posledním cílem práce bylo shrnutí možných ochran proti analýze a útoku postranním kanálem.

KLÍČOVÁ SLOVA

Postranní kanály, proudový postranní kanál, jednoduchá proudová analýza, diferenciální proudová analýza, neuronové sítě, proudová analýza pomocí neuronových sítí

ABSTRACT

Side channels fundamentally changes the view of the cryptographic system security in cryptography. It is not enough to analyze the security algorithm only from a mathematical point of view using abstract models but it is necessary to focus on the implementation of the algorithms. The introduction of the thesis deals with the basic terms, principles of side channel attacks and basic clasification of side channels. The following chapter describes the objectives of the thesis. The main goal of the thesis is to propose and experimentally verify a new power analysis method whish will use the neural network. This main goal was based on the realized analyzes presented in the following chapters. These chapters contain a detailed analysis of currently used power analysis and analysis of AES encryption algorithm. AES was selected becouse the algorithm is resistant to the conventional cryptoanalysis. The following section describes the experimental results of the optimization of existing methods, the influence of the parameters affecting power consumption and the results of the proposed analysis using neural networks. This section includes the discussion of the results. This type of side channel attack has not been published yet thus it is a completely new idea. The final goal of the thesis was to summarize the possible countermeasures protecting against the side channel attacks.

KEYWORDS

Side channels, power side channel, simple power analysis, differential analysis, neural networks, power analysis using neural networks

MARTINÁSEK, Zdeněk *Kryptoanalýza postranními kanály*: dizertační práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2013. 129 s. Vedoucí práce byl doc. Ing. Václav Zeman, Ph.D.

PROHLÁŠENÍ

Prohlašuji, že svou doktorskou práci na téma „Kryptoanalýza postranními kanály“ jsem vypracoval samostatně pod vedením vedoucího doktorské 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 doktorské práce dále prohlašuji, že v souvislosti s vytvořením této doktorské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.

Brno ....

(podpis autora)

Děkuji vedoucímu dizertační práce Doc. Ing. Václavu Zemanovi, Ph.D. za užitečnou metodickou pomoc v průběhu celého studia a za cenné rady při zpracování dizertační práce. V neposlední řadě bych chtěl poděkovat rodičům a kolegům za podporu během celého studia.

OBSAH

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

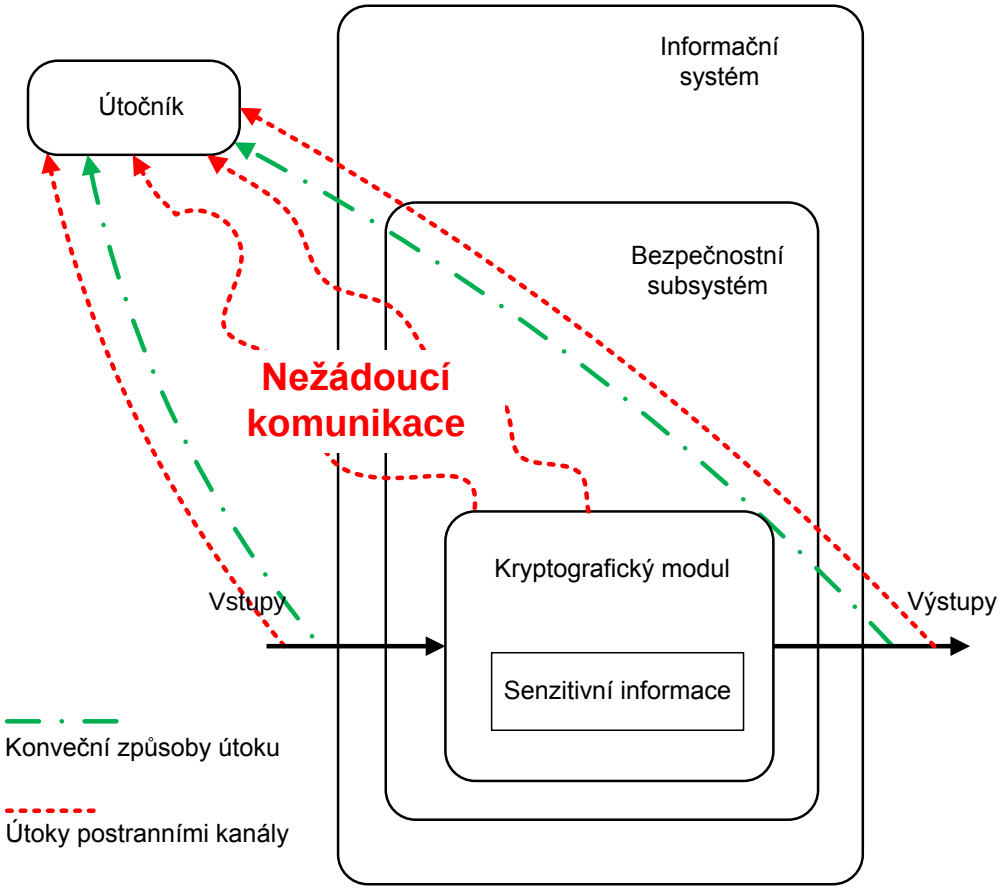
ÚVOD

Se stále zrychlujícím se vývojem moderních komunikačních a počítačových systémů se objevila řada nových typů útoků na kryptografické systémy. Útočníci neodposlouchávají pasivně přenosový kanál, ale využívají stále sofistikovanějších metod kryptoanalýzy. Díky tomuto trendu se požadavky na bezpečnost stále zvyšují. Hlavním úkolem pro kryptografické systémy je zajištění bezpečnosti. V současnosti používané kryptografické prostředky v informačních systémech poskytují k zajištění bezpečnosti tyto služby:

- důvěrnost (confidentiality) - utajení informace před neoprávněnými uživateli,
- autentičnost (authentication) - příjemce je schopen ověřit autora zprávy,
- integritu dat (integrity) - příjemce dokáže jednoznačně rozpoznat zda byla zpráva během přenosu modifikována,
- nepopíratelnost (non-repuditation) - odesílatel nemůže popřít, že danou zprávu odeslal.

Tyto kryptografické služby v celém systému zajišťuje kryptografický modul, který bývá součástí bezpečnostního subsystému (obr.1). Tento modul je v podstatě fyzickou implementací konkrétního kryptografického algoritmu popř. protokolu. Realizace kryptografického modulu může být hardwarová, softwarová nebo kombinovaná. Během činnosti kryptografického modulu probíhají uvnitř procesy, které jsou spojeny s šifrováním, dešifrováním, ověřením, autentizací atd. Během těchto činností pracuje modul se senzitivními daty (např. tajný klíč), které bývají uloženy v paměti modulu. Z toho plyne, že praktická realizace kryptografického modulu, která v sobě obsahuje všechny pravidla, klíče a další senzitivní materiál, do značné míry ovlivňuje bezpečnost celého systému.

Dosavadní konvenční způsoby kryptoanalýzy (viz obr.1, zelená čerchovaná čára) se soustředily na objevení slabiny v matematické podstatě kryptografických algoritmů. Proti v současnosti používaným šifrovacím algoritmům je tento způsob neefektivní a časově prakticky nerealizovatelný. Nový způsob kryptoanalýzy, využívající postranní kanály (Side Channels), se soustředí na konkrétní implementace algoritmů a protokolů. Při konstrukci modulu se předpokládá jediná možná komunikace modulu s okolím a to jen prostřednictvím přesně definované vstupů a výstupů. V reálném prostředí modul během své činnosti komunikuje se svým okolím i jiným, nežádoucím způsobem. Modul může vyzařovat do svého okolí např. tepelné nebo elektromagnetické záření, každý reálný modul při své činnosti odebírá určitý proud ze zdroje, každá jeho operace způsobuje různé časové zpoždění, na konkrétní situaci reaguje modul stavovým nebo chybovým hlášením, klávesnice modulu může být mechanicky opotřebená atd. Všechny tyto projevy modulu jsou neodmyslitelně spojeny s jeho činností, při které mohou být vyneseny některé ze senzitivních informací.



Obr. 1: Model útoku využívající postranní kanály.

Každý nežádoucí způsob výměny informace mezi kryptografickým modulem a jeho okolím se nazývá postranním kanálem.

Analýzou postranního kanálu (Side Channels analysis) je označován postup, při kterém je možné získat užitečné informace, které lze odvodit ze signálu přicházejícím po tomto kanálu. Útok vedený pomocí postranního kanálu je založen na využití takto získané informace k napadení daného kryptografického modulu a získání tak senzitivních informací. Blokové schéma kryptoanalýzy využívající postranní kanály je zobrazeno na obr. 1 červenou čárkovanou čárou.

Koncept útoku postranními kanály, tak jak je chápán v dnešní době, zadefinoval a popsal Kocher v práci [55] v roce 1999. Princip útoku byl proveden na algoritmus Data Encryption Standard metodou založenou na rozdílu středních hodnot. Postranní kanály se prakticky využívaly dříve, kdy se definice postranních kanálů nepoužívala. Akustický postranní kanál patří k nejstarším používaným postranním kanálům, např. v roce 1956 Britové získávají informace z egyptského šifrátoru odposlechem zvuků klávesnice a v roce 1961 Američané provádějí akustický odposlech prostřednictvím ústředního topení. Elektromagnetický postranní kanál byl ve své

historii také nejdříve využíván v armádě a tajných službách. Tyto organizace se odborně zabývaly studiem problematiky parazitních emisí, která označovaly termínem TEMPEST. Hlavním zájmem vojenských organizací bylo zamezení nežádoucích emisí a naopak využití tohoto vyzařování k špionážní činnosti. Pojem TEMPEST vznikl na přelomu 60. a 70.let dvacátého století a označuje i skupinu vojenských standardů, ve kterých jsou stanoveny maximální povolené limity elektromagnetického záření v různých elektronických systémech. Ve veřejném sektoru se o významný posuv na poli elektromagnetických útoků zasloužil nizozemský vědec van Eck [33], který jako první dokázal, že je možné zachytit a změřit velikost elektromagnetického pole počítačových monitorů a z naměřených průběhů extrahovat snímaný obraz. První veřejně publikovanou prací na téma EM analýzy integrovaných obvodů a výpočetních jednotek provádějících kryptografické operace, byla v roce 2001 práce [38].

Postranní kanály zcela mění celkový pohled na bezpečnost systému. Již nestačí zvolit kvalitní šifru ale je nezbytné velkou pozornost věnovat i její implementaci. Návrháři a konstruktéři kryptografického modulu často neví a ani nemohou vědět o existenci všech nežádoucích postranních kanálů. Existují ovšem některé postranní kanály, které jsou schopni minimalizovat. V současné době neexistuje žádný konkrétní návod pro návrh zcela imunního kryptografického modulu vůči postranním kanálům, ale existují testy které otestují navrhovaný modul na některé konkrétní typy postranních kanálů a na množství unikající informace.

1 POSTRANNÍ KANÁLY V KRYPTOGRAFII

Cílem kapitoly je uvést základní přehled postranních kanálů a definovat používané odborné pojmy. Úvod kapitoly vysvětluje pomocí časového postranního kanálu princip útoku postranním kanálem a následně je pozornost věnována detailnímu popisu proudového postranního kanálu, na který je práce zaměřena. Ostatní typy postranních kanálů jsou uvedeny pro úplnost a přehlednost problematiky a to jen stručným popisem.

Každý typ postranního kanálu je založen na konkrétní měřitelné informaci a často mívají tyto informace podobu fyzikální veličiny, kterou je schopen potencionální útočník změřit. Základní dělení postranních kanálů vychází z fyzikální veličiny nebo typu informace prosakující prostřednictvím postranního kanálu. Kryptoanalytici považují v současné době za hlavní druhy postranních kanálů následující typy (pro lepší orientaci s odbornou literaturou je v závorce uveden anglický název):

- časový (timing),
- výkonový (power),
- elektromagnetický (electromagnetic),
- optický (optical),
- chybový (fault),
- ostatní typy postranních kanálů.

Prostřednictvím postranního kanálu prosakují informace obsahující senzitivní údaje z kryptografického modulu. Útočník je musí v rámci útoku zpracovat a vyhodnotit. Zpracování a vyhodnocení informací je v kryptografii souhrnně nazýváno analýzou kanálu. Existují dvě základní analýzy:

- jednoduchá (Simple Analysis, SA),
- diferenciální (Differential Analysis, DA).

Jednoduchá analýza představuje základní způsob zpracování výsledků. Informace získané z postranního kanálu jsou útočníkem přímo pozorovány a vyhodnoceny. Diferenciální analýza vyžaduje použití matematického aparátu. Často však umožňuje nalézt citlivé informace i z postranních kanálů, kde jejich přítomnost není zřejmá. Na každý typ postranního kanálu můžeme aplikovat oba způsoby analýz např. pro proudový postranní kanál je to jednoduchá proudová analýza (Simple power analysis, SPA) nebo diferenciální proudová analýza (Diferen power analysis, DPA). Útok postranním kanálem lze chápat jako proces využití analyzované informace získané postranním kanálem k napadení kryptografického modulu.

Do kategorie ostatní typy postranních kanálů spadají ty jejichž existence byla prokázána, ale na využití ještě nebyla soustředěna dostatečná pozornost a tudíž se zatím nedostaly do popředí. Patří sem například postranní kanál akustický [113], frekvenční [105], využívající viditelné světlo [57] a využívající skenovací řetězec [111]

atd. Tyto postranní kanály nebudou v práci analyzovány. Při současné tendenci výzkumu lze také předpokládat, že budou objeveny další dosud neznámé postranní kanály.

1.1 Časový postranní kanál

Časový postranní kanál je prvním publikovaným a typickým příkladem postranního kanálu. Základní myšlenka útoku byla publikována na vědecké konferenci již v roce 1996 [55]. S existencí možnosti vedení útoku na kryptografický modul přišel známý americký kryptolog Paul Carl Kocher.

Časový útok (Timing Analysis, TA) je založen na měření času, který je potřebný k vykonání určitých operací ve sledovaném modulu. Pod pojmem určité operace uvnitř kryptografického modulu chápeme například procesorové instrukce (násobení, dělení, sčítání, rotace . . .), větvení programu (tj. vykonání podmíněných příkazů), operace čtení a zápisu dat do paměti atd. Jedná se v podstatě o provádění kryptografického algoritmu, kde prováděné operace mají různou délku trvání v závislosti na vstupních datech a tajném šifrovacím klíči. Z uvedeného vyplývá, že útok časovým postranním kanálem je použitelný k napadení každého kryptografického modulu, ve kterém existuje přímá souvislost mezi hodnotou klíče a dobou výpočtu. Konkrétní realizace útoků využívají různé operace příslušného kryptografického modulu a různé sofistikované statistické nástroje pro vyhodnocování naměřených časových údajů. Ve většině odborných publikací bývá TA vysvětlována na implementaci algoritmu RSA (iniciály autorů Rivest, Shamir, Adleman)[1] [55]. V této práci je pro změnu představena jednoduchá časová analýza na implementaci algoritmu verifikace hesla [32]. Tento příklad jednoduché analýzy je uveden k vysvětlení principu útoku postranními kanálem. Příkladem diferenciální časové analýzy na algoritmus AES (Advanced Encryption Standard), je útoku využívající vyrovnávací paměť procesorů [14, 90, 85, 83].

Útok na verifikaci hesla

Některé aplikace vyžadují po uživateli při procesu autentizace heslo, aby umožnily přístup k informacím nebo datům jen pro oprávněné uživatele (proces autorizace). Například soubor vyžaduje po uživateli heslo pro otevření obsahu nebo na webovém rozhraní se musí uživatel přihlásit zadáním jména a hesla. Má-li heslo délku 8 bajtů,

[1] Algoritmus RSA je založen na matematické operaci modulární mocniny, na implementaci výpočtu se často používá tzv. algoritmus square and multiply, který je založen na postupném zpracování jednotlivých bitů soukromého klíče, doba výpočtu je závislá na hodnotě klíče.

```
Input: PU =(P[0],..., P[7]), PS =(P[0],...,P[7]),
Output: 'true'or'false'

1: for j = 0 to 7, j++
2:      if (PU[j] ~= PS[j])
3:       return 'false'
4: end
5: return 'true'
```

Obr. 1.1: Algoritmus verifikace hesla.

útok hrubou silou[2] by vyžadoval realizovat $256^8 = 2^{64}$ pokusů. Osobní počítač s procesor o frekvenci jádra 3GHz by na útok hrubou silou potřeboval cca 71168 dní a to odpovídá 195 rokům. Pro běžného uživatele je tato výpočetně prokazatelná bezpečnost dostatečná.

Verifikace hesla bývá v praxi implementována algoritmem, který je zobrazen na obr. 1.1, kde `PU` představuje osmi bajtové heslo zadané uživatelem a `PS` značí správné heslo [51]. Princip algoritmu je na první pohled zřejmý, jednoduchý podmíněný cyklus vrací `true` v případě, že heslo bylo zadáno korektně a v opačném případě vrací hodnotu `false`. Útočník může měřit čas potřebný k vykonání algoritmu a z průběhu algoritmu plyne, že doba vykonání algoritmu bude delší při zadání správného hesla, ve srovnání s dobou, kdy se heslo bude lišit např. hned ve druhém bajtu. Na této skutečnosti je založen následující útok využívající jednoduchou časovou analýzu:

1. útočník si nejprve vytvoří sadu 256-ti hesel délky 8-mi bajtů následujícím způsobem: $PU = \{n, 0, 0, 0, 0, 0, 0, 0\}$, kde $0 \geq n \leq 256$ představuje hodnotu prvního bajtu hesla. Útočník měří odpovídající čas vykonání algoritmu $\tau[n]$ pro všechny vytvořené hesla.
2. Následně vybere maximální naměřenou hodnotu:

$$\tau[n_0] := \underbrace{max}_{0 \geq n \leq 256} \tau[n], \tag{1.1}$$

   a tím získá první hodnotu bajtu hesla $PU[0]$, která odpovídá n_0.
3. Hodnota prvního bajtu je odtajněna a útočník pokračuje stejným způsobem pro druhý bajt hesla $PU = \{P[0], n, 0, 0, 0, 0, 0, 0\}$. Tento postup opakuje dokud nezíská hodnoty všech bajtů hesla.

[2]Při útoku hrubou silou útočník zkouší postupně všechny možné kombinace tajného klíče dokud nenalezne správnou kombinaci.

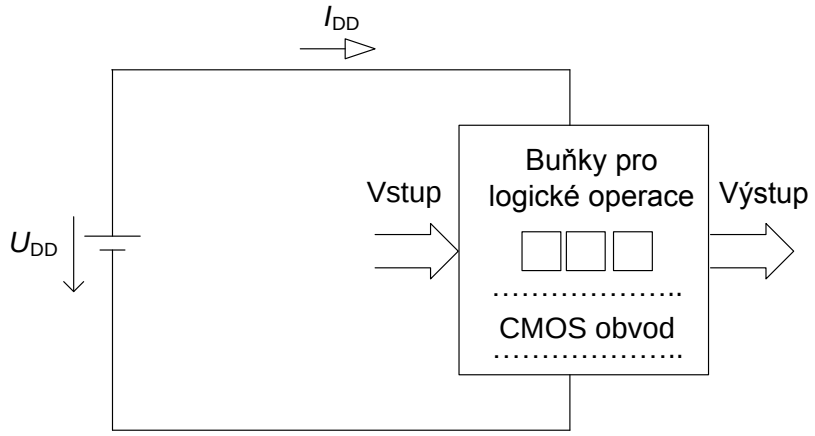
Útok výše popsaným způsobem je velice efektivní, pro získání hesla potřebujeme maximálně $256 \cdot 8 = 2048$ realizací verifikačního algoritmu. Útočník je schopen otestovat jednu realizaci algoritmu na běžně dostupném počítači během 1s, z toho plyne, že celý útok může trvat cca. 30 minut. I pro běžného uživatele je tato výpočetně prokazatelná bezpečnost zcela nedostatečná. Z porovnání časů potřebných na prolomení hesla, kdy útočník použije metodu hrubé síly a TA je vidět vysoká efektivita a síla postranních kanálů.

V praxi se hesla neukládají přímo, ale pouze jejich haše (otisky). Haše nijak neodhalují hesla (díky jednocestnosti hašovacích funkcí), z nichž jsou vypočteny, ale přitom umožňují kontrolu jejich správnosti při přihlašování uživatelů. Aby se vyloučil slovníkový útok, kdy si útočník předvypočítá haše pro často používaná slova a výrazy, používá se tzv. metoda solení. Při ní se vždy při ukládání otisku hesla vygeneruje i náhodný řetězec (tzv. sůl), který se poté hašuje dohromady s vlastním heslem. Do databáze hašů se pak ukládá dvojice (sůl, hash(heslo, sůl)).

Jako nejjednodušší protiopatření proti TA, můžeme použít vložení náhodného zpoždění před návratovou hodnotu algoritmu. To však nezabrání útočníkovi v podobném útoku. Útok v tomto případě bude probíhat úplně stejným způsobem, jen v prvním kroku útočník zkouší heslo $PU = \{n, 0, 0, 0, 0, 0, 0, 0\}$ t-krát a měří odpovídající průměrné časy $\overline{\tau}[n]$ pro $0 \geq n \leq 256$. Obtížnost útoku se zvýšila o faktor t. Jediným správným protiopatřením TA je realizace konstantní časové implementace algoritmu bez závislosti vstupních dat.

1.2 Proudový postranní kanál

Výkonová analýza (Power analysis, PA) studuje proudovou spotřebu kryptografického zařízení v závislosti na jeho činnosti, byla představena v roce 1999 panem Kocherem [55]. Průběh proudové spotřeby elektronického zařízení (kryptografického modulu) není s časem konstantní a na první pohled připomíná nahodilý šum. Většina moderních kryptografických zařízení bývá založena na technologii CMOS (Complementary Metal Oxide Semiconductor). Celková proudová spotřeba CMOS obvodu je dána součtem proudových spotřeb jednotlivých buněk, z kterých se skládá obvod. Z tohoto důvodu proudová spotřeba závisí na počtu logických buněk v obvodu, na spojení mezi nimi a na tom jak jsou buňky tvořeny. Obvody CMOS jsou většinou napájeny konstantním napájecím napětím $\mathrm{U_{DD}}$, jak ukazuje obr. 1.2, při práci obvodu logické buňky zpracovávají vstupní data a odebírají proud ze zdroje napětí. Celkový okamžitý proud označíme $i_{\mathrm{DD}}(t)$ a okamžitou výkonovou spotřebu $p_{\mathrm{obv}}(t)$. Potom průměrná výkonová spotřeba P_{obv} obvodu za čas T může být vypočítána



Obr. 1.2: Obecné schéma proudové spotřeby CMOS obvodu.

podle vztahu:

$$P_{\text{obv}} = \frac{1}{T}\int_0^T p_{\text{obv}}(t)dt = \frac{\text{U}_{\text{DD}}}{T}\int_0^T i_{\text{DD}}(t)dt. \tag{1.2}$$

Základním stavebním logickým prvkem (buňkou) založeným na CMOS technologii je invertující člen (obr. 1.3). Pomocí invertoru bude podrobně vysvětlen princip odebírání proudu ze zdroje, tento princip platí i pro všechny složitější buňky. Invertor obsahuje jen dva tranzistory řízené napětím s opačným typem vodivosti. Složitější buňky pracují na stejném principu, ale obsahují více těchto tranzistorů. Branami CMOS tranzistorů u invertoru protékají tři různé druhy proudů [89], první se nazývá svodový proud (direct path current), druhý proud nabíjející/vybíjecí (charge/discharge current) parazitní kapacity a třetí se nazývá zbytkový proud (leakage current). Vznik jednotlivých proudů a popis činnosti invertoru je podrobněji popsán v následujícím textu.

Obecně lze celkovou výkonovou spotřebu invertoru rozdělit do dvou základních částí. První částí je statická výkonová spotřeba P_{stat}, která je odebírána invertorem ve stabilních stavech. Druhou částí spotřeby je dynamická spotřeba P_{dyn}, která nastane dojde-li k přepnutí stavu na vstupu nebo výstupu invertoru. Celková spotřeba invertoru je následně dána součtem P_{stat} a P_{dyn}. Pro krátký časový úsek a výstup invertoru můžeme obecně definovat čtyři přechody viz tab. 1.1. Pro dva přechody $0 \rightarrow 0$ a $1 \rightarrow 1$ je výkonová spotřeba dána statickou spotřebou a pro zbylé přechody $0 \rightarrow 1$ a $1 \rightarrow 0$ je celková výkonová spotřeba dána součtem statistické a dynamické výkonové spotřeby. Obecně platí $P_{00} \approx P_{11} \ll P_{01}, P_{10}$ a z toho plyne, že dynamická spotřeba je závislá na právě zpracovávaných datech.

Tab. 1.1: Přechody výstupu invertoru a odpovídající výkonová spotřeba.

Přechod	Výkonová spotřeba	Typ výkonové spotřeby
$0 \rightarrow 0$	P_{00}	statická
$0 \rightarrow 1$	P_{01}	statická + dynamická
$1 \rightarrow 0$	P_{10}	statická + dynamická
$1 \rightarrow 1$	P_{11}	statická

1.2.1 Statická výkonová spotřeba

Invertor obsahuje dva tranzistory T1 (PMOS) a T2 (NMOS) řízené napětím s opačným typem vodivosti (viz obr.1.3). Invertor pracuje následovně:

- je-li vstupní napětí (U_{IN}) v logické úrovni 1, je otevřen tranzistor T2 a T1 je uzavřen,
- je-li vstupní napětí (U_{IN}) v logické úrovni 0, je otevřen tranzistor T1 a T2 je uzavřen.

V obou těchto stabilních stavech je výkonová spotřeba minimální. Například při vstupní hodnotě napětí U_{IN} menší než hodnota prahového napětí tranzistoru je tranzistor T2 (NMOS) uzavřen (cutoff region). Invertorem neprotéká prakticky žádný proud. Přesněji řečeno invertorem protéká velmi malý zbytkový proud odpovídající podprahovému proudu NMOS tranzistoru a závěrným proudům P-N přechodů oblastí emitoru a kolektoru. Označme zbytkový proud I_{zbytk}, potom statická spotřeba P_{stat} může být spočítána dle vztahu:

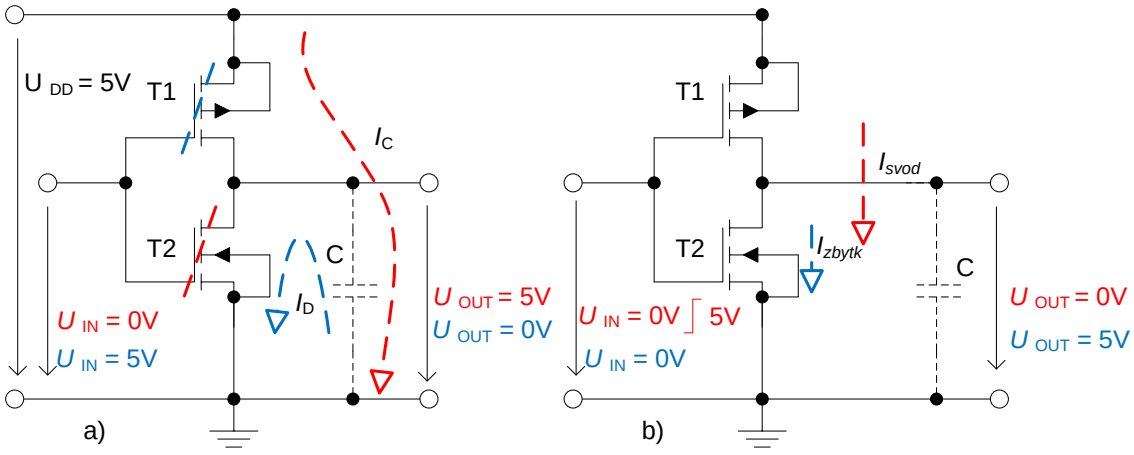
$$P_{\mathrm{stat}} = I_{\mathrm{zbytk}} \cdot \mathrm{U_{DD}}. \tag{1.3}$$

Zbytkový proud pro MOS tranzistor je typicky v rozsahu 10^{-12}A, tato hodnota se neustále zvyšuje pro moderní výrobní technologie. Zbytkový proud pro samostatný tranzistor NMOS je dán vztahem:

$$I_{\mathrm{zbytk,NMOS}} = I_0 \frac{W}{L} e^{-U_{\mathrm{TH}}/nkT/q}, \tag{1.4}$$

kde U_{TH} je prahové napětí transistoru, I_0 a n jsou parametry závislé na technologii, W/L je poměr tranzistoru (W - šířka kanálu a L - délka kanálu), T je teplota, k Boltzmanova konstanta a q náboj elektronu. Analogický vztah platí pro PMOS tranzistor. Velikost zbytkového proudu (rovnice 1.4) je silně závislá na teplotě a velikosti prahového napětí U_{TH}.

Zbytkový proud CMOS invertoru odpovídá zbytkovému proudu transistoru NMOS nebo PMOS (rovnice 1.4) v závislosti na tom, který z nich je právě uzavřen. Jak je popsáno výše je-li vstupní napětí (U_{IN}) v logické úrovni 1, je otevřen tranzistor



Obr. 1.3: Model invertoru a) nabíjení b) vybíjení parazitní kapacity.

T2 (NMOS) a T1 (PMOS) je uzavřen a je-li vstupní napětí v logické úrovni 0, je otevřen tranzistor T1 a T2 je uzavřen. Velikost prahového napětí u reálné CMOS technologie je rozdílná pro NMOS a PMOS tranzistor tedy i odpovídající zbytkové proudy jsou rozdílné (rovnice 1.4). Z předchozích tvrzení vyplývá, že velikost zbytkového proudu silně závisí na vstupu digitálního obvodu [3]. Útočník může dát do souvislosti vstupní otevřený text a průběhy zbytkových proudů a zjisti tímto informace o tajném klíči. Zbytkový proud může být měřen obdobným způsobem jako proud u dynamické výkonové spotřeby v tradiční proudové analýze.

Označení statické analýzy je LPA (Leakage Power Analysis) vychází tedy z existence zbytkového proudu. Obecná existence zbytkových a závěrných proudů byla poprvé publikována v článku [41] z této publikace vychází práce [62], která se zabývá simulacemi zbytkových proudů a na výsledky aplikuje znalosti z klasických dynamické analýzy. Tyto publikace nerozebírají teoretické základy LPA a konkrétní reálný útok. LPA byla poprvé analyzována systematickým způsobem v r. 2010 a to z pohledu teoretického i experimentálního panem Aliotem [9]. Útok si klade za cíl , stejně jak klasická dynamická výkonová analýza, zjištění tajného klíče z kryptografické modulu z výsledků měření statické výkonové spotřeby (dána velikostí zbytkového proudu). Tento typ útoku vychází z faktu, že pro nové výrobní technologie čipů již není dynamický výkon hlavní částí z celkového výkonu čipu, ale mnohem rychleji vzrůstá výše zmiňovaný statický výkon [9]. Například u výrobní technologie 65nm je statický výkon polovina celkové výkonové spotřeby čipu a plánuje se další zvyšování v následujících generacích.

1.2.2 Dynamická výkonová spotřeba

Dynamická výkonová spotřeba nastává vždy, když je změněn vnitřní stav nebo výstupní stav invertoru, obecně buňky obvodu. Proudová spotřeba odpovídající přepnutí vnitřních stavů buňky se zpravidla zanedbává, protože je daleko nižší ve srovnání s proudovou spotřebou pro přepnutí výstupního stavu buňky. Existují dvě příčiny pro dynamickou výkonovou spotřebu. První příčina nastává při přepnutí stavu invertoru, kdy po krátký čas jsou otevřeny oba tranzistory (T1, T2) a napájení je přes ně zkratováno k zemi (svodový proud). Velikost proudové špičky je úměrná počtu právě přepínaných tranzistorů v celém integrovaném obvodu. Průměrná výkonová spotřeba P_{svod} která je způsobena svodovým proudem během časového okamžiku T může být vypočítána dle následujícího vztahu:

$$P_{\mathrm{svod}} = \frac{1}{T}\int_0^T p_{\mathrm{svod}}(t)dt = P_{0\to1} \cdot f \cdot \mathrm{U_{DD}} \cdot I_{\mathrm{svod}} \cdot t_{\mathrm{svod}}, \tag{1.5}$$

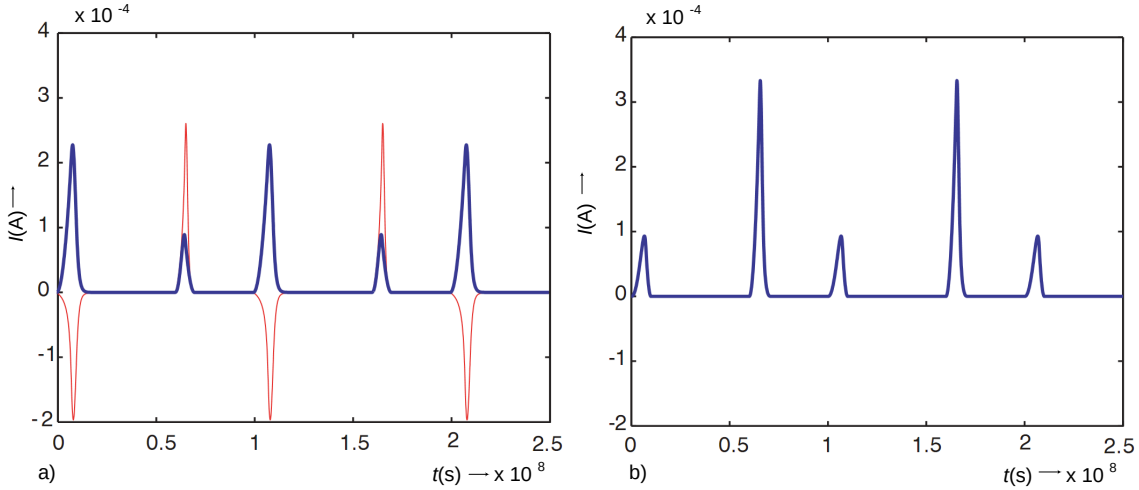
kde $p_{\mathrm{svod}}(t)$ je okamžitá výkonová spotřeba invertoru během krátkého časového úseku T, I_{svod} je svodový proud a t_{svod} je doba po kterou jsou otevřeny oba transistory.

Druhá příčina je nabíjení parazitní kapacity proudem I_{C} a vybíjení parazitní kapacity proudem I_{D} (vybíjecí/nabíjecí proud) což je dominantní zdroj výkonové spotřeby. Tato parazitní kapacita představuje kapacity řídicích elektrod následujících tranzistorů a spoje mezi buňkami. Její velikost záleží na fyzikálních vlastnostech použitého materiálu, výrobní technologii, délkách spojů atd. Velikost parazitní kapacity se pohybuje typicky mezi 10^{-3} až 1 pF. Dynamická výkonová spotřeba invertoru způsobena nabíjením parazitní kapacity lze vyjádřit vztahem [89]:

$$P_{\mathrm{dyn}} = \frac{1}{T}\int_0^T p_{\mathrm{nab}}(t)dt = P_{0\to1} \cdot f \cdot C \cdot \mathrm{U_{DD}^2}, \tag{1.6}$$

kde $p_{\mathrm{nab}}(t)$ je okamžitá výkonová spotřeba invertoru během nabíjení parazitní kapacity za čas T, C je parazitní kapacita, $P_{0\to1}$ je pravděpodobnost přechodu mezi stavy $0 \to 1$, f je kmitočet spínání a $\mathrm{U_{DD}}$ je napájecí napětí.

Pokud měříme výkonovou spotřebu (na zemnící nebo napájecí svorce invertoru) bude největší špička během nabíjení parazitní kapacity, protože během vybíjení můžeme měřit jen svodový proud. Výsledky simulací invertoru zobrazeného na obr. 1.3 znázorňují grafy na obr. 1.4, které potvrzující předchozí tvrzení. Vstupní signál ($\mathrm{U_{IN}}$) představoval sled pulsů. Obr. 1.4 a) zobrazuje velikost proudu transistorem NMOS (modrá barva) a proud parazitní kapacitou (červená barva). Obr. 1.4 b) zachycuje proud procházející svorkou $\mathrm{U_{DD}}$ nebo GND, hodnoty proudu odpovídají součtu předchozích průběhů. Důsledkem výše popsaných příčin výkonových změn je, že výkonová spotřeba kryptografického modulu je přímo závislá na zpracovávaných datech a probíhajících operacích.

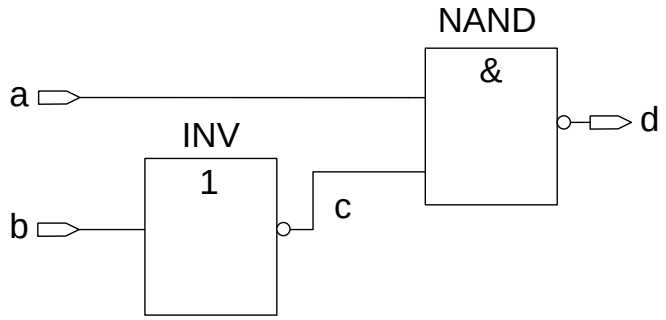


Obr. 1.4: Výsledky simulací nabíjení a vybíjení parazitní kapacity [89].

Za 14 let své existence útoky jednoduchou a diferenční proudovou analýzou jsou obsáhle publikovány, například útok na algoritmus DES (Data Encryption Standard) [55, 111, 47, 7, 102, 101], RSA [51, 47, 77, 13] a AES [103, 11, 10, 107, 69, 87, 88]. Tato četnost vychází z faktu, že zařízení na měření výkonové spotřeby je cenově dostupné prakticky komukoli, útočník nemusí mít k dispozici žádné drahé speciální zařízení, ale postačí měřicí karta do počítače nebo osciloskop. Pro shrnutí, v klasické PA se zkoumá závislost dynamické výkonové spotřeby (viz rovnice 1.6) v závislosti na vstupním otevřeném textu, který je šifrován kryptografickým modulem za pomoci tajného klíče. Hodnoty okamžité výkonu spotřeby jsou zaznamenávány měřicím zařízením v průběhu šifrování pro jednotlivé otevřené texty. Z výkonových průběhů útočník odečte senzitivní informace přímo (SPA) nebo je následně matematicky zpracovává pomocí DPA. Podrobný popis konkrétních analýz je předmětem následujícího textu (kapitola 3).

1.2.3 Hazardní stavy v CMOS obvodech

V CMOS obvodech je typicky spojeno více logických kombinačních buněk za sebou tzn. že výstup jedné kombinační buňky je vstupem druhé kombinační buňky. Výstup druhé je použit jako vstup třetí kombinační buňky atd. To je nazýváno jako vícestupňový logicky kombinační obvod. Na obr. 1.5 je zobrazen jednoduchý vícestupňový logicky kombinační obvod, pro který může nastat situace, že jednotlivé vstupní hodnoty kombinačních buněk nemusí vždy přijít ve stejný časový okamžik. Tato situace je způsobena nenulovým šířením signálu na propojovacích vodičích při šeření signálu z výstupu první buňky na vstup druhé buňky. Určitou dobu trvá také přepnutí výstupního stavu buňky při změně vstupního stavu buňky. Různé časy



Obr. 1.5: Jednoduchý kombinační obvod u kterého může nastat hazardní stav.

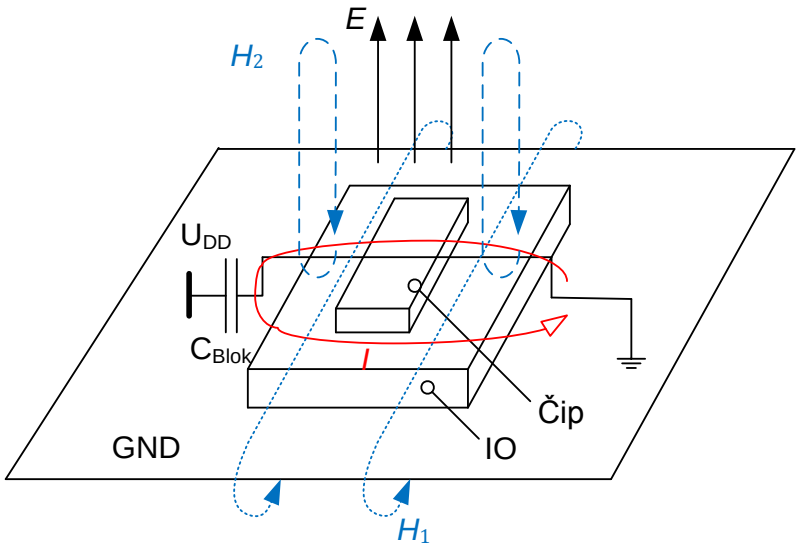
příchodu vstupních signálů způsobí dočasný výstupní stav buňky. Tento dočasná výstupní hodnota (výstupní stav) kombinační buňky je nazýván hazardním stavem kombinačního obvodu (glitches) a způsobuje změny ve výstupech logických buněk v celém obvodu [93, 104].

Počet takto vzniklých hazardních stavů v integrovaném obvodu roste s počtem stupňů kombinačního obvodu a vzniklé dočasné stavy se zde šíří jako lavina. Pokud například nastane dočasný stav buňky v prvním stupni kombinačního obvodu tak tato jednoduchá chyba vede k chybám na výstupech všech integrovaných obvodů, které jsou na tuto buňku napojeny. Ve složitějších CMOS obvodech se tyto chyby mohou stát dominantním prvkem proudové spotřeby. Z principu vzniku hazardních stavů plyne závislost na zpracovávaných datech a použití v proudové analýze [100].

1.3 Elektromagnetický postranní kanál

V předchozí kapitole jsou pomocí invertujícího členu logiky CMOS (obr. 1.3) popsány zdroje změn výkonové spotřeby. Tento popis je vhodný i pro vysvětlení příčin vzniku elektromagnetického postranního kanálu. Následkem nabíjení a vybíjení parazitní kapacity C, vzniká v obvodu skoková změna proudu, projevující se emitací elektromagnetického pole v blízkém okolí invertoru. Dnešní integrované obvody jsou složeny z milionů tranzistorů a spojů, ve kterých protékají proudy, které jsou závislé na přenášených datech. Tyto proudy generují proměnné elektromagnetické (EM) pole, které může být v okolí měřeno pomocí sond. Způsoby, jakými se projevuje EM záření emitované integrovanými obvody, jsou následující:

- Vodivá emise - se projevuje na pinech integrovaného obvodu, respektive v cestách na ně připojených, kdy se vlivem skokové změny proudu tyto cesty mohou chovat jako antény emitující rušení.
- Elektrická a magnetická emise v blízkém poli - EM pole je generováno vli-



Obr. 1.6: Princip přímé emise magnetického a elektrického pole IO.

vem proudových smyček v integrovaném obvodu. Magnetická složka pole lze rozdělit na dvě části $H1$ a $H2$, viz obr. 1.6. Pole $H1$ se uzavírá kolem zemnícího kontaktu tištěného spoje, pole $H2$ je generováno proudy ve vnitřních kondenzátorech a uzavírá se v oblasti nad povrchem IO v dosahu přibližně do 10 mm. Magnetické pole $H2$ je výrazně větší než pole $H1$.

Elektrické pole se nachází v okolí součástí pod napětím. V IO jsou zdrojem elektrického pole vnitřní vodivé spoje. Na obr. 1.6 je zobrazena emise elektrickým polem E způsobená hodinovým signálem. Většina toku se uzavírá do země, ale část toku je vyzářena do okolí.

Vyjdeme-li z předpokladu, že integrovaný obvod generuje elektromagnetické pole, pak je možné charakterizovat elektromagnetickou emisi obvodu, pomocí měření těchto polí. Tato měření se realizují pomocí elektrických a magnetických sond [89, 39, 66, 96, 82]. Měření pomocí rozměrově malé magnetické sondy slouží k zjištění velikosti magnetické složky blízkého EM pole. Výhodou těchto sond je, že mohou být umístěny co nejblíže ke zdroji záření a zvyšují tak přesnost měření. Pokud se sonda umístí do větší vzdálenosti, pak bývá zachycen u mikroprocesorů hodinový signál CLK. Důvodem je, že CLK signály jsou v uvedených zařízeních dominantní a jejich úroveň významně převyšuje úroveň ostatních signálů. Pokud sondu umístíme do blízkosti některého zařízení, je možné pozorovat emisi konkrétního části zařízení (např. CPU, sběrnice, paměti apod.). Užitečné EM signály, které jsou závislé na zpracovávaných datech lze zachytit v oblastech procesoru a pamětí kryptosystému [89, 84]. K zachování věrnosti měření by měla veškerá měření probíhat v blízké zóně,

tedy ve vzdálenosti do maximálně délky vlny od zdroje. V této zóně všechny signály mohou být považovány za kvazistatické. Proto lze definovat Biot-Savartův zákon popisující magnetickou indukci pole $\vec{B}$:

$$d\vec{B} = \frac{\mu I \, \vec{dl} \times \hat{r}}{4\pi |\vec{r}|^2}, \tag{1.7}$$

kde μ je permeabilita prostředí, I je proud, $\vec{dl}$ je vektor jehož rozměr určuje délku diferenciálního elementu a jeho směr určuje směr konvenčního proudu a $\vec{r}$ je vektor specifikující vzdálenost mezi zdrojem záření a bodem měření, pro $\hat{r}$ platí ($\hat{r} = \vec{r} / |\vec{r}|$). Pomocí Faradayova zákona lze vyjádřit hodnotu magnetomotorického napětí, které se bude v sondě indukovat:

$$U_{emf} = -N\frac{d\phi}{dt}, \tag{1.8}$$

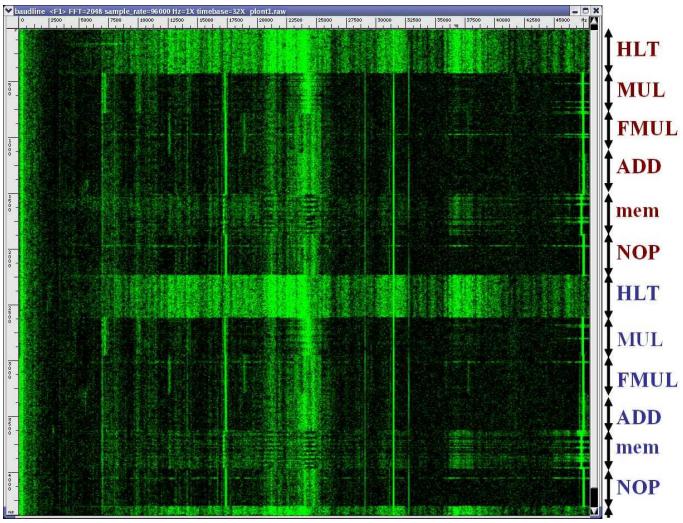
$$d\phi = \int_{surface} \vec{B} \cdot d\vec{S}, \tag{1.9}$$

kde U_{emf} je magnetomotorické napětí, N je počet závitů sondy (cívky) a $d\phi$ vyjadřuje změnu magnetického toku za dobu dt. Z výše uvedeného vyplývá, že bude potřeba volit kompromis v počtu závitů cívky měřicí sondy, jelikož velikost magnetomotorického napětí je přímo úměrná počtu závitů cívky. Na druhou stranu z teorie snímání EM pole vyplývá požadavek na co nejkratší měřicí cívku.

Elektromagnetický postranní kanál byl ve své historii nejdříve využíván v armádě a tajných službách. Tyto organizace se odborně zabývaly studiem problematiky parazitních emisí, která označovaly termínem TEMPEST [3]. Hlavním zájmem vojenských organizací bylo zamezení nežádoucích emisí a naopak využití tohoto vyzařování k špionážní činnosti. Pojem TEMPEST vznikl na přelomu 60. a 70.let dvacátého století a označuje i skupinu vojenských standardů, ve kterých jsou stanoveny maximální povolené limity elektromagnetického záření v různých elektronických systémech.

Ve veřejném sektoru se o významný posuv na poli elektromagnetických útoků zasloužil nizozemský vědec van Eck [33], který jako první dokázal, že je možné zachytit a změřit velikost elektromagnetického pole počítačových monitorů a z naměřených průběhů extrahovat snímaný obraz. Obranu proti tomuto útoku vynalezli vědci Kuhn a Anderson [58], jednalo se o speciální stínící fólii, která snižovala elektromagnetické záření monitoru. První veřejně publikovanou prací na téma EM analýzy

[3]Termín TEMPEST je krycím jménem vytvořeno NSA v 60. letech pro operace snažící se zabezpečit elektronická komunikující zařízení proti odposlechu, vláda USA uvedla že tato zkratka nemá žádný význam i když několik bylo navrhováno př. Transmitted Electro-Magnetic Pulse / Energy Standards & Testing.



Obr. 1.7: Spektra různých operací mikroprocesoru [99].

integrovaných obvodů a výpočetních jednotek provádějících kryptografické operace, byla v roce 2001 práce Electomagnetic Analysis: Concrete Results autorů Gandolfiho, Mourtela, Oliviera [38]. Útok prováděli pomocí několika antén umístěných v blízkosti výpočetních integrovaných obvodů čipové karty. Tento útok byl invazivní, což znamená, že vyžadoval porušení pouzdra čipové karty, tak aby bylo možné umístit antény co nejblíže pasivační vrstvě. Na tuto práci následně navázali o rok později Agrawal, Archambeault, Rao a Rohatgi [4], kteří ve své práci The EM-Side-Channels: Attacks and Assessment Methodologies využili odtajněných materiálů z projektu TEMPEST a ukázali, že útoky EM postranním kanálem na kryptografická zařízení jsou prakticky realizovatelné a zároveň, že některé informace unikající prostřednictvím EM kanál, bylo možné získat z proudového postranního kanálu pouze velmi obtížně. Dále jsou v práci uvedeny nové možnosti útoků využívajících nepřímých EM záření vznikající vazbami mezi jednotlivými částmi kryptografického systému.

1.4 Akustický postranní kanál

Akustický postranní kanál patří k nejstarším postranním kanálům. Byl používán již v době, kdy se definice postranních kanálů ještě nepoužívala, např. v roce 1956 Britové získávají informace z egyptského šifrátoru odposlechem zvuků klávesnice a v roce 1961 Američané provádějí akustický odposlech prostřednictvím ústředního topení. Další akustické útoky byly prováděny na klávesy psacího stroje a jehličkové tiskárny. Nyní se tento postranní kanál zaměřuje na zvuky vydávané počítačovými

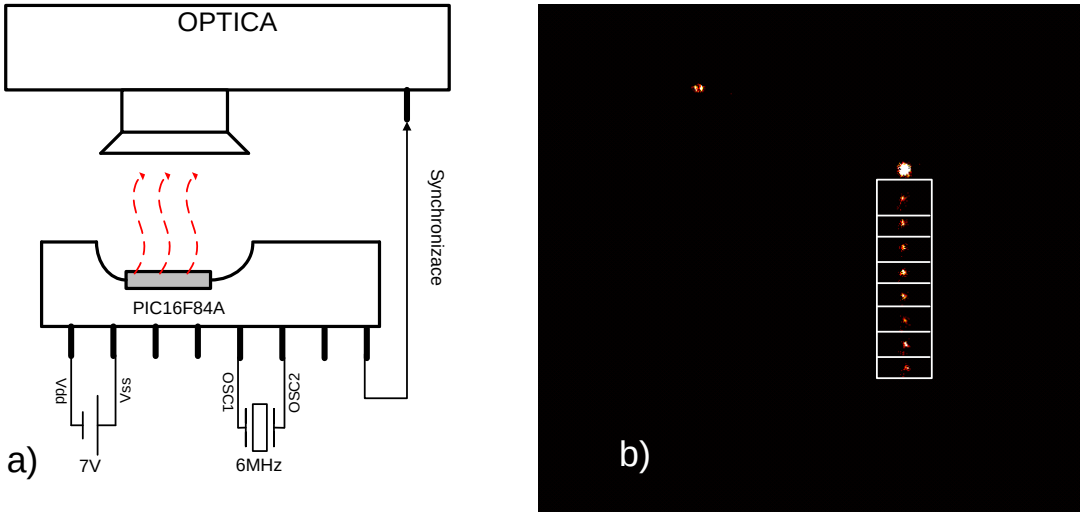
klávesnicemi [113]. V jednotlivých útocích na klávesnice byly použity různé metody pro převod zvukového signálu klávesnice zpět na text. Např. v práci „Dictionary Attacks Using Keyboard Acoustic Emanations“ se pro zpětnou analýzu využíval anglický slovník a neuronová síť [113]. V práci [36] byly počítány rozdíly v době příchodu zvukové vlny na dva mikrofony a ke klasifikaci byly použity neuronové sítě. Vypočítané spektrogramy pro jednotlivé tlačítka byly použity v pracích [119] a [64], kde pro klasifikaci byla použita také neuronová síť. Zajímavostí je, že žádná z metod není schopna rekonstruovat stisk více kláves najednou.

Akustický postranní kanál se dá využít také na vnitřní komponenty PC (mikroprocesor, koprocesor . . .) [99]. Obr. 1.7 zachycuje výsledky experimentu, který se snaží rozlišit spektra různých operací mikroprocesoru.

1.5 Optický postranní kanál

Optický postranní kanál byl poprvé představen v práci [57], kde na základě sledování rozptýleného odrazu světla na překážce, jehož zdrojem je CRT zobrazovací jednotka, byl rekonstruován původní obraz. V práci je poukázáno na možnost použití stejné techniky i pro rekonstrukci obrazu z LED zobrazovacích jednotek. Optický postranní kanál v moderním pojetí byl poprvé představen v roce 2008 dvojicí autorů Ferrignem a českým spoluautorem Hlaváčem [35]. Základní myšlenka optického postranního kanálu je přímo geniálně prostá a účinná, a proto se předpokládá, že bude mít určitě ještě v kryptoanalýze pokračování.

Integrované obvody se skládají z tranzistorů, jejichž stavy reprezentují hodnotu 0 nebo 1. Pokaždé, když tranzistor v integrovaném obvodu změní svůj stav, vyzáří několik fotonů. Jedna z možných technik, která je schopna detekovat polohu a čas vyzářených fotonů se nazývá pikosekundová zobrazovací obvodová analýza (picosecond imaging circuit analysis, PICA [106]). Máme-li důkladně provedenou synchronizaci s algoritmem, který je prováděn na pozorovaném zařízení (mikrokontroléru), je PICA schopna identifikovat přepínající se transistory v paměti. Zařízení OPTICA, využívající výše popsanou měřící techniku, měli autoři k dispozici v laboratoři CNES. Blokové schéma měřicího zařízení představuje obr.1.8 add a). Pro sledování informací prosakujících prostřednictvím optického postranního kanálu byla použita jednoduchá implementace části zdrojového kódu algoritmu AES (část `AddRoundKey`), běžící na PIC16F84A mikrokontroléru. Tyto mikrokontroléry nemají žádné protiopatření proti prosakování informací prostřednictvím postranních kanálů a taktovací frekvence není nijak modifikována, což je nutné k snadné synchronizaci s měřicím zařízením OPTICA. Ve fázi `AddRoundKey` jsou bajty otevřeného textu xorovány s bajty prvního rundovního klíče. Obsah paměťové buňky (tranzistoru) reprezen-



Obr. 1.8: a)Zařízení OPTICA b)Příklad naměřených dat pomocí PICA [35].

tující bit otevřeného textu se změní, pokud příslušný bit klíče je 1. Pokud je bit klíče 0, operace `XOR` (Exclusive OR) ponechá bit otevřeného textu nezměněn. Když tranzistory pozorujete speciálním zařízením v době, kdy dochází k této operaci tranzistory na nás zablikají nebo nezablikají v závislosti na tom zda byl bit klíče 1 nebo 0. Prostřednictvím optického postranního kanálu tedy tranzistory tajný klíč AES přímo útočníkovi „vyblikají“ (obr.1.8 add b) bajt tajného klíče 11111111). Opakované spouštění kódu mikrokontroléru se stejnými vstupními daty vedlo k identickým (optickým) výsledkům získaných pomocí optického postranního kanálu. Výsledky experimentu ukázaly, že oba přechody tranzistorů jak přechod z $1 \rightarrow 0$ tak přechod $0 \rightarrow 1$ jsou detekovatelné. Můžeme také předpokládat, že výsledky budou podobné pro další instrukce (`addwf, andwf, iorwf, subwf`). Tato možnost rozšíří eventuální použití optického postranního kanálu k útoku na další šifrovací algoritmy. Nevýhodou postranního kanálu je, že zařízení určené pro snímání fotonů (př. OPTICA) není tak snadno dostupné, jako zařízení pro získávání informací z jiných postranních kanálů. Cena tohoto zařízení se pohybuje v řádu desítek milionů českých korun. K realizaci útoku je nutná přesná synchronizace. Protiopatření implementovaná v nových čipech např. náhodné provádění instrukcí znesnadní realizaci útoku. Dalším problémem realizace je, že za účelem pozorování vyzářeného světla z přístroje potřebujeme zeslabit vrstvu silikonu na zadní straně čipu minimálně na 100 μm [35]. Po zeslabení je nutné vrstvu vyčistit speciálním leštícím systémem. Ztenčení a leštění vrstev silikonu dovolí IR světlu proniknout zadní stranou čipu, a tak může být provedena detekce záření. Výše popsané aspekty činní optický kanál atraktivní pro případné útočníky a vědce, ale také realizovatelný zatím jen v precizních laboratorních podmínkách.

1.6 Chybový postranní kanál

Vznik výše popsaných postranních kanálů je založen na určité fyzikální vlastnosti kryptografického modulu, ale existuje i postranní kanál, který využívá jiný typ komunikace. Podstatou chybového postranního kanálu [30, 54] je existence komunikace kryptografického modulu s okolím při vzniku chyby. Několik let se všichni kryptologové plně věnovali bezpečnosti kryptografických algoritmů a považovali chybová hlášení jako vedlejší činnost systému, která nemůže mít vliv na bezpečnost a pro případného útočníka nemá význam. Přesto se chybová hlášení stávají dalším postranním kanálem a umožňují napadení kryptografických modulů.

Typickým příkladem útoku založeném na zdánlivě neškodném chybovém hlášení je útok na implementace algoritmů v módu CBC (Cipher Block Chaining). U tohoto módu je třeba vyřešit šifrování posledního bloku dat. Možné řešení popisuje například norma PKCS#5 [49], kde je použito techniky doplnění (padding) na předepsanou délku bloku. Pokud nastane situace, kdy doplněk neodpovídá pravidlům doplnění, systém vyhodnotí chybu přenosu a o jejím výskytu informuje odesílatele. Právě toto, na první pohled korektní a neškodné chybové hlášení, vytváří postranní kanál, jehož důsledné využití vede až k získání celého otevřeného textu.

Nejčastěji se lze setkat s využitím útoku chybovým kanálem u implementace asymetrických algoritmů, mezi které patří zejména algoritmy RSA, DSA [75, 16]. Často zmiňovaným příkladem chybového útoku je útok na schéma digitálního podpisu RSA. Podstata útoku spočívá v napadení operace dešifrování, která využívá k výpočtu čínskou větu o zbytcích. S další možností využití chybového postranního kanálu přišel v roce 1998 švédský kryptolog Daniel Bleichenbacher [17]. Princip tohoto útoku spočívá v úpravě formátu dané šifrované zprávy do takové podoby, kdy dešifrovací zařízení reaguje na tento stav chybovým hlášením, ze kterého je útočník schopen určit žádané informace. Pro integritní kontrolu platnosti zformátované zprávy se využívá speciálního kódování, jehož činnost umožňuje útočníkovi získat konkrétní části původního nešifrovaného textu. Z důvodu možnosti realizace Bleichenbacherova útoku byla navržena nová dokonalejší formátovací metoda OAEP, která do celé zprávy určitým způsobem vnáší náhodné prvky a tím zabraňuje realizaci útoku švédského kryptologa. Na možné chyby formátovací metody OAEP upozornil v roce 2001 kryptolog Manger. Manger dokázal existenci postranního kanálu využívající nedokonalosti implementace metody OAEP [68].

2 CÍLE DISERTACE

Disertační práce bude zaměřena na jednoduchou a diferenciální analýzu proudovým postranním kanálem. Hlavním cílem disertační práce bude návrh nové metody proudové analýzy, která umožní určit hodnotu šifrovacího klíče jen z jednoho proudového průběhu a to u algoritmů, které jsou odolné vůči jednoduché proudové analýze. Do jisté míry bude navržená metoda spojovat vlastnosti jednoduché a diferenciální proudové analýzy. Většina proudových analýz využívá různé statistické metody (rozdíl středních hodnot, korelační koeficient atd. viz kapitola 3) k určení závislosti proudové spotřeby a hledané senzitivní informace. Navrhovaná metoda bude založena na odlišném způsobu a plánuje využít neuronové sítě ke klasifikaci senzitivní informace z naměřených průběhů proudové spotřeby. Tento typ útoku proudovým postranním kanálem, který bude zaměřen na klasifikaci konkrétní hodnoty šifrovacího klíče, nebyl dosud publikován. Navržená metoda bude pracovat odlišným způsobem než klasické metody, a proto se předpokládají odlišné vlastnosti např. potřebný počet naměřených proudových průběhů, úspěšnost určení šifrovacího klíče atd. Analýza těchto parametrů je dalším cílem disertační práce. K ověření funkce nové metody je nutné navrhnout a ověřit metodu pro snímání a záznam proudového odběru kryptografického modulu. Způsob a přesnost měření proudového odběru má zásadní vliv na správnou funkci navrhované metody kryptoanalýzy. Z tohoto důvodu je nutné sestavit experimentální pracoviště včetně kryptografického modulu s implementovaným kryptografickým algoritmem, a navrhovanou metodu analýzy implementovat a prakticky ověřit její funkčnost. Proudové analýzy jsou testovány většinou na implementaci algoritmu AES, proto bude navržená metoda proudové analýzy využívající neuronové sítě také testována pomocí implementace kryptografického algoritmu AES (viz příloha A.1). Posledním důležitým dílčím cílem práce je analýza možných ochran proti proudové analýze a útoku postranním kanálem. Z analyzovaných ochran vybrat vhodné řešení zabraňující použití navržené metody.

Cíle disertační práce mohou být shrnuty do následujících bodů:

- analýza současného stavu problematiky (proudová analýza, protiopatření proti proudové analýze, neuronové sítě),
- návrh a realizace metody pro měření proudového odběru, způsob snímání proudu má zásadní vliv na úspěšnost navrhované metody proudové analýzy,
- návrh a implementace nové metody proudové analýzy využívající neuronové sítě,
- analýza a zhodnocení dosažených výsledků klasifikace.

3 SOUČASNÝ STAV PROBLEMATIKY

Cílem kapitoly je detailně popsat jednoduchou a diferenciální proudovou analýzu používanou při útoku na kryptografické zařízení. Na podobných principech jsou založeny všechny jednoduché a diferenciální analýzy postranním kanálem. Další část kapitoly popisuje techniky protiopatření vedoucí k znesnadnění a nebo plnému znemožnění útoku proudovým postranním kanálem. Závěrečná část kapitoly rozebírá použití neuronových sítí v kryptografii a v oblasti postranních kanálů.

3.1 Jednoduchá proudová analýza SPA

V této kapitole bude vysvětlen princip jednoduché proudové analýzy a následně budou popsány různé typy SPA. Jednoduchá proudová analýza byla definována Kocherem [55] následovně: jednoduchá proudová analýza je technika, která představuje přímé interpretování proudové spotřeby měřené během provozu kryptografického zařízení. Jinými slovy se útočník snaží určit tajný klíč přímo ze změřených průběhů proudové spotřeby. To činí SPA pro potencionální útočníky atraktivní technikou, ale ti většinou potřebují detailní znalost implementovaného algoritmu a kryptografického zařízení. SPA můžeme rozdělit do dvou základních skupin a to na analýzu jen jednoho proudového průběhu (single-shot SPA) a analýzu několika proudových průběhů (multiple-shot). Analýza jednoho proudového průběhu představuje extrémní případ, kdy útočník zaznamenal a zkoumá jen jeden průběh proudové spotřeby odpovídající jednomu vstupnímu textu. Ve většině případů je nutné použít statistických metod k extrakci užitečného signálu. Při jednoduché analýze několika proudových průběhů má útočník k dispozici více naměřených proudových průběhů, a to pro stejný nebo různý vstupní text, které použije k redukci šumu v naměřených datech. Pro oba typy útoku SPA je nutné, aby v kryptografickém zařízení, na které je prováděn útok, existovala výrazná (přímá nebo nepřímá) závislost proudové spotřeby na hodnotě šifrovacího klíče.

3.1.1 Přímé interpretování proudové spotřeby

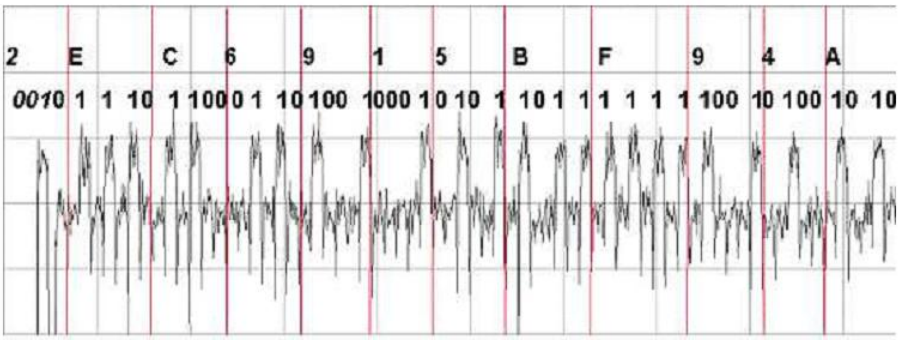
Přímé interpetování proudové spotřeby (Visual Inspections) je založeno na následujících skutečnostech. Všechny algoritmy, které běží na kryptografickém zařízení jsou vykonávány postupně v přesně definovaném pořadí. Kryptografický algoritmus se ve většině případů skládá z několika dílčích funkcí (operací), které jsou následně po implementaci přeloženy do podporovaných instrukcí zařízení. Například algoritmus AES může být popsán funkcemi podrobně rozebranými v kapitole A.1 (expanze klíče, přičtení klíče, nelineární bajtová substituce, rotace řádků a násobení

maticí). Algoritmus AES může být implementován do mikroprocesoru, kde jsou jednotlivé funkce implementovány pomocí instrukcí podporovaných mikroprocesorem. Mikroprocesory mají instrukční sadu, která obsahuje aritmetické instrukce (sčítání), logické instrukce (`XOR`), instrukce práce s daty (uložení, přesun) a instrukce větvení programu (skok nebo podmínka). Každá instrukce pracuje s různým počtem bitů popřípadě bajtů a používá k tomu různé části obvodu, například aritmeticko-logickou jednotku, čtení popřípadě zápis do interní nebo externí paměti, vstupní a výstupní porty. Tyto komponenty mikroprocesoru jsou fyzicky odděleny a liší se funkčností a realizací. Z tohoto důvodu mají charakteristickou proudovou spotřebu, která vede k charakteristickému proudovému vzoru (otisku) v proudové spotřebě. Možnost rozlišit jednotlivé instrukce v proudové spotřebě vede k vážné bezpečnostní hrozbě pokud posloupnost instrukcí závisí přímo na hodnotě šifrovacího klíče.

Pokud v průběhu zpracování algoritmu je určitá instrukce zpracovávána v případě, že bit klíče je 1 a odlišná instrukce je zpracovávána pokud bit klíče je 0, pak je možné určit šifrovací klíč přímo analýzou naměřeného průběhu na základě posloupnosti vykonávaných instrukcí. Typickým příkladem tohoto typu SPA je implementace asymetrického kryptosystému RSA [51, 47, 37]. Asymetrický algoritmus RSA je založen na matematické operaci modulární mocniny. Pro výpočet modulární mocniny existuje více metod, ale pro implementaci v kryptografických modulech se často používá tzv. algoritmus *square and multiply*, který je založen na postupném zpracování jednotlivých bitů soukromého klíče k. Princip algoritmu je zobrazen v následujícím textu, kde b představuje počet bitů soukromého klíče k, n modul algoritmu RSA a $k(i) = k_0 \dots k_{b-1}$ představuje binární reprezentaci soukromého klíče.

```
%Algoritmus square and multiply
1.R = m
2.for i = 0:(b - 1) {
3.        R = (R*R)mod(n)
4.        if (k(i)==1) {
5.            R=(R*m)mod(n)  }
6. }
7. return R
```

Z naměřené proudové spotřeby prováděného algoritmu *square and multiply* může útočník určit sekvenci prováděných instrukcí, tedy v kterém kroku byl prováděn řádek 5 v závislosti na hodnotě soukromého klíče. Přímá interpretace proudové spotřeby algoritmu *square and multiply* na hodnotu soukromého klíče je zobrazeno na obr. 3.1.



Obr. 3.1: Proudová spotřeba algoritmu square and multiply [47].

3.1.2 Útoky pomocí šablon

Útoky pomocí šablon (Template Based Attacks) využívají toho, že průběh proudové spotřeby je závislí na právě zpracovávaných datech. Proudové průběhy jsou charakterizovány vícerozměrným normálním rozdělením, které je plně definováno vektorem středních hodnot a kovarianční maticí $(\mathbf{m}, \mathbf{C})$. Tuto dvojici $(\mathbf{m}, \mathbf{C})$ označíme *šablonou*. Útočník předpokládá, že může zařízení charakterizovat pomocí těchto šablon pro jednotlivé instrukce (posloupnost instrukcí). Například útočník má k dispozici a plně pod kontrolou stejný typ zařízení, na které hodlá provádět útok. Na tomto zařízení spustí sekvence instrukcí pro různá vstupní data d_i a různý šifrovací klíč k_j a zaznamená si proudové průběhy. Následně seskupí odpovídající průběhy (d_i, k_j) a vypočte vektor středních hodnot a kovarianční matici vícerozměrného normálního rozdělení. Výsledkem získá vzory (šablony) pro všechny dvojce dat a klíčů $(d_i, k_j) : h_{d_i,k_j} = (\mathbf{m}, \mathbf{C})_{d_i,k_j}$. Později útočník použije šablony a naměřenou proudovou spotřebu k určení šifrovacího klíče zařízení, na které útočí. Vypočte funkci hustoty pravděpodobnosti vícerozměrného normálního rozdělení $(\mathbf{m}, \mathbf{C})_{d_i,k_j}$ a proudového průběhu. Jinými slovy vypočtou se pravděpodobnosti pro změřenou proudovou spotřebu pro všechny šablony dle následujícího vztahu:

$$p(t; (\mathbf{m}, \mathbf{C})_{d_i,k_j}) = \frac{exp(-\frac{1}{2} \cdot (\mathbf{t} - \mathbf{m})' \cdot \mathbf{C}^{-1} \cdot (\mathbf{t} - \mathbf{m}))}{\sqrt{(2 \cdot \pi)^T \cdot det(\mathbf{C})}}. \tag{3.1}$$

Pravděpodobnosti ukazují jak dobře šablona odpovídá změřenému průběhu. Intuitivně by nejvýší pravděpodobnost měla odpovídat správné šabloně, protože každá šablona je asociována s tajným klíčem, určí tímto způsobem útočník hodnotu tajného klíče. Tato metoda vychází z teorie popsané v [50].

3.1.3 Jednoduchá proudová analýza - shrnutí

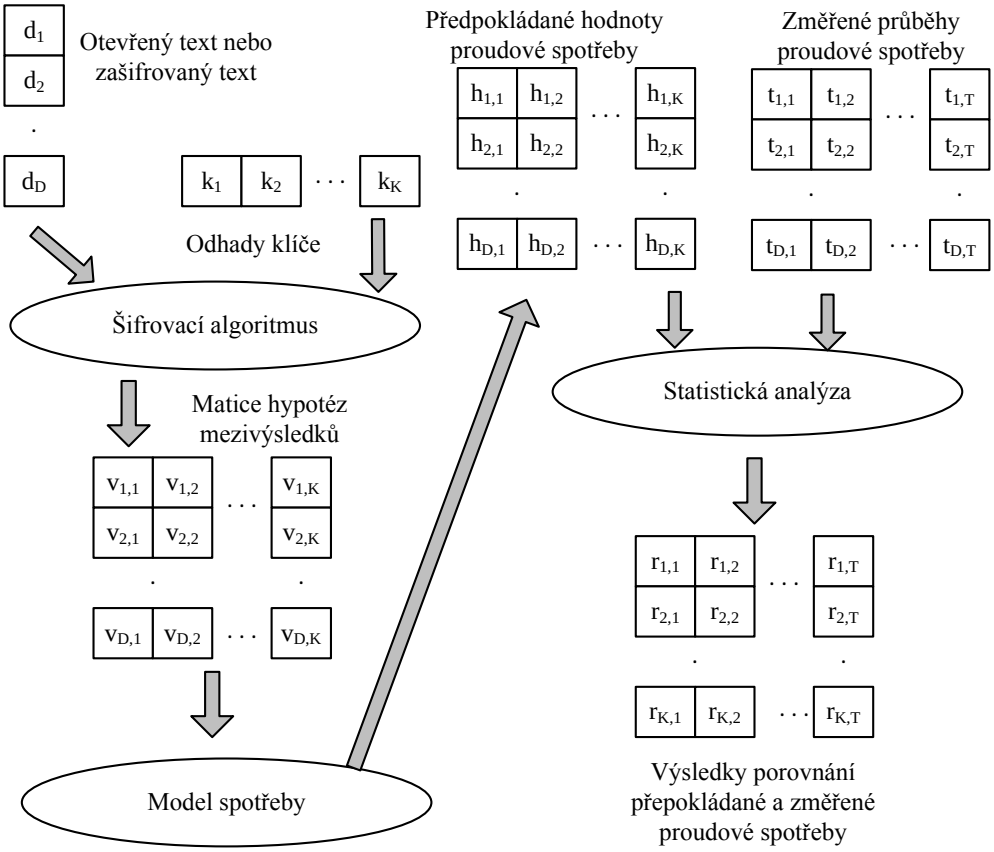
Kocher [55] byl první, který představil proudovou analýzu kryptografických algoritmů a také poukázal na závislost proudové spotřeby na prováděných instrukcích. Jako příklad byla uvedena analýza algoritmu RSA a DES. V práci [74] byla diskutována SPA na algoritmus DES, který byl implementován na 8 bitový mikrokontrolér. Útok byl zaměřen na určení Hammingovy váhy šifrovacího klíče, který redukoval prostor potřebný na útok hrubou silou. Podobný útok SPA založený na určení Hammingovy váhy z proudového průběhu instrukce `MOV` je uveden v práci [71]. V práci [15] byl nastíněn útok na implementaci algoritmu AES, který využíval analýzu proudového průběhu, který byl naměřen během vykonávání operací substituce při přístupu do vyrovnávací paměti. Mangard v práci [65] diskutoval SPA analýzu algoritmu AES, která je zaměřena na operaci expanze klíče. Útoky využívající šablony byly definovány a představeny poprvé v práci [22]. Praktické aspekty těchto útoků byly rozebrány v [95, 5, 43, 86].

3.2 Diferenciální proudová analýza DPA

Cílem útoků DPA je získat šifrovací klíč kryptografického zařízení na základě znalosti velkého počtu proudových spotřeb, které byly zaznamenány útočníkem během provádění operace šifrování nebo dešifrování pro různá vstupní data. Hlavní výhodou diferenciální proudové analýzy ve srovnání s SPA je, že útočník nepotřebuje detailní znalost kryptografického zařízení a šifrovacího algoritmu. Dalším důležitým rozdílem mezi těmito analýzami způsob zpracování naměřených dat. V SPA jsou proudové průběhy zpracovávány většinou v časové ose. Útočník se zde pokouší v jednom proudovém průběhu najít vzor, známý otisk instrukce nebo šablonu. Naproti tomu, tvar proudového průběhu v časové oblasti není v DPA důležitý. DPA analyzuje závislost proudové spotřeby v určitý konstantní časový okamžik na právě zpracovávaných datech. V následujícím textu bude popsán detailněji postup získání šifrovacího klíče metodou DPA. Všechny DPA útoky využívají prakticky stejného postupu, který se skládá z pěti kroků. Toto obecné schéma je názorně zobrazeno na obr. 3.2.

První krok: Volba mezivýsledku algoritmu

Prvním krokem DPA je volba mezivýsledku kryptografického algoritmu, který je vykonáván zařízením. Mezivýsledek musí být funkcí $f(d, k)$, kde d jsou známá nekonstantní data a k je malá část šifrovacího klíče (např. první bajt). Ve většině případů DPA útoku d představuje otevřený text nebo šifrovaný text. Takto definovaný mezivýsledek může být použit k určení části šifrovacího klíče k.



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

Druhý krok: Měření proudové spotřeby

Druhým krokem DPA útoku je měření proudové spotřeby kryptografického zařízení 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 prvním kroku. Hodnoty známých dat tvoří vektor $\mathbf{d} = (d_1, \ldots, d_D)'$, kde d_i označuje hodnotu i-tého kroku šifrování nebo dešifrování. V průběhu každého tohoto kroku si útočník zaznamenává proudovou spotřebu. Průběhy proudové spotřeby, korespondující s bloky dat d_i, označíme $t_i' = (t_{i,1}, \ldots, T_{i,T})$, kde T označuje délku naměřené proudové spotřeby. Útočník měří proudovou spotřebu pro každý blok dat D, a proto naměřené průběhy mohou být zapsány do matice $\mathbf{T}$ o velikosti $D \times T$. Pro DPA útok je klíčové, aby naměřené proudové průběhy byly přesně zarovnané. To znamená, že hodnoty pro jednotlivé sloupce t_j matice $\mathbf{T}$ musí odpovídat stejné operaci. K získání takto zarovnaných dat je nutná správná synchronizace s měřicím zařízením, nebo je zapotřebí zarovnat data softwarově pomocí nalezení několika markantů (otisků v proudovém průběhu).

Třetí krok: Výpočet hypotetických mezivýsledků

Dalším krokem útoku je výpočet hypotetických mezivýsledků pro všechny možné hodnoty šifrovacího klíče k. Všechny možnosti klíče lze zapsat jako vektor $\mathbf{k} = (k_1, \ldots, k_K)$, kde K označuje celkový počet možných klíčů. V DPA jsou jednotlivé prvky vektoru $\mathbf{k}$ označovány za hypotézy klíče nebo odhady klíče. Z vektoru známých dat $\mathbf{d}$ a vektoru hypotéz všech klíčů může útočník jednoduše vypočítat hodnotu mezivýsledku $f(d, k)$ pro všechny šifrovací operace D a pro všechny hypotézy klíče K. Výsledkem výpočtu je matice $\mathbf{V}$ o rozměrech $D \times K$ vypočtená dle následujícího vztahu:

$$v_{i,j} = f(d_i, k_j)\ i = 1, \ldots, D\ j = 1, \ldots, K \tag{3.2}$$

Sloupec j matice $\mathbf{V}$ obsahuje mezivýsledky, které byly vypočítány dle hypotéz klíče k_j. Je zřejmé, že jeden sloupec matice $\mathbf{V}$ obsahuje takové mezivýsledky, které byly vypočítány v zařízení během šifrování a dešifrování. Jinými slovy, jednotlivé sloupce matice $\mathbf{V}$ obsahují mezivýsledky pro všechny klíče, tedy i pro klíč který byl použit v zařízení. Tento index bude označen ck, tedy k_{ck} označuje hledaný tajný klíč. Cílem DPA je nalezení odpovídajícího sloupce, který byl zpracováván při operacích šifrování a dešifrování v zařízení a tedy nalezení k_{ck}.

Čtvrtý krok: Mapování hypotetických mezivýsledků na hodnoty proudové spotřeby

Čtvrtým krokem DPA útoku je namapování matice hypotetických mezivýsledků $\mathbf{V}$ na matici $\mathbf{H}$ reprezentující předpokládané hodnoty proudové spotřeby. V tomto bodě se využívá simulace proudové spotřeby kryptografického zařízení. Použitý model spotřeby přiřadí každému hypotetickému mezivýsledku $v_{i,j}$ předpokládanou hodnotu proudové spotřeby $h_{i,j}$. Správnost výsledků simulace silně závisí na útočníkových znalostech o zařízení a činní DPA efektivnější. Mezi často používané modely přiřazení hodnot $\mathbf{V}$ na $\mathbf{H}$ patří model Hammingovy vzdálenosti a Hammingovy váhy (viz kapitola 3.2.5).

Pátý krok: Porovnání hypotetických hodnot s naměřenými průběhy proudové spotřeby

V posledním kroku útočník porovná předpokládané hodnoty proudové spotřeby závislé na odhadu klíče (hodnoty ve sloupci h_i matice $\mathbf{H}$) se změřenými průběhy proudové spotřeby (hodnoty ve sloupci t_j matice $\mathbf{T}$). Výsledkem je matice $\mathbf{R}$ velikosti $K \times T$, kde každý element $r_{i,j}$ představuje výsledek porovnání sloupců h_i a t_j. Samotné porovnání je realizováno různými metodami, které jsou detailněji popsány

v následujícím textu (kapitoly 3.2.1, 3.2.2 a 3.2.3). Společná vlastnost všech postupů je, že hodnota $r_{i,j}$ je větší pro lepší shodu sloupců h_i a t_j. Určení tajného klíče využívá následujících skutečností.

- Proudové průběhy odpovídají proudové spotřebě zařízení během provádění algoritmu šifrování nebo dešifrování pro různá vstupní data.
- Mezivýsledek, který byl vybrán v prvním kroku, je částí tohoto algoritmu.

Z těchto důvodu počítá zařízení hodnotu mezivýsledku v_{ck} v průběhu šifrování nebo dešifrování pro různá vstupní data. V důsledku toho jsou naměřené průběhy v určitých časových okamžicích závislé na hodnotě mezivýsledku. Označíme toto místo naměřených průběhů jako ct (to znamená, že sloupec t_{ct} obsahuje hodnoty proudové spotřeby, které závisí na hodnotě mezivýsledku v_{ck}). Hypotetické hodnoty proudové spotřeby h_{ck} byly simulovány útočníkem na základě hodnot v_{ck}. Proto jsou sloupce h_{ck} a t_{ct} na sobě silně závislé. Ve skutečnosti tyto dva sloupce vedou k největší hodnotě v $\mathbf{R}$, to znamená, že největší hodnota matice $\mathbf{R}$ je hodnota $r_{ck,ct}$. Další prvky matice $\mathbf{R}$ mají malou hodnotu, protože ostatní sloupce $\mathbf{H}$ a $\mathbf{T}$ nejsou na sobě silně závislé. Útočník tak může určit index pro správný klíč ck a časový okamžik ct jednoduše a to nalezením největší hodnoty v matici $\mathbf{R}$. Příslušné indexy této hodnoty jsou pak výsledkem DPA útoku.

V praxi se může stát, že všechny hodnoty z $\mathbf{R}$ si budou prakticky rovny. V tomto případě útočník nezměřil dostatečné množství proudových průběhů k ustanovení vztahu mezi řádky $\mathbf{H}$ a $\mathbf{T}$. Čím více průběhů útočník změří, tím více elementů budou mít sloupce $\mathbf{H}$ a $\mathbf{T}$ a tím lépe může útočník určit vztah mezi sloupci.

3.2.1 Útok založený na korelačním koeficientu

Korelační koeficient (Correlation coefficient) patří k nejznámější metodě k určení lineární závislosti mezi dvěma náhodnými proměnnými. Proto je to také vhodná metoda pro provedení DPA útoku. Existuje velmi dobře definovaná teorie pro korelační koeficient, který může být použit k modelování statických vlastností DPA útoků. Korelační koeficient je definován pro veličiny X a Y pomocí kovariance vztahem:

$$\rho(X,Y) = \frac{Cov(X,Y)}{\sqrt{\sigma^2(X) \cdot \sigma^2(Y)}}, \tag{3.3}$$

kde $Cov(X,Y)$ označuje kovarianci tedy střední hodnotu součinu odchylek obou náhodných veličin X, Y od jejich středních hodnot a $\sigma^2(X)$ a $\sigma^2(Y)$ označují rozptyl těchto veličin. Veličina ρ je bezrozměrná a může nabývat hodnot $-1 \leq \rho \leq 1$. Hodnota -1 korelačního koeficientu značí nepřímou závislost (změna v jedné skupině je provázena opačnou změnou ve skupině druhé). Hodnota 0 korelačního koeficientu značí, že mezi hodnotami obou skupin neexistuje žádná statisticky zjisti-

telná lineární závislost. Při nulovém korelačním koeficientu na sobě veličiny mohou záviset, ale tento vztah nelze vyjádřit lineární funkcí. Jestliže korelační koeficient je roven 1, značí to přímou závislost, dokonalou korelaci mezi hodnotami obou skupin. Výpočet ρ se liší podle typu zkoumaných statistických proměnných. V případě, že náhodné veličiny X a Y jsou kvantitativní náhodné veličiny se společným dvourozměrným normálním rozdělením, je pro konkrétní hodnoty (x_1, y_1), (x_2, y_2), ..., (x_n, y_n) výběrový korelační koeficient dán vztahem:

$$r = \frac{\sum_{i=1}^{n}(x_i - \overline{x}) \cdot (y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \cdot \sum_{i=1}^{n}(y_i - \overline{y})^2}}. \tag{3.4}$$

V DPA je korelační koeficient použit k určení lineární závislosti mezi sloupci h_i a t_j pro $i = 1, \ldots, K$ a $j = 1, \ldots, T$. Výsledkem je matice $\mathbf{R}$ obsahující korelační koeficienty. Označíme každou hodnotu jako $r_{i,j}$ na základě elementů D ze sloupců h_i a t_j. Použijeme-li předchozí definici korelačního koeficientu, můžeme vztah 3.4 vyjádřit:

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i}) \cdot (t_{d,j} - \overline{t_j})}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i})^2 \cdot \sum_{d=1}^{D}(t_{d,j} - \overline{t_j})^2}}, \tag{3.5}$$

kde $\overline{h_i}$ a $\overline{t_j}$ označují průměrné hodnoty sloupců h_i a t_j.

3.2.2 Útok založený na rozdílu středních hodnot

Základem statistické metody založené na rozdílu středních hodnot (Difference of mean) je srovnání dvou skupin naměřených hodnot (distribucí) výpočtem rozdílu středních hodnot těchto skupin. Systematický popis metody je uveden v práci [105] a graficky je postup metoda zobrazen na obr. 4.18. Tato metoda používá jiný způsob k určení závislosti mezi sloupci matice $\mathbf{H}$ a $\mathbf{T}$. Útočník vytvoří binární matici $\mathbf{H}$, která rozdělí naměřené proudové průběhy do dvou skupin. Posloupnost nul a jedniček v každém sloupci $\mathbf{H}$ je funkcí vstupních dat d a odhadů klíče k_i. Za účelem ověření zda odhad klíče k_i je správný útočník může rozdělit matici $\mathbf{T}$ na dva soubory řádků (tzn. dvě sady proudových spotřeb podle h_i). První soubor obsahuje ty řádky matice $\mathbf{T}$, kde index odpovídá pozici nul ve vektoru h_i. Druhý soubor obsahuje zbylé řádky $\mathbf{T}$. Následně útočník vypočítá průměr řádků. Vektor m'_{0i} značí průměr řádků prvního souboru a m'_{1i} označuje průměr druhého souboru. Odhad klíče k_i je správný pokud existuje výrazný rozdíl mezi m'_{0i} a m'_{1i}. Rozdíl mezi m'_{0i} a m'_{1i} indikuje vztah mezi h_{ck} a některým ze sloupců $\mathbf{T}$. Stejně tak jako v předchozím případě tato diference označuje časový okamžik kdy jsou mezivýsledky odpovídající h_{ck} zpracovávány. V jiných okamžicích je diference mezi vektory rovna nule. Výsledkem útoku je tedy matice $\mathbf{R}$, kde každý řádek odpovídá rozdílu mezi vektory m'_{0i} a m'_{1i}

pro jeden odhad klíče. Rovnice výpočtu $\mathbf{R}$ je dána vztahem:

$$m'_{1i,j} = \frac{1}{n_{1i}} \cdot \sum_{l=1}^{n} h_{l,i} \cdot t_{l,j}, \tag{3.6}$$

$$m'_{0i,j} = \frac{1}{n_{0i}} \cdot \sum_{l=1}^{n} (1 - h_{l,i}) \cdot t_{l,j}, \tag{3.7}$$

$$n_{1,i} = \sum_{l=1}^{n} h_{l,i}, \tag{3.8}$$

$$n_{0i} = \sum_{l=1}^{n} (1 - h_{l,i}), \tag{3.9}$$

$$\mathbf{R} = \mathbf{M_1} - \mathbf{M_0}, \tag{3.10}$$

kde n značí počet řádků matice $\mathbf{H}$ (tzn. počet naměřených proudových spotřeb).

3.2.3 Útok založený na vzdálenosti středních hodnot

Tato metoda založená na vzdálenosti středních hodnot (Distance of Means) je vylepšení předchozí metody, protože bere v úvahu směrodatné odchylky. Metoda je založena na běžně používaném testu k porovnání rovnosti středních hodnot dvou různých rozdělení. Útok rozdělí matici $\mathbf{T}$ na dvě skupiny řádků pro každý odhad klíče stejně jak bylo popsáno v předchozí kapitole 3.2.2. Rozdíl od předchozí metody spočívá v tom, že střední hodnoty jsou porovnány podle testu vzdálenosti středních hodnot. Prvky matice $\mathbf{R}$ jsou vypočítány dle následujícího vztahu:

$$r_{i,j} = \frac{m_{1i,j} - m_{0i,j}}{s_{i,j}}, \tag{3.11}$$

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

3.2.4 Útok pomocí šablon

Útoky popsané v předchozích kapitolách 3.2.1, 3.2.2 a 3.2.3 předpokládaly, že útočník má jen velmi omezené znalosti o proudové spotřebě kryptografického zařízení, na které provádí útok. Ve většině případů bývají používány jednoduché modely proudové spotřeby a to model Hammingovy váhy a model Hammingovy vzdálenosti (podrobněji popisuje následující kapitola 3.2.5). Je zřejmé, že čím lépe bude odpovídat model skutečnosti, tím se útok stává efektivnějším. Při útoku DPA využívající šablony útočník popisuje proudovou spotřebu pomocí šablon (Template Based Attack). Tento typ útoku DPA byl poprvé představen v práci [4], kde jsou v podstatě rozšířeny šablony z SPA útoků.

Pro vysvětlení útoku bude pozornost zaměřena nejprve na jeden proudový průběh tzn. t'_i z matice $\mathbf{T}$. V tomto případě útočníka zajímá odpověď na otázku: jaká je

pravděpodobnost toho, že klíč v zařízení se rovná k_j pro $j = 1, \ldots, K$ s ohledem na naměřený průběh $t_i^{'}$. Tato podmíněná pravděpodobnost $p(k_j|t_i^{'})$ může být spočtena dle Bayesova teorému [50]. Bayesova věta umožní spočítat pravděpodobnost $p(k_j|t_i^{'})$ v závislosti na pravděpodobnosti a priori $p(k_l)$ a pravděpodobnosti $p(t_i^{'}|k_l)$ pro $l = 1, \ldots, K$ vztahem:

$$p(k_j|t_i^{'}) = \frac{p(t_i^{'}|k_j) \cdot p(k_j)}{\sum_{l=1}^{K}(p(t_i^{'}|k_l) \cdot p(k_l))}. \tag{3.12}$$

Pravděpodobnosti a priori jsou pravděpodobnosti pro různé klíče bez ohledu na průběh $t_i^{'}$. Bayesův teorém pak může být vnímán jako funkce, kde vstupem jsou pravděpodobnosti a priori $p(k_l)$ bez ohledu na průběh $t_i^{'}$ a výstupem jsou pravděpodobnosti a posteriori $p(k_j|t_i^{'})$ s ohledem na $t_i^{'}$. Pro jediný proudový průběh $t_i^{'}$ platí, že správný odhad klíče je klíč k_j s nejvyšší pravděpodobností $p(k_j|t_i^{'})$. Útoky založené jen na jednom proudovém průběhu nemohou být vždy úspěšné, protože v jednom průběhu nemusí být dostatek informací k nalezení klíče. Z tohoto důvodu rozšíříme tento přístup pro více naměřených proudových průběhů a to na určení pravděpodobnosti $p(k_j|\mathbf{T})$. Rozšíření $p(k_j|t_i^{'})$ na $p(k_j|\mathbf{T})$ není složité, protože proudové průběhy jsou statisticky nezávislé a tak můžeme pravděpodobnosti pro odpovídající průběhy násobit a rovnici 3.12 přepsat následujícím vztahem:

$$p(k_j|\mathbf{T}) = \frac{\left(\prod_{i=1}^{D} p(t_i^{'}|k_j)\right) \cdot p(k_j)}{\sum_{l=1}^{K}\left(\left(\prod_{i=1}^{D} p(t_i^{'}|k_l)\right) \cdot p(k_l)\right)}. \tag{3.13}$$

Rovnice 3.13 představuje podstatu útoku a výsledkem jsou pravděpodobnosti $p(k_j|\mathbf{T})$ pro $j = 1, \ldots, K$, které jsou použity k určení tajného klíče. Důležitou otázkou zůstává jak útočník může určit všechny pravděpodobnosti $p(k_j)$ a $p(t_i^{'}|k_j)$ potřebné k určení $p(k_j|\mathbf{T})$. Nalezení pravděpodobností a priori $p(k_j)$ je většinou snadné a útočník stanoví, že všechny klíče mají stejnou pravděpodobnost $p(k_j) = 1/K$. Určení pravděpodobnosti $p(t_i^{'}|k_j)$ je mnohem složitější záležitostí a jsou zde používány šablony. Útočník může použít v podstatě stejné šablony jak při SPA útoku, viz kapitola 3.1.2, tedy šablony odpovídající páru klíč a otevřený text.

3.2.5 Modely proudové spotřeby

V diferenciální proudové analýze je nutné mapování mezivýsledků na hodnoty proudové spotřeby (viz kapitola 3.2, čtvrtý krok útoku). Jedná se v podstatě o simulaci proudové spotřeby zařízení. Tyto simulace jsou typicky jednoduché a to z toho důvodu, že útočník většinou nedisponuje podrobnějšími znalostmi o zařízení, na které útočí.

Model Hammingovy váhy

Model Hammingovy váhy (HW, Hamming weight) je nejjednodušším modelem a je obvykle použit pokud útočník nemá žádné znalosti o vnitřní struktuře zařízení nebo nezná právě zpracovávaná data. Při použití HW modelu útočník předpokládá, že proudová spotřeba je přímo úměrná počtu právě nastavených nenulových bitů ve zpracovávaných datech. Datové hodnoty, které byly zpracovávány před nebo následně po této hodnotě jsou ignorovány. Z tohoto důvodu tento model není teoreticky vhodný pro popis CMOS obvodů, ale experimentální výsledky z praxe však ukazují, že HW zpracovávaných dat je závislá na proudové spotřebě CMOS obvodů a můžeme tento model použít.

Model Hammingovy vzdálenosti

Při použití modelu Hammingovy vzdálenosti (HD, Hamming Distance) útočník předpokládá, že proudová spotřeba je úměrná počtu změněních míst ve zpracovávaných datech. Podrobněji je v těchto simulacích model Hammingovy vzdálenosti použit k mapování přechodů, které nastávají na výstupu buněk vnitřního zapojení, na hodnoty proudové spotřeby. Útočník nemá většinou přístup k vnitřnímu zapojení, a tak nemůže provést kompletní simulaci. V praxi má útočník většinou některé informace o částech vnitřního zapojení a může tak provést simulace těchto částí. Například je-li kryptografické zařízení mikroprocesor, je velice pravděpodobné, že mikroprocesor je vyroben podobným způsobem jako veřejně známe mikroprocesory. Bude tak obsahovat registry, datové sběrnice, aritmetiko logickou jednotku a nějaké komunikační rozhraní atd. Tyto komponenty obvodu mají své typické vlastnosti, např. datová sběrnice je dlouhá a připojena k mnoha dalším částem obvodu. Proto je kapacitní zátěž této sběrnice velká a významně přispívá k celkové proudové spotřebě mikroprocesoru. Na datové sběrnici nenastávají chyby způsobené vícestupňovou strukturou (viz kapitola 1.2.3). Lze také předpokládat, že kapacitní zátěž všech jednotlivých vodičů datové sběrnice si je rovna. Na základě těchto tvrzení je zřejmé, že model HD se velmi dobře hodí k popsání proudové spotřeby datových sběrnic. Útočník tak může mapovat data, která jsou přenášena datovou sběrnicí na hodnoty proudové spotřeby bez znalosti vnitřního zařízení. Proudová spotřeba, která je způsobena změnou datové sběrnice z hodnoty v_0 na hodnotu v_1, je úměrná $HD(v_0, v_1) = HW(v_0 \oplus v_1)$. Podobně toto tvrzení může být aplikováno na další sběrnice, např. adresní a také na popis proudové spotřeby registrů. Registry jsou řízeny hodinovým signálem, a proto mění hodnotu jen jednou v průběhu hodinového cyklu. Útočník tak může simulovat proudovou spotřebu registrů počítáním HD dat, které jsou uleženy v registrech pro po sobě jdoucí hodinové cykly. Obecně lze tedy HD model použít k simulaci proudové spotřeby části zařízení dle vnitřního

zapojení pokud útočník zná zpracovávána data. Pro případ kombinačních buněk, kde útočník neví hodnoty zpracovávaných dat z důvodu vícestupňové struktury, je tento přístup nevhodný. Model HD je proto hlavně používán k simulaci registrů a sběrnic.

3.2.6 Diferenciální proudová analýza - shrnutí

Koncept útoku DPA byl poprvé popsán v práci [55]. Útok byl proveden na algoritmus DES metodou založenou na rozdílu středních hodnot. Následně pak byly diskutovány možné aplikovatelné statistické testy v [26]. Proudové modely byly poprvé definovány v práci [20] a v [6] byly základní proudové modely analyzovány v kontextu čipových karet. Simulační modely jsou stále modifikovány k zvýšení efektivity PA [89, 78, 34]. V [19] je popsáno použití korelačního koeficientu jako statistické metody. Ve vědecké literatuře se stalo populárním zavádět nové názvy pro DPA útoky, které jsou drobnou variací obecného schématu (viz kapitola 3.2), např. používají jen jiný statistický test (Korelační analýza [19, 24]). V kontextu DPA je důležité si uvědomit, že pojem DPA útok je nezávislý na použitém statistickém testu nebo použitém mezivýsledku. Pokročilé metody DPA jsou uvedeny v práci [109]. V práci [97] byla představena koncepce stochastických modelů. Útoky vyššího řádu (Higher-order DPA) kombinují DPA útoky pro několik zvolených mezivýsledků algoritmu. Tato myšlenka byla zmíněna již v prvním článku o DPA [55], ale až práce [73] popisuje konkrétní implementaci. Navazující práce [8, 108] popisují metody maskování a útoky vyššího řádu umožňující získat senzitivní data. Důležitá otázka vlivu předzpracování naměřených dat na efektivitu DPA byla prezentována v pracích [48, 44].

3.3 Protiopatření proti proudové analýze

Cílem kapitoly je popsat techniky protiopatření (countermeasure methods) vedoucí k znesnadnění a nebo plnému znemožnění útoku proudovým postranním kanálem. Zpravidla není nezbytné odstranit kanál samotný, ale postačuje zajistit, aby kanálem neunikaly z modulu citlivé informace nebo útočníkovy znesnadnit následnou analýzu naměřených dat. Přesto je odstranění kanálu samotného také žádoucí, neboť hrozí nalezení nové techniky a opětovného využití kanálu k útoku. Hlavním cílem protiopatření je zajistit, aby proudová spotřeba byla nezávislá na hodnotě mezivýsledku a operacích právě zpracovávaných kryptografickým modulem. Metody a techniky, kterými lze tuto nezávislost více či méně docílit jsou detailněji popsány v následující kapitole. Obecně lze techniky protiopatření kryptografického modulu proti útoku postranním kanálem rozdělit do dvou velkých skupin a to techniky skrývání (hiding)

a maskování (masking). Tyto dvě skupiny se následně dělí do dvou podskupin dle implementace na hardwarová a softwarová protiopatření.

3.3.1 Skrývání

Cílem skrývání je zajistit, aby proudová spotřeba byla nezávislá na hodnotě mezivýsledků, operacích právě zpracovávaných kryptografickým modulem a hodnotě dat. V podstatě existují dva způsoby jak docílit této požadované nezávislosti. První způsob je vyrobit zařízení takovým způsobem, aby proudová spotřeba byla náhodná. To znamená, že v každém hodinovém taktu bude proudová spotřeba různá i pro stejné instrukce. Druhý způsob je vyrobit zařízení takovým způsobem, že proudová spotřeba bude konstantní pro všechny operace a všechny datové hodnoty, tzn. proudová spotřeba bude v každém hodinovém cyklu stejná. Ideálním cílem skrývání, kterého však v praxi nelze dosáhnout, je realizace takového zařízení. Existuje několik návrhů a řešení jak se k tomuto ideálnímu stavu proudové spotřeby alespoň přiblížit. Tyto řešení můžeme rozdělit do dvou skupin. První skupina návrhů znáhodní proudovou spotřebu provedením operací kryptografického algoritmu v různých časových okamžicích při každém spuštění algoritmu. Tyto návrhy mají vliv na **časovou oblast proudové spotřeby**. Druhá skupina návrhů si klade za cíl učinit proudovou spotřebu náhodnou nebo konstantní a to přímo změnou charakteristické proudové spotřeby prováděných operací. Z tohoto důvodu mají tyto návrhy vliv na **okamžitou velikost proudové spotřeby**.

Ovlivnění časové oblasti proudové spotřeby

V kapitole 3.2 byl uveden důležitý požadavek DPA a to že naměřené proudové průběhy musí být správně zasynchronizovány. To znamená, že otisky pro jednotlivé instrukce by se měly nacházet vždy na stejné pozici v proudovém průběhu. Pokud tento požadavek není splněna, útočník k DPA útoku potřebuje podstatně více naměřených průběhů. Tento podmět je motivací k vytvoření kryptografického zařízení, které provádí instrukce algoritmu v náhodném pořadí. Výsledkem tohoto opatření je, že proudová spotřeba se jeví náhodná. Čím více je provádění jednotlivých instrukcí v čase náhodné tím více je obtížné provést útok na zařízení. Nejpoužívanějšími technikami k znáhodnění prováděných instrukcí je náhodné vkládání prázdných operací (dummy operations) nebo přesouvání operací (shuffling of operations).

Technika *vkládání prázdných operací* spočívá v náhodném vkládání prázdných instrukcí do kryptografického algoritmu. Pokaždé když je algoritmus spuštěn jsou vygenerovány náhodné čísla, které slouží k určení pozic vložení prázdných instrukcí. Je důležité, aby celkový počet vložených prázdných instrukcí byl vždy stejný. V tomto

případě útočník nemůže zjistit žádnou informaci o počtu vložených instrukcí opětovným měřením času vykonání celého algoritmu. V implementaci chráněnou touto technikou je pozice každé instrukce závislá na počtu vložených prázdných operací. Čím více je tato pozice proměnná, tím více je celková proudová spotřeba náhodná. Tedy znáhodnění proudové spotřeby přímo závisí na počtu vložených prázdných instrukcí. Je také zřejmé, že čím více je vloženo prázdných instrukcí, tím klesá rychlost provádění algoritmu (degraduje se optimalizace algoritmu). V praxi je proto optimální volit kompromis mezi rychlostí a znáhodněním proudové spotřeby pro každou konkrétní implementaci algoritmu.

Alternativní technikou k vkládání prázdných operacích je technika *přesouvání operací* popřípadě instrukcí kryptografického algoritmu. Základní myšlenkou techniky je znáhodnění provádění instrukcí algoritmu, u kterého lze instrukce a operace provádět v libovolném pořadí. Například u algoritmu AES při operaci `SubByte` je prováděna substituce dle substituční tabulky (viz příloha A.1). Těchto 16 substitucí je na sobě nezávislých a mohou být provedeny v libovolném pořadí. Při použití techniky přesouvání operací jsou vygenerována náhodná čísla při každém spuštění algoritmu, které jsou použity k určení pořadí provádění 16 substitucí. Přesouvání operací znáhodní proudovou spotřebu jednoduchým způsobem tak jako vkládání prázdných operací. Přesouvání operací nemá vliv na výkon algoritmu, tedy nedegraduje rychlost provádění algoritmu, ale nejde použít na všechny operace algoritmu a počet operací je tak omezen. Tento počet se odvíjí od použitého kryptografického algoritmu a konkrétní implementace. V praxi se tyto dvě techniky často používají současně.

Ovlivnění okamžité velikosti proudové spotřeby

Tato technika na rozdíl od předchozích přímo mění charakteristické proudové otisky prováděných instrukcí a operací. Základní myšlenkou je snížit množství unikající informace prostřednictvím postranního kanálu snížením odstupu signálu od šumu prováděných operací. Optimálním výsledkem této metody je hodnota SNR (Signal-to-noise ratio) rovna 0. Toho může být teoreticky docíleno buď snížením užitečného signálu na nulu nebo zvýšením šumu v měřeném signálu na nekonečno. V praxi tyto předpoklady můžeme realizovat jen do určité míry.

Nejjednodušším způsobem jak zvýšit šum jednotlivých instrukcí je provádět několik nezávislých operací paralelně. Proto jsou vhodnější hardwarové architektury kryptografických algoritmů s širokou strukturou datových cest. Například je daleko složitější provést útok na jeden bit proudové spotřeby AES v 128 bitové architektuře než v 32 bitové architektuře. Druhým způsobem zvýšení šumu je použití speciálních šumových rušiček. Tyto rušičky provádějí náhodné přepínání aktivních

a paralelních operací.

Naopak snížení užitečného signálu na nulu docílíme pokud bude proudová spotřeba stejná pro všechny instrukce a všechny datové hodnoty. Na první pohled se může zdát úkol zkonstruovat zařízení s konstantní proudovou spotřebou pro všechny instrukce a data snadno realizovatelný. Pokud se však tento úkol prostuduje podrobněji je jasné, že splnění není triviální. Existují dva způsoby jak se tomuto cíli přiblížit. Nejpoužívanější metodou je použití specializovaných logických buněk. Celková proudová spotřeba zařízení je součtem proudové spotřeby všech buněk logické struktury. Pokud budou všechny buňky vytvořeny tak, že jejich proudová spotřeba bude konstantní, potom bude výsledná proudová spotřeba také konstantní. Druhým způsobem snížení užitečného signálu je filtrování proudové spotřeby. Základní myšlenkou této metody je odstranit datovou závislost a závislost na operacích používaných komponent z proudové spotřeby za použití filtrů.

3.3.2 Maskování

Cílem všech opatření je, aby proudová spotřeba byla nezávislá na hodnotě mezivýsledku a operacích právě zpracovávaných kryptografickým modulem. Maskování přistupuje k tomuto problému znáhodněním mezivýsledku zpracovávaným kryptografickým modulem. Výhodou této metody je, že může být implementována na úrovni algoritmu bez nutnosti změn charakteristické proudové spotřeby zařízení. Jinými slovy, maskování umožní vytvořit proudovou spotřebu nezávislou na mezivýsledcích a to i v případě, že zařízení má proudovou spotřebu závislou na zpracovávaných datech.

Při maskování je každý mezivýsledek v zamaskován náhodnou hodnotou m, kterou nazýváme maska $v_m = v * m$. Maska m je generována interně v kryptografickém modulu a pro každé spuštění má jinou hodnotu, proto její hodnotu útočník nezná. Operace $*$ je typicky definovaná dle operací, které jsou použity algoritmem. Tato operace je většinou v kryptografickém modulu realizována exklusivním součtem XOR značeným symbolem $\oplus$ (aditivní maskování) nebo násobením značeným symbolem $\otimes$ (multiplikativní maskování). Ve většině případů jsou masky používány přímo na otevřený text nebo tajný klíč. Při použití maskování musí být implementace algoritmu mírně modifikována s ohledem na maskované mezivýsledky. Výsledkem kryptografického algoritmu je také maska, kterou je nutno odstranit k získání kryptogramu. Typické maskovací schéma popisuje jak jsou všechny mezivýsledky maskovány a jak se masky mění v algoritmu a následně konečné odstranění masek. Následující text uvede základní typy maskování, detailnější přehled technik maskování a implementace maskování je uveden v [67, 18, 87].

Rozlišujeme mezi *booleanovským* a *aritmetickým* maskováním. Při použití boolean maskování je mezivýsledek skryt exklusivním součtem s maskou $v_m = v \oplus m$. Při implementaci aritmetického maskování je mezivýsledek skryt aritmetickou operací sčítáním nebo násobením často v modulu n, $v_m = v + m (\mathrm{mod}\ n)$. Některé algoritmy jsou založeny na boolean a aritmetických operacích, a proto je nutné použít oba typy maskování. To je problematické, protože přechod mezi maskováními vyžaduje značné množství přídavných operací [25, 42].

Kryptografické algoritmy využívají lineární a nelineární funkce. Lineární funkce je například pro boolean maskování vyjádřena předpisem $f(x \oplus m) = f(x) \oplus f(m)$. Z toho plyne, že lineární operace jsou snadno maskovány s booleanovským maskováním. Příklad nelineární operace je AES S-box $S(x \oplus m) \neq S(x) \oplus S(m)$. V tomto případě se maska mění daleko složitějším způsobem, a proto není snadné používat boolean maskování na nelineární operace. Substituce S-box je založena na výpočtu multiplikativního inverzního prvku $f(x) = x^{-1}$ (viz příloha A.1.1) a to je vhodné pro multiplikativní maskování $f(x \times m) = (x \times m)^{-1} = f(x) \times f(m)$. Efektivní schéma pro přechod mezi multiplikativními a booleanovskými maskami je uvedeno např. [7]. Velkou nevýhodou multiplikativního maskování je nemožnost maskování mezivýsledku s nulovou hodnotou.

V předchozím textu bylo uvedeno, že při maskování je každý mezivýsledek v zamaskován náhodnou hodnotou m, kterou nazýváme maska, např. při použití booleanovského maskování $v_m = v \oplus m$. Hodnota mezivýsledku v může být vypočtena za pomoci v_m a m. Jinými slovy hodnota mezivýsledku v je reprezentována dvěma parametry (v_m, m). Znalost jednoho parametru nevede k žádné informaci o v, útočník potřebuje znát oba parametry k určení v (*maskování sdílením tajemství*). Následně maskování založené na sdílením tajemství používá tyto dva sdílené parametry. Sdílení tajemství s několika maskami je základní metoda maskování, tzn. že několik masek je použito pro jednu hodnotu mezivýsledku [21].

U asymetrických algoritmů je aditivní maskování nebo multiplikativní maskování velice efektivní. Aplikace aritmetického maskování je v kontextu asymetrické kryptografie nazýváno oslepení. Například u algoritmu RSA u procesu dešifrování je aplikovatelné multiplikativní maskování na vstupní zprávu $v : v_m = v \times m^e$. Výsledek v^d může být jednoduše získán na konci algoritmu protože platí: $(v_m)^d \equiv (v^d \times) (\mathrm{mod}\ n)$. Tato technika se nazývá *oslepení zpráv* (message blinding).

3.3.3 Způsoby implementace protiopatření

Podle toho na které úrovni architektury je implementováno protiopatření, rozlišujeme softwarové a hardwarové implementace. Je zřejmé, že ekonomicky výhodnější jsou softwarová protiopatření, a to z důvodu jejich snadné implementace do stávajících

kryptografických zařízení. Mnohem nákladnější hardwarová protiopatření, ale přináší zpravidla vyšší úroveň bezpečnosti.

Softwarové implementace

Možnosti jak měnit průběh proudové spotřeby pomocí softwarové implementace jsou značně omezené. Charakteristické proudové spotřeby jednotlivých instrukcí jsou definovány hardwarem na nižší úrovni architektury. Nejpoužívanějším protiopatřením je výše popsané skrývání v časové oblasti, jedná se o vkládání prázdných instrukcí a přesouvání instrukcí popřípadě celých operací. Tyto protiopatření jsou snadno implementovatelné v programu, avšak potřebují ke své činnosti generátor náhodných čísel, který musí být implementován v zařízení. Obecně lze konstatovat, že vkládání prázdných instrukcí a přesouvání operací nepřináší velký stupeň ochrany proti útokům proudovým postranním kanálem.

Softwarově lze měnit i okamžitá velikost proudové spotřeby. Charakteristické proudové spotřeby jednotlivých instrukcí jsou definovány hardwarem na nižší úrovni architektury, ale je to program, který určuje konkrétní instrukce k implementaci algoritmu. Výběrem instrukcí, které jsou použity k implementaci algoritmu, lze přímo ovlivňovat okamžitou velikost proudové spotřeby. Toto řešení zabrání útokům SPA při přímé interpretaci proudové spotřeby, ale není většinou dostatečné k zamezení provedení DPA útoku.

Hardwarové implementace

Na úrovni hardwaru existuje podstatně více možností k ochraně kryptografického zařízení pomocí skrývání. V časové oblasti navíc existují dvě skupiny protiopatření, které náhodně mění časový okamžik provádění instrukcí. První skupina je stejná jako při softwarové implementaci, tj. vkládání prázdných instrukcí a přesouvání operací. Tyto implementace jsou provedeny stejně tak jak na softwarové úrovni. Druhá skupina protiopatření ovlivňuje hodinový signál. Tyto protiopatření náhodně mění hodinový signál tak, aby znesnadnili synchronizaci proudové spotřeby. Mezi nejčastěji používané techniky k změně hodinového signálu patří:

- vynechání hodinového pulzu,
- náhodná změna frekvence hodinového signálu,
- existence více zdrojů hodinového signálu a přepínaní mezi nimi.

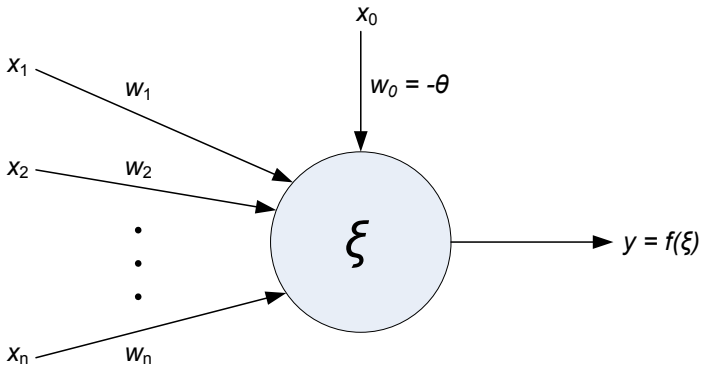
Okamžitá velikost proudové spotřeby může být ovlivněna na hardwarové úrovni snadněji než na softwarové úrovni. Proudová spotřeba může být rovna pro všechny operace a hodnoty dat filtrováním nebo může být zvyšována hodnota šumu vytvořenými šumovými rušičkami. V praxi je filtrování proudové spotřeby provedeno spínacími kapacitory, zdroji konstantního proudu a dalšími prvky obvodu, které

regulují proudovou spotřebu. Rušičky bývají vytvořeny na základě generátoru náhodných čísel. S cílem získat co největší vliv na proudovou spotřebu bývají generátory připojeny k síti velkých kapacitorů. Náhodné nabíjení a vybíjení této sítě vede k zvýšení šumu v proudové spotřebě, která činí útok PA obtížnější.

Při hardwarové implementaci je důležité si uvědomit, že SNR nezávisí jen na kryptografickém zařízení, ale také na měřicím pracovišti a metodě, které jsou použity k analýze proudové spotřeby. Pokud protiopatření sníží SNR pro jednu konfiguraci měřicího pracoviště, nesnižuje tím SNR pro všechny ostatní konfigurace měřicích pracovišť. Například metoda filtrování proudové spotřeby značně ztíží měření proudové spotřeby při použití proudového bočníku. Pokud by útočník měřil prosakující informaci prostřednictvím elektromagnetického postranního kanálu, nemělo by toto protiopatření výrazný efekt. Ze stejného důvodu by rušičky měly být rozprostřeny rovnoměrně v celém obvodu a neměly by být umístěny na jednom místě.

3.3.4 Protiopatření proti proudové analýze - shrnutí

Základní myšlenka skrývání byla zmíněná již v prvním článku o proudové analýze [55]. V posledních letech bylo publikováno několik článků zabývajícími se návrhy skrývání a to hardwarovými i softwarovými, nyní bude uveden stručný přehled publikovaných metod. V článku [27] je analyzován efekt RLC filtrů, které byly vloženy do napájecích cest kryptografického modulu. Shamir prezentoval v práci [98] myšlenku oddělení napájení od kryptografického zařízení pomocí dvou kapacitorů, které jsou periodicky přepínány tak, že zařízení není nikdy přímo propojeno s napájením. Metoda potlačení užitečného signálu je popsána [94], tato metoda používala speciální obvod na potlačení signálu. Obdobné metody potlačení užitečného signálu jsou uvedeny v pracích [80, 72]. Kromě protiopatřeních mající vliv na okamžitou hodnotu proudové spotřeby byly publikovány práce na časovou oblast, tedy znáhodnění vykonávání kryptografického algoritmu. V práci [70] je využit procesor, který náhodně mění sekvenci vykonávaných instrukcí a v článku [112] je využíváno změny frekvence hodinového signálu a napájecího napětí k znáhodnění proudové spotřeby. Efektivita znáhodnění provádění instrukcí kryptografického algoritmu byla analyzována v práci [21]. Dále byly publikovány práce popisující útoky na maskování využívající různé techniky zarovnání naměřených průběhů [23]. Detailnější přehled o skrývání u symetrických a asymetrických algoritmů a útoky na skrývání je uveden v knize [67]. V dnešní době jsou tyto základní metody stále používány a vylepšovány z pohledu implementace [79, 31, 52, 40].



Obr. 3.3: Formální neuron.

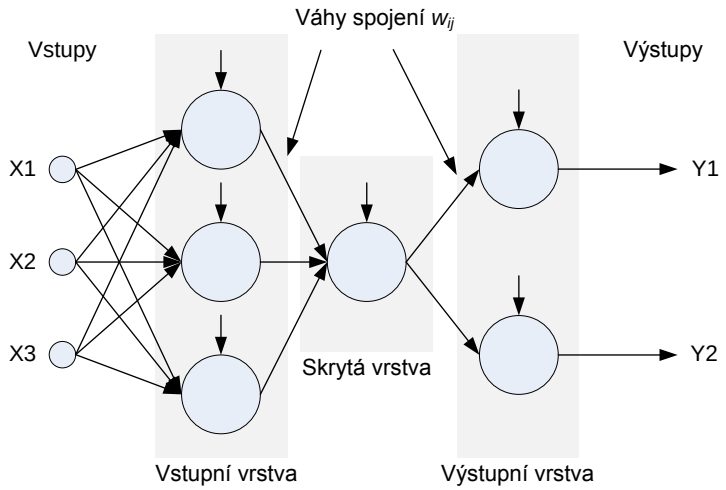
3.4 Neuronové sítě v kryptografii

Tato kapitola se v zabývá stručným úvodem do problematiky neuronových sítí. Nejprve je zadefinován formální neuron a následně je uveden základní popis třívrstvé neuronové sítě včetně učícího algoritmu využívající zpětné šíření chyby. Teto neuronová síť je využita v praktické části disertační práce. Závěr kapitoly se zabývá problematikou použití neuronových sítí v kryptografii a při analýze postranním kanálem.

Umělá neuronová síť je postavena na základech biologické neuronové sítě. Dochází nejprve k učení a následně potom k aplikaci naučených znalostí. Poznatky získané ze studia biologických neuronových sítí byly aplikovány do matematického modelu, který stejně jako biologická předloha zakládá svůj princip učení na změnách spojů mezi základními stavebními prvky sítě – neurony. Základní prvek umělé neuronové sítě je formální neuron, v literatuře též označován jako perceptron. Základní model je znázorněn na obr. 3.3. Formální neuron obsahuje x_i vstupů, které jsou násobeny vahami w_i, kde $i = 1$ až n. Vstup x_0 s váhou $w_0 = -\theta$ určuje prahovou hodnotu neuronu. Při učení neuronu dochází k přepočtu hodnot vah za účelem co nejbližšího přiblížení se k požadované výstupní hodnotě. Nejprve probíhá výpočet postsynaptického potenciálu, který je definován jako vnitřní funkce neuronu následujícím vztahem:

$$\xi = \sum_{i=1}^{n} x_i w_i - \theta \tag{3.14}$$

a má nejčastěji sigmoidní průběh. Z výsledné funkce ξ je pak vypočtena výstupní hodnota neuronu $y = f(\xi)$. Jeden formální neuron není schopen řešit složitější problémy, a proto jsou využívány neuronové sítě. Neurony jsou uspořádány do vrstev, jako je tomu na obr. 3.4. První vrstva je vstupní nebo také rozdělovací. Má za úkol hodnoty ze vstupu každého neuronu přivést na vstup každého neuronu další



Obr. 3.4: Obecná struktura třívrstvé neuronové síě.

vrstvy. Vnitřní vrstva se nazývá skrytá a může sdružovat i více vrstev, jejichž počet závisí na složitosti úkolu a zvoleném typu sítě. Poslední vrstva (výstupní) je odezvou celého systému na vstupní vzory. Celou funkci umělé neuronové sítě lze shrnout do dvou základních fází. První fází je fáze učení, v literatuře také označována za trénování neuronové sítě. V podstatě se jedná o matematický postup úpravy jednotlivých vah tak, aby výstup sítě odpovídal zadanému vzoru. K tomuto účelu je například často používaná metoda obecně nazývána jako zpětné šíření chyby (Backpropagation).

Neuronové sítě v kryptografii (Neuro-Cryptography nebo Neural Cryptography) je obor kryptologie věnovaný analýze využití statistických algoritmů, zejména neuronových sítí v kryptografii a kryptoanalýze. Neuronové sítě jsou dobře známé pro svou schopnost selektivně prozkoumat prostor řešení daného problému. Tato funkce jde přirozeně využít v oblasti kryptoanalýzy. Neuronové sítě také nabízí nový přístup k šifrování a dešifrování založený na zásadě, že každá funkce může být reprezentována pomocí neuronové sítě. Neuronové sítě jsou také výkonný výpočetní nástroj, který může být použit k nalezení inverzní funkce šifrovacího algoritmu. Neuronové sítě se nejčastěji využívají v kryptografii v následujících oblastech:

- obdoba asymetrických šifrovacích algoritmů [63],
- problematika distribuce klíčů [53],
- hašovací funkce [61],
- generátory náhodných čísel [110],
- protokol na výměnu klíčů [76] (obdoba Diffie-Hellman protokolu).

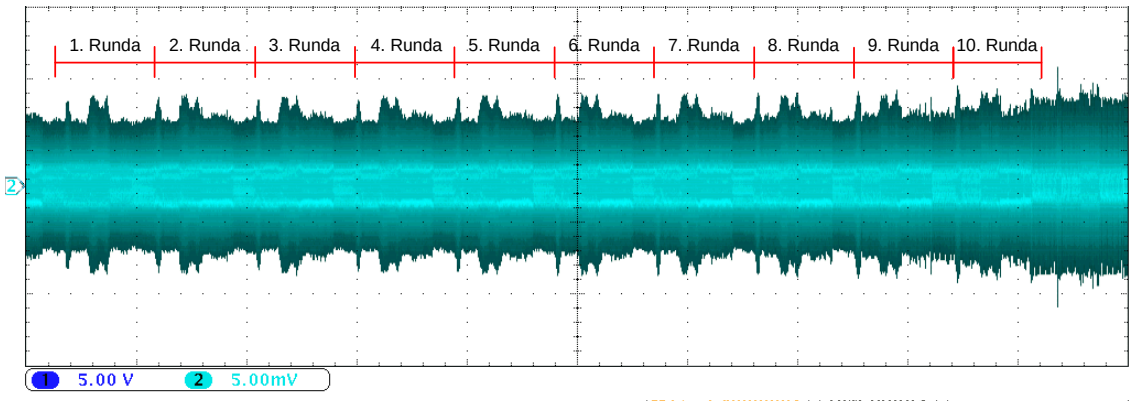
Neuronové sítě byly poprvé použity při klasifikaci informací unikajících prostřednictvím akustického postranního kanálu viz kapitola 1.4, např. [36, 64, 113]. Při analýze proudovým postranním kanálem byla možnost využití neuronových sítí poprvé publikována v práci [92]. Následně na tuto práci navazovali další autoři např. [59, 56], kteří se zabývali klasifikací proudových otisků, tedy přiřazením specifických proudových otisků jednotlivým prováděným instrukcím kryptografického modulu. Jednalo se v podstatě o metody zpětného inženýrství využívající proudovou spotřebu k určení vykonávaného kryptografického algoritmu. Použití neuronových sítí při analýze proudovým postranním kanálem a při klasifikaci konkrétní hodnoty bajtu tajného klíče bylo doposud velice málo publikováno a testováno. Práce zabývající se touto problematikou jsou založeny na algoritmech podpůrných vektorů (Support vector machines (SVM) [45, 12, 46, 60] a nevyužívají klasické vícevrstvé neuronové sítě s algoritmy učení.

4 VLASTNÍ ŘEŠENÍ - PROUDOVÁ ANALÝZA

Cílem kapitoly je přehledně shrnout dosažné výsledky. Text se průběžně odkazuje na publikované články v odborných časopisech a na konferencích, ve kterých jsou dosažené výsledky podrobně popsány. Dle definovaných cílů práce byla nejprve pro ověření teoretických znalostí testována metoda měření proudovým postranním kanálem. Na metodě měření byly testovány různé konfigurace a nastavení, tak aby výsledky měření byly co nejmarkantnější. Nejprve byly porovnávány různé metody měření proudového odběru kryptografického modulu: proudový bočník, proudová sonda, diferenciální sonda, pasivní sondy a pasivní sonda s oddělovacím transformátorem [117]. Následně byl zkoumán vliv parametrů vybrané metody měření využívající odporový bočník na výslednou proudovou analýzu [122, 121]. Zkoumané parametry vycházely z nastudované teorie proudového postranního kanálu (kapitola 1.2): vliv velikosti napájecího napětí, vliv odporu bočníku (při měřicí metodě proudové spotřeby využívající vložený odporový bočník [117]), frekvence hodinového signálu a velikost kapacity blokovacích kondenzátorů. Získané znalosti a zkušenosti byly nezbytné ke správné analýze proudové spotřeby kryptografického modulu a návrhu nové proudové analýzy využívající neuronové sítě.

Metoda měření proudovým postranním kanálem byla přizpůsobena pomocí sondy na měření blízkého elektromagnetického pole [120]. Byl porovnán proudový a elektromagnetický průběh šifrovacího algoritmu AES [124] a průběhy byly podrobeny detailní analýze v závislosti na implementovaném algoritmu. Získané elektromagnetické a proudové průběhy byly využity na realizaci útoků elektromagnetickým a proudovým postranním kanálem využívající jednoduchou i diferenciální analýzu [120, 125, 124, 114, 115]. Teoretický návrh optimalizace základní metody diferenciální proudové analýzy využívající rozdíl středních hodnot byl popsán v [116]. Následné experimenty se zaměřily na způsoby klasifikace pomocí neuronových sítí [118, 119]. Kompletní výsledky těchto analýz jsou detailně popsány ve výše uvedených pracích a tato kapitola shrnuje jen nejdůležitější výsledky.

Stěžejní částí této kapitoly je popis navržené analýzy proudovým postranním kanálem využívající neuronovou síť, který odpovídá hlavnímu cíli disertační práce [123]. Konkrétní návrh metody byl rozdělen do tří fází, které jsou postupně detailně popsány včetně naměřených proudových průběhů, implementace a získaných výsledků klasifikace. Závěrečná část kapitoly je popis optimalizace navržené metody, která umožnila zvýšení pravděpodobnosti správné klasifikace šifrovacího klíče, testování opakovatelnosti (realizovatelnosti metody) na větším počtu naměřených proudových průběhů a zhodnocení metody.

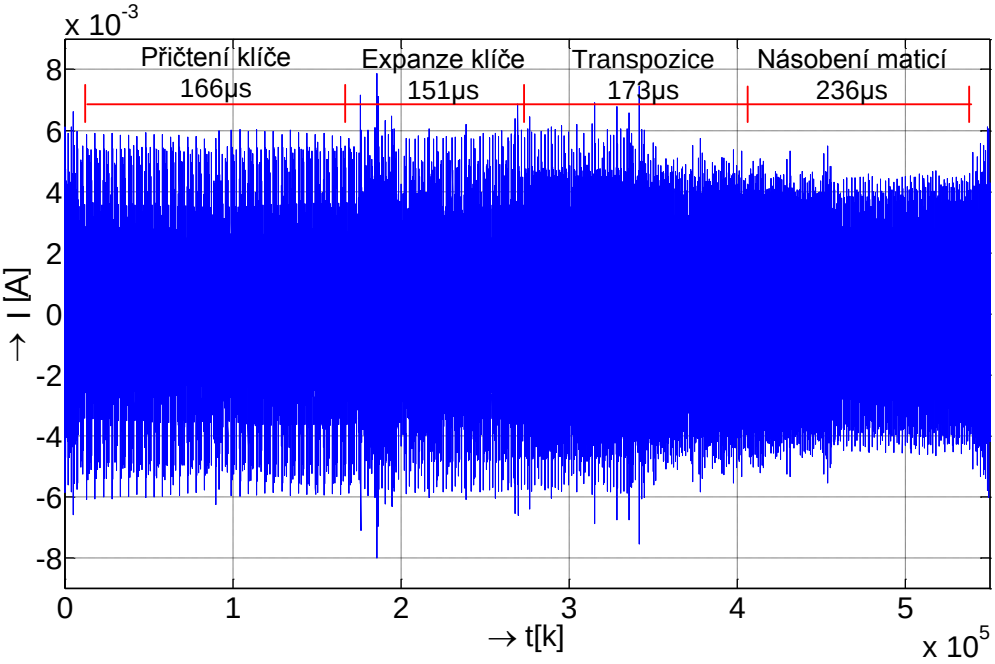


Obr. 4.1: Průběh proudové spotřeby algoritmu AES.

4.1 Proudový odběr algoritmu AES

Do kryptografického modulu (mikroprocesor PIC16F84A) byl implementován algoritmus AES, který byl vytvořen v jazyku symbolických adres panem Edimem Permadimem [91]. Tento program byl pouze upravován dle potřeb měření (např. vložen synchronizační signál pro konkrétní operace popřípadě jednotlivé instrukce atd.). Obr. 4.1 zobrazuje celkový průběh proudové spotřeby algoritmu AES. Na průběhu jsou zřetelně viditelné jednotlivé rundy šifrovacího algoritmu. Z úvodního měření proudové spotřeby bylo zřejmé, že proudová spotřeba odpovídala teoretickému předpokladu.

Pro podrobnější analýzu naměřeného průběhu zobrazuje obr. 4.2 detail proudové spotřeby a to konkrétně průběh první rundy algoritmu AES. Hrubé obrysy jednotlivých operací algoritmu AES jsou vidět i pouhým okem (viz příloha A.1). Algoritmus začíná počáteční fázi, kde dochází operací `AddRoundKey` k přičtení tajného klíče a tato fáze zabere mikroprocesoru 166μs. Následuje operace expanze klíče (`KeyExpansion`) s délkou 151μs, která je kvůli nedostatku paměťového místa na mikroprocesoru prováděna v každém cyklu a nedochází tak k výpočtu všech rundovních subklíčů před začátkem šifrování. Následují transpoziční operace nelineární bajtové substituce (`SubBytes`) a rotace řádků (`ShiftRows`), které modul vykoná během 151μs. Poslední operací je násobení maticí (`MixColumns`), které je poměrně časově náročná a zabere mikroprocesoru 236μs. Celková doba trvání jedné rundy této konkrétní implementace šifrovacího algoritmu AES-128 je 726μs. Po detailní analýze celkové proudové spotřeby algoritmu AES se pozornost soustředila na zkoumání parametrů metody měření ovlivňující proudovou spotřebu.

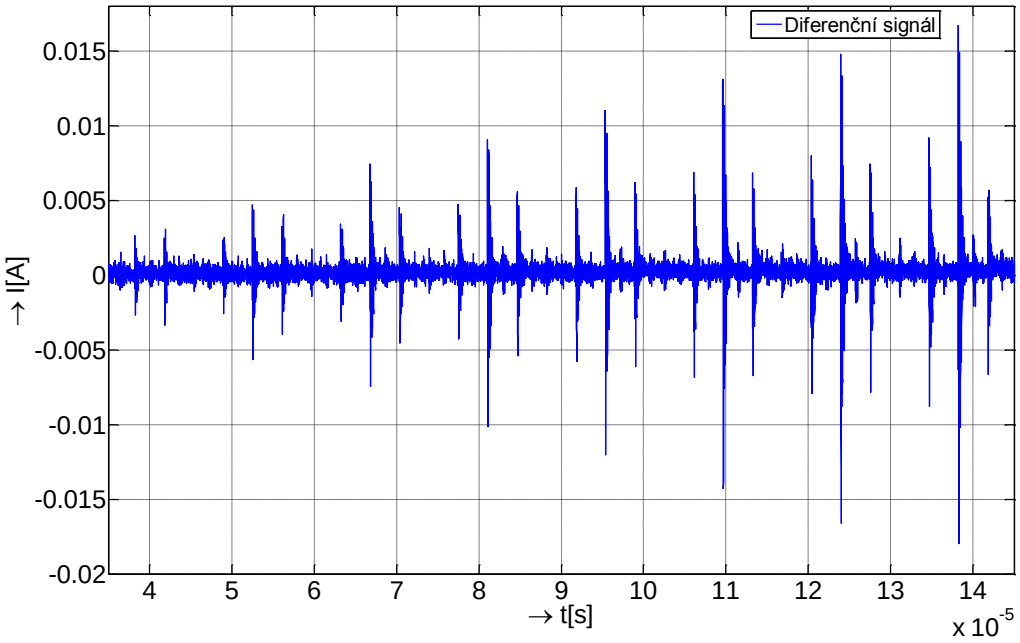


Obr. 4.2: Detail první rundy AES.

4.2 Parametry ovlivňující proudovou spotřebu

Dle poznatků popsaných v kapitole 3.3 je jednou z metod pro ochranu proti analýze postranním kanálem snížení užitečného signálu v měřeném průběhu. Tato kapitola se zabývá analýzou parametrů ovlivňující naměřenou proudovou spotřebu kryptografického modulu. Tyto výsledky jsou použitelné k znesnadnění analýzy proudovým postranním kanálem pro případného útočníka nebo k optimalizace nastavení parametrů metody měření. Experimentální poznatky z těchto měření byly použity k optimalizaci metody měření a k ověření opakovatelnosti a věrnosti výsledků měření proudové spotřeby během testování navržené metody proudové analýzy. Způsob a přesnost měření proudového odběru kryptografického modulu má zásadní vliv při proudové analýze. Detailní popis získaných výsledků je uveden v článcích [122, 121].

Během zkoumání nastavení parametrů metody měření nebyla soustředěna pozornost na celkový průběh proudové spotřeby kryptografického algoritmu AES, ale pozornost byla soustředěna pouze na počáteční fázi algoritmu, konkrétně na operaci `AddRoundKey`, která může být považována za kritickou, protože během této operace kryptografický modul pracuje s tajným klíčem, ze kterého jsou pak následně vypočítány všechny rundovní klíče (viz příloha A.1, operace expanze klíče) používané během celého procesu šifrování. Pokud by se útočníkovi podaří zjisti hodnotu tohoto klíče může zachycené šifrované zprávy bez problémů dešifrovat. Pro porovnání velikosti vlivu různých parametrů metody měření na výsledky proudové spotřeby byl



Obr. 4.3: Diferenciální průběh proudové spotřeby `AddRoundKey`.

určen diferenciální průběh proudové spotřeby operace `AddRoundKey`, který sloužil jako reference (obr. 4.3). Ve výsledku to znamenalo, že byl zkoumán vliv nastavení parametrů metody měření na proudovou spotřebu prováděných instrukcí `MOV` a `XOR` pro právě zpracovávaná data kryptografickým modulem. Hodnoty proudové spotřeby byly naměřeny metodou měření s parametry nastavenými následujícím způsobem: U_{CC}= 10 odpor bočníku R_{B}= 1 Ω, frekvence hodinového signálu f_{OSC1}= 4 MHz a hodnota blokovacího kondenzátoru C_1= 100 nF.

Určení diferenciálního průběhu bylo provedeno následujícím způsobem. Do kryptografického modulu byla implementována funkce `AddRoundKey` a to ve dvou variantách. Tato operace provádí XOR nad blokem (maticí) otevřeného textu uloženém v registru *Stav* (označíme maticí **S**) a blokem tajného klíče (**K**) a ukládá výsledek zpět do registru *Stav*. Ve své původní podobě pracuje AES algoritmus s bloky dat o délce 128 bitů tedy matice 4x4 bajty (viz kapitola A.1). Rovnici A.10 můžeme pro jednoduchost vyjádřit v maticové podobě:

$$\mathbf{S}' = \mathbf{S} \otimes \mathbf{K}. \tag{4.1}$$

V první variantně byla matice otevřeného textu $\mathbf{S_1}$ a matice tajného klíče $\mathbf{K_1}$ nulová (tzn. každé slovo má hodnotu 00h). V druhé variantě byla matice $\mathbf{S_2}$ opět nulová, ale matice $\mathbf{K_2}$ nabývala hodnot od 01h do FFh, kdy Hammingova váha w prvku následujícího je vždy větší o 1 oproti prvku předchozímu. Hodnota prvního slova klíče $k_{0,0}$ je rovna 01h (B'00000001'), tedy $w(k_{0,0}) = 1$. Následující prvek

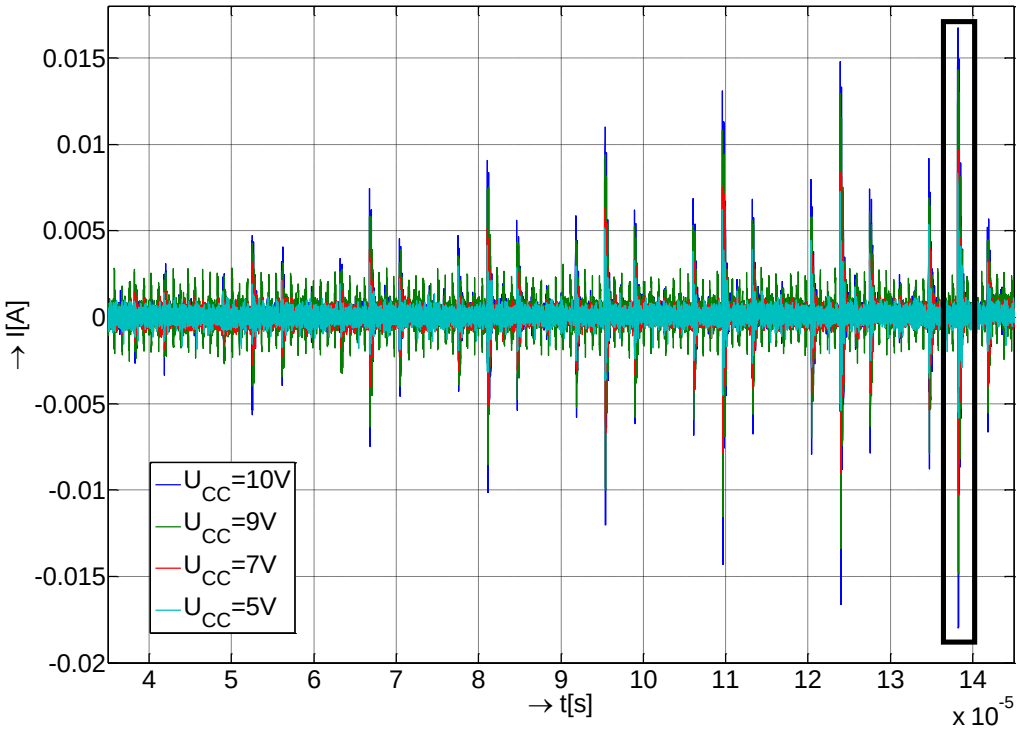
v matici má hodnotu 03h (B'00000011'), tedy Hammingova váha $w(k_{0,1}) = 2$. Poslední prvek $k_{3,3}$ pak nabývá hodnoty FFh (B'11111111'), kde $w(k_{3,3}) = 8$. Matice tajného klíče pro obě varianty bude vypadat následovně (hexadecimální zápis):

$$\mathbf{K_1} = \begin{pmatrix} 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 \end{pmatrix}, \mathbf{K_2} = \begin{pmatrix} 01 & 03 & 07 & 0F \\ 1F & 3F & 7F & FF \\ 01 & 03 & 07 & 0F \\ 1F & 3F & 7F & FF \end{pmatrix}. \tag{4.2}$$

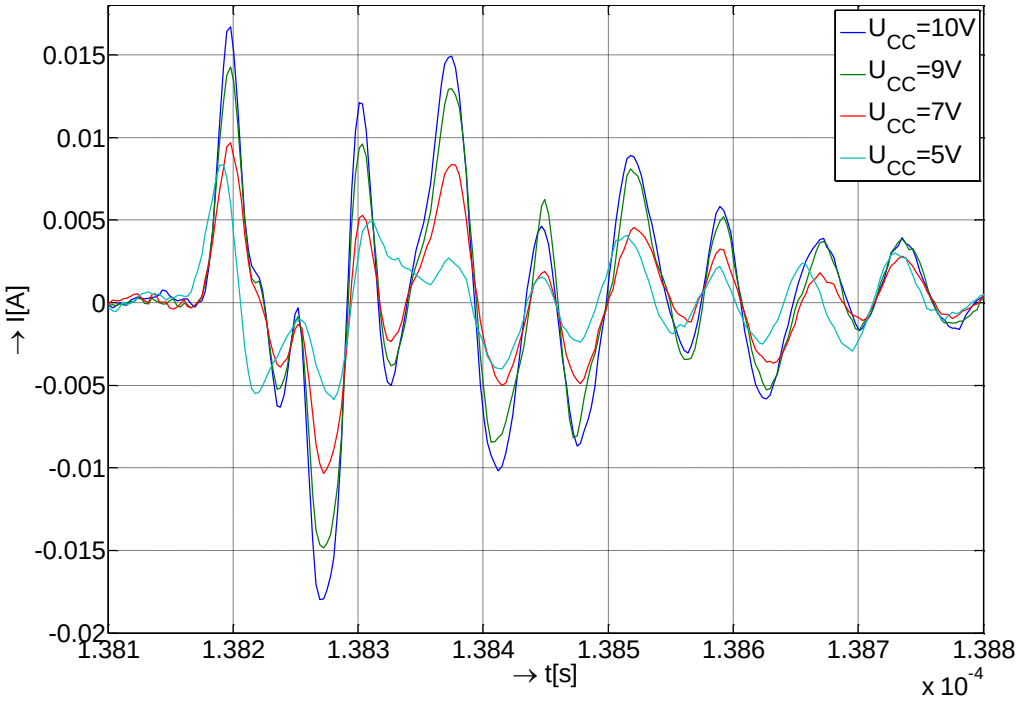
Proudová spotřeba během provádění algoritmu v první a druhé variantě byla zaznamenána. Z teorie vyplývá, že v první variantě nedochází k žádné operaci s daty, tedy přepnutí stavů tranzistorů, ukládání změn do paměti atd. V druhé variantě dochází k práci s daty dle popsaných pravidel v kapitole 1.2. Diferenciální signál se následně vypočte prostým rozdílem průběhů proudové spotřeby odpovídající těmto dvou fází, tzn. průběh proudové spotřeby z první fáze se odečte od průběhu proudové spotřeby z druhé fáze. Výsledkem této jednoduché operace je průběh proudové spotřeby zobrazený na obr. 4.3, který by měl zobrazovat rozdíl proudové spotřeby kryptografického modulu během operace `AddRoundKey` kdy jsou zpracovávána data a kdy jsou zpracovávaná data rovny nule. Na obrázku jsou patrné proudové špičky odpovídající práci s daty. Důležitá je viditelnost závislosti právě zpracovávaných dat na proudovou spotřebu během provádění operace `AddRoundKEy`, která odpovídá Hammingově váze nebo Hammingově vzdálenosti. Tento fakt není možné určit, protože matice otevřeného textu $\mathbf{S_1}$ a $\mathbf{S_2}$ byly nulové a proto model HW a HD jsou ekvivalentní.

Vliv napájecího napětí

Na obr. 4.4 jsou zobrazeny naměřené diferenciální průběhy proudové spotřeby pro různé hodnoty napájecího napětí U_{CC}. Z průběhů jsou patrny proudové špičky odpovídající operacím `MOV` a `XOR` pro jednotlivé bajty matic otevřeného textu a tajného klíče. Černým obdélníkem je označena největší proudová špička odpovídající operaci `XOR` pro datové hodnoty 00 a 0F, která bude sloužit k detailnějšímu porovnání průběhů proudové spotřeby. Obr. 4.5 znázorňuje tento detail označený černým obdélníkem na obr. 4.4. Stejný detail (největší proudová špička) bude použit i pro zkoumání jiných parametrů v následujícím textu. Z obou obrázků je patrné, že s rostoucím napájecím napětím U_{CC} narůstá hodnota proudu odebíraného mikroprocesorem. Tento proud je přímo úměrný napětí na bočníku. Pro U_{CC}= 5 V byla velikost proudu 8 mA. Pro U_{CC}= 10 V byla hodnota proudu dvojnásobná, tj. 16 mA. Se vzrůstajícím napájecím napětím mikroprocesoru se hodnota proudové



Obr. 4.4: Diferenciální průběh pro různá napájecí napětí.



Obr. 4.5: Detail průběhů proudů pro různé velikosti napájecího napětí.

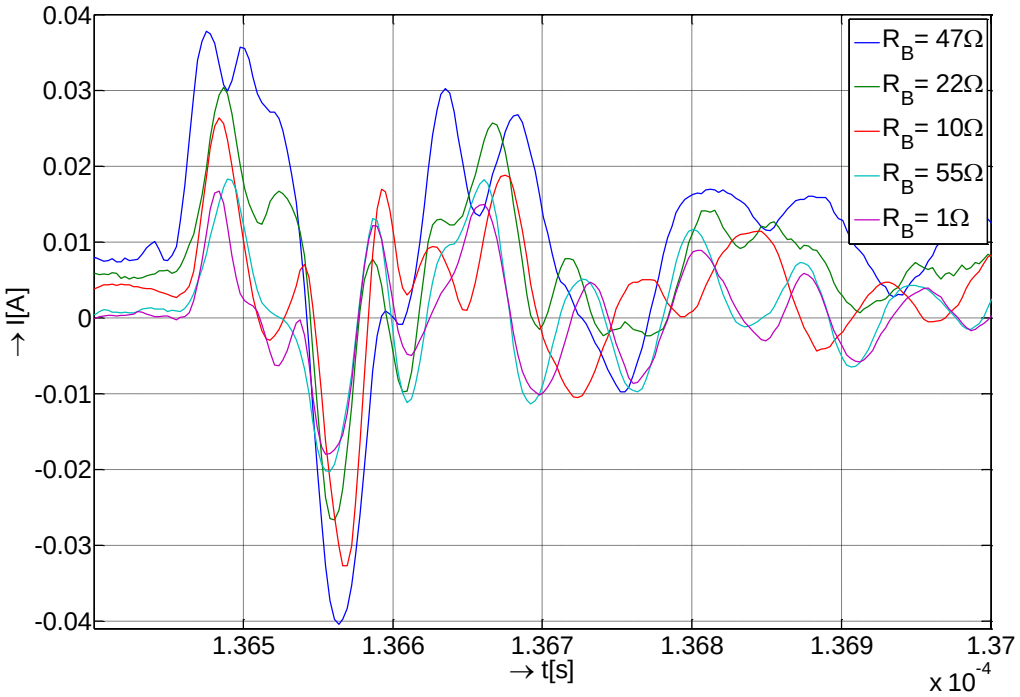
spotřeby zvyšovala, avšak šumová složka se prakticky nezměnila. S vyššími hodnotami napájecího napětí se tedy hodnota SNR zvyšuje a tedy kryptoanalýza proudového postranního kanálu je mnohem účinnější. V opačném případě při snižování U_{CC} odstup užitečného signálu od šumu klesá a proudová analýza je složitěji realizovatelná. Hodnoty proudové spotřeby byly naměřeny metodou měření s parametry odporu bočníku $R_B = 1\,\Omega$, frekvenci hodinového signálu $f_{OSC1} = 4\,\text{MHz}$ a hodnoty blokovacího kondenzátoru $C_1 = 100\,\text{nF}$.

Vliv odporu bočníku

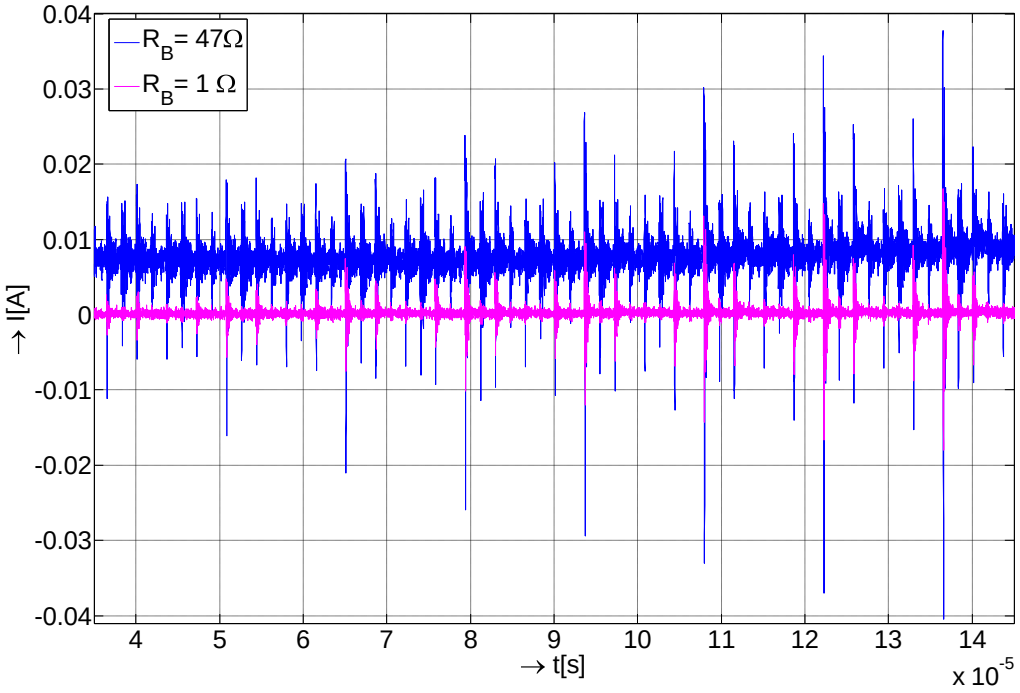
Na obr. 4.6 je zobrazen opět detail proudové špičky pro diferenciální průběhy proudové spotřeby pro různé velikosti odporu bočníku R_B. Proudové spotřeby kryptografického modulu byly naměřeny pro metodu měření s parametry: $U_{CC} = 10\,\text{V}$, frekvenci hodinového signálu $f_{OSC1} = 4\,\text{MHz}$ a kapacitu blokovacího kondenzátoru $C_1 = 100\,\text{nF}$. Dle předpokladů (Ohmův zákon) byly hodnoty proudové spotřeby vyšší s rostoucí hodnotou odporu bočníku, ale velikost nebyla přímo úměrná velikosti R_B. Pro $R_B = 1\,\Omega$ byla maximální hodnota referenčního signálu $I = 16{,}72\,\text{mA}$ a pro $R_B = 47\,\Omega$ byla maximální hodnota referenčního signálu $I = 37{,}76\,\text{mA}$. Z grafu zobrazeného na obr. 4.6 je patrné, že se vzrůstajícím odporem bočníku se zvyšovala Peak-to-Peak hodnota diferenciálního průběhu proudové spotřeby na bočníku. Přechody ze stavu logická 1 do stavu logická 0 byly sice výraznější, avšak klesala hodnota odstupu užitečného signálu od šumu (SNR) v naměřených průbězích. Tato vlastnost je patrná z porovnání naměřených diferenciálních proudových průběhů operace `AddRoundKey` na obr. 4.7 pro odpor bočníku $R_B = 1\,\Omega$ a $R_B = 47\,\Omega$.

Vliv frekvence hodinového signálu

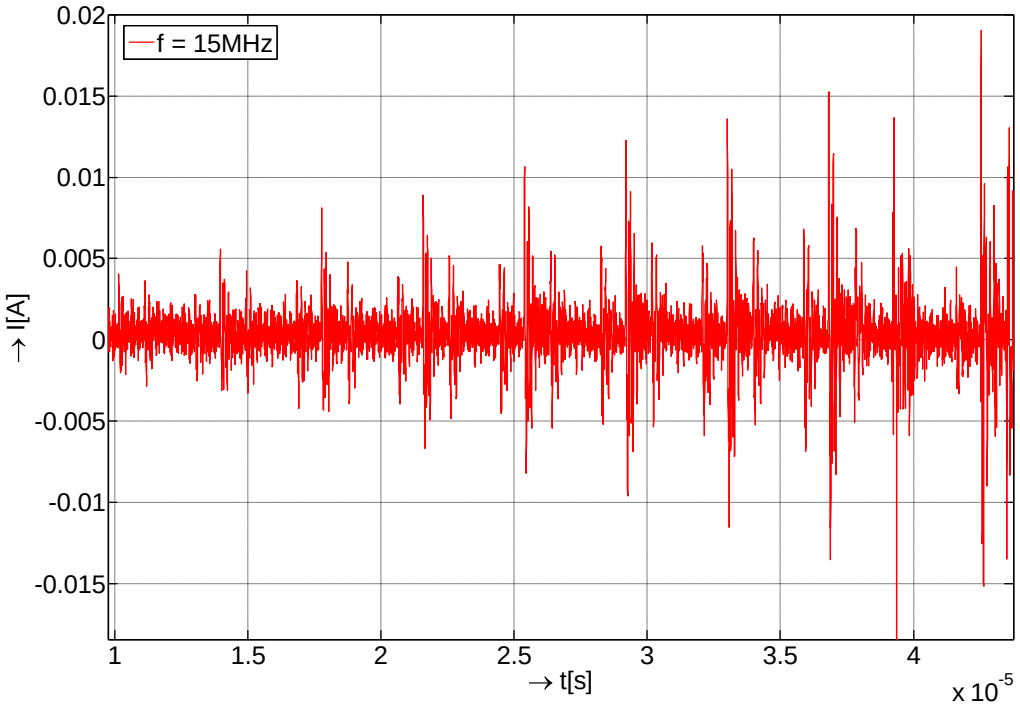
Na obr. 4.8 je zobrazen diferenciální průběh proudové spotřeby pro odlišný takt hodinového signálu na vstupu OSC1 mikroprocesoru. Proudový průběh byl naměřen pro metodu měření s parametry: odpor bočníku $R_B = 1\,\Omega$, napájecí napětí $U_{CC} = 10\,\text{V}$ a kapacitu blokovacího kondenzátoru $C_1 = 100\,\text{nF}$ a frekvence hodinového signálu $f_{OSC1} = 15\text{MHz}$. Při porovnání diferenciálních průběhů proudové spotřeby pro frekvence 4 a 15 MHz viz obr. 4.3 a 4.8 jsou patrné výraznější přechody mezi stavy vnitřních obvodů mikroprocesoru pro nižší frekvenci oscilátoru. Rozdíl v maximální hodnotě proudové spotřeby byl $\Delta I_{15,4} = 19{,}5 - 14{,}3 = 5{,}2\,\text{mA}$. S rostoucí frekvencí se ovšem zvýrazňuje oscilace napěťového signálu na bočníku, což je pro analýzu signálu nežádoucí. Výsledky měření nejsou tak čitelné, některé proudové špičky jsou matoucí, např proudová špička v čase 3.8s na obr. 4.8. Z hlediska bezpečnosti je proto výhodnější použití oscilátoru s vyšším taktem. Při vyšším taktu se také zvyšují požadavky na vzorkovací frekvenci osciloskopu (měřicího zařízení) a tím



Obr. 4.6: Detail průběhů proudů pro různé velikosti odporu bočníku.



Obr. 4.7: Srovnání průběhů diferenciálního signálu pro R_{B}= 1 Ω a R_{B}= 47 Ω.

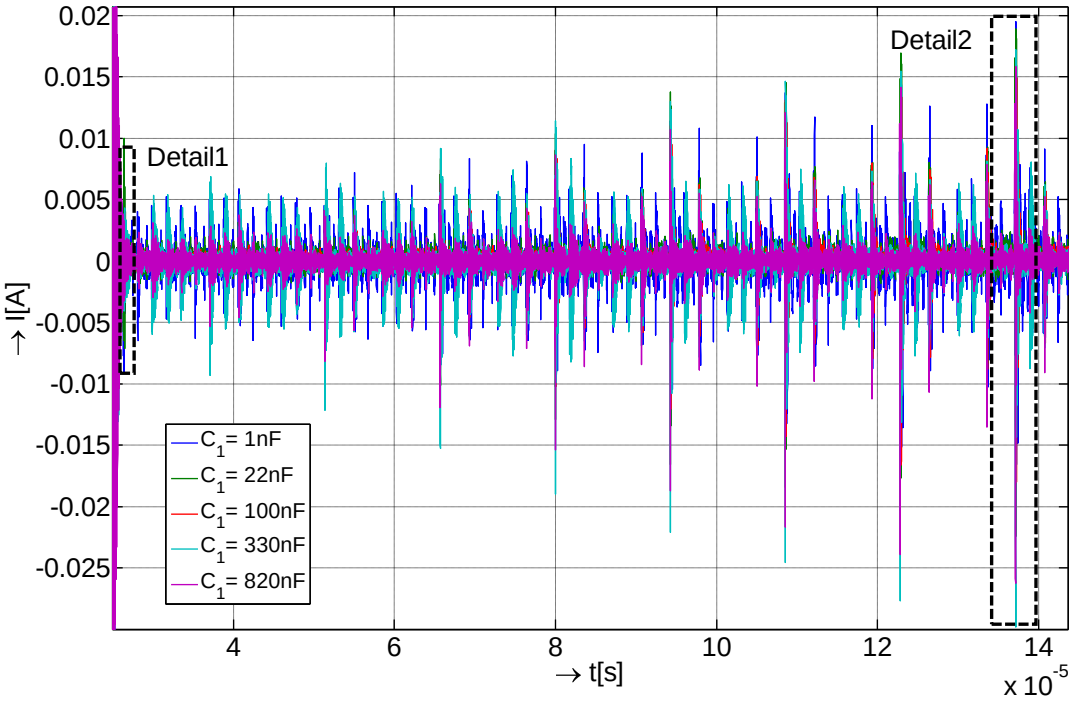


Obr. 4.8: Diferenciální průběh pro frekvenci hodinového signálu 15 MHz.

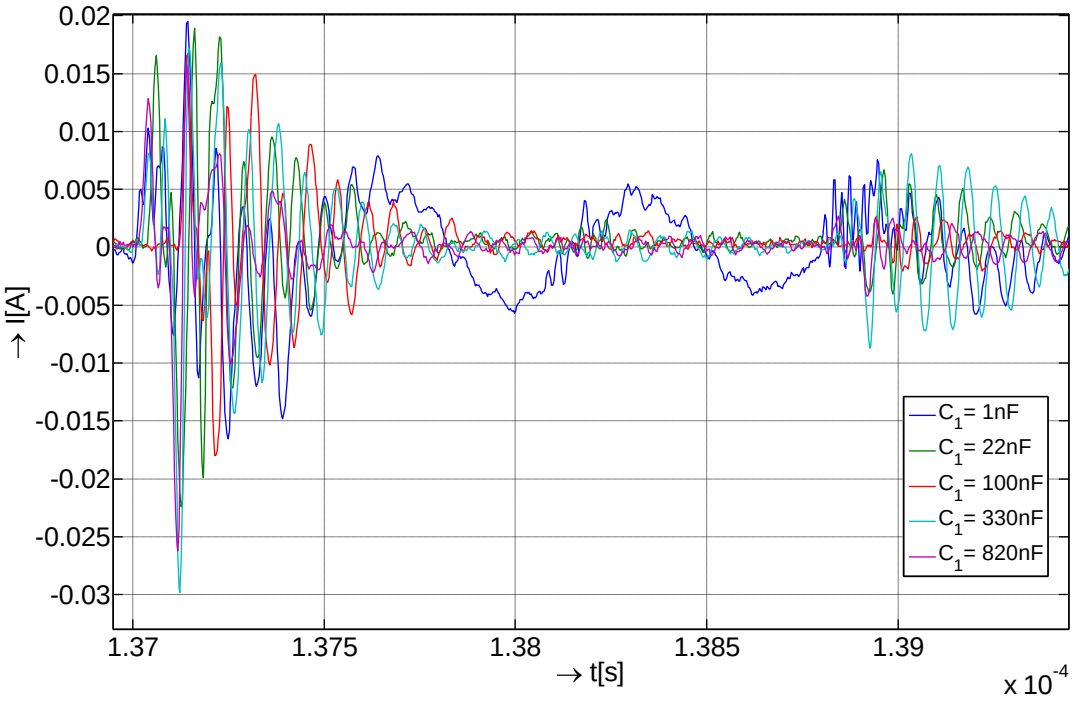
kladou vyšší nároky potenciálnímu útočníkovi. V ideálním případě by pro zvýšení bezpečnosti bylo výhodné použití takové frekvence taktu hodinového signálu, která by při analýze útočníkem způsobovala aliasing rekonstruovaného signálu.

Vliv kapacity blokovacího kondenzátoru

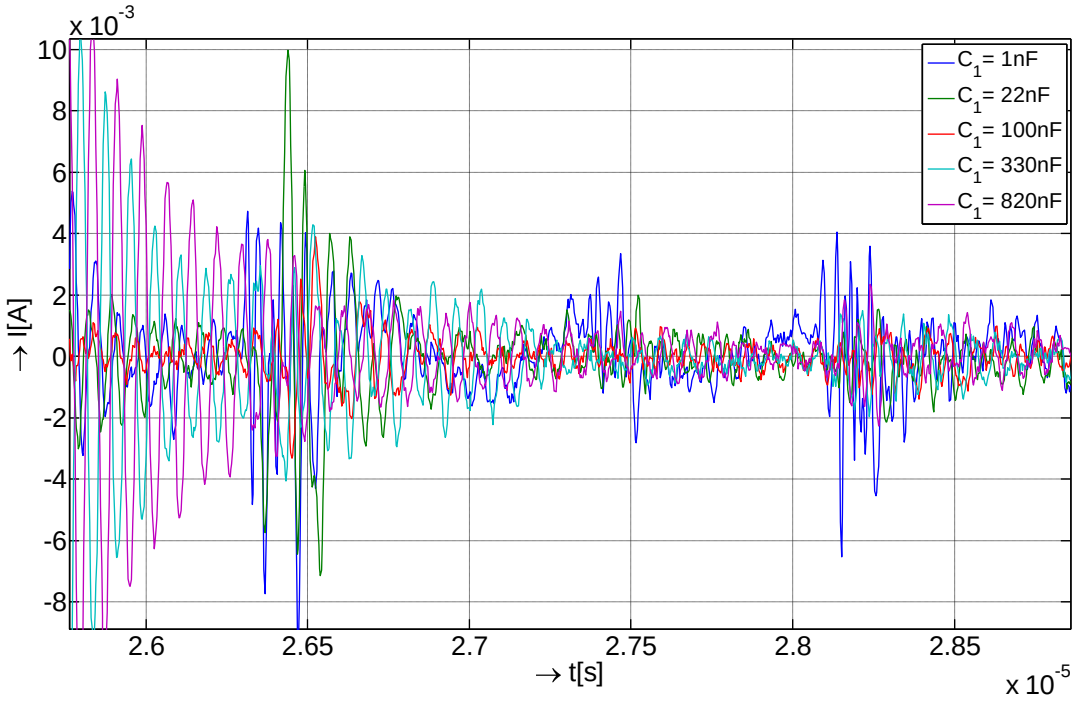
Obr. 4.9 zobrazuje průběhy diferenciálního průběhu proudové spotřeby pro různé hodnoty kapacity blokovacího kondenzátoru C_1. Proudové spotřeby byly naměřeny pro metodu měření s parametry: napájecí napětí U_{CC}= 10 V, frekvence hodinového signálu f_{OSC1}= 4 MHz a odpor bočníku R_B= 1 Ω. Kapacity blokovacího kondenzátoru byly voleny v hodnotách 1, 22, 100, 330 a 820 nF. Doporučená hodnota kapacity pro blokovací kondenzátor je C_1= 100 nF pro použitý kryptografický modul. Na obr. 4.11 je zobrazen první detail diferenciálních průběhů. Konkrétně se jedná o proudový průběh vykonání instrukce `bsf RB0`, která zajišťuje synchronizační signál. Pin RB0 mikroprocesoru se nastaví na hodnotu log. 1 (napájecí napětí mikroprocesoru), což je doprovázeno vysokou proudovou špičkou v průběhu proudové spotřeby mikroprocesoru. Pro velikost kapacity výrazně vyšší než hodnota 100nF je patrná výrazná oscilace signálu. Stejně tak to platí pro hodnoty nižší. V druhém detailu zobrazeném na obr. 4.10, který opět zobrazuje detail odpovídající největší proudové špičce v diferenciálním signálu je výrazná oscilace signálu při C_1= 1 nF. Volba této hodnoty



Obr. 4.9: Diferenciální průběhy pro různé velikosti blokovacího kondenzátoru.



Obr. 4.10: Detail 2 diferenciálního průběhu.



Obr. 4.11: Detail 1 diferenciálního průběhu.

je ovšem z konstrukčního hlediska naprosto nevhodná. Z pohledu ochrany proti proudové analýze by bylo vhodné použití vyšších hodnot kapacit blokovacích kondenzátorů.

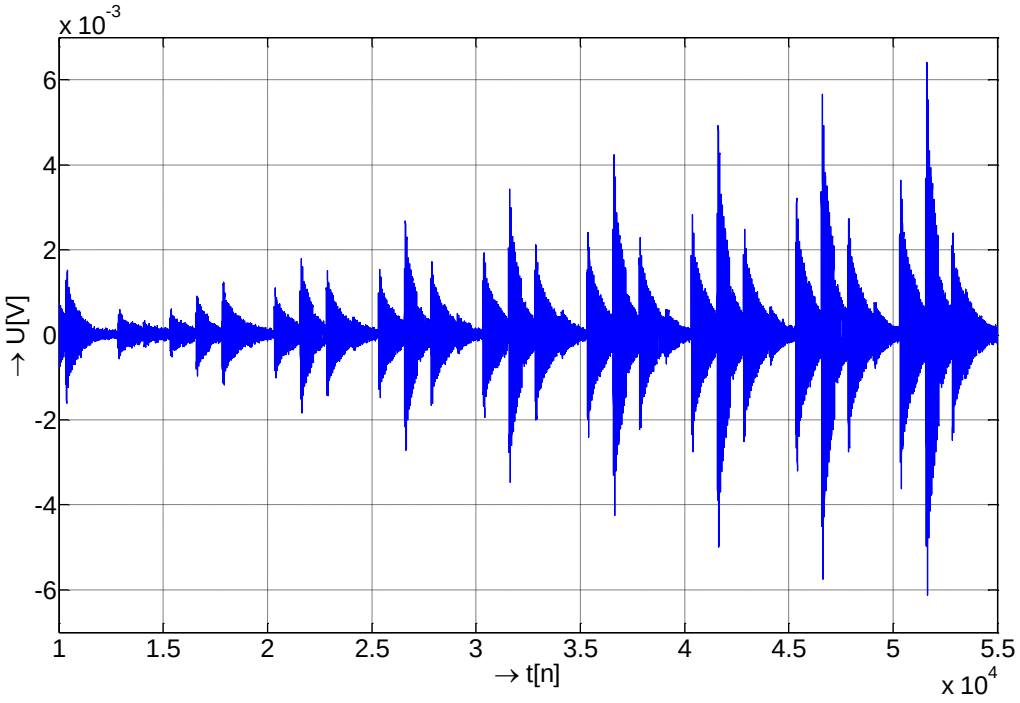
Zhodnocení výsledků

Proudový odběr mikroprocesoru byl měřen v závislosti na nastavení parametrů metody měření. Sledovanými parametry byla velikosti napájecího napětí, velikosti odporu bočníku, frekvence hodinového signálu a velikosti kapacity blokovacího kondenzátoru. Experimentální poznatky z těchto měření byly klíčové v návrhu a realizaci proudové analýzy využívající neuronové sítě. Se vzrůstajícím napájecím napětím mikroprocesoru se hodnoty proudové spotřeby úměrně zvyšovaly, avšak šumová složka se prakticky nezměnila. S vyššími hodnotami napájecího napětí může být proudová analýza mnohem účinnější, protože odstup signálu od šumu v naměřených datech klesá. Volba velikosti odporu bočníku měla dle předpokladů také značný vliv na výsledky měření. S rostoucí hodnotou velikosti odporu se sice Peak-to-Peak hodnoty diferenciálního signálu zvyšovaly, avšak rovněž narůstala velikost šumové složky ve výsledném průběhu, tj. klesal poměr odstupu užitečného signálu od šumu. Z těchto důvodů volba menší hodnoty odporu bočníku usnadňuje analýzu proudového postranního kanálu. Dle získaných zkušeností je vhodná hodnota odpo-

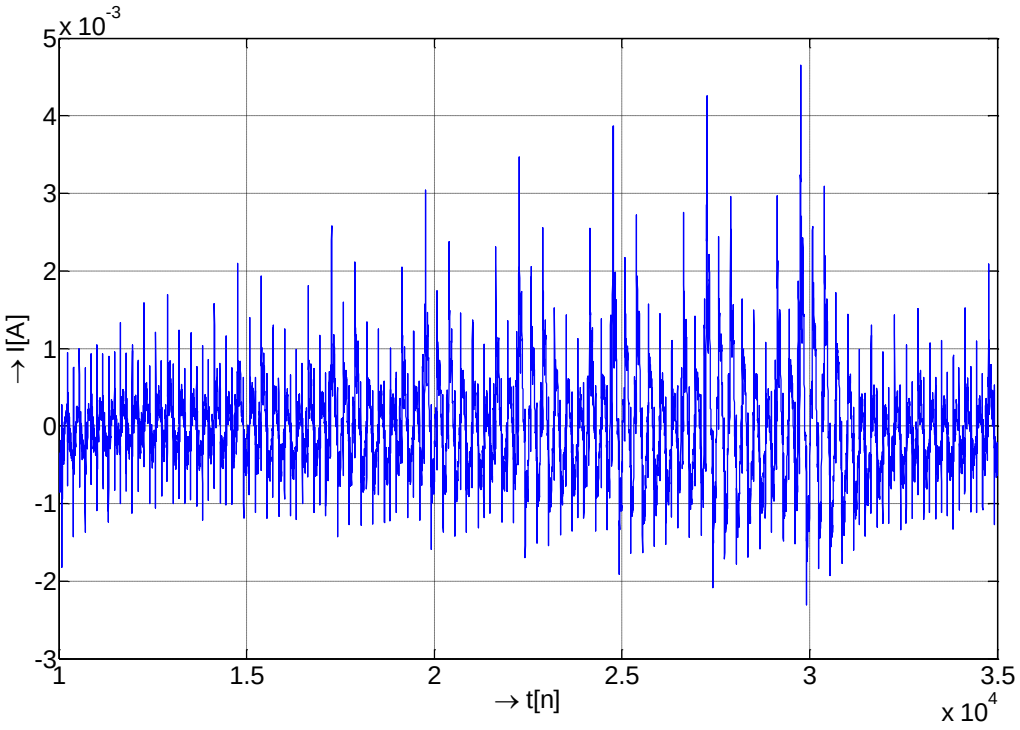
rového bočníku 1 Ω. Snímané diferenciální napětí pak odpovídá dle Ohmova zákonu přímo proudu a není nutný přepočet. Jako vhodná metoda měření diferenciálního napětí na bočníku o velikosti 1 Ω je například diferenciální sonda nebo pasivní sondy s oddělovacím transformátorem [117]. Volba frekvence hodinového signálu se rovněž výrazným způsobem projevovala na výsledném průběhu diferenciálního signálu. Pro proudovou analýzu je vhodnější použití nižší frekvence, která klade nižší nároky na měřicí zařízení. Oproti tomu z pohledu ochrany je vhodnější volba frekvence hodinového signálu co možná nejvyšší, popřípadě nejvhodnější volba je proměnná hodnota hodinového signálu. S vyššími hodnotami kmitočtu je diferenciální průběh napětí více zarušen šumem a jsou patrny výraznější oscilace napětí na bočníku. V případě analýzy vlivu velikosti kapacity blokovacích kondenzátorů byl průběh diferenciálního signálu nejméně zarušen při použití doporučené velikosti kapacity 100 nF (rozdílný výsledek měření než v práci [30]). V případě této volby je pak výsledný průběh diferenciálního signálu nejméně zarušen šumem a přechody mezi stavy vnitřních obvodů mikroprocesoru jsou pak mnohem zřetelnější. U příliš nízkých, nebo naopak vysokých hodnot kapacit, je u diferenciálního signálu výraznější oscilace při změnách stavů obvodů v mikroprocesoru.

4.3 Metoda měření - elektromagnetické pole

Tato kapitola porovnává diferenciální průběh proudové spotřeby funkce `AddRoundKey` kryptografického modulu pro různé metody měření. Byly porovnávány metody měření založené na odporovém bočníku, proudové sondě CT-6 a elektromagnetické sondě. Proudová spotřeba kryptografického modulu při použití odporového bočníku je zobrazena na obr. 4.3. Měřící metoda byla modifikována původní elektromagnetickou sondou pro snímání magnetické složky blízkého elektromagnetického pole. Podrobnější informace o konstrukci sondy a modifikacích pracoviště jsou uvedeny v práci [124, 125]. Diferenciální průběh elektromagnetického pole je zobrazen na obr. 4.12. Při vizuálním porovnání naměřených průběhů (obr. 4.3 a 4.12) je zřejmé, že průběhy jsou si prakticky totožné a výsledky měření odpovídají teoretickým předpokladům o vzniku blízkého elektromagnetického pole uvedených v kapitole 1.3. Diferenciální průběh získaný proudovou sondou CT-6 [2] je zobrazen na obr. 4.13. Z průběhu je opět jasně čitelná závislost proudové spotřeby na právě zpracovávaných datech kryptografickým modulem. Načítání a ukládání dat do registrů je zřetelné a průběh proudové spotřeby obsahuje více proudových šiček tedy obsahuje více informací (porovnání obr. 4.3 a 4.13). Šumová složka je také dobře potlačena v porovnání s předchozími průběhy. Z těchto důvodů byla metoda měření využívající proudovou sondu CT-6 vybrána pro následující měření.



Obr. 4.12: Diferenciální průběh `AddRoundKey` - elektromagnetická sonda.



Obr. 4.13: Diferenciální průběh `AddRoundKey` - proudová sonda CT-6.

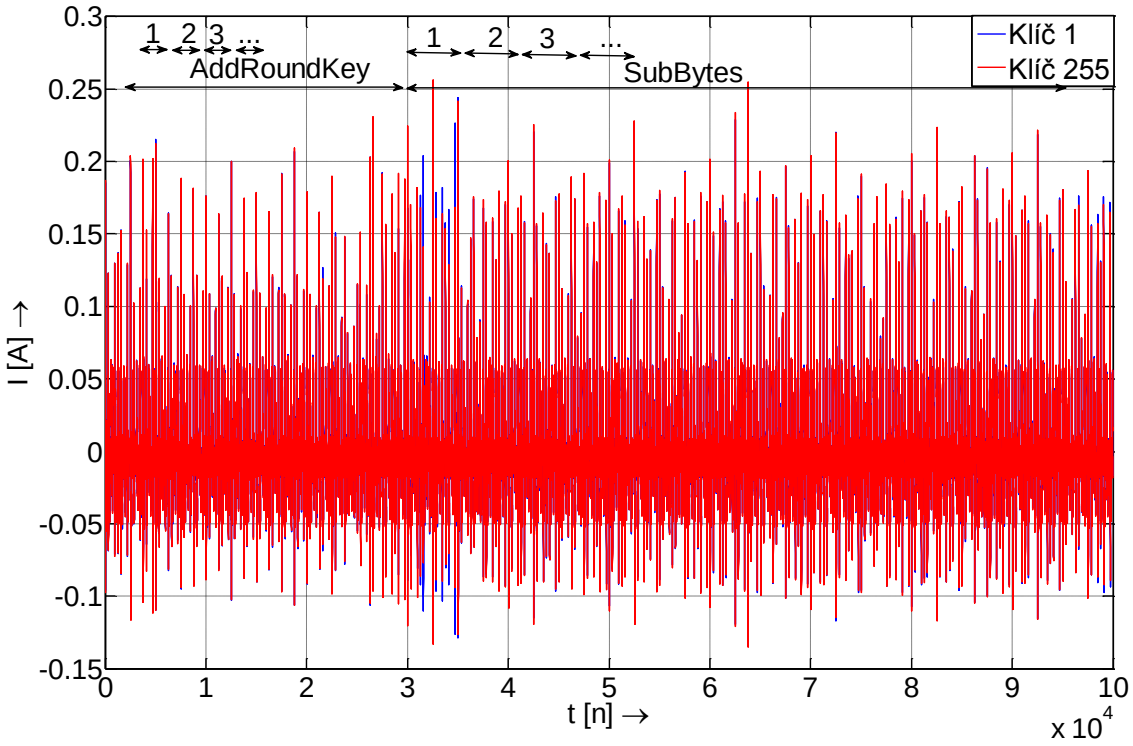
4.4 Realizace DPA útoku

Následující kapitola popisuje realizovaný útok DPA využívající korelační koeficient. Jsou postupně popsány jednotlivé kroky s ohledem na obecné schéma útoku. Do kryptografického modulu byl implementován algoritmus AES, který byl vytvořen v jazyku symbolických adres panem Edimem Permadimem [91]. Tento program byl upraven dle potřeb měření, byl vložen synchronizační signál pro konkrétní operace a zdrojový kód byl doplněn o komunikaci přes port RS-232. Po detailní analýze celkové proudové spotřeby algoritmu AES a verifikaci, že synchronizační signál a komunikace neovlivňuje proudovou spotřebu se pozornost soustředila pouze na počáteční fázi algoritmu (a to na operaci `AddRoudnKey` a operaci `SubBytes`), která je považována za nejdůležitější, protože během této operace kryptografický modul pracuje s tajným klíčem. Je vhodné pro útok DPA volit některou nelineární operaci, protože lineární operace ve většině případů nevynášejí dostatečné množství informace. Diferenciální analýza byla zaměřena jen na nelineární transformaci `SubBytes` v inicializační fázi algoritmu. Nelineární bajtová substituce zpracovává nezávisle každý bajt vstupních dat dle substituční tabulky, viz kapitola A.1.

V kryptografickém modulu byl nastaven **první bajt tajného klíče na hodnotu 71**. Otevřený text byl nastaven tak, že změna nastávala vždy jen v prvním bajtu. Z teoretických předpokladů plyne [67], že proudové průběhy se budou lišit výrazněji jen v místech, kde dochází k práci s prvním bajtem otevřeného textu. Zbylé odlišnosti jsou způsobeny elektronickým šumem [67], protože kryptografický modul zpracovává stále stejná (neměnná data). Dva proudové průběhy operací `AddRoudnKey` a operace `SubBytes` pro datové hodnoty 0 a 255 jsou zobrazeny na obr. 4.14. Na obr. 4.14 jsou naznačeny oblasti, v kterých probíhají operace `AddRoudnKey` a `SubBytes` a číslovkami jsou označeny místa korespondující práci s jednotlivými bajty. Obr. 4.15 zobrazuje náhodně vybraných 250 naměřených průběhů proudové spotřeby pro různá data, obrázek potvrzuje teoretické předpoklady. Následující text popisuje jednotlivé kroky útoku. Realizovaný útok **odpovídal obecnému schématu** popsanému v kapitole 3.2 a podrobněji jsou jednotlivé kroky a výpočty znázorněny na obr. 4.16.

V **prvním kroku** DPA útoku byla nejprve zvolena vnitřní hodnota algoritmu AES. Tato hodnota musí být závislá na známých datech a části klíče. Vnitřní hodnotou byl zvolen první výstupní bajt operace `SubBytes` v první rundě AES. Tedy první bajt otevřeného textu je nejprve přičten operací XOR ke klíči a následně je hodnota substituována substituční tabulkou.

V **druhém kroku** byly naměřeny průběhy proudové spotřeby kryptografického modulu pro operace `AddRoundKey` a `SubBytes` při šifrování 2500 náhodných bloků otevřeného textu. Naměřené průběhy korespondují s průběhy zobrazenými na obr.



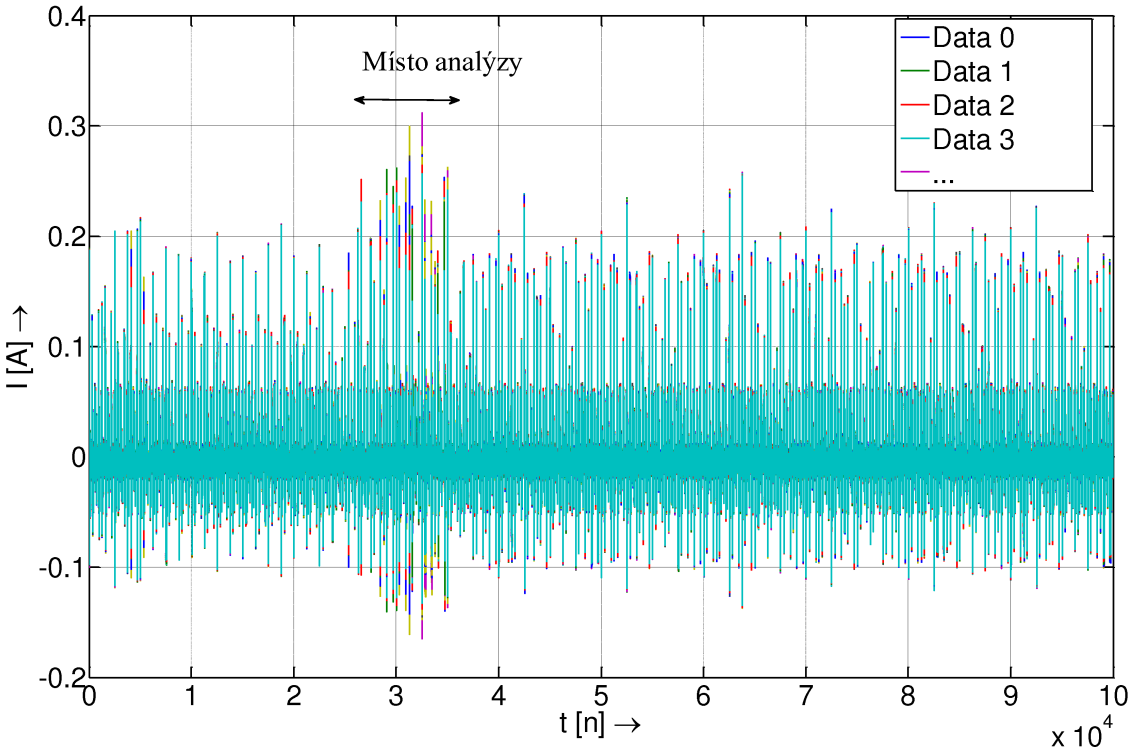
Obr. 4.14: Průběh proudové spotřeby operací `AddRoundKey` a `SubBytes`.

4.15. Pro snazší práci a výpočty v programu MATLAB byly proudové průběhy redukovány jen na místo práce s prvním bajtem u operace substituce viz obr. 4.15. Naměřené a redukované průběhy proudové spotřeby byly uloženy do matice $\mathbf{T}$ znázorněnou na obr. 4.17 a příslušné vstupy do proměnné `inputs`.

Třetí krok DPA útoku spočíval v stanovení hypotéz vnitřních hodnot algoritmu AES. Tyto hodnoty jsou určeny pro 2500 bloků známého otevřeného textu a všechny hodnoty tajného klíče (0 až 255). Výsledkem byla matice $\mathbf{V}$. Pro její prvky platí $v_{i,j} = S(d_i \oplus k_j)$, kde $d_1, \ldots, d_{2500}$ je první bajt z každého bloku otevřeného textu a $k_j = j-1$ pro $j = 1, \ldots, 256$. Matice $\mathbf{V}$ obsahuje celkově 2500×256 hypotéz vnitřních hodnot. V prostředí MATLAB byla tato operace implementována následovně:

```
[m,n] = size(T);
key = [0:255];
V = zeros(m,256);
for i=1:m
   V(i,:) = SubBytes(bitxor(inputs(i),key)+1);
end,
```

kde `SubBytes` odpovídá vektoru dekadických hodnot ze substituční tabulky (99, 124, 119, 123, 242, ...).



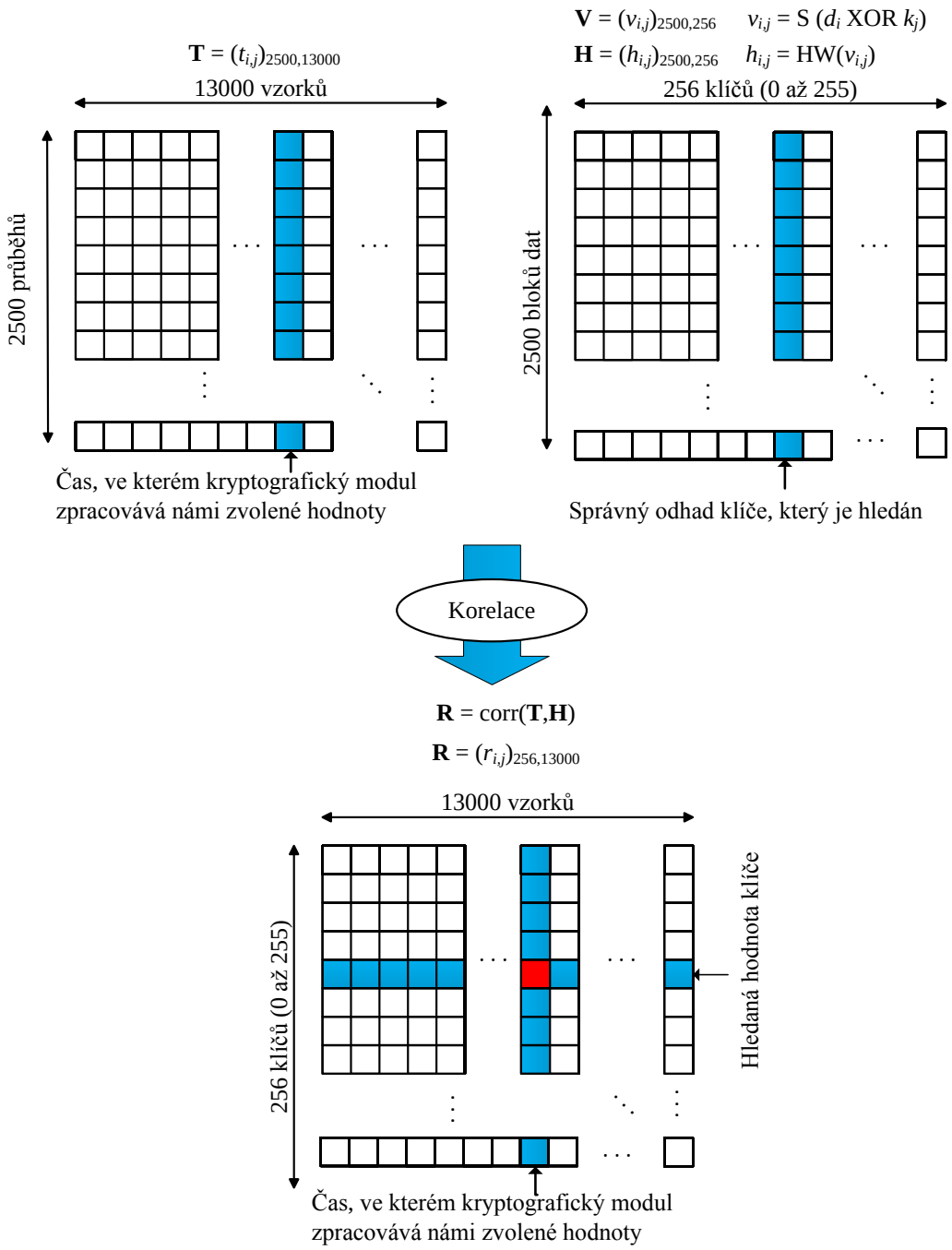
Obr. 4.15: Naměřené průběhy proudové spotřeby pro různá data.

Ve **čtvrtém kroku** je každé hypotetické vnitřní hodnotě $v_{i,j}$ z matice $\mathbf{V}$ přiřazena hypotetická hodnota proudové spotřeby $h_{i,j}$. K tomu je nutné vytvořit simulaci proudové spotřeby kryptografického zařízení, tzv.model spotřeby (viz kapitola 3.2.5). Zpravidla má útočník jen omezené množství informací o napadeném zařízení, a proto se snaží využít, co možná nejjednodušší model HW nebo HD. V uvedeném příkladě je použit model Hammingovy váhy. Jeho aplikací na matici $\mathbf{V}$ hypotetických vnitřních hodnot vznikne matice $\mathbf{H}$ pro hypotézy proudové spotřeby. Implementace uvedených modelů byla provedena následovně:

```
% model HW
  H = HW(V+1);
%model HD
for i=1:m
  H(i,:) = HW(bitxor(inputs(i),V(i,:))+1);
end,
```

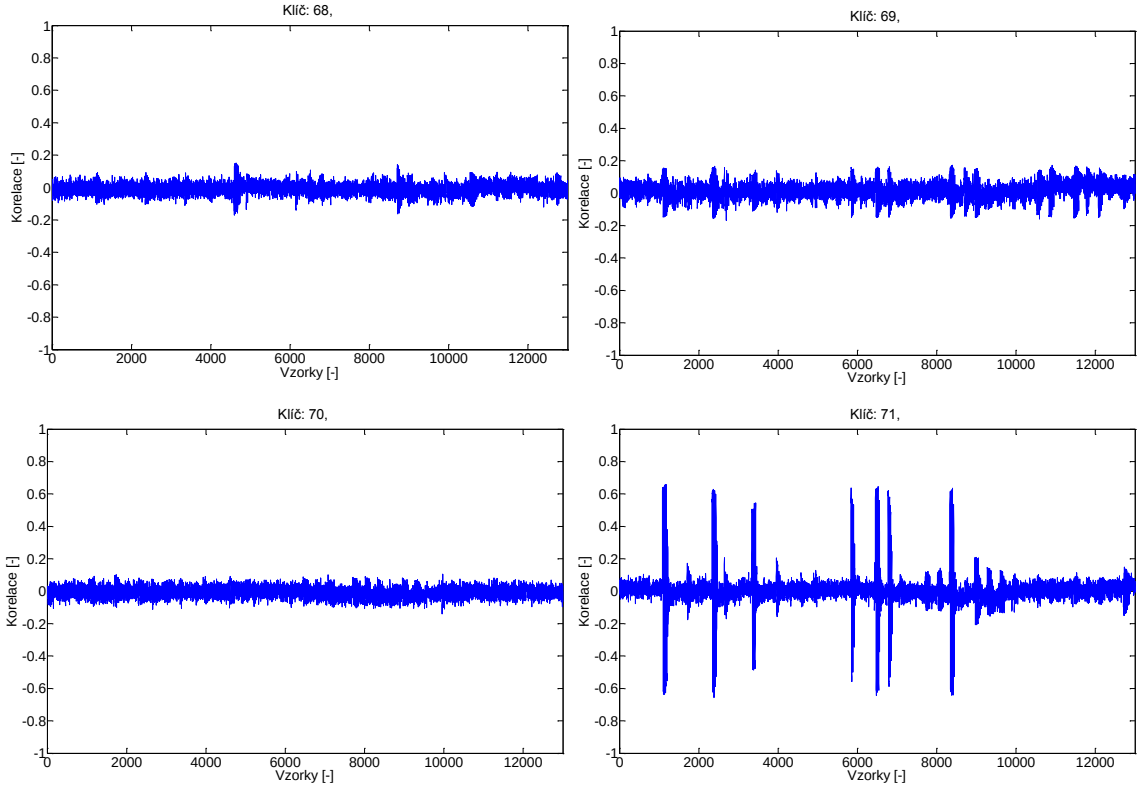
kde `HW` odpovídá vektoru dekadických hodnot Hammingových vah pro všechny hodnoty bajtu 0 až 255 (0, 1, 1, 2, 1, 2. . . .).

Pátý krok DPA útoku slouží k vyhodnocení míry lineární závislosti (korelace) hypotéz proudové spotřeby pro všechny odhady klíče (hodnoty ve sloupci $\mathbf{h}_i$ matice



Obr. 4.16: Průběh jednotlivých kroků DPA.

H) a změřených průběhů (hodnoty ve sloupci $\mathbf{t}_j$ matice **T**). K vyhodnocení míry lineární závislosti mezi sloupci $\mathbf{h}_i$ a $\mathbf{t}_j$, kde $i = 1, \ldots, K$ a $j = 1, \ldots, T$, se využívá různých metod viz kapitola 3.2.1, 3.2.2 a 3.2.3. Ze zmíněných metod je nejčastěji používán korelační koeficient. Výsledkem této metody je matice **R** korelačních koe-



Obr. 4.17: Výsledky korelační matice.

ficientů. Každá hodnota korelačního koeficientu $r_{i,j}$ je určena následovně:

$$r_{i,j} = \frac{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i}) \cdot (t_{d,j} - \overline{t_j})}{\sqrt{\sum_{d=1}^{D}(h_{d,i} - \overline{h_i})^2 \cdot \sum_{d=1}^{D}(t_{d,j} - \overline{t_j})^2}}, \tag{4.3}$$

kde hodnoty $\overline{h_i}$ a $\overline{t_j}$ označují průměrné hodnoty prvků ve sloupcích $\mathbf{h}_i$ a $\mathbf{t}_j$. D udává počet těchto prvků. Matici $\mathbf{R}$ lze zobrazit jako množinu grafů. Každý graf znázorňuje jeden řádek matice $\mathbf{R}$, tedy jednu hypotézu (odhad) klíče. Na obr. 4.17 jsou zachyceny grafické průběhy pro hypotézy klíče 68 až 71 právě provedeného útoku. V průběhu pro hypotézu klíče 71 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 informaci, že první bajt tajného klíče má hodnotu 71. Z počtu špiček daného průběhu dále vyplývá, že mikrokontrolér s touto vnitřní hodnotou pracuje v několika instrukcích. To je zpravidla obvyklé pro softwarovou implementaci šifrovacího algoritmu, kdy při práci s hledanou vnitřní hodnotou dochází k přesunu této hodnoty mezi pamětí a registrem mikrokontroléru a obráceně.

4.5 Návrh optimalizace DPA

Tato kapitola se zabývá teoretickým návrhem optimalizace základní metody diferenciální proudové analýzy založené na rozdílu středních hodnot. Návrh výpočetní optimalizace je podrobněji popsáno v článku [116] pro proudovou analýzu a práce [120] popisuje optimaliyace pro elektromagnetickou analýzu.

Pro vysvětlení návrhu optimalizace bude uveden konkrétní příkladu útoku metodou využívající rozdílu středních hodnot (obecný popis metody viz kapitola 3.2) na šifrovací algoritmus AES. Schéma útoku DPA touto metodou lze názorně popsat blokovým schématem zobrazeném na obr. 4.18. Útočník si připraví sadu otevřených textů $d[1 \dots D]$, kde první stavový bajt (prvních 8 bitů) má náhodný charakter a zbylých 120 bitů je konstantní. Toto uspořádání zaručuje, že diferenciální průběhy, které budou následně počítány, budou vykazovat diferenci pouze v místech, kde bude pracováno právě s prvními osmi bity stavu (tajný klíč je po celou dobu konstantní).

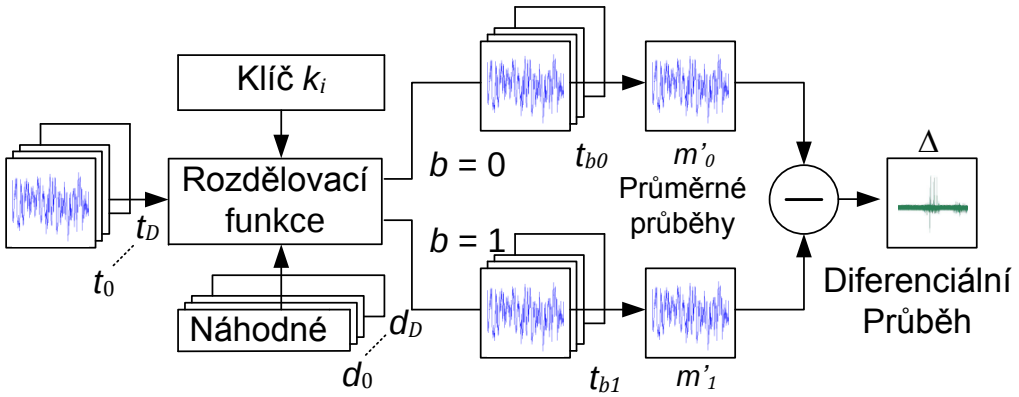
Následně útočník změří proudové průběhy odpovídající mezivýsledku, například operaci `Add Round Key` v inicializační fázi algoritmu AES s připravenými otevřenými texty pro několik stovek až tisícovek opakování. Měření D průběhů elektromagnetického pole navzorkuje na K hodnot. Tyto průběhy lze pak reprezentovat jako dvourozměrné pole $t[0 \dots D][0 \dots K]$, kde první index určuje pořadí operace a druhý definuje konkrétní vzorek. Korespondující otevřené texty jsou reprezentovány v poli $P[0 \dots D]$.

Další fází útoku je zpracování naměřených dat. Výstup operace `Add Round Key` je bajt vstupující do substituční tabulky S-BOX, na jejímž výstupu dostáváme jinou osmibitovou kombinaci. Libovolný bit na výstupu substituční tabulky poslouží jako rozdělovací bit b rozdělovací funkce. Naměřené průběhy proudové spotřeby lze následně rozdělit pro každý odhad klíče do dvou podmnožin na základě hodnoty b bitu. Podle hodnoty b bitu rozdělíme naměřené průběhy do dvou skupin - pro $b = 0$ do podskupiny t_{b0} a pro $b = 1$ do podskupiny t_{b1}. Získané podmnožiny lze definovat vztahy:

$$t_{b0} = \{t_i : b = 0\}, \tag{4.4}$$

$$t_{b1} = \{t_i : b = 1\}. \tag{4.5}$$

Toto rozdělení naměřených průběhů je nutno provést pro všechny odhady klíče (pro AES128 je to $256 \cdot 16$ možných variant). V případě, že otevřené zprávy budou náhodné, bude rozložení průběhu v obou podmnožinách rovnoměrné. Každá podmnožina je dále reprezentována průměrným průběhem, který je definován pro každou podmnožinu pro $j = 1 \dots k$ takto (dle vztahů 3.10):



Obr. 4.18: Blokové schéma útoku založeném na rozdílu středních hodnot.

$$m_1^{'}[j] = \frac{1}{|t_{b1}|} \sum_{t_i \in t_{b1}} t_i[j], \tag{4.6}$$

$$m_0^{'}[j] = \frac{1}{|t_{b0}|} \sum_{t_i \in t_{b0}} t_i[j], \tag{4.7}$$

,kde $|t_{b0}| + |t_{b1}| = D$ a $t_i[j]$ představuje j-tou hodnotu z měřené výkonové spotřeby t_i.

Poslední fázi je výpočet diferenciálního průběhu. Diferenciální průběh je získán rozdílem těchto průměrných průběhů. Diferenciální průběh lze zapsat následovně pro odhad klíče $j = 1 \ldots k$:

$$\Delta[j] = m_1^{'}[j] - m_0^{'}[j]. \tag{4.8}$$

Průměrné průběhy budou rozdílné pouze v časových okamžicích, na které má vliv b bit, jelikož vliv ostatních bitů na proudový průběh je zastoupen v obou podmnožinách stejně. Na základě toho lze pro diferenciální průběh v časových okamžicích j^*, kdy jsou prováděny operace s bitem b definovat diferenci:

$$E[t_i[j^*]|b = 1] - E[t_i[j^*]|b = 0] = \epsilon. \tag{4.9}$$

V časových okamžicích, $j \neq j^*$, kdy jsou proudové průběhy na bitu b nezávislé platí rovnice:

$$E[t_i[j^*]|b = 1] - E[t_i[j^*]|b = 0] = 0. \tag{4.10}$$

V případě, že je k dispozici dostatek naměřených průběhů proudové spotřeby, tak $m_0^{'}[j]$ a $m_1^{'}[j]$ konverguje k $E[t_i[j^*]|b = 0]$ a $E[t_i[j^*]|b = 1]$, pak lze psát pro $j = j^*$:

$$lim_{D\to\infty}\Delta[j] = lim_{D\to\infty}m_1^{'}[j] - lim_{D\to\infty}m_0^{'}[j] = \epsilon \tag{4.11}$$

Vztah 4.11 popisuje diferenciální průběh v případě správného rozdělení naměřených průběhů. Takovýto výpočet obsahuje několik zákmitů (diferencí) v místech působnosti b bitu, okolní získané zákmity mají výrazně menší úroveň. V případě, že jsme zvolili špatný odhad, tak vztah 4.11 neplatí a diferenciální průběh bude dosahovat nulové hodnoty. Správnost přiřazení b bitu tak rozhoduje o podobě diferenciálního průběhu.

Optimalizace DA se soustředí na fázi vypočítání matice hypotéz mezivýsledků, viz kapitola 3.2, kde útočník počítá mezivýsledky pro všechny odhady klíče. Berme v úvahu DPA založenou na rozdílu středních hodnot, kde jsou naměřené průběhy rozděleny pro každý odhad klíče do dvou podmnožin. Toto rozdělení proudových tras je nutno provést pro všechny odhady klíče. Aby metoda nebyla degradována na metodu „hrubé síly“ postupuje se po jednotlivých bajtech. Nejprve se vezme v potaz první bajt klíče, což odpovídá 256 možným variantám a následně pro zbylé bajty klíče, tedy pro celý klíč $256 \cdot 16$. Celkový počet nutných výpočtů je dán vztahem:

$$q = n \cdot i \cdot l, \tag{4.12}$$

kde n je počet otevřených textů (viz kapitola 3.2 cca 1000) a i je počet možných klíčů (256 pro první bajt) a l je délka klíče ($l = 16$ pro AES182). Pro realizaci druhé faze je zapotřebí dle vztahu 4.12:

$$q = n \cdot 256 \cdot 16 = n \cdot 4096 \tag{4.13}$$

výpočtů a pak následné rozdělení, průměrování a výpočet diference dle dle vztahů 4.7, 4.6, 4.8 a 4.11.

Předpokládejme tajný klíč vyjádřený bajtově $K = \{n, n, n, n, n, n, n, n\}$ pro $0 \leq n \leq 255$. Z matematického pohledu každý bajt klíče představuje skupinu osmi prvků obsahující dvě skupiny prvků (určitý počet jedniček a nul). Pak počet všech možných kombinací tajného klíče i (parametr ve vztahu 4.12) je dán vztahem pro permutace s opakováním:

$$i_{m_1,m_2,\ldots,m_k} = \frac{m!}{m_1! \cdot m_2! \cdot \cdots \cdot m_k!}. \tag{4.14}$$

Počet jedniček (počet první skupiny prvků) můžeme označit Hammingovou váhou klíče w $0 \leq w \leq 8$. Při znalosti Hammingovy váhy klíče můžeme zredukovat počet všech možných variant klíče dle vztahu 4.14. Tab. 4.1 zobrazuje počet kombinací klíče dle rovnice 4.14 v závislosti na Hammingově váze klíče.

Z tab. 4.1 je patrno, že při znalosti Hammingovy váhy klíče dojde k výrazné redukci počtu potřebných výpočtů. V nejhorším případě pro $w = 4$ je nutno provést 1120 výpočtů z teoreticky 4096 a dojde tedy k redukci o 73%. Tato optimalizace lineárně roste s počtem měření proudových průběhů. Dle kapitoly 3.2 je počet roven stovkám až tisícům tedy pro $n = 1000$ dle vztahu 4.13 dojde k ušetření 2976000

Tab. 4.1: Permutace tajného klíče v závislosti na Hammingově váze.

Hamingova váha w (max 8)	Permutace i (max 256)	Počet výpočtů (max 4096)
1	8	128
2	28	448
3	56	896
4	70	1120
5	64	1024
6	28	448
7	8	128
8	1	16

výpočtů. Hammingova váha tajného klíče lze zjistit přidáním pouze jednoho měření proudového průběhu a určení diferenciálního signálu, viz kapitola 4.2 a obr. 4.12 a obr. 4.3. Pokud jsou hodnoty otevřeného textu nulové, nezáleží jestli kryptografický modul odpovídá modelu HW nebo HD. Útočník si nejprve změří referenční proudový průběh T_{ref} odpovídající operaci `Add Round Key` v inicializační fázi algoritmu AES, kde hodnoty klíče budou nulové a hodnoty stavu také. Následně pokračuje v měření proudových průběhů pro náhodné texty (viz kapitola 3.2.2) a vypočte průměr pro prvních $n = 10$ měření:

$$T_{prum} = \frac{1}{n} \sum_{i=1}^{n} T_i. \tag{4.15}$$

Nakonec útočník vypočte rozdílový průběh dle následujícího vztahu:

$$T_{dif} = T_{prum} - T_{ref}. \tag{4.16}$$

Z rozdílového průběhu je schopen odečíst Hammingovu váhu klíče a určit tak možné varianty klíče dle vztahu 4.14 a redukovat tím počet potřebných výpočtů.

Předpokládejme příklad kdy kryptografický modul pracuje s tajným klíčem popsaným v kapitole 4.2, tedy že Hammingova váha bajtů je rovna $w(k) = 1$ až 8. Výsledkem měření popsaného předchozí kapitolou je diferenciální signál zobrazený na obr. 4.3. Na proudovém průběhu jsou patrné špičky odpovídající práci s daty. Důležitá je viditelnost zvyšující se Hammingovy váhy tajného klíče pro všechny hodnoty $w(k) = 1$ až 8. V tomto případě má klíč Hammingovu váhu 72.

Do tab. 4.2 jsou vepsány zjištěné hodnoty Hammingovy váhy tajného klíče pro jednotlivé bajty a počet nutných výpočtů útoku (tabulka je uvedena pro první polovinu klíče, druhá polovina je stejná). Z tabulky je patrno, že při znalosti Hammingovy váhy klíče dojde k redukci počtu potřebných výpočtů o 57120 z teoreticky 65536, tedy je nutných pouze 8416, což odpovídá 13%. V nejhorším případě, pro

Tab. 4.2: Zhodnocení naměřených dat.

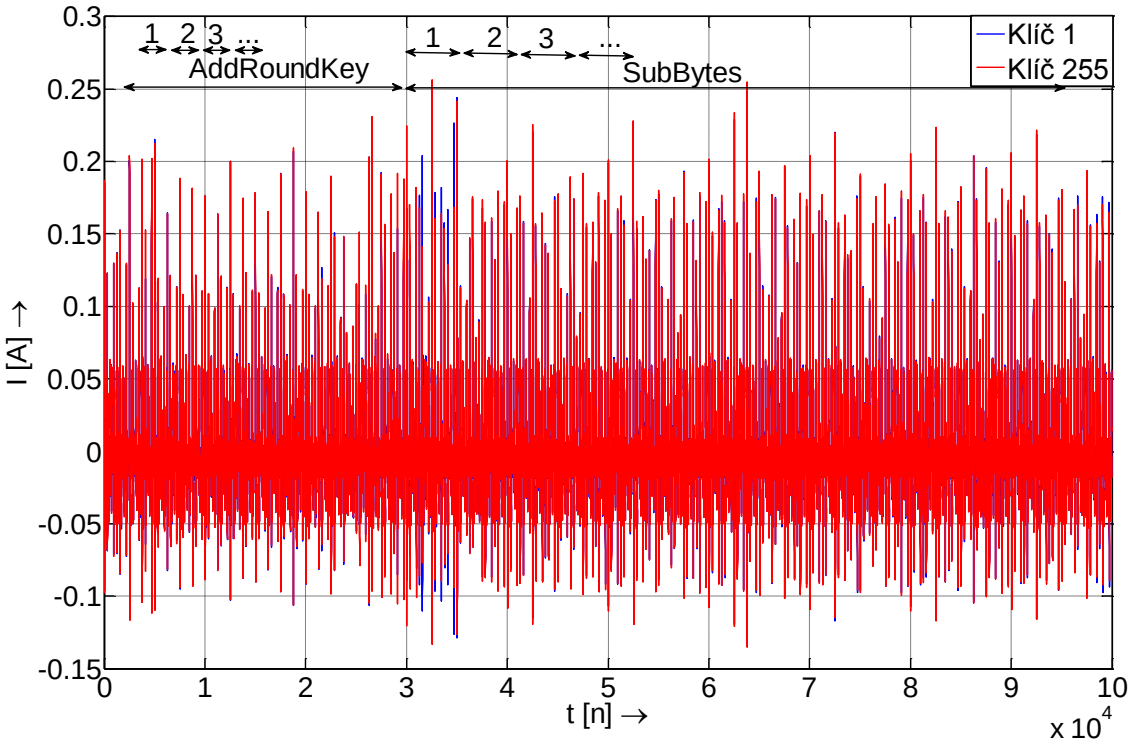
Hammingova váha **K**	Počet permutací	Počet výpočtů	Ušetřené výpočty
1	8	128	3968
2	28	448	3648
3	56	896	3200
4	70	1120	2976
5	64	1024	3072
6	28	448	3648
7	8	128	3968
8	1	16	4080
celkem			
36	263	4208	28560

$w = 4$, pro všechny bajty by bylo nutno provést 17920 výpočtů z teoreticky 65536 a dojde tedy k redukci o 73%. Tato optimalizace lineárně roste s počtem naměřených proudových průběhů. Dle kapitoly 3.2 je počet roven stovkám až tisícům tedy pro $n = 1000$ dle vztahu 4.13 dojde v tomto kokrétním případě k ušetření 57120000 výpočtů.

4.6 Proudová analýza využívající neuronové sítě

Hlavním cílem disertační práce je návrh a implementace analýzy postranním kanálem, která bude využívat neuronové sítě [123]. Analýza bude zaměřena jen na důležité operace algoritmu AES (operaci `AddRoudnKey` a operaci `SubBytes`), z kterých lze přímo určit tajný klíč algoritmu. Tento typ útoku proudovým postranním kanálem nebyl dosud publikován, jedná se tedy o zcela novou myšlenku. Předpokládá se, že k provedení útoku nebude zapotřebí velké množství měření proudové spotřeby jako například u DPA viz kapitola 3.2. Tato výhoda je stěžejní v porovnání s SPA a DPA a umožní v extrémním případě určení tajného klíče z jednoho proudového průběhu u algoritmů, které jsou odolné vůči SPA. Útočník tak může provést útok na modul, který se mu podařilo získat na krátký čas.

Navrhovaná metoda bude pracovat „per partes“, stejně jako většina analýz postranním kanálem, tedy cílem je určení tajného klíče po jednotlivých bajtech, kde tajný klíč je vyjádřený bajtově $K = \{k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8\}$ pro $0 \leq k_i \leq 255$ a pro kroky metody označené $i = 0$ až 8. Navrhovaná metoda tedy v prvním cyklu určí hodnotu prvního bajtu k_1 v dalším cyklu druhého k_2 a v posledním kroku hodnotu posledního bajtu tajného klíče k_8. Rozdíl mezi jednotlivými kroky bude



Obr. 4.19: Průběh proudové spotřeby operací `AddRoundKey` a `SubBytes`.

v rozdělení naměřené proudové spotřeby na části odpovídající jednotlivým bajtům tajného klíče.

Na obr. 4.19 je zobrazen naměřený proudový průběh odpovídající operacím `AddRoudnKey` a `SubBytes`, kdy došlo ke změně pouze v prvním bajtu tajného klíče k_1 a to z hodnoty 1 na hodnotu 255. Na průběhu jsou jasně patrny úseky, na které má vliv první bajt. Čísla označují jednotlivé části, které odpovídají jednotlivým krokům metody. V následujícím textu bude pojmem tajný klíč (K_{taj}) označen klíč uložený v kryptografickém modulu, na který je prováděn útok. Pojmem odhad klíče (K_{odh}) bude myšlena hodnota odhadu tajného klíče, kterou klasifikuje navrhovaná metoda. Cílem metody je, aby si na konci analýzy hodnota tajného klíče a odhadu klíče byla rovna. Měření proudových průběhů bylo prováděno pomocí metody měření, která byla důkladně otestováno viz kapitola 4.2 a byla založena na proudové sondě CT-6. Kompletní implementace navržené a testování byly provedeny v prostředí MATLAB. Toto prostředí poskytuje široké možnosti pro implementaci matematických metod, zpracování signálů, simulace a testování. Velkou výhodou je také dostupnost tzv. toolboxů, souborů funkcí a skriptů, které řeší konkrétní problém. Pro implementaci neuronových sítí byla použita neuronová síť vytvořená pomocí Netlab Neural Network toolbox. Autory tohoto toolboxu jsou Ian Nabney a Christopher Bishop z Aston University v Birminghamu. Toolbox je volně ke stažení [81].

Navržené obecné schéma metody

Dle výše popsaných skutečností byla navržena a realizována metoda využívající neuronové sítě sestávající se z několika fází:

- fáze přípravy vzorů pro tajné klíče k_i,
- fáze vytvoření a trénování neuronové sítě vytvořenými vzory,
- fáze útoku, určení odhadu klíče.

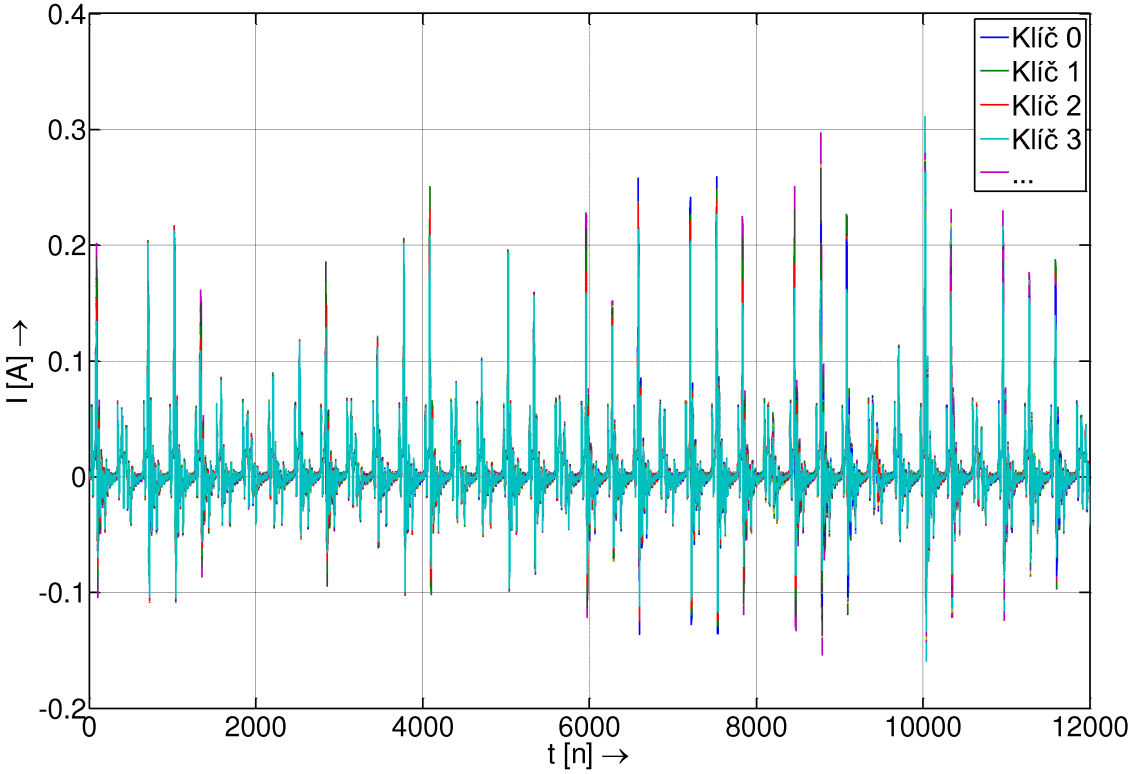
Provedení těchto fází umožní útočníkovi realizovat jeden krok analýzy, tedy určení jednoho bajtu tajného klíče k_i. V první fázi si útočník připraví trénovací množinu dat, kterými bude následně učit neuronovou síť. Útočník musí znát typ kryptografického modulu, na který hodlá útočit a musí stejný typ modulu mít zcela pod kontrolou (například plánuje-li útoky na čipovou kartu obsahující mikrokontrolér PIC16F84 musí tuto kartu vlastnit). Na kryptografický modul implementuje požadovaný kryptografický algoritmus a zaznamená proudové průběhy pro operace `AddRoundKey` a `SubBytes` pro všechny varianty tajného klíče k_i (256 možných variant).

Naměřené průběhy proudové spotřeby odpovídající práci s bajtem k_i použije útočník k natrénování neuronové sítě, která bude dané průběhy přiřazovat k hodnotám tajného klíče. Po úspěšném natrénování neuronové sítě může útočník pokračovat poslední fází a to fází útoku, kdy využije natrénovanou neuronovou síť k napadení kryptografického modulu.

V poslední fázi útoku útočník naměří proudovou spotřebu kryptografického modulu, na který útočí a přivede ji na vstup naučené neuronové sítě. Neuronová síť následně přiřadí proudovou spotřebu k odhadům tajného klíče a odhad klíče s největší pravděpodobností bude odpovídat hodnotě tajného klíče a tím dojde k určení hodnoty k_i. V následujícím textu budou popsány jednotlivé fáze navržené analýzy včetně implementace a dosažených výsledků.

Příprava vzorů

Cílem této fáze je získat trénovací vzory proudové spotřeby pro operaci `AddRoundKey` a `SubBytes` pro všechny varianty tajného klíče k_1 (256 možných variant). Do kryptografického modulu byl implementovány operace `AddRoundKey` a `SubBytes` dle předem ověřených znalostí o algoritmu AES a kryptografickém modulu (kapitoly 4.1 a A.1). Program pracoval ve smyčce a před započetím každé smyčky byla načtena data klíče k_i do paměti tak, aby smyčka vždy pracovala se stejnými vstupními proměnnými. Program umožňoval inkrementovat nebo dekrementovat hodnotu klíče a indikovat tuto operaci odesláním aktuální hodnoty klíče pomocí sériové linky do počítače. Synchronizační signál a komunikace s PC neměla na zkoumanou proudovou spotřebu vliv. Stěžejní části implementovaného programu jsou přiloženy v příloze



Obr. 4.20: Proudové vzory spotřeby pro všechny hodnoty klíče.

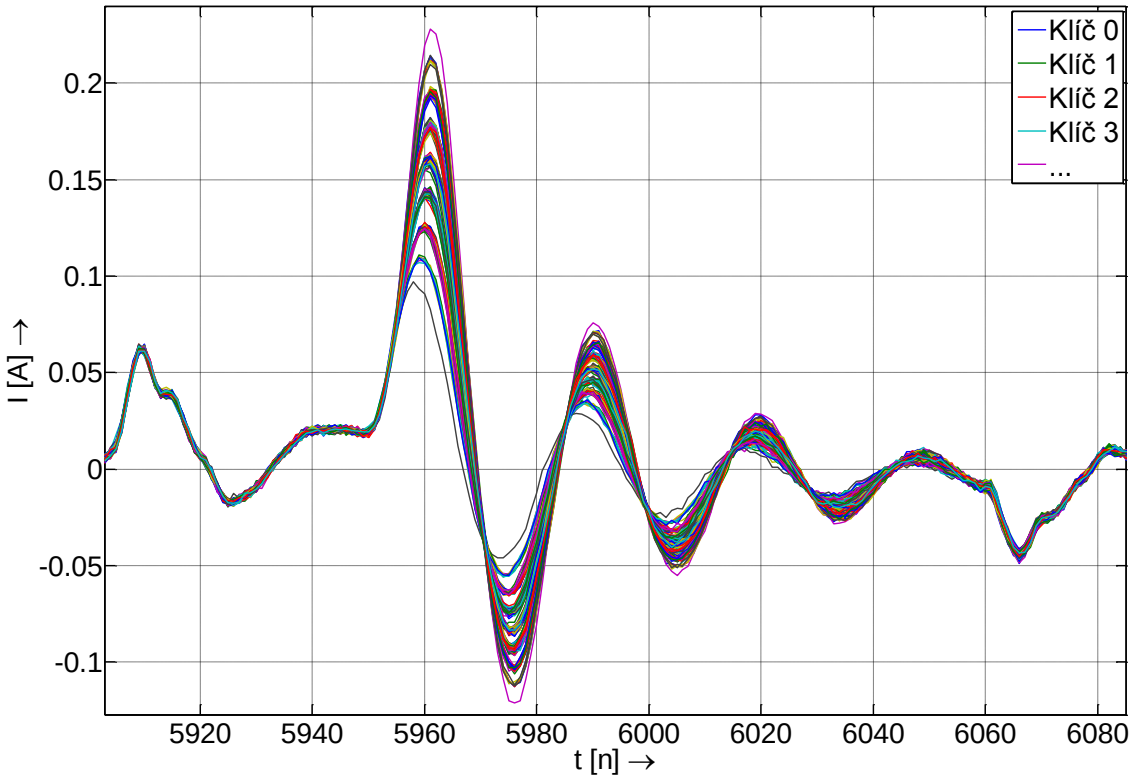
A.3. Stejně tak jak v předchozí kapitole vyjádříme operaci `AddRoundKey` pro přehlednost a jednoduchost v maticové podobě:

$$\mathbf{S}' = \mathbf{S} \otimes \mathbf{K}. \tag{4.17}$$

V průběhu měření byly hodnoty otevřeného textu $\mathbf{S}$ nastaveny na konstantní hodnoty. Hodnoty prvního bajtu tajného klíče $\mathbf{K}$ nabývaly postupně hodnotu 0 až 255 a zbylé hodnoty byly nulové. Matice tajného klíče a otevřeného textu vypadaly následovně (hexadecimální zápis):

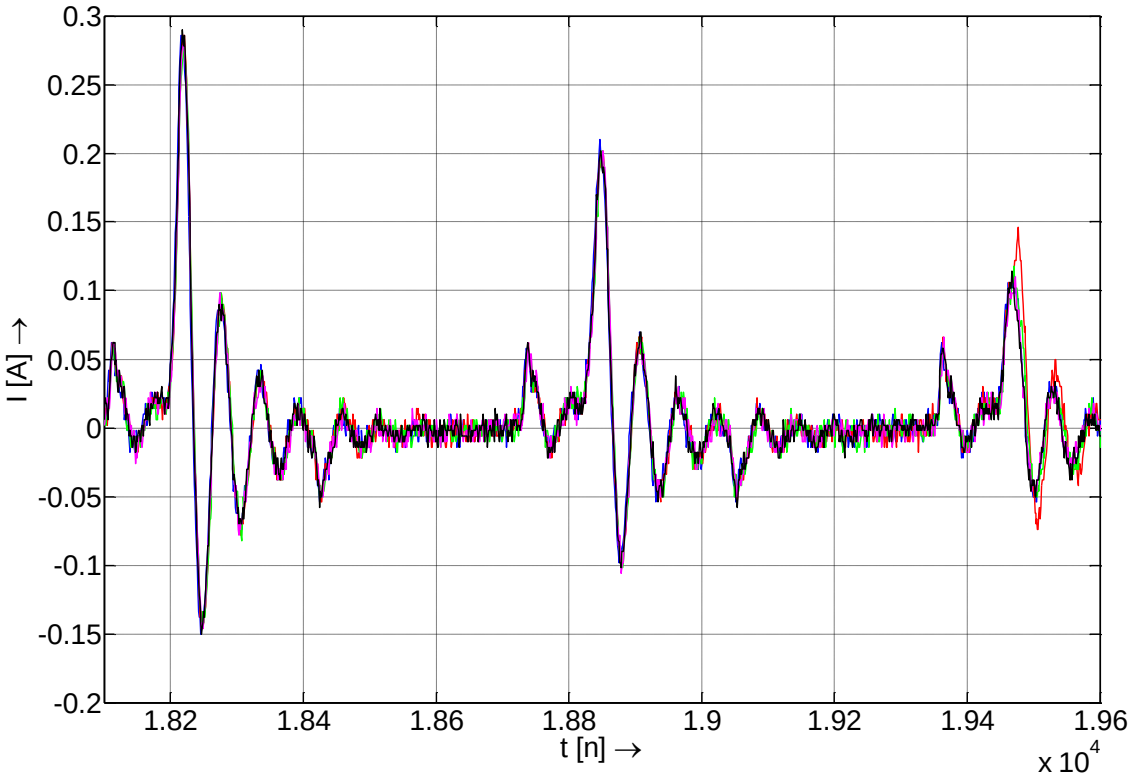
$$\mathbf{S} = \begin{pmatrix} 01 & 03 & 07 & 0F \\ 1F & 3F & 7F & FF \\ 01 & 03 & 07 & 0F \\ 1F & 3F & 7F & FF \end{pmatrix}, \mathbf{K} = \begin{pmatrix} 00 \ldots FF & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 \\ 00 & 00 & 00 & 00 \end{pmatrix}. \tag{4.18}$$

Obr. 4.19 zobrazuje proudové průběhy pro operace `AddRoundKey` a `SubBytes` pro hodnoty klíče 1 a 255. Proudové průběhy jsou si takřka identické s vyjímkou začátku průběhu, který odpovídá načtení registru a operaci `XOR` otevřeného textu s hodnotou tajného klíče a části pro čas $t = 35000$, která odpovídá operacím prováděných



Obr. 4.21: Detail proudové spotřeby pro všechny hodnoty klíče.

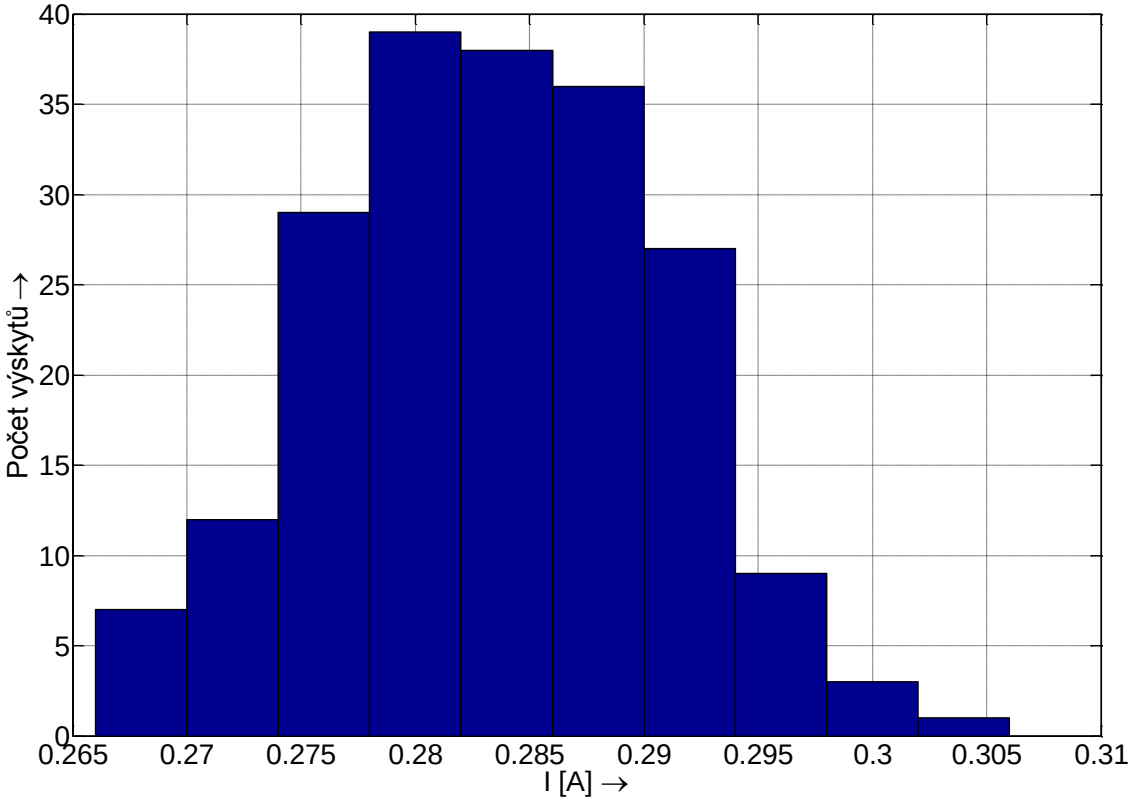
během substituce (viz A.3). Je zřejmé, že je zbytečné a značně neefektivní učit neuronovou síť na celé průběhy, a proto byly všechny průběhy redukovány na místa práce s prvním bajtem tajného klíče. Takto redukované a připravené proudové průběhy pro všechny hodnoty tajného klíče určených pro neuronovou síť zobrazuje obr. 4.20. Detail první proudové špičky je zobrazen na obr. 4.21 a je patrné, že proudové průběhy jsou rozděleny do několika skupin, které dle podrobnějšího zkoumání odpovídají HW tajného klíče. Neuronové sítě, které byly použity při klasifikaci akustického signálu [113], byly naučeny (natrénovány) na konkrétní průběhy akustických signálů. Tato metoda předpokládá dostatečné rozdíly mezi jednotlivými průběhy. U proudové analýzy je pravděpodobné, že tento postup povede k neúspěšné klasifikaci instrukcí a to ze dvou základních vlastností PA. První vlastností je, že proudové průběhy jednotlivých instrukcí jsou si velice podobné [11]. Druhou typickou vlastností je, že měříme-li výkonový průběh konkrétní instrukce opakovaně, průběhy nejsou zcela identické a to v důsledku změn pomocných registrů kryptografického modulu (čítač instrukcí atd.). Tato vlastnost bývá nazývána jako elektronický šum (Electronic Noise), který závažným způsobem ovlivňuje výsledky PA. Při přípravě vzorů i během fáze útoku je nutné snížit elektronický šum na minimální hodnotu, jinak bude docházet ke špatné klasifikaci tajného klíče. Výsledkem měření by byl



Obr. 4.22: Prvních 5 proudových průběhů operace ukládání dat do registru.

proudový průběh zařazen v chybné skupině, porovnání obrázku obr. 4.21 a obr. 4.22.

K určení elektronického šumu byla naměřena opakovaně proudová spotřeba kryptografického modulu zpracovávajícího stále stejná data. Kryptografický modul na experimentálním pracovišti zpracovával 200 krát datovou hodnotu 170, kterou ukládal do registru. Obr. 4.22 zobrazuje prvních 5 proudových průběhů operace. Průběhy jsou si velice podobné, protože jsou stále zpracovávány stejné instrukce a data. Rozdíly mezi průběhy způsobuje elektronický šum. S cílem získat lepší představu o rozložení bodů proudové spotřeby bude následující analýza zaměřena pouze na jeden bod z proudové spotřeby. Vezmeme v potaz například bod $n = 18219$ odpovídající proudové špičce. Obr. 4.23 zobrazuje vypočítaný histogram pro daný bod, který zobrazuje jak často byly jednotlivé hodnoty proudu naměřeny. Z obrázku je patrno, že největší výskyt bodů je pro hodnotou proudu kolem 280 mA a velmi malý výskyt pro hodnoty 305 a 265 mA. Pokud bude zobrazen histogram pro jakýkoli jiný bod proudové spotřeby, tvar histogramu bude vždy obdobný. Tvary histogramů indikují, že body naměřených průběhů se řídí normálním rozdělením. Normální rozdělení pravděpodobnosti s parametry μ a σ, pro $-\infty < \mu < \infty$ a $\sigma > 0$, je pro



Obr. 4.23: Histogram pro zvolený bod proudové spotřeby.

$-\infty < x < \infty$ definováno hustotou pravděpodobnosti ve tvaru Gaussovy funkce:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{4.19}$$

parametry μ a σ jsou nazývány střední hodnota a směrodatná odchylka. Mocninou směrodatné odchylky je rozptyl:

$$\begin{aligned} \mathrm{E}(X) &= \mu, \\ \mathrm{D}(X) &= \sigma^2. \end{aligned} \tag{4.20}$$

Normální rozdělení se většinou značí $\mathrm{N}(\mu, \sigma)$. V našem experimentu, proměnná X definuje proudovou spotřebu definovanou zvoleným bodem. V normálním rozdělení je střední hodnota nejpravděpodobnější výsledek měření, tzn. je to výsledek vyskytující se nejčastěji. Navíc z definice normálního rozdělení plyne, že většina výsledků experimentu se pohybuje blízko střední hodnoty, a proto můžeme určit $\mu = \mathrm{E}(X)$ jako průměr hodnoty $\bar{x}$. V uvedeném příkladu vychází průměrná hodnota $\bar{x} = 55,3\mathrm{mA}$, což odpovídá středu histogramu. Lze také vypočítat směrodatnou odchylku

$\sigma = 7,5$mA. Důležitou vlastností je, že $68,3\%$ hodnot se pohybuje v mezích směrodatné odchylky ($\pm\sigma$) a $95,5\%$ všech hodnot se pohybuje v rozmezí dvou směrodatných odchylek ($\pm 2\sigma$). V provedeném experimentu byly zpracovávána vždy stejná data a stejné instrukce, proto se dá předpokládat, že rozptyl výkonové spotřeby závislé na datech je nulový. Za tohoto předpokladu platí, že elektronický šum je rozložen dle normálního rozdělení s parametry $\mu = 0$mA a $\sigma = 7,5$mA. V praxi se elektronický šum řídí u většiny kryptografických zařízení dle normálního rozdělení se specifickou směrodatnou odchylkou pro každé zařízení. Dle výše popsaných poznatků chování elektronického šumu a také odborné literatury např. [67] je nejlepším způsobem snížení elektronického šumu opakované měření proudových spotřeb a následné vypočtení průměrné hodnoty. Proto byly proudové spotřeby pro různé hodnoty dat měřeny vícekrát a následně byl vypočítán průměrný průběh proudové spotřeby, s kterým se bude následně pracovat. Experimentálně bylo ověřeno, že optimální hodnota průměrování proudových průběhů je 16. Průběhy proudové spotřeby jsou funkcí s diskrétním časem, označme proudové průběhy odpovídající jednotlivým bajtům klíče $k_i[n] = f[n]$ pro $[n] = \{0, \ldots, t\}$ a každé měření je opakováno s-krát, kde $s = 16$. Potom průměrná spotřeba, která je použita jako vzorová data je definována:

$$\bar{k_i}[n] = \frac{1}{s} \sum_{j=0}^{s} k_i^j[n]. \tag{4.21}$$

U průběhů proudové spotřeby zobrazených na obr.4.20 a obr.4.21 je použito průměrování dle vztahu 4.21.

Fáze vytvoření a trénování neuronové sítě

Naměřené průběhy byly importovány a uloženy do matice $\mathbf{K_{vzor}}$ v programu MATLAB pro následné zpracování. Pro vytvoření neuronové sítě, jak již bylo řečeno, byl zvolen NETLAB Toolbox. Tato kapitola popisuje základní vlastnosti a implementaci neuronové sítě. Kompletní implementace je přiložena v příloze A.4.

Vytvořená neuronová síť v prostředí MATLAB je typická třívrstvá, jak popisuje kapitola 3.4. Struktura sítě je zobrazena na obr. 3.4. Metoda učení byla zvolen algoritmus využívající zpětné šíření chyby (Backpropagation), která patří k nejpoužívanějším principům učení neuronových sítí. Tato metoda je popsána následujícími kroky.

- **Krok 1:** Počáteční inicializace vah w_{ij} a prahů θ_i jednotlivých neuronů.
- **Krok 2:** Přivedení vstupního vektoru $\mathbf{X} = [x_1, \ldots, x_N]^T$ a definice požadované výstupní odezvy $\mathbf{D} = [d_1, \ldots, d_M]^T$.

- **Krok 3:** Výpočet aktuálního výstupu podle následujících vztahů:

$$y_l(t) = f_s(\sum_{k=1}^{N_2} w_{kl}^{''}(t)x_k^{''}(t) - \theta_l^{''}),\ 1 \leq l \leq M,\ \text{výstupní vrstva,} \quad (4.22)$$

$$x_k^{''}(t) = f_s(\sum_{j=1}^{N_1} w_{jk}^{'}(t)x_j^{'}(t) - \theta_k^{'}),\ 1 \leq k \leq N_2,\ \text{skrytá vrstva,} \quad (4.23)$$

$$x_j^{'}(t) = f_s(\sum_{i=1}^{N} w_{ij}(t)x_i(t) - \theta_j),\ 1 \leq j \leq N_1,\ \text{vstupní vrstva.} \quad (4.24)$$

  Výpočet platí pro třívrstvou neuronovou síť uvedenou na obr. 3.4.
- **Krok 4:** Adaptace vah a prahů dle následujících vtahů:

$$w_{ij}(t+1) = w_{ij}(t) + \eta\delta_j x_i,\ \text{popř.} \quad (4.25)$$

$$w_{ij}(t+1) = w_{ij}(t) + \eta\delta_j x_i + \alpha(w_{ij}(t) - w_{ij}(t-1)). \quad (4.26)$$

  Nastavení vah začíná u výstupních neuronů a postupuje rekurzivně směrem ke vstupním neuronům. V uvedených vztazích jsou w_{ij} váhy mezi i-tým skrytým neuronem popřípadě vstupním a uzlem j-tým v čase t. Výstup i-tého neuronu je označen $x_i^{'}$, η je koeficient učení, α je tzv. momentový koeficient a δ_j je chyba, pro kterou platí následující vztahy:

$$\delta_j = y_j(1-y_j)(d_j - y_j),\ \text{pro výstupní neurony,} \quad (4.27)$$

$$\delta_j = x_j^{'}(1-x_j^{'})(\sum \delta_k w_{jk}),\ \text{pro skryté neurony,} \quad (4.28)$$

  kde k se mění přes všechny neurony vrstvy, které následují za uzlem j.
- **Krok 5:** Opakování kroků 3 až 5, dokud chyba není menší než předem stanovená hodnota.

V následujících kapitolách při použití neuronové sítě bude brána v úvahu právě popsaná třívrstvá neuronová síť s metodou učení založeném na zpětném šíření chyby.

Vytvořená neuronová síť v prostředí MATLAB má tyto parametry: vstupní vrstva obsahuje stejný počet neuronů jako je počet vzorků v průběhu, tedy 3000. Výstupní vrstva klasifikuje vstup na jednotlivé klíče, tedy musí obsahovat 256 neuronů pro všechny kombinace klíče 0 až 255. Skrytá vrstva může mít libovolný počet neuronů v závislosti na složitosti řešeného problému. V implementaci je počet možno konfigurovat od 128 do 256 neuronů. S těmito počty bylo provedeno testování a dosahovalo se nejlepších výsledků. Typ aktivační funkce byl zvolen `logistic`, což odpovídá standardní sigmoidě. Následující text obsahuje nejdůležitější část programu implementace neuronové sítě. Uvedené řádky odpovídají postupně vytvoření, konfiguraci a následné trénování neuronové sítě.

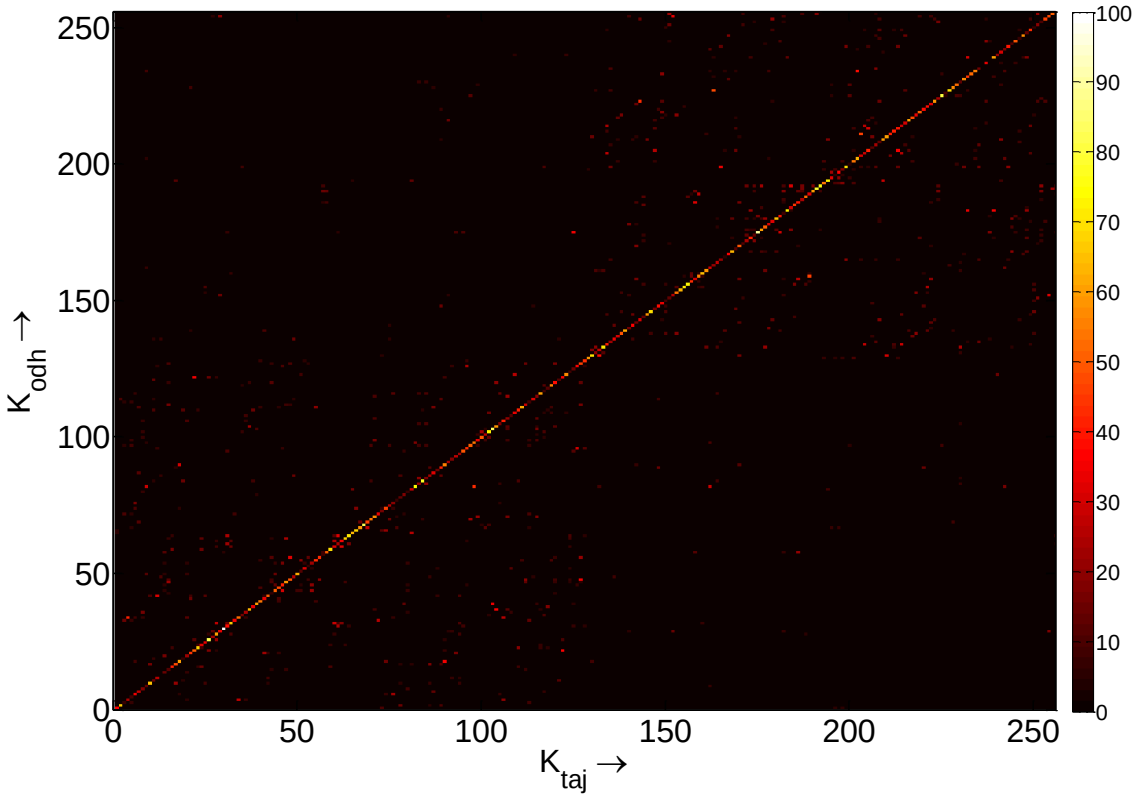
```
%Vytváření neuronové sítě
nn = mlp(pocet_vzorku, pocet_neuronu, pocet_mereni,'logistic');
%Nastavení parametrů neuronové sítě
options = zeros(1,18);              % Reset konfiguračního pole
options(1) = vypis;                 % Výpis chyby během učení
options(14) = pocet_iteraci;        % Počet trénovacích cyklů
%Trénování neuronové sítě
[nn, options] = netopt(nn, options, K_vzor, clas, 'scg');
```

Vzorová data uložená v $\mathbf{K_{vzor}}$ obsahují 256 průběhů a každý z nich má 3000 vzorků. Pro tyto průběhy je nutné vytvořit klasifikační matici, která určí správnou klasifikaci daného vstupu na příslušný klíč. Tato matice má rozměry 256×256 a jednotlivé řádky odpovídají naměřeným průběhům a příslušné sloupce přiřazují výsledný klíč hodnotou 1. Výsledkem je jednotková klasifikační matice, která jednotlivým průběhům pro klíče 0 až 255 přiřadí klíče 0 až 255. Po úspěšném natrénování neuronové sítě následuje fáze útoku.

Fáze útoku

Po úspěšném naučení je neuronová síť připravena k rozpoznávání dat. V reálném útoku by útočník naměřil proudové spotřeby kryptografického modulu odpovídající tajnému klíči a pokusil se izolovat operaci `AddRoundKey`. Tato část je považována za kritickou a je důležité, aby data sloužící k útoku byla stejně synchronizována jako data vzorová. Ideální metodou je vložení identického synchronizačního signálu jako při měření vzorových dat. Pokud tuto možnost útočník nemá nezbývá, než naměřit celý průběh proudové spotřeby kryptografického algoritmu, na který je prováděn útok a stejně, tak jak popisuje kapitola 4.1 následnou postupnou analýzu průběhu určit jednotlivé fáze algoritmu a synchronizovat operaci `AddRoundKey` na například pomocí první proudovou špičky. (práce s prvním bajtem tajného klíče). Důležitým faktorem je také stejná implementace kryptografického algoritmu, pokud by byl algoritmus implementován odlišným způsobem (jiné instrukce, jiná posloupnost instrukcí), výsledky klasifikace by byly chybné. Důležitým faktorem je také dodržení stejného postupu snížení elektronického šumu, tedy průměrování naměřených průběhů dle vztahu 4.21. Po korektním naměření proudové spotřeby kryptografického modulu je provedena klasifikace neuronovou sítí a je určen první bajt tajného klíče jako odhad klíče s největší pravděpodobností.

Pro ověření metody byly naměřeny a uloženy do matice **test** proudové průběhy odpovídající všem hodnotám tajného klíče k_1. Tato matice byla postupně klasifikována řádek po řádku neuronovou sítí. Tímto způsobem se získaly výsledky pro všechny hodnoty tajného klíče k_1 a představa do jaké míry je metoda úspěšná.



Obr. 4.24: Grafické znázornění kompletních výsledků klasifikace $\mathbf{V_{cel}}$.

Následující část zdrojového kódu zobrazuje klasifikaci proudových průběhů neuronovou sítí.

```
load neuronova_sit;          %nacteni neuronove site
test = test(:,3000:6000-1); %matice proudových prubehu
V_cel = [];
for i=1:256                  %cyklus klasifikujici jednotlive prubehy
V_cel = [V_cel; mlpfwd(nn,test(i,:))];
end
```

Výsledkem analýzy pro všechny proudové průběhy korespondující se všemi hodnotami tajného klíče byla matice $\mathbf{V_{cel}}$ o rozměrech 255×255. Hodnota indexu řádku odpovídala hodnotě tajného klíče, pro který byla měřena proudová spotřeba a index sloupce představoval odhad klíče přiřazeného neuronovou sítí. Neuronová síť přiřadila každému průběhu proudové spotřeby vektor obsahují pravděpodobnosti pro jednotlivé odhady klíče (řádek matice $\mathbf{V_{cel}}$). Celkové výsledky klasifikace jsou graficky znázorněny na obr. 4.24 a pro lepší představu je část výsledné matice číselně zapsána do tab. 4.3. Z tabulky je patrno, že neuronová síť klasifikovala například

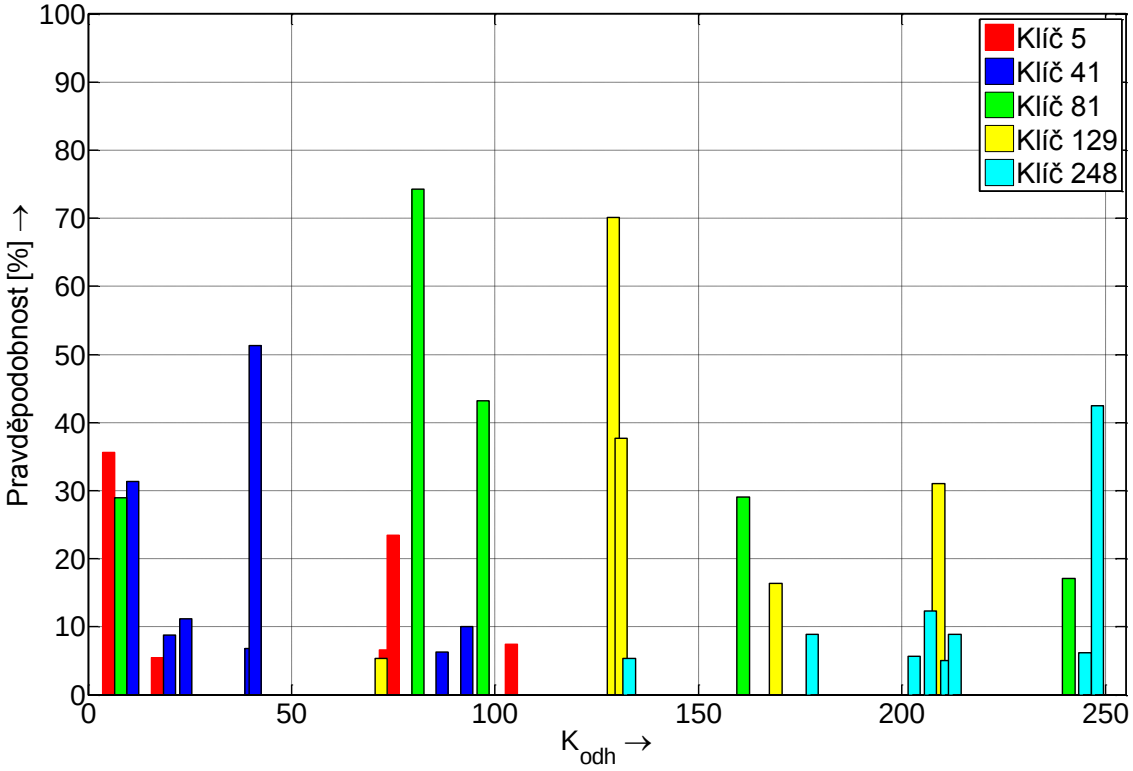
Tab. 4.3: Výsledky analýzy - část matice $\mathbf{V_{cel}}$.

		Pravděpodobnost odhadu klíče K_{odh}						
		0	1	2	3	4	5	6
Hodnota tajného klíče K_{taj}	⋮	...	...	...	...	...	...	...
	6	0,00%	0,00%	0,00%	0,00%	0,03%	0,00%	27,21%
	5	0,00%	0,00%	0,08%	0,00%	0,00%	35,61%	0,00%
	4	0,00%	0,00%	0,00%	0,00%	7,91%	0,00%	0,00%
	3	0,00%	0,00%	0,00%	23,79%	0,00%	0,00%	0,00%
	2	0,00%	0,00%	6,44%	0,00%	6,98%	0,00%	0,00%
	1	0,00%	66,42%	0,00%	0,00%	0,00%	0,00%	0,00%
	0	36,77%	0,00%	0,00%	0,00%	0,00%	0,00%	1,37%

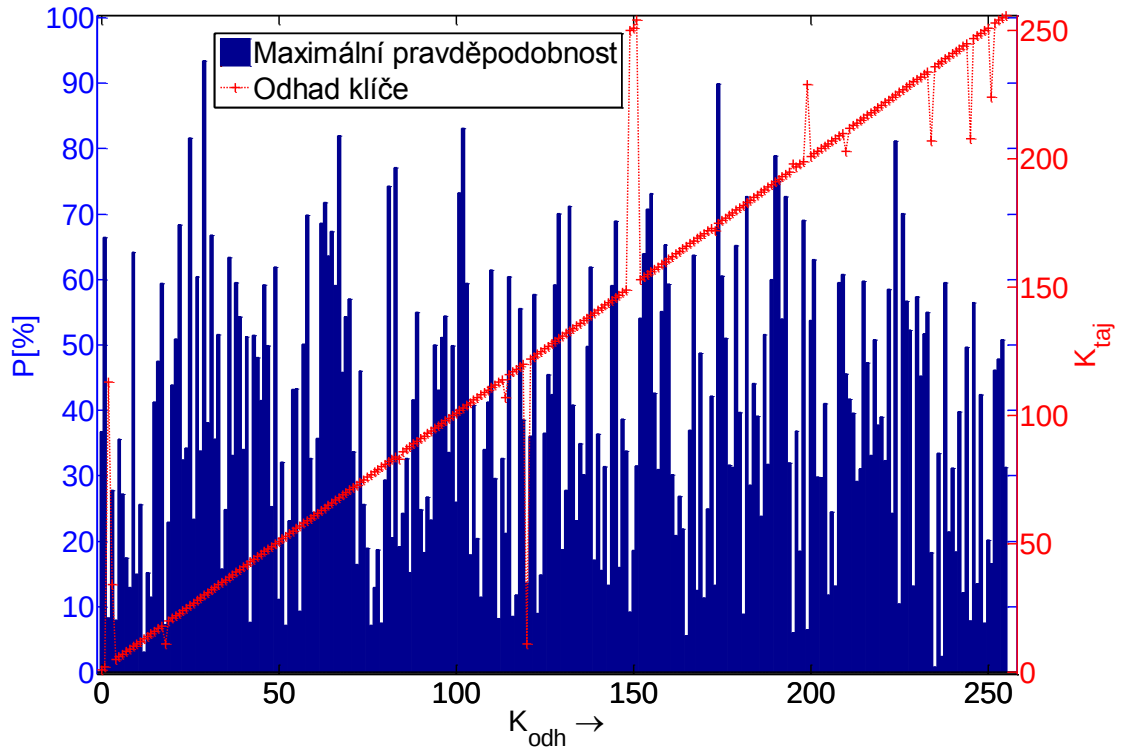
proudovou spotřebu pro tajný klíč s hodnotou 0 s pravděpodobností $36,77\%$ pro odhad klíče 0 a pro proudový průběh odpovídající hodnotě tajného klíče 1 klasifikovala odhad klíče 1 s pravděpodobností $66,42\%$.

Pro získaní lepší představy o výsledcích klasifikace neuronové sítě a rozložení pravděpodobnosti odhadů klíčů jsou na obr. 4.25 zobrazeny výsledky klasifikace pro 5 náhodně vybraných proudových průběhů korespondující s pěti hodnotami tajných klíčů. Na ose `x` jsou zobrazeny odhady klíče, tzn. výstup neuronové sítě a osa `y` udává s jakou pravděpodobností se odhad klíče rovná tajnému. Barevně jsou vyznačeny jednotlivé průběhy odpovídající tajnému klíči, tedy byly vybrány proudové průběhy pro hodnoty tajného klíče 5, 41, 81, 129 a 248 (dekadický zápis). Z obr.4.25 je patrno, že pravděpodobnost odhadu klíče 5 pro proudový průběh s hodnotu tajného klíče 5 byla 35% a ostatní pravděpodobnosti určené neuronovou sítí byly: 5% pro odhad klíče 18, 6% pro odhad klíče 74, 23% pro odhad klíče 76 a 7% pro odhad klíče 105. Analogicky lze vyčíst rozložení pravděpodobností pro ostatní vybrané hodnoty tajného klíče. Zobrazené hodnoty korespondují s maticí výsledků $\mathbf{V_{cel}}$ a tab. 4.3. Pro náhodně vybrané hodnoty tajného klíče největší pravděpodobnost odhadu klíče odpovídala vždy hodnotě tajného klíče. Z těchto dílčích výsledků plyne dobrá funkčnost metody.

Pro podrobnější analýzu funkčnosti metody zobrazuje obr. 4.26 **maximální hodnoty pravděpodobností** odhadu klíče pro jednotlivé hodnoty tajného klíče. Graf ukazuje jaký odhad klíče byl klasifikován neuronovou sítí s největší pravděpodobností pro konkrétní proudový průběh korespondující s hodnotou tajného klíče. Graf je zobrazen se dvěma osami `y` a to pro lepší přehlednost a názornost. Osa `x` představuje odhady klíčů a modrá osa `y` příslušné maximální pravděpodobnosti. Červená osa `y` koresponduje s hodnotou tajného klíče.



Obr. 4.25: Výsledky klasifikace pro 5 náhodně vybraných klíčů.



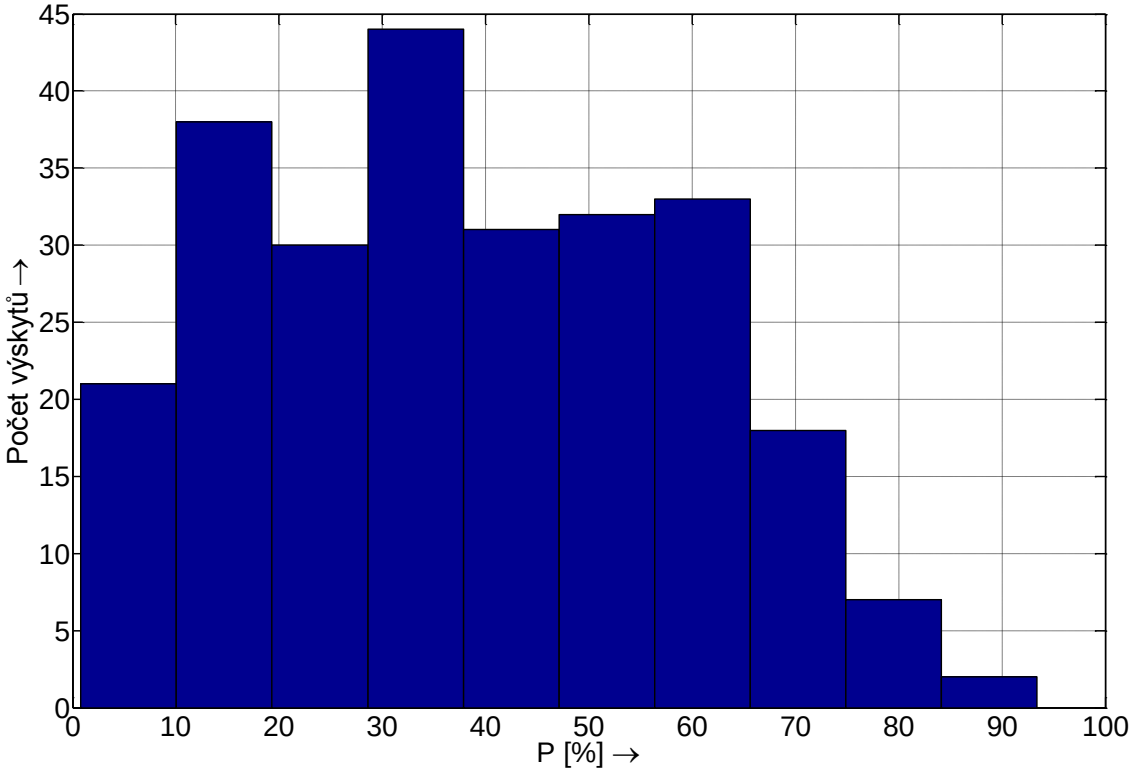
Obr. 4.26: Maximální hodnoty pravděpodobnosti a určené odhady klíče.

Z požadavků metody je zřejmé, aby odhad klíče byl roven tajnému klíči, tedy v ideálním případě platí funkce $K_{odh} = K_{taj}$. Průběh funkce $K_{odh} = K_{taj}$ je markantní na první pohled. Hladký průběh funkce ruší body, které indikují chyby klasifikace. Jedná se o odhady klíče, které byly chybně klasifikovány, tedy kdy vybraný odhad klíče s největší pravděpodobností nekorespondoval s hodnotou tajného klíče v kryptografickém modulu ($K_{odh} \neq K_{taj}$). Seznam všech chybně klasifikovaných tajných klíčů je zapsán do tab. 4.4. Z naměřeného souboru, který byl určen pro ověření metody, neuronová síť přiřadila šestnáctkrát špatný odhad klíče. Ze všech možných testovaných variant tajného klíče 256 to odpovídá $6,27\%$ chybných klasifikací. Navržená metoda určila správnou hodnotu tajného klíče v **93,72%** případů.

Tab. 4.4: Chybně určené odhady klíčů.

K_{taj}	2	3	18	84	114	120	149	150
K_{odh}	112	33	10	82	106	10	249	250
P[%]	8,32	27,77	7,31	19,15	21,23	13,60	9,20	18,59
K_{taj}	151	173	195	199	210	234	245	251
K_{odh}	253	171	197	228	202	206	207	223
P[%]	31,57	13,23	6,02	6,44	45,59	18,27	7,82	16,60

Při opětovném experimentálním testování metody se dosahovalo obdobných výsledků, kdy metoda dosahoval okolo 85 až 90% úspěšnosti správné klasifikace s chybami, které se vyskytovaly u odhadů klíčů u nichž je maximální hodnota pravděpodobnosti nízká. Tento poznatek potvrzují data v tab. 4.4, medián pravděpodobností, které vedly ke špatné klasifikaci je zde 15% a průměrná hodnota 17%. Při klasifikaci je tedy požadavek na co největší pravděpodobnost u správného odhadu klíče. Z obr. 4.26, který zobrazuje maximální hodnoty pravděpodobností je patrné, že maximální pravděpodobnosti 14%, 18% a 20% nejsou vyjímkou. Proto bylo přistoupeno k další analýze výsledků a obr. 4.27 zobrazuje histogram všech maximálních pravděpodobností klasifikace. Z histogramu lze vyčíst, že pravděpodobnosti do 10% se vyskytují dvacetjednakrát a pravděpodobnosti 10% až 20% se vyskytují třicetosmkrát, což součtu odpovídá 23%. Pravděpodobnosti 20% až 60% se vyskytují nejčastěji a to stosedmdesátkrát, což odpovídá 66% z cekového počtu. Maximální pravděpodobnosti 70% až 90% se vyskytují jen 27 krát, což z celkového počtu klíčů odpovídá deseti procentům. Celkový počet potencionálně náchylných klíčů k chybné klasifikaci je asi 23%, což by znamenalo, že navržená metoda by pracovala asi s **80%** úspěšností. Z výše popsané analýzy výsledků klasifikace navržené metody byla navržena optimalizace, která umožní snížení chybné klasifikace a to tím, že se pokusí zvýšit maximální pravděpodobnost klasifikace a snížit pravděpodobnost chybné klasifikace.

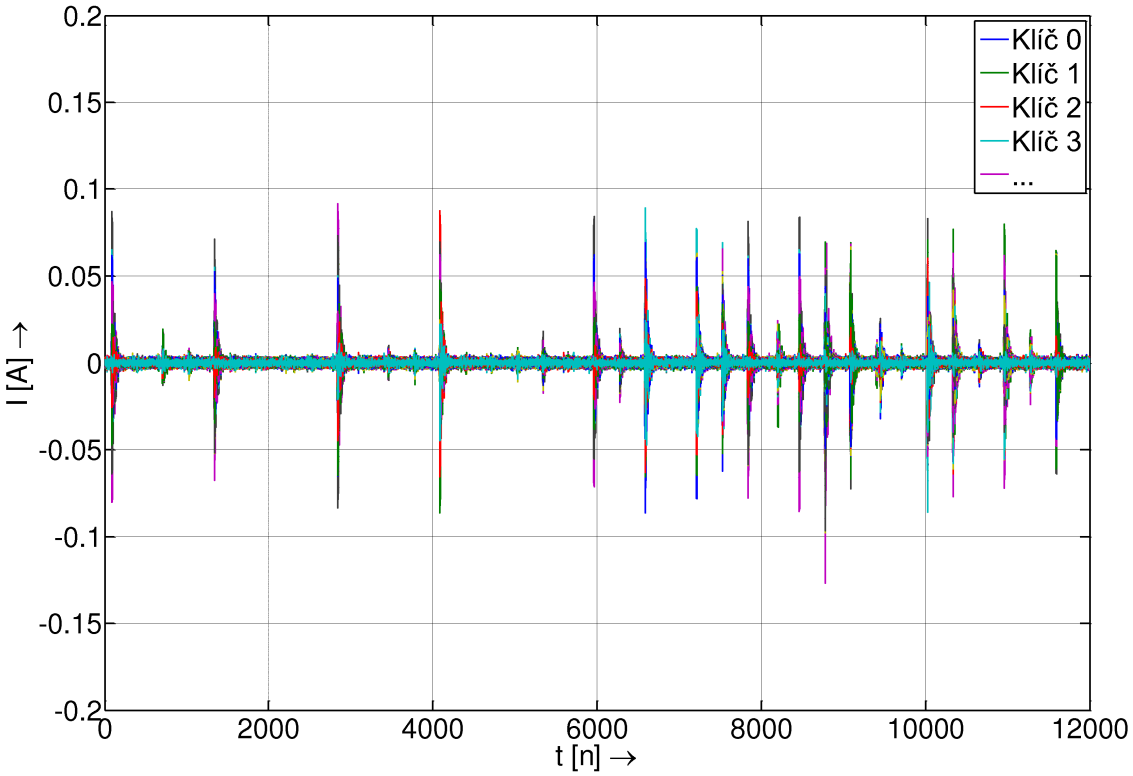


Obr. 4.27: Histogram maximálních hodnot pravděpodobností.

Optimalizace navržené metody

Cílem optimalizace je získat trénovací vzory proudové spotřeby pro všechny varianty tajného klíče k_i s většími diferencemi pro jednotlivé průběhy. Zvýšení diference mezi jednotlivými průběhy umožní přesnější klasifikaci. Z obr. 4.20 je patrné, že proudové průběhy jsou si velmi podobné a liší se v místech práce s registry. Pro zvýšení diference mezi průběhy byla použita metoda, která byla implementována k zvýraznění proudových průběhu jednotlivých instrukcí mikroprocesoru. Metoda byla navržena a testována v pojednání o disertační práci, ale jen pro několik průběhů tří instrukcí mikroprocesoru, konkrétně se jednalo o instrukci `XOR`, `SWAP` a `AND`. Následně probíhalo i testování této metody s neuronovými sítěmi [118], které podnítilo návrh optimalizace.

Samotné zvýšení diferencí je docíleno předzpracováním naměřených dat. Naměřené průběhy proudové spotřeby ve fázi přípravy vzorů jsou zpracovány následujícím způsobem. Nejprve je vypočten průměrný průběh proudové spotřeby pro všechny hodnoty tajného klíče. Průběhy proudové spotřeby jsou funkcí s diskrétním časem, označme proudové průběhy odpovídající jednotlivým bajtům klíče $k_i[n] = f[n]$ pro $[n] = \{0, \ldots, t\}$ a každý bajt může nabývat hodnotu 0 až 255, tedy 256 průběhů pro



Obr. 4.28: Průběh proudové spotřeby `AddRoundKey` pro všechny hodnoty klíče.

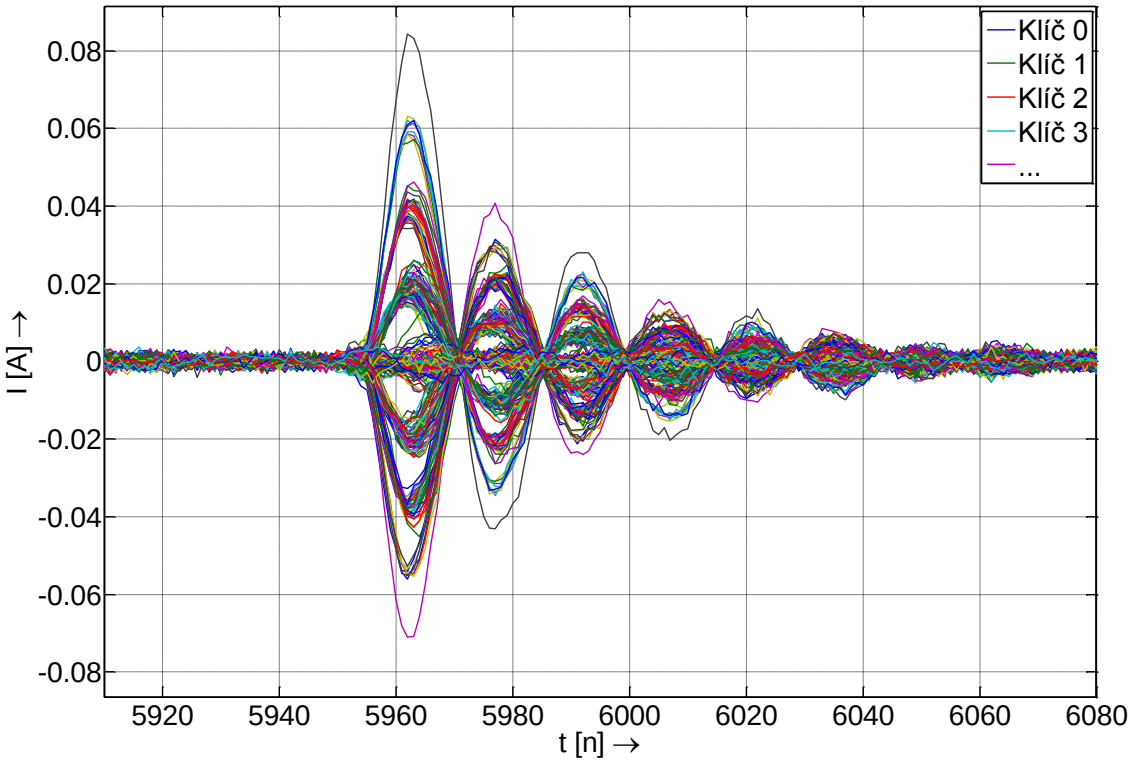
první bajt. Průběh průměrné proudové spotřeby je definován:

$$\bar{K}_i[n] = \frac{1}{256} \sum_{j=0}^{256} k_i^j[n]. \tag{4.29}$$

Následně jsou vypočítány učící vzory jako rozdíly $\bar{K}_i$ a proudových spotřeb pro jednotlivé tajné klíče. Ve skutečnosti jsou brány opět průměry proudových spotřeb a to kvůli snížení elektronického šumu. Celkový výpočet vzorů tedy můžeme vyjádřit:

$$V_I = \bar{P} - \frac{1}{s} \sum_{l=0}^{s} k_i^l[n] = \frac{1}{256} \sum_{j=0}^{256} k_i^j[n] - \frac{1}{s} \sum_{l=0}^{s} k_i^l[n], \tag{4.30}$$

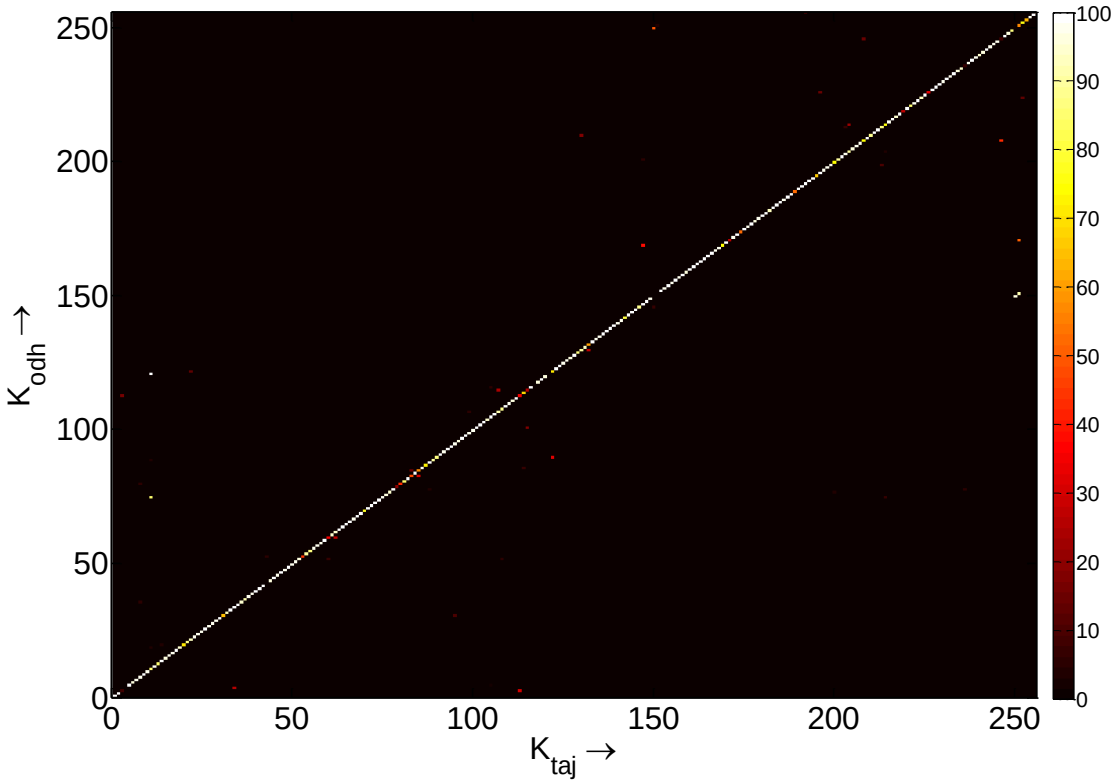
kde s je počet měření jednotlivých proudových průběhů (16). Tímto výpočtem se docílí požadované zvětšení diference mezi jednotlivými proudovými průběhy. Obr. 4.28 zobrazuje kompletní sadu naměřených a následně početně upravených proudových průběhů. Z porovnání průběhů 4.20 a 4.28 je zřejmé, že jsou zobrazeny jen diference mezi jednotlivými průběhy. Stejně tak, jak v předchozím případě, je zobrazen detail proudové špičky na obr. 4.29. Dle předpokladu jsou proudové průběhy rozděleny do skupin a to i v záporných hodnotách. Vytvoření neuronové sítě a učení probíhalo stejným způsobem jak v předchozí kapitole. Pro ověření metody byly



Obr. 4.29: Průběh proudové spotřeby `AddRoundKey` pro všechny hodnoty klíče.

opět použity naměřené proudové průběhy odpovídající všem hodnotám tajného klíče a následně byly tyto průběhy analyzovány neuronovou sítí. Z důvodu porovnání metod byla použita stejná sada měření jak v předchozí kapitole. Tímto způsobem se získaly opět výsledky pro všechny možné hodnoty klíče a představa do jaké míry je metoda úspěšná.

Výsledkem analýzy pro všechny hodnoty klíče byla matice $\mathbf{VD_{cel}}$ o rozměrech 256×256. Hodnota indexu řádku odpovídala hodnotě tajného klíče, pro který byla měřena proudová spotřeba a index sloupce představoval odhad klíče přiřazeného neuronovou sítí. Část výsledné matice je zobrazena v tab. 4.5 a celá matice je graficky zobrazena na obr.u 4.30. Z tabulky je patrno, že neuronová síť klasifikovala proudovou spotřebu pro tajný klíč s hodnotou 0 s pravděpodobností $96,00\%$ pro odhad klíče 0 a pro proudový průběh odpovídající hodnotě tajného klíče 1 klasifikovala odhad klíče 1 s pravděpodobností $99,87\%$. Z porovnání části výsledků pro obě metody zobrazených v tab. 4.3 a 4.5 je patrné navýšení pravděpodobnosti pro správné odhady klíče. Například pro správné odhady klíče 0 a 1 byla pravděpodobnost navýšena z $36,77\%$ a $66,42\%$ na hodnoty $96,00\%$ a $99,87\%$. Z porovnání obrázku obr. 4.24 a 4.30 je také patrno na první pohled markantní zlepšení výsledků klasifikace a funkce $K_{odh} = K_{taj}$ je tvořena hodnotami pravděpodobnosti mezi 90% a



Obr. 4.30: Grafické znázornění kompletních výsledků klasifikace $\mathbf{VD_{cel}}$.

100%. Z obrázků je také patrné snížení alternativních variant klasifikace, tedy absence rovnoběžných úseček s funkcí $K_{odh} = K_{taj}$, které jsou jasně patrné na obr. 4.24. Z těchto dílčích výsledků je patrné zlepšení klasifikace, ale je nezbytné zhodnotit všechny výsledky a všechny pravděpodobnosti matice $\mathbf{VD_{cel}}$, jestli nedošlo ke zvýšení pravděpodobností u nesprávných odhadů.

Pro ověření o změnách ve výsledcích klasifikace neuronové sítě je zobrazen obr. 4.31, který udává výsledky klasifikace pro stejně vybraných pět proudových spotřeb kryptografického modulu korespondující s pěti hodnotami tajného klíče jako v předchozí kapitole. Na ose `x` jsou zobrazeny odhady klíče, tzn. výstup neuronové sítě a osa `y` koresponduje s jakou pravděpodobností se odhad klíče rovná tajnému. Barevně jsou zobrazeny jednotlivé průběhy odpovídající tajnému klíči. Z porovnání obrázků 4.31 a 4.25 je na první pohled zřejmé, že došlo ke zlepšení klasifikace. Například pro tajný klíč 5 byl správný odhad tajného klíče upraven z 35% na 96% a ostatní varianty tajného klíče byly zcela potlačeny. Tato žádaná vlastnost, tedy potlačení potencionálních možných variant klíče se potvrdila i u ostatních 3 tajných klíčů. U tajného klíče 129 byly alternativní varianty kromě jedné také potlačeny, ale byla také zvýšena maximální pravděpodobnost z 70% na 90%.

Tab. 4.5: Výsledky analýzy - část matice $\mathbf{VD_{cel}}$.

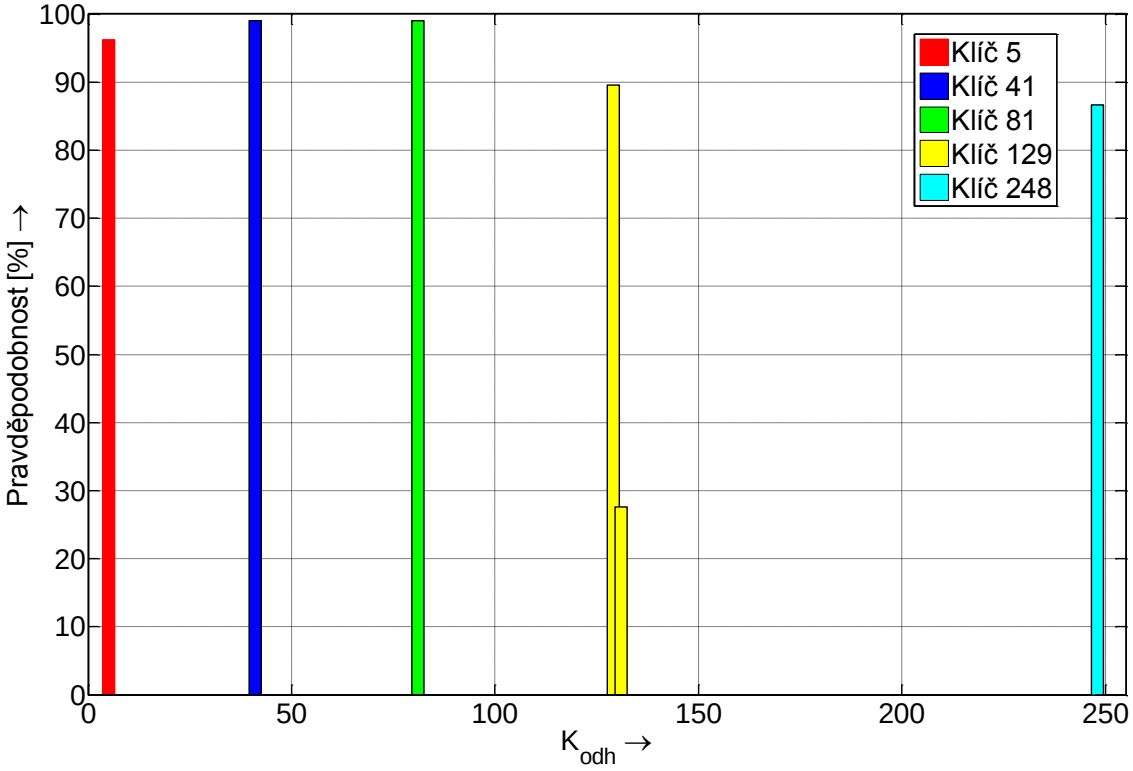
		Pravděpodobnost odhadu klíče K_{odh}						
		0	1	2	3	4	5	6
Hodnota tajného klíče K_{taj}	⋮	...	...	...	...	...	...	...
	6	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	98,39%
	5	0,00%	0,00%	0,00%	0,00%	0,00%	96,24%	0,00%
	4	0,00%	0,00%	0,00%	0,00%	99,09%	0,00%	0,00%
	3	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
	2	0,00%	0,00%	9,28%	0,00%	0,00%	0,00%	0,00%
	1	0,00%	99,87%	0,00%	0,00%	0,00%	0,00%	0,00%
	0	96,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%

Obr. 4.32 ukazuje maximální hodnoty pravděpodobností odhadů klíče pro jednotlivé hodnoty tajného klíče pro optimalizovanou metodu. Graf je zobrazen se dvěma osami stejně jako při první implementaci metody (viz graf na obr. 4.26). Z porovnání výsledků pro obě metody (obr. 4.26 a obr. 4.32) je na první pohled patrné požadované zvýšení maximální pravděpodobnosti u všech odhadů klíčů. Průběh funkce $K_{odh} = K_{taj}$ je takřka hladký a obsahuje jen devět chybně klasifikovaných klíčů, tzn. snížení počtu chyb z 16 na 9, to odpovídá zlepšení o 43%. Z těchto výsledků se jasně prokázala funkčnost a vhodnost předzpracování naměřených proudových průběhů pro klasifikaci neuronovou sítí. Navržená a optimalizovaná metoda klasifikovala devět odhadů klíčů chybně, všechny hodnoty jsou uvedeny v následující tab. 4.6.

Tab. 4.6: Chybně určené odhady klíčů.

K_{taj}	2	3	116	120	149	150	170	247	249
K_{odh}	112	33	118	10	249	250	250	207	149
P[%]	29,75	26,51	0,11	99,62	95,43	92,12	50,84	15,47	48,48

Z výsledku klasifikace se potvrdilo, že chyby se vyskytly opět u odhadů klíčů, které byly klasifikovány s nižší pravděpodobností. Uvedená metoda způsobila i zvýšení pravděpodobnosti u chybně klasifikovaných klíčů a to na průměrnou hodnotu 50%. Z celého souboru naměřených proudových průběhů neuronová síť klasifikovala devět odhadů chybně. Ze všech možných testovaných variant tajného klíče to odpovídá $3,5\%$ chybných klasifikací. Navržená metoda určila hodnotu tajného klíče v **96,5%** případů. Při opětovném testování dosahovala optimalizovaná metoda obdobných výsledků klasifikace a to **95 - 98%**.

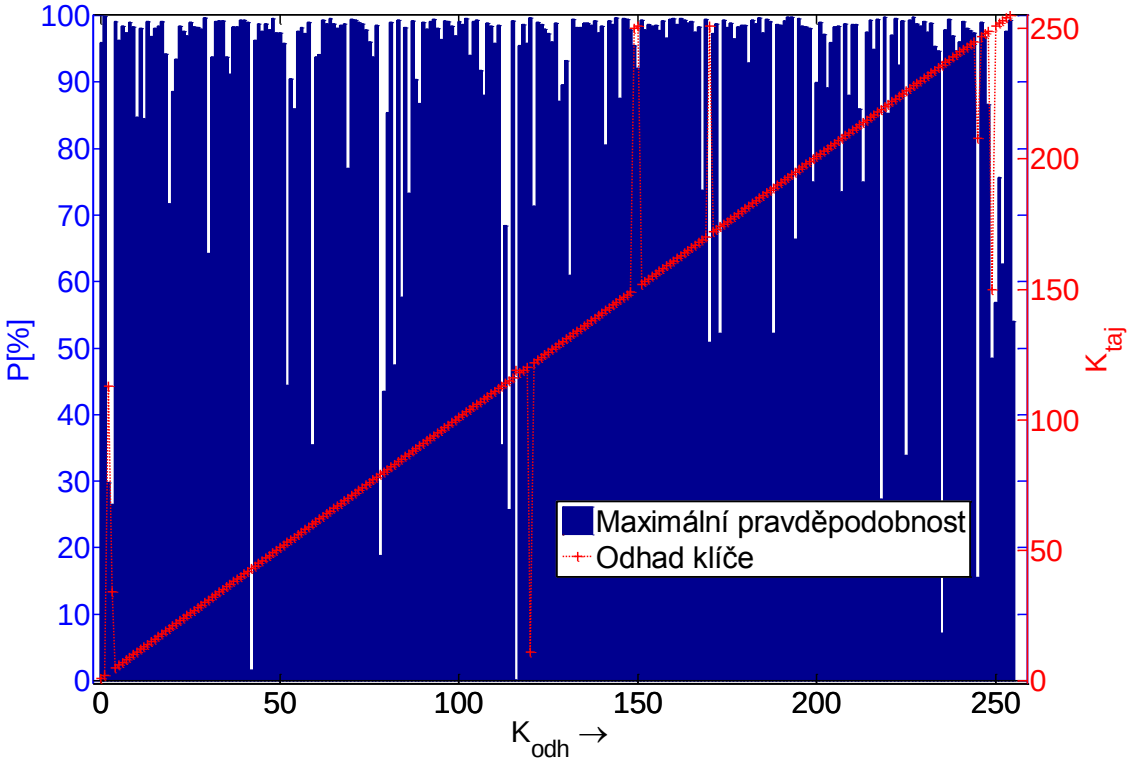


Obr. 4.31: Výsledky klasifikace pro 5 náhodně vybraných klíčů.

Pro detailnější analýzu maximálních pravděpodobností obr. 4.33 zobrazuje histogram všech maximálních pravděpodobností klasifikace optimalizované metody. Z histogramu lze vyčíst, že pravděpodobnosti od 10% do 70% se vyskytují každá jen pět krát. Pravděpodobnosti 70% až 80% se vyskytují desetkrát a patnáctkrát a největší zastoupení ve vybraných maximálních pravděpodobnostech mají pravděpodobnosti 90% až 100%, které se vyskytují dvěstěpětkrát. Průběh histogramu opět potvrzuje zvýšení maximálních pravděpodobností, tedy zvýšení počtu výskytů pravděpodobností nad 90%. Z celkového počtu 256 testovaných klíčů to odpovídá 80% kíčů klasifikovaných s pravděpodobností nad 90%. Celkový počet potencionálně náchylných klíčů k chybné klasifikaci je snížen po optimalizaci z cca 20% na 5%.

Opakovatelnost měření a zhodnocení metody

Pro analýzu opakovatelnosti a realizovatelnosti metody byl naměřen větší soubor proudových spotřeb kryptografického modulu a metoda byla otestována. Cílem bylo ověřit zda i pro opakované měření proudových spotřeb dojde ke správné klasifikaci tajného klíče. Bylo naměřeno 2560 průběhů proudové spotřeby, odpovídající všem hodnotám tajného klíče a tyto průběhy nebyly měřeny postupně, ale náhodně v jiné dny (z důvodu eliminace vlivu metody měření). Pro každou hodnotu tajného



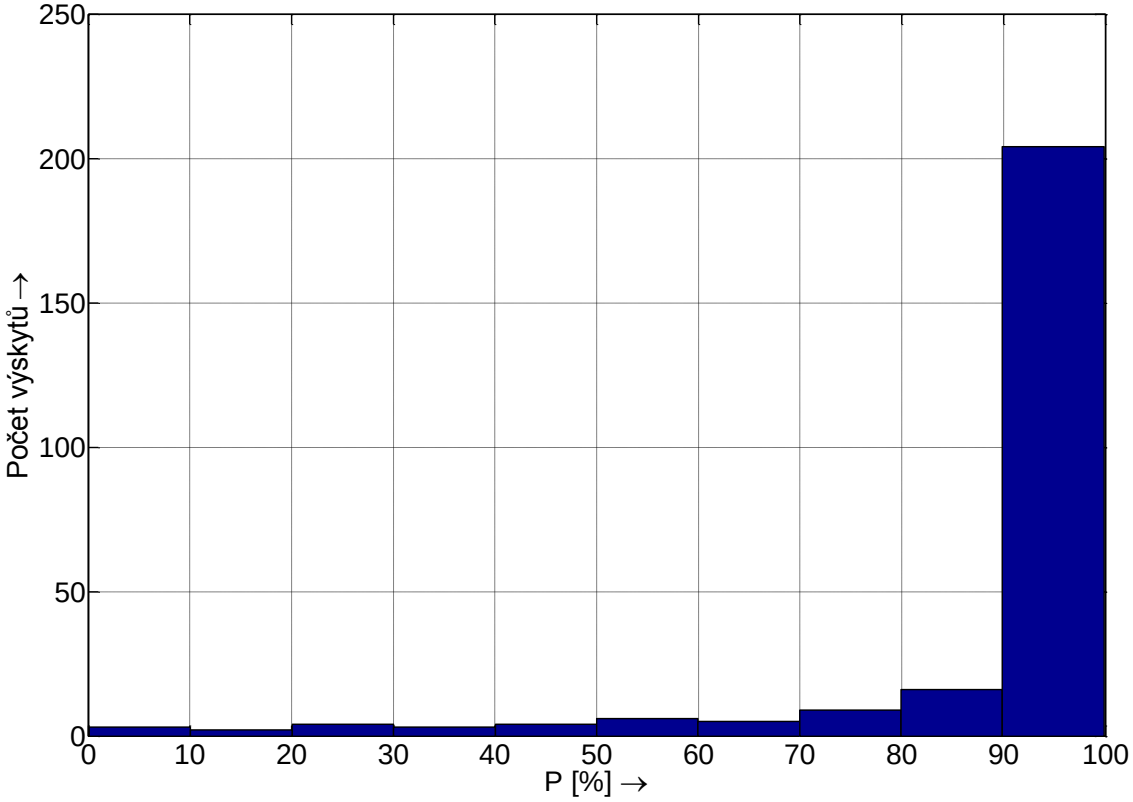
Obr. 4.32: Maximální hodnoty pravděpodobnosti a určené odhady klíče.

klíče bylo nezávisle uloženo 10 průběhů a následně byly tyto průběhy klasifikovány neuronovou sítí. Tímto způsobem se získaly výsledky pro všechny možné hodnoty klíče opakovaně z nezávislých měření a představa do jaké míry je metoda úspěšná i pro opakované měření a klasifikaci. Výsledky klasifikace tohoto objemu dat shrnuje následující tab. 4.7.

Tab. 4.7: Výsledky klasifikace pro 2560 proudových průběhů.

Použitá metoda klasifikace	Počet chybných klasifikací z celkového počtu 2560	Úspěšnost [%]
bez optimalizace	378	**85,23**
s optimalizací	139	**94,57**

Výsledky potvrdily, že opakované měření nemá na výsledky klasifikace vliv a je tedy možné metodu použít. Výsledky potvrdily získané dílčí výsledky z předchozích analýz, které byly provedeny se souborem 256 proudových spotřeb. Neoptimalizovaná metoda dosáhla 85% úspěšné klasifikace a optimalizovaná metoda klasifikovala tajné klíče s 95% úspěšností i z obsáhlého souboru naměřených dat. Pro doplnění výsledků klasifikace je uvedena tab. 4.8, která udává výsledky klasifikace pro sedm



Obr. 4.33: Histogram maximálních hodnot pravděpodobností po optimalizaci.

vybraných klíčů. Prvních pět vybraných klíčů (5, 41, 81, 129 a 248) koresponduje s klíči vybranými v předchozích kapitolách a určené maximální pravděpodobnosti jsou takřka totožné. Jako příklad chybné klasifikace jsou zobrazeny klíče 19 a 20, kde se vyskytovaly řádově nižší pravděpodobnosti a byly zde větší rozdíly v hodnotách pravděpodobností. Neuronová síť klasifikovala z těchto 20 průběhů 12 chybně a po optimalizaci 8. Opět se potvrdila funkčnost zvýšení diference mezi jednotlivými proudovými průběhy.

Při porovnání metody využívající neuronové sítě s používanými metodami DPA a SPA je hlavní výhoda v tom, že i pro algoritmus odolný proti konvenční analýze je metoda schopna určit první bajt tajného klíče s pravděpodobností kolem **96% pomocí jednoho průběhu proudové spotřeby.** DPA útoky potřebují k realizaci několik stovek nebo tisíc měření proudových spotřeb. Tento typ útoku proudovým postranním kanálem, který je zaměřen na určení hodnoty tajného klíče nebyl dosud publikován, jedná se tedy o zcela novou myšlenku. Podobná myšlenka byla publikována, ale byla zaměřena jen na rozpoznávání jednotlivých otisků proudových spotřeb jednotlivých instrukcí mikroprocesoru [118] a [59]. Metoda využívá snížení elektronického šumu v naměřeném proudovém průběhu a zvyšuje diferenci mezi

Tab. 4.8: Výsledky opakované klasifikace pro 7 klíčů.

K_{taj}	**5**	**41**	**81**	**129**	**248**	**19**	**20**
Bez optimalizace P_{max}[%]	28,74	38,20	79,92	67,46	30,07	7,67	23,49
	27,21	41,48	79,99	67,68	39,26	17,27	13,49
	27,10	39,78	80,51	67,87	36,55	11,49	18,23
	28,96	42,19	80,15	69,02	31,05	9,30	25,07
	23,03	41,37	79,93	68,02	38,97	7,73	17,90
	28,81	31,34	77,83	67,94	37,95	12,63	25,07
	23,75	36,28	80,03	67,83	34,38	8,73	25,94
	26,95	37,32	77,32	66,56	36,04	13,85	28,05
	22,02	33,39	80,30	67,91	39,25	8,27	26,28
	28,44	38,25	80,43	67,99	34,82	7,81	13,73
S optimalizací P_{max}[%]	98,28	98,99	98,25	97,14	81,85	28,43	76,20
	97,99	98,98	98,40	98,13	99,42	98,77	5,61
	98,52	99,07	99,29	97,87	98,96	92,58	32,79
	98,19	99,04	99,06	77,21	87,86	76,18	73,42
	97,04	99,12	98,16	96,65	99,45	49,10	40,15
	98,48	98,37	82,53	96,63	99,25	95,40	85,24
	97,04	99,01	98,15	97,01	97,55	69,45	87,14
	98,56	99,00	61,90	98,90	98,58	96,86	92,34
	95,05	98,70	98,96	97,90	99,44	39,62	85,76
	98,74	98,95	98,92	98,28	97,22	51,72	7,44

proudovými průběhy pomocí předzpracování naměřených průběhů. Navržená metoda může pracovat co nejrychleji a útočník může provést útok i na modul, který se mu podařilo získat jen na krátký čas. Dosavadní metody předpokládají plnou kontrolu nad modulem.

Nevýhodou je nutnost prvotního trénování neuronové sítě, kdy útočník musí vytvořit trénovací množinu proudových spotřeb. Počet proudových spotřeb musí odpovídat všem možným kombinacím tajného klíče, v našem případě se jednalo o první bajt AES, tzn. 256 proudových průběhů. Pro následující útoky již stačí jen jeden konkrétní proudový průběh. V reálném útoku je za kritickou část považována synchronizace naměřených proudových průběhů. Ideální metodou je vložení identického synchronizačního signálu jako při měření vzorových dat. Pokud tuto možnost útočník nemá, nezbývá než naměřit celý průběh zkoumaného algoritmu stejně tak jak popisuje kapitola 4.1 a následnou postupnou analýzu průběhu určit důležité operace a ty synchronizovat na první proudovou špičku. Důležitým faktorem je taká stejná

implementace algoritmu, pokud by byl algoritmus implementován odlišnými instrukcemi, výsledky analýzy by byly chybné. Další pokračování v práci spočívá v ověření funkčnosti metody pro různé kryptografické moduly a pro následující bajty tajného klíče.

Přínosy a nevýhody metody lze shrnout do následujících bodů:

- Přínosy
  - klasifikace se provádí z jednoho naměřeného proudového průběhu stejně jako u SPA útoků,
  - metoda je aplikovatelná na algoritmy odolné proti SPA,
  - ne nutnost měření stovek proudových průběhů jako u DPA,
  - realizace útoku velmi rychlá v porovnání s DPA (předpoklad naučená neuronová síť),
  - implementovaná metoda určila první bajt tajného klíče algoritmu AES s pravděpodobností kolem 96%,
  - metoda je opakovatelná, tedy prakticky realizovatelná (testováno na 2560 proudových průbězích s úspěšností 95%),
  - při předzpracování proudových průběhů (optimalizace) jsou minimalizovány chybné klasifikace odpovídající podobným proudovým průběhům,
- Nevýhody
  - nevýhoda metody spočívá v přípravě trénovací množiny pro neuronovou síť,
  - pro specifický kryptografický modul (stejný typ procesoru, čipové karty atd.) je zapotřebí mít naučenou neuronovou síť,
  - za kritickou část útoku se považuje správná synchronizace naměřených proudových průběhů, toho se dá využít při implementaci protiopatření ovlivňující časovou oblast proudové spotřeby.

5 ZÁVĚR

Disertační práce se zabývá problematikou postranních kanálů, které umožňují útočníkovi z kryptografického modulu získat senzitivní informace netradiční cestou. Nedílnou součástí práce je také rozbor protiopatření, které tomuto útoku zabraňují. V úvodu práce je uveden souhrn dosavadních metod kryptoanalýzy postranními kanály, je provedeno jejich zhodnocení a klasifikace. V uvedené oblasti zatím neexistuje jednotná terminologie, je proto nutné jasně definovat jednotlivé typy postranních kanálů a jejich základní principy. Podrobněji je v práci rozebrán proudový postranní kanál, ze kterého posléze vychází nově navržená metoda kryptoanalýzy. Návrh a experimentální ověření nové metody je hlavním cílem disertační práce. Navrhovaná metoda využívá neuronové sítě k odhalení hodnoty šifrovacího klíče. Myšlenka využití neuronových sítí v kryptoanalýze proudovým postranním kanálem je původní, poprvé byla autorem publikována v roce 2010 [118]. Další vývoj v oblasti proudové analýzy ukázal, že neuronové sítě jsou vhodným nástrojem [92, 12].

Pro správnou funkci nově navržené metody kryptoanalýzy je stěžejní způsob snímání proudové spotřeby kryptografického modulu. Při nevhodném způsobu měření může dojít v snímání proudového odběru k odfiltrování senzitivních informací. Proto byly pro ověření teoretických znalostí navrženy a experimentálně ověřeny různé způsoby měření. Metody měření jsou popsány v kapitole 4 a byly publikovány v odborných časopisech i na tuzemských a mezinárodních konferencích [116, 124, 117, 121, 122, 115, 120].

K návrhu nové metody a jejímu testování byl vybrán algoritmus AES a to z důvodu jeho známe odolnosti proti konvenčnímu způsobu kryptoanalýzy. Implementace metody byla provedena v programovém prostředí MATLAB, získané výsledky jsou detailně popsány v kapitole 4.6. Navržená metoda určila hodnotu tajného klíče algoritmu AES v 93% případů, ale z opakovaných testů a podrobné analýzy výsledků klasifikace vyplynula teoretická funkčnost metody jen 80%, a proto byla navržená metoda dále optimalizována. Optimalizace metody byla založena na zvýšení diference mezi jednotlivými průběhy proudové spotřeby. Pro zvýšení diference bylo použito předzpracování proudových průběhů využívající rozdíl jednotlivých průběhů od vypočteného průměrného průběhu proudové spotřeby. Takto optimalizovaná metoda úspěšně klasifikovala hodnotu tajného klíče v 96% případů.

Následně byla provedena analýza opakovatelnosti a realizovatelnosti obou metod. Bylo naměřeno 2560 průběhů proudové spotřeby odpovídající všem hodnotám tajného klíče a tyto průběhy byly klasifikovány neuronovými sítěmi. Tímto způsobem byly získány výsledky klasifikace pro všechny možné hodnoty tajného klíče opakovaně z nezávislých měření. Úspěšnost klasifikace potvrdila získané dílčí výsledky z předchozích analýz, které byly provedeny se souborem 256 proudových spotřeb.

Neoptimalizovaná metoda dosáhla 85% úspěšné klasifikace a optimalizovaná metoda klasifikovala tajné klíče s 95% úspěšností i z obsáhlejšího souboru proudových průběhů. Z výsledků je patrný pozitivní vliv předzpracování proudových průběhů na úspěšnost klasifikace.

Při porovnání navržené metody využívající neuronové sítě s obecně používanými metodami DPA a SPA je hlavní výhoda nové metody v tom, že i pro algoritmus odolný proti konvenční analýze je metoda schopna určit první bajt tajného klíče s pravděpodobností kolem 96% pomocí jen jednoho průběhu proudové spotřeby. Navržená metoda může pracovat rychle a útočník může provést útok i na kryptografický modul, který se mu podařilo získat jen na krátký čas. Nevýhodou metody je nutnost prvotního trénování neuronové sítě, kde útočník musí vytvořit trénovací množinu proudových spotřeb pro konkrétní kryptografický modul. Za kritickou část je považována správná synchronizace naměřených proudových průběhů. Tohoto faktu lze využít k implementaci protiopatření zabraňující kryptoanalýze, lze např. znemožněním správné synchronizace ovlivněním časové oblasti proudové spotřeby. Další pokračování v práci spočívá v ověření funkčnosti metody pro různé kryptografické moduly (stejný typ) a pro následující bajty tajného klíče bez nutnosti trénování neuronové sítě. Předpokládá se identická implementace algoritmu pro následující bajty tajného klíče viz algoritmus AES operace `AddRoundKey`. Všechny stanovené cíle disertační práce považuji za splněné a dosažené výsledky byly publikovány v odborných časopisech i na tuzemských a mezinárodních konferencích [123, 114, 125, 121].

LITERATURA

[1] Federal Information Processing Standards Publication (FIPS 197). Advanced Encryption Standard (AES). 2001.

[2] AC Current Probes CT1, CT2, CT6 Data Sheet. Data sheet, Tektronix, 2011.
URL `http://www.tek.com/sites/tek.com/files/media/media/resources/60W_12572_2.pdf`

[3] Abdollahi, A.; Fallah, F.; Pedram, M.: Leakage current reduction in CMOS VLSI circuits by input vector control. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, ročník 12, č. 2, feb. 2004: s. 140 – 154, ISSN 1063-8210.

[4] Agrawal, D.; Archambeault, B.; Rao, J.; aj.: The EM Side-Channel(s). In *Cryptographic Hardware and Embedded Systems - CHES 2002*, *Lecture Notes in Computer Science*, ročník 2523, Springer Berlin / Heidelberg, 2003, ISBN 978-3-540-00409-7, s. 29–45.
URL `http://dx.doi.org/10.1007/3-540-36400-5_4`

[5] Agrawal, D.; Rao, J. R.; Rohatgi, P.; aj.: Templates as Master Keys. In *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*, *Lecture Notes in Computer Science*, ročník 3659, editace J. R. Rao; B. Sunar, Springer, 2005, ISBN 3-540-28474-5, s. 15–29, doi:http://dx.doi.org/10.1007/11545262_2.

[6] Akkar, M.-L.; Bevan, R.; Dischamp, P.; aj.: Power Analysis, What Is Now Possible... In *Advances in Cryptology - ASIACRYPT 2000*, *Lecture Notes in Computer Science*, ročník 1976, editace T. Okamoto, Springer Berlin Heidelberg, 2000, ISBN 978-3-540-41404-9, s. 489–502.
URL `http://dx.doi.org/10.1007/3-540-44448-3_38`

[7] Akkar, M.-L.; Giraud, C.: An Implementation of DES and AES, Secure against Some Attacks. In *Cryptographic Hardware and Embedded Systems - CHES 2001*, ročník 2162, Springer Berlin / Heidelberg, 2001, ISBN 978-3-540-42521-2, s. 309–318.
URL `http://dx.doi.org/10.1007/3-540-44709-1_26`

[8] Akkar, M.-L.; Goubin, L.: A Generic Protection against High-Order Differential Power Analysis. In *Fast Software Encryption, 10th International Workshop, FSE 2003, Lund, Sweden, February 24-26, 2003, Revised Papers*, *Lecture Notes in Computer Science*, ročník 2887, Springer, 2003, s. 192–205.
URL `http://www.iacr.org/cryptodb/archive/2003/FSE/2955/2955.pdf`

[9] Alioto, M.; Giancane, L.; Scotti, G.; aj.: Leakage Power Analysis Attacks: A Novel Class of Attacks to Nanometer Cryptographic Circuits. *Circuits and Systems I: Regular Papers, IEEE Transactions on*, ročník 57, č. 2, feb. 2010: s. 355 –367, ISSN 1549-8328.

[10] Alioto, M.; Poli, M.; Rocchi, S.: Power analysis attacks to cryptographic circuits: a comparative analysis of DPA and CPA. In *Microelectronics, 2008. ICM 2008. International Conference on*, dec. 2008, s. 333 –336, doi:10.1109/ICM.2008.5393827.

[11] Ambrose, J.; Aldon, N.; Ignjatovic, A.; aj.: Anatomy of Differential Power Analysis for AES. In *Symbolic and Numeric Algorithms for Scientific Computing, 2008. SYNASC '08. 10th International Symposium on*, sept. 2008, ISBN 978-0-7695-3523-4, s. 459 –466.
URL http://dx.doi.org/10.1109/SYNASC.2008.8

[12] Bartkewitz, T.; Lemke-Rust, K.: Efficient 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, ISBN 978-3-642-37287-2, s. 263–276.
URL http://dx.doi.org/10.1007/978-3-642-37288-9_18

[13] Bayam, K.; Ors, B.: Differential Power Analysis resistant hardware implementation of the RSA cryptosystem. In *ISCAS*, IEEE, 2008, s. 3314–3317.
URL http://dblp.uni-trier.de/db/conf/iscas/iscas2008.html#BayamO08

[14] Bernstein, D. J.: Cache-timing attacks on AES. Technická zpráva, 2005.

[15] Bertoni, G.; Zaccaria, V.; Breveglieri, L.; aj.: AES Power Attack Based on Induced Cache Miss and Countermeasure. In *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume I - Volume 01*, ITCC '05, Washington, DC, USA: IEEE Computer Society, 2005, ISBN 0-7695-2315-3, s. 586–591.
URL http://dx.doi.org/10.1109/ITCC.2005.62

[16] Berzati, A.; Canovas, C.; Dumas, J.-G.; aj.: Fault Attacks on RSA Public Keys: Left-To-Right Implementations Are Also Vulnerable. In *Proceedings of the The Cryptographers' Track at the RSA Conference 2009 on Topics in Cryptology*, CT-RSA '09, Berlin, Heidelberg: Springer-Verlag, 2009, ISBN 978-3-642-00861-0, s. 414–428, doi: 10.1007/978-3-642-00862-7_28.
URL http://dx.doi.org/10.1007/978-3-642-00862-7_28

[17] Bleichenbacher, D.: Chosen Ciphertext Attacks Against Protocols Based on the RSA Encryption Standard PKCS #1. In *CRYPTO*, 1998, s. 1–12.
URL http://dx.doi.org/10.1007/BFb0055716

[18] Blömer, J.; Guajardo, J.; Krummel, V.: Provably secure masking of AES. In *Proceedings of the 11th international conference on Selected Areas in Cryptography*, SAC'04, Springer-Verlag, 2005, ISBN 3-540-24327-5, 978-3-540-24327-4, s. 69–83.
URL http://dx.doi.org/10.1007/978-3-540-30564-4_5

[19] Brier, E.; Clavier, C.; Olivier, F.: Correlation Power Analysis with a Leakage Model. In *CHES*, 2004, s. 16–29.

[20] Chari, S.; Jutla, C.; Rao, J. R.; aj.: A Cautionary Note Regarding Evaluation of AES Candidates on Smart-Cards. In *In Second Advanced Encryption Standard (AES) Candidate Conference*, s. 133–147.

[21] Chari, S.; Jutla, C. S.; Rao, J. R.; aj.: Towards sound approaches to counteract power-analysis attacks. In *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, Springer-Verlag, 1999, ISBN 3-540-66347-9, s. 398–412.
URL `http://dl.acm.org/citation.cfm?id=646764.703964`

[22] Chari, S.; Rao, J. R.; Rohatgi, P.: Template Attacks. In *CHES*, 2002, s. 13–28.

[23] Clavier, C.; Coron, J.-S.; Dabbous, N.: Differential Power Analysis in the Presence of Hardware Countermeasures. In *Proceedings of the Second International Workshop on Cryptographic Hardware and Embedded Systems*, 2000, ISBN 3-540-41455-X, s. 252–263.
URL `http://dx.doi.org/10.1007/3-540-44499-8_20`

[24] Clavier, C.; Feix, B.; Gagnerot, G.; aj.: Improved Collision-Correlation Power Analysis on First Order Protected AES. In *Cryptographic Hardware and Embedded Systems - CHES 2011*, *Lecture Notes in Computer Science*, ročník 6917, editace B. Preneel; T. Takagi, Springer Berlin Heidelberg, 2011, ISBN 978-3-642-23950-2, s. 49–62, doi: 10.1007/978-3-642-23951-9_4.
URL `http://dx.doi.org/10.1007/978-3-642-23951-9_4`

[25] Coron, J.-S.; Goubin, L.: On Boolean and Arithmetic Masking against Differential Power Analysis. In *Proceedings of the Second International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '00, London, UK, UK: Springer-Verlag, 2000, ISBN 3-540-41455-X, s. 231–237.
URL `http://dl.acm.org/citation.cfm?id=648253.752404`

[26] Coron, J.-S.; Naccache, D.; Kocher, P.: Statistics and secret leakage. *ACM Trans. Embed. Comput. Syst.*, ročník 3, č. 3, Srpen 2004: s. 492–508, ISSN 1539-9087.
URL `http://doi.acm.org/10.1145/1015047.1015050`

[27] Coron, J.-S.; Naccache, D.; Kocher, P.: Statistics and secret leakage. *ACM Trans. Embed. Comput. Syst.*, ročník 3, č. 3, Srpen 2004: s. 492–508, ISSN 1539-9087, doi: 10.1145/1015047.1015050.
URL `http://doi.acm.org/10.1145/1015047.1015050`

[28] Daemen, J.; Rijmen, V.: AES Proposal: Rijndael. 1999.

[29] Daemen, J.; Rijmen, V.: *The design of Rijndael: AES — the Advanced Encryption Standard*. Springer-Verlag, 2002, ISBN 3-540-42580-2, 238 s.

[30] Daněček, P.: *Útoky na kryptografické moduly*. Dizertační práce, Vysoké učení technické v Brně, fakulta elektrotechniky a komunikačních technologií, may 2007.

[31] Debraize, B.: Efficient and Provably Secure Methods for Switching from Arithmetic to Boolean Masking. In *Cryptographic Hardware and Embedded Systems - CHES 2012*, *Lecture Notes in Computer Science*, ročník 7428, editace E. Prouff; P. Schaumont, Springer Berlin Heidelberg, 2012, ISBN 978-3-642-33026-1, s. 107–121, doi:10.1007/978-3-642-33027-8_7.
URL `http://dx.doi.org/10.1007/978-3-642-33027-8_7`

[32] Dhem, J.-F.; Koeune, F.; Leroux, P.-A.; aj.: A Practical Implementation of the Timing Attack. Januar 1998.

[33] Eck, W. V.; Laborato, N.: Electromagnetic Radiation from Video Display Units: An Eavesdropping Risk? *Computers & Security*, 1985: s. 269–286, ISSN 0167-4048.
URL `http://dx.doi.org/10.1016/0167-4048(85)90046-X`

[34] Fei, Y.; Luo, Q.; Ding, A.: A Statistical Model for DPA with Novel Algorithmic Confusion Analysis. In *Cryptographic Hardware and Embedded Systems - CHES 2012*, *Lecture Notes in Computer Science*, ročník 7428, editace E. Prouff; P. Schaumont, Springer Berlin Heidelberg, 2012, ISBN 978-3-642-33026-1, s. 233–250, doi:10.1007/978-3-642-33027-8_14.
URL `http://dx.doi.org/10.1007/978-3-642-33027-8_14`

[35] Ferrigno, J.; Hlavac, M.: When AES blinks: introducing optical side channel. *Information Security, IET*, ročník 2, č. 3, september 2008: s. 94 –98, ISSN 1751-8709.

[36] Fiona, A. H. Y.: *ERG4920CM Thesis II Keyboard Acoustic Triangulation Attack*. Dizertační práce, Department of Information Engineering the Chinese University of Hong Kong, 2006.

[37] Fouque, P.-A.; Kunz-Jacques, S.; Martinet, G.; aj.: Power Attack on Small RSA Public Exponent. In *Cryptographic Hardware and Embedded Systems - CHES 2006, 8th International Workshop*, *Lecture Notes in Computer Science*, ročník 4249, Springer, 2006, s. 339–353, doi:10.1007/11894063_27.
URL `http://www.iacr.org/cryptodb/archive/2006/CHES/27/27.pdf`

[38] Gandolfi, K.; Mourtel, C.; Olivier, F.: Electromagnetic Analysis: Concrete Results. In *CHES '01: Proceedings of the Third International Workshop on Cryptographic Hardware and Embedded Systems*, London, UK: Springer-Verlag, 2001, ISBN 3-540-42521-7, s. 251–261.

[39] Gandolfi, K.; Naccache, D.; Paar, C.; aj.: Electromagnetic Analysis: Concrete Results. 2001.

[40] Genelle, L.; Prouff, E.; Quisquater, M.: Thwarting higher-order side channel analysis with additive and multiplicative maskings. In *Proceedings of the 13th international conference on Cryptographic hardware and embedded systems*, CHES'11, Berlin, Heidelberg: Springer-Verlag, 2011, ISBN 978-3-642-23950-2, s. 240–255.
URL `http://dl.acm.org/citation.cfm?id=2044928.2044949`

[41] Giorgetti, J.; Scotti, G.; Simonetti, A.; aj.: Analysis of data dependence of leakage current in CMOS cryptographic hardware. In *GLSVLSI '07: Proceedings of the 17th ACM Great Lakes symposium on VLSI*, New York, NY, USA: ACM, 2007, ISBN 978-1-59593-605-9, s. 78–83.

[42] Goubin, L.: A Sound Method for Switching between Boolean and Arithmetic Masking. In *Proceedings of the Third International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '01, London, UK, UK: Springer-Verlag, 2001, ISBN 3-540-42521-7, s. 3–15.
URL `http://dl.acm.org/citation.cfm?id=648254.752571`

[43] Hanley, N.; Tunstall, M.; Marnane, W. P.: Using templates to distinguish multiplications from squaring operations. *Int. J. Inf. Sec.*, ročník 10, č. 4, 2011: s. 255–266.

[44] Herbst, C.; Oswald, E.; Mangard, S.: An AES Smart Card Implementation Resistant to Power Analysis Attacks. In *Applied Cryptography and Network Security, Second International Conference, ACNS 2006, volume 3989 of Lecture Notes in Computer Science*, Springer, 2006, s. 239–252.

[45] Heuser, A.; Zohner, M.: Intelligent Machine Homicide - Breaking Cryptographic Devices Using Support Vector Machines. In *COSADE*, 2012, s. 249–264.

[46] Hospodar, G.; Gierlichs, B.; Mulder, E. D.; aj.: Machine learning in side-channel analysis: a first study. *J. Cryptographic Engineering*, ročník 1, č. 4, 2011: s. 293–302.

[47] Joye, M.; Olivier, F.: Side-Channel Analysis. In *Encyclopedia of Cryptography and Security (2nd Ed.)*, 2011, s. 1198–1204.
URL `http://dx.doi.org/10.1007/978-1-4419-5906-5_516`

[48] Joye, M.; Paillier, P.; Schoenmakers, B.: On Second-Order Differential Power Analysis. In *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop*, Springer, 2005, s. 293–308.

[49] Kaliski, B.: RFC 2898 - PKCS #5: Password-Based Cryptography Specification Version 2.0. Technická zpráva, IETF, Září 2000.
URL `http://tools.ietf.org/html/rfc2898`

[50] Kay, S. M.: *Fundamentals of Statistical Signal Processing, Volume I: Estimation Theory (v. 1)*. Prentice Hall, první vydání, Duben 1993, ISBN 0133457117.
URL http://www.worldcat.org/isbn/0133457117

[51] Çetin Kaya Koç; Rothatgi, P.; Schindler, W.; aj. (editoři): *Cryptographic Engineering*. 2009, ISBN 978-0-387-71816-3.

[52] Kim, H.; Hong, S.; Lim, J.: A fast and provably secure higher-order masking of AES S-box. In *Proceedings of the 13th international conference on Cryptographic hardware and embedded systems*, CHES'11, Berlin, Heidelberg: Springer-Verlag, 2011, ISBN 978-3-642-23950-2, s. 95–107.
URL http://dl.acm.org/citation.cfm?id=2044928.2044937

[53] Kim, H.-M.; Kang, D.-J.; Kim, T.-H.: Flexible Key Distribution for SCADA Network using Multi-Agent System. *Bio-inspired, Learning, and Intelligent Systems for Security, ECSIS Symposium on*, 2007: s. 29–34.

[54] Klima, V.; Rosa, T.: Side Channel Attacks on CBC Encrypted Messages in the PKCS7 Format. Cryptology ePrint Archive, Report 2003/098, 2003.

[55] Kocher, P. C.; Jaffe, J.; Jun, B.: Differential Power Analysis. In *CRYPTO '99: Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, London, UK: Springer-Verlag, 1999, ISBN 3-540-66347-9, s. 388–397.

[56] Kolofík, J.: *Optický Postranní kanál*. bakalařská práce, Vysoké učení technické v Brně, fakulta elektrotechniky a komunikačních technologií, 2010.

[57] Kuhn, M. G.: Optical Time-Domain Eavesdropping Risks of CRT Displays. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Washington, DC, USA: IEEE Computer Society, 2002, ISBN 0-7695-1543-6, str. 3.

[58] Kuhn, M. G.; Anderson, R. J.: Soft tempest: Hidden data transmission using electromagnetic emanations. In *Proc. 2nd Workshop on Information Hiding*, Springer-Verlag, 1998, s. 124–142.

[59] Kur, J.; Smolka, T.; Svenda, P.: Improving Resiliency of Java Card Code Against Power Analysis. In *Mikulaska kryptobesidka, Sbornik prispevku*, 2009, s. 29–39.

[60] Lerman, L.; Bontempi, G.; Markowitch, O.: Side channel attack: an approach based on machine learningn. In *COSADE 2011 - Second International Workshop on Constructive Side-Channel Analysis and Secure Design*, 2011, s. 29–41.

[61] Lian, S.; Sun, J.; Wang, Z.: One-way Hash Function Based on Neural Network. *CoRR*, ročník abs/0707.4032, 2007.
URL http://dblp.uni-trier.de/db/journals/corr/corr0707.html#abs-0707-4032

[62] Lin, L.; Burleson, W.: Leakage-based differential power analysis (LDPA) on sub-90nm CMOS cryptosystems. In *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, may 2008, s. 252 –255, doi:10.1109/ISCAS.2008.4541402.

[63] Liu, N.; Guo, D.: Security Analysis of Public-key Encryption Scheme Based on Neural Networks and Its Implementing. In *Computational Intelligence and Security, 2006 International Conference on*, ročník 2, nov. 2006, s. 1327 –1330, doi:10.1109/ICCIAS.2006.295274.

[64] Machů, P.: *Nové postranní kanály v kryptografii.* diplomová práce, Vysoké učení technické v Brně, fakulta elektrotechniky a komunikačních technologií, 2010.

[65] Mangard, S.: A Simple Power-Analysis (SPA) attack on implementations of the AES key expansion. In *ICISC 2002, LNCS 2587*, Springer-Verlag, 2002, s. 343–358.

[66] Mangard, S.: Exploiting Radiated Emissions EM Attacks on Cryptographic ICs. 2003, presentation.

[67] Mangard, S.; Oswald, E.; Popp, T.: *Power Analysis Attacks: Revealing the Secrets of Smart Cards (Advances in Information Security).* Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007, ISBN 0387308571.

[68] Manger, J.: *A Chosen Ciphertext Attack on RSA Optimal Asymmetric Encryption Padding (OAEP) as Standardized in PKCS #1 v2.0*, ročník 2139. Springer-Verlag, 2001, s. 230–238.
URL `http://portal.acm.org/citation.cfm?id=646766.704143`

[69] Markantonakis, K.; Tunstall, M.; Hancke, G.; aj.: Attacking smart card systems: Theory and practice. *Information Security Technical Report*, ročník 14, č. 2, 2009: s. 46 – 56, ISSN 1363-4127, doi:DOI:10.1016/j.istr.2009.06.001, smart Card Applications and Security.
URL `http://www.sciencedirect.com/science/article/pii/S136341270900017X`

[70] May, D.; Muller, H.; Smart, N.: Non-deterministic Processors. In *Information Security and Privacy*, *Lecture Notes in Computer Science*, ročník 2119, editace V. Varadharajan; Y. Mu, Springer Berlin / Heidelberg, 2001, ISBN 978-3-540-42300-3, s. 115–129, 10.1007/3-540-47719-5_11.
URL `http://dx.doi.org/10.1007/3-540-47719-5_11`

[71] Mayer-Sommer, R.: Smartly Analyzing the Simplicity and the Power of Simple Power Analysis on Smartcards. In *Proceedings of the Second International Workshop on Cryptographic Hardware and Embedded Systems*, CHES '00, London, UK, UK: Springer-Verlag, 2000, ISBN 3-540-41455-X, s. 78–92.
URL `http://dl.acm.org/citation.cfm?id=648253.752540`

[72] Mesquita, D.; Techer, J.-D.; Torres, L.; aj.: Current mask generation: a transistor level security against DPA attacks. In *SBCCI*, 2005, s. 115–120.

[73] Messerges, T.: Using Second-Order Power Analysis to Attack DPA Resistant Software. In *Cryptographic Hardware and Embedded Systems - CHES 2000*, *Lecture Notes in Computer Science*, ročník 1965, editace e. KoĂ§; C. Paar, Springer Berlin Heidelberg, 2000, ISBN 978-3-540-41455-1, s. 238–251.
URL http://dx.doi.org/10.1007/3-540-44499-8_19

[74] Messerges, T. S.; Dabbish, E. A.; Sloan, R. H.; aj.: Investigations of power analysis attacks on smartcards. In *In USENIX Workshop on Smartcard Technology*, 1999, s. 151–162.

[75] Michéle, B.; Krämer, J.; Seifert, J.-P.: Structure-Based RSA fault attacks. In *Proceedings of the 8th international conference on Information Security Practice and Experience*, ISPEC'12, Berlin, Heidelberg: Springer-Verlag, 2012, ISBN 978-3-642-29100-5, s. 301–318, doi:10.1007/978-3-642-29101-2_21.
URL http://dx.doi.org/10.1007/978-3-642-29101-2_21

[76] Mislovaty, R.; Perchenok, Y.; Kanter, I.; aj.: Secure key-exchange protocol with an absence of injective functions. *Phys. Rev. E*, ročník 66, Dec 2002: str. 066102, doi:10.1103/PhysRevE.66.066102.
URL http://link.aps.org/doi/10.1103/PhysRevE.66.066102

[77] Miyamoto, A.; Homma, N.; Aoki, T.; aj.: Enhanced power analysis attack using chosen message against RSA hardware implementations. In *International Symposium on Circuits and Systems (ISCAS 2008), 18-21 May 2008, Sheraton Seattle Hotel, Seattle, Washington, USA*, IEEE, 2008, s. 3282–3285, doi:http://dx.doi.org/10.1109/ISCAS.2008.4542159.

[78] Moradi, A.; Salmasizadeh, M.; Manzuri Shalmani, M. T.; aj.: Vulnerability modeling of cryptographic hardware to power analysis attacks. *Integr. VLSI J.*, ročník 42, č. 4, Září 2009: s. 468–478, ISSN 0167-9260, doi:10.1016/j.vlsi.2009.01.001.
URL http://dx.doi.org/10.1016/j.vlsi.2009.01.001

[79] Moss, A.; Oswald, E.; Page, D.; aj.: Compiler Assisted Masking. In *Cryptographic Hardware and Embedded Systems - CHES 2012*, *Lecture Notes in Computer Science*, ročník 7428, editace E. Prouff; P. Schaumont, Springer Berlin Heidelberg, 2012, ISBN 978-3-642-33026-1, s. 58–75, doi:10.1007/978-3-642-33027-8_4.
URL http://dx.doi.org/10.1007/978-3-642-33027-8_4

[80] Muresan, R.; Vahedi, H.; Zhanrong, Y.; aj.: Power-smart system-on-chip architecture for embedded cryptosystems. In *Proceedings of the 3rd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, CODES+ISSS '05, New York, NY, USA: ACM, 2005, ISBN 1-59593-161-9, s. 184–189,

doi:10.1145/1084834.1084883.
URL http://doi.acm.org/10.1145/1084834.1084883

[81] Nabney, I. T.: *NETLAB: algorithms for pattern recognition*. Advances in Pattern Recognition, New York, NY, USA: Springer-Verlag New York, Inc., 2002, ISBN 1-85233-440-1.

[82] Nečas, O.: *Útok elektromagnetickým postranním kanálem*. diplomová práce, Vysoké učení technické v Brně, fakulta elektrotechniky a komunikačních technologií, 2011.

[83] Neve, M.; pierre Seifert, J.; Wang, Z.: Cache time-behavior analysis on AES.

[84] Ostermann, T.; Gut, W.; Bacher, C.; aj.: Measures to Reduce the Electromagnetic Emission of a SoC. In *VLSI-SOC*, 2003, s. 31–.

[85] Osvik, D. A.; Shamir, A.; Tromer, E.: Cache Attacks and Countermeasures: the Case of AES. In *Topics in Cryptology - CT-RSA 2006, The Cryptographersd'ż″ Track at the RSA Conference 2006*, Springer-Verlag, 2005, s. 1–20.

[86] Oswald, D.; Paar, C.: Breaking mifare DESFire MF3ICD40: power analysis and templates in the real world. In *Proceedings of the 13th international conference on Cryptographic hardware and embedded systems*, CHES'11, Berlin, Heidelberg: Springer-Verlag, 2011, ISBN 978-3-642-23950-2, s. 207–222.
URL http://dl.acm.org/citation.cfm?id=2044928.2044947

[87] Oswald, E.; Mangard, S.; Pramstaller, N.; aj.: A Side-Channel Analysis Resistant Description of the AES S-Box. In *FSE*, 2005, s. 413–423.

[88] Oswald, M. E.; Mangard, S.; Herbst, C.; aj.: Practical Second-Order DPA Attacks for Masked Smart Card Implementations of Block Ciphers. In *Topics in Cryptology - CT-RSA 2006*, *Lecture Notes in Computer Science*, ročník 3860, editace D. Pointcheval, Springer, 2006, s. 192 – 207.

[89] Peeters, E.; Standaert, F.-X.; Quisquater, J.-J.: Power and electromagnetic analysis: Improved model, consequences and comparisons. *Integration, the VLSI Journal*, ročník 40, č. 1, 2007: s. 52 – 60, ISSN 0167-9260, doi:DOI:10.1016/j.vlsi.2005.12.013, embedded Cryptographic Hardware.
URL http://www.sciencedirect.com/science/article/B6V1M-4J3NWY2-1/2/0197aa6143d75a8303ace31403077841

[90] Percival, C.: Cache missing for fun and profit. In *Proc. of BSDCan 2005*, 2005.

[91] Permadim, E.: PIC Microcontoller Math Library Methods. Prosinec 2010.
URL http://www.piclist.com/techref/microchip/math/index.htm

[92] Quisquater, J.-J.; Samyde, D.: Automatic code recognition for smart cards using a Kohonen neural network. In *Proceedings of the 5th conference on Smart Card*

Research and Advanced Application Conference - Volume 5, CARDIS'02, Berkeley, CA, USA, 2002, s. 6–6.
URL http://dl.acm.org/citation.cfm?id=1250988.1250994

[93] Rabaey, J. M.; Chandrakasan, A. P.; Nikolic, B.: *Digital integrated circuits : a design perspective*. Prentice Hall electronics and VLSI series, Pearson Education, druhé vydání, 2003, ISBN 0130909963.
URL http://www.worldcat.org/isbn/0130909963

[94] Ratanpal, G. B.; Williams, R. D.; Blalock, T. N.: An On-Chip Signal Suppression Countermeasure to Power Analysis Attacks. *IEEE Trans. Dependable Secur. Comput.*, ročník 1, č. 3, Červenec 2004: s. 179–189, ISSN 1545-5971, doi:10.1109/TDSC.2004.25.
URL http://dx.doi.org/10.1109/TDSC.2004.25

[95] Rechberger, C.; Oswald, E.: Practical Template Attacks. In *Information Security Applications, 5th International Workshop, WISA 2004, Jeju Island, Korea, August 23-25, 2004, Revised Selected Papers, volume 3325 of Lecture Notes in Computer Science*, Springer, 2004, s. 443–457.

[96] Sauvage, L.; Guilley, S.; Mathieu, Y.: Electromagnetic Radiations of FPGAs: High Spatial Resolution Cartography and Attack on a Cryptographic Module. *ACM Trans. Reconfigurable Technol. Syst.*, ročník 2, č. 1, Březen 2009: s. 4:1–4:24, ISSN 1936-7406, doi:10.1145/1502781.1502785.
URL http://doi.acm.org/10.1145/1502781.1502785

[97] Schindler, W.; Lemke, K.; Paar, C.: Paar: A Stochastic Model for Differential Side Channel Cryptanalysis. In *Cryptographic Hardware and Embedded Systems - CHES 2005, Springer, LNCS 3659*, Springer, 2005, s. 30–46.

[98] Shamir, A.: Protecting Smart Cards from Passive Power Analysis with Detached Power Supplies. In *CHES*, 2000, s. 71–77.
URL http://dx.doi.org/10.1007/3-540-44499-8_5

[99] Shamir, A.; Tromer, E.: Acoustic cryptanalysis On nosy people and noisy machines @ONLINE. 2011.
URL http://people.csail.mit.edu/tromer/acoustic/

[100] Shum, W. W.-K.; Anderson, J. H.: FPGA glitch power analysis and reduction. In *Proceedings of the 17th IEEE/ACM international symposium on Low-power electronics and design*, ISLPED '11, Piscataway, NJ, USA: IEEE Press, 2011, ISBN 978-1-61284-660-6, s. 27–32.
URL http://dl.acm.org/citation.cfm?id=2016802.2016811

[101] Standaert, F.-X.; Örs, S. B.; Quisquater, J.-J.; aj.: Power Analysis Attacks Against FPGA Implementations of the DES. In *FPL*, 2004, s. 84–94.

[102] Standaert, F.-X.; Rouvroy, G.; Quisquater, J.-J.: FPGA Implementations of the DES and Triple-DES Masked Against Power Analysis Attacks. In *FPL*, 2006, s. 1–4.

[103] Sugawara, T.; Homma, N.; Aoki, T.; aj.: Differential power analysis of AES ASIC implementations with various S-box circuits. In *Circuit Theory and Design, 2009. ECCTD 2009. European Conference on*, aug. 2009, s. 395 –398, doi:10.1109/ECCTD.2009.5275004.

[104] Thompson, S.; Mycroft, A.: Abstract interpretation of combinational asynchronous circuits. *Sci. Comput. Program.*, ročník 64, č. 1, Leden 2007: s. 166–183, ISSN 0167-6423.
URL `http://dx.doi.org/10.1016/j.scico.2006.03.007`

[105] Tiu, C. C.; Tiu, C. C.: A New Frequency-Based Side Channel Attack for Embedded Systems. Master degree thesis, Deparment of Electrical and Computer Engineering,University of Waterloo,Waterloo. Technická zpráva, 2005.

[106] Tsang, J. C.; Kash, J. A.; Vallett, D. P.: Picosecond imaging circuit analysis. *IBM J. Res. Dev.*, ročník 44, č. 4, 2000: s. 583–603, ISSN 0018-8646.

[107] Velegalati, R.; Yalla, P. S. V. V. K.: Differential Power Analysis Attack on FPGA Implementation of AES. 2008: s. 1–5.

[108] Waddle, J.; Wagner, D.: Towards Efficient Second-Order Power Analysis. In *Cryptographic Hardware and Embedded Systems - CHES 2004: 6th International Workshop Cambridge, MA, USA, August 11-13, 2004. Proceedings*, *Lecture Notes in Computer Science*, ročník 3156, Springer, 2004, s. 1–15.

[109] Walter, C. D.; Çetin Kaya Koç; Paar, C. (editoři): *Cryptographic Hardware and Embedded Systems - CHES 2003, 5th International Workshop, Cologne, Germany, September 8-10, 2003, Proceedings*, *Lecture Notes in Computer Science*, ročník 2779, Springer, 2003, ISBN 3-540-40833-9.

[110] Wang, Y.-h.; Shen, Z.-d.; Zhang, H.-g.: Pseudo Random Number Generator Based on Hopfield Neural Network. Srpen 2006, s. 2810–2813.
URL `http://dx.doi.org/10.1109/ICMLC.2006.259003`

[111] Yang, B.; Wu, K.; Karri, R.: Scan Based Side Channel Attack on Dedicated Hardware Implementations of Data Encryption Standard. In *ITC '04: Proceedings of the International Test Conference on International Test Conference*, Washington, DC, USA: IEEE Computer Society, 2004, ISBN 0-7803-8581-0, s. 339–344.

[112] Yang, S.; Wolf, W.; Vijaykrishnan, N.; aj.: Power Attack Resistant Cryptosystem Design: A Dynamic Voltage and Frequency Switching Approach. *Design, Automation and Test in Europe Conference and Exhibition*, ročník 3, 2005: s. 64–69, ISSN 1530-1591, doi:http://doi.ieeecomputersociety.org/10.1109/DATE.2005.241.

[113] Zhuang, L.; Zhou, F.; Tygar, J. D.: Keyboard acoustic emanations revisited. In *Proceedings of the 12th ACM conference on Computer and communications security*, CCS '05, New York, NY, USA: ACM, 2005, ISBN 1-59593-226-7, s. 373–382, doi: http://doi.acm.org/10.1145/1102120.1102169.
URL `http://doi.acm.org/10.1145/1102120.1102169`

VYBRANÉ PUBLIKACE AUTORA

[114] Martinasek, Z.; CLupek, V.; Trasy, K.: General Scheme of Differential Power Analysis. In *In 36nd International Conference on Telecommunications and Signal Processing - TSP' 20013*, in print.

[115] Martinasek, Z.; Clupek, V.; Zeman, V.; aj.: Základní metody diferenciální proudové analýzy. *Elektrorevue - Internetový ăasopis (http://www.elektrorevue.cz*, ročník 2013, č. 3, 2013: s. 1 – 10, ISSN 1213-1539.
URL `http://www.elektrorevue.cz`

[116] Martinasek, Z.; Macha, T.; Raso, O.; aj.: Optimization of differential power analysis. *PRZEGLAD ELEKTROTECHNICZNY*, ročník 87, č. 12, 2011: s. 140 – 144, ISSN 0033-2097.
URL `http://pe.org.pl/articles/2011/12a/28.pdf`

[117] Martinasek, Z.; Macha, T.; Stancikk, P.: Power side channel information measurement. In *Research in telecommunication technologies RTT2010*, September 2010.

[118] Martinasek, Z.; Macha, T.; Zeman, V.: Classifier of power side channel. In *Proceedings of NIMT2010*, September 2010, ISBN 978-80-214-4126- 2.

[119] Martinasek, Z.; Machu, P.: New side channle in cryptography. In *Proceedings of the 17th Conference Student EEICT 2011*, April 2011, ISBN 978-80-214-4273- 3.

[120] Martinásek, Z.; Nečas, O.; Zeman, V.; aj.: Diferenciální elektromagnetická analýza. *Elektrorevue - Internetový časopis (http://www.elektrorevue.cz*, ročník 2011, č. 60, 2011: s. 1 – 6, ISSN 1213-1539.
URL `http://www.elektrorevue.cz`

[121] Martinasek, Z.; Petrik, T.; Stancik, P.: Conditions affecting the measurement of power analysis. In *Research in telecommunication technologies RTT2011*, September 2011.

[122] Martinásek, Z.; Petřík, T.; Stančík, P.: Parametry ovlivňující proudovou analýzu mikroprocesoru vykonavajícího funkci AddRoundKey. *Elektrorevue - Internetovy časopis (http://www.elektrorevue.cz*, ročník 2011, č. 51, 2011: s. 1 – 6, ISSN 1213-1539.
URL `http://www.elektrorevue.cz`

[123] Martinasek, Z.; Zeman, V.: Innovative Method of the Power Analysis. *Radioengineering*, 2013, ISSN 1210-2512, in print.

[124] Martinasek, Z.; Zeman, V.; Sysel, P.; aj.: Near electromagnetic field measurement of microprocessor. *PRZEGLAD ELEKTROTECHNICZNY*, ročník 89, č. 2a, 2013: s. 203 – 207, ISSN 0033-2097.
URL `http://pe.org.pl/articles/2013/2a/44.pdf`

[125] Martinasek, Z.; Zeman, V.; Trasy, K.: Simple Electromagnetic Analysis in Cryptography. *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, ročník 1, č. 1, 2012: s. 1 – 6, ISSN 1805-5443.
URL http://www.ijates.org/domains/ijates.org/index.php/ijates/article/view/6/5

SEZNAM SYMBOLŮ, VELIČIN A ZKRATEK

SA Jednoduchá analýza – Simple Analysis

DA Diferenciální analýza – Differential Analysis

SPA Jednoduchá proudová analýza – Simple Power Analysis

DPA Diferenciální proudová analýza – Differential Power Analysis

TA Časová analýza – Timing Analysis

LPA Proudová statická analýza – Leakage Power Analysis

PICA Picosekundová zobrazovací obvodová analýza – Picosecond Imaging Circuit Analysis

EM Elektromagnetické – ElectroMagnetic

AES Pokročilý šifrovací standart – Advanced Encryption Standard

XOR Bitová nonekvivalence – Exclusive OR

CRT Zobrazovají jednotka na principu katodové trubice – Cathode Ray Tube

LED Světlo vyzařující dioda – Light Emitting Diode

CMOS Výrobní technologie integrovaných obvodů – Complementary Metal Oxide Semiconductor

RISC Procesory s redukovanou instrukční sadou – Reduced Instruction Set Computer

RSA Asymetrický šifrovací algoritmus, iniciály autorů Rivest, Shamir, Adleman

DSA Algoritmus digitálního podpisu – Digital Signature Algorithm

DES Šifrovací algoritmus – Data Encryption Standard

RC5 Šifrovací algoritmus – Rivest Cipher

STFT Krátkodobá Fourierova transformace – Short-time Fourier transform

CBC Řetězení šifrového textu – Cipher Block Chaining

HD Hammingova vzdálenost – Hamming Distance

SNR Poměr mezi úrovní užitečného signálu a úrovní šumu – Signal-to-Noise Ratio

SEZNAM PŘÍLOH

A PŘÍLOHA

A.1 Šifrovací algoritmus AES

Algoritmus AES je konkrétní realizací obecného algoritmu Rijndael, pojmenovaného podle jeho tvůrců Joana Daemena a Vincenta Rijmena [29, 28, 1]. Vstup a výstup algoritmu AES tvoří datový blok o délce 128 bitů. Délka šifrovacího klíče je u algoritmu AES volena z trojice možných hodnot 128, 195 a 256 bitů. Jiné délky vstupu, výstupu a šifrovacího klíče nejsou standardem povoleny. Počet N_r tzv. rund (opakovaných sekvencí blokových operací) závisí na délce šifrovacího klíče. Povolené kombinace délky klíče, bloku a počtu rund pro algoritmus AES jsou shrnuty v tab. A.1.

Tab. A.1: Standardy AES.

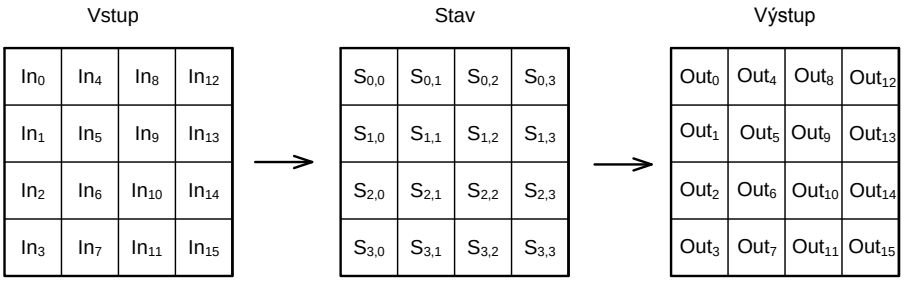
Standard	Délka šifrovacího klíče N_k bitů/slov	Velikost bloku N_b bitů/slov	Počet rund N_r
AES-128	128/4	128/4	10
AES-192	192/6	128/4	12
AES-256	256/8	128/4	14

Základní jednotkou pro zpracování algoritmem AES je bajt sekvence osmi bitů. Každý bajt je reprezentován jako posloupnost jednotlivých bitů $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$. Tyto bity jsou pak reprezentovány v polynomiálním tvaru v konečném poli:

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{i=0}^{7} b_i x^i \tag{A.1}$$

Například $\{01100011\}$ specifikuje prvek konečného pole $x^6 + x^5 + x + 1$, který můžeme označit v hexademcimální soustavě $\{63\}$.

Jak proces šifrování, tak i inverzní proces dešifrování je založen na několika základních transformacích. Jsou to konkrétně čtyři bajtově orientované transformace: nelineární bajtová substituce, rotace řádků, násobení maticí a přičtení rundovního klíče. Tyto operace jsou detailněji popsány v následujícím textu. Prováděny jsou na dvojrozměrném bloku mezivýsledků o velikosti 4x4 bajty označovaným jako *Stav* (State Array obr. A.1). Pole bajtů *Stav* označíme s a každý jednotlivý bajt je indexován dvěma indexy, r číslo řádku a c číslo sloupce ($s_{r,c}$). Na začátku šifrovacího nebo dešifrovacího procesu je vstupní blok nakopírován do pole *Stav* a po finální rundě nakopírován z pole *Stav* do výstupního bloku dat viz. obr. A.1 (následně je pak prováděno zřetězení výstupních bloků - bloková šifra). *Stav* a pole šifrovacího klíče bývají někdy reprezentovány jako jednorozměrné pole 32 bitových slov (sloupců), $w_0 \dots w_3$, kde číslo řádku c bude indexem pole. Potom příklad stavu



Obr. A.1: Pole bajtů stav, výstupní a vstupní pole bajtů.

z obr. A.1 bude reprezentován čtyřmi slovy:

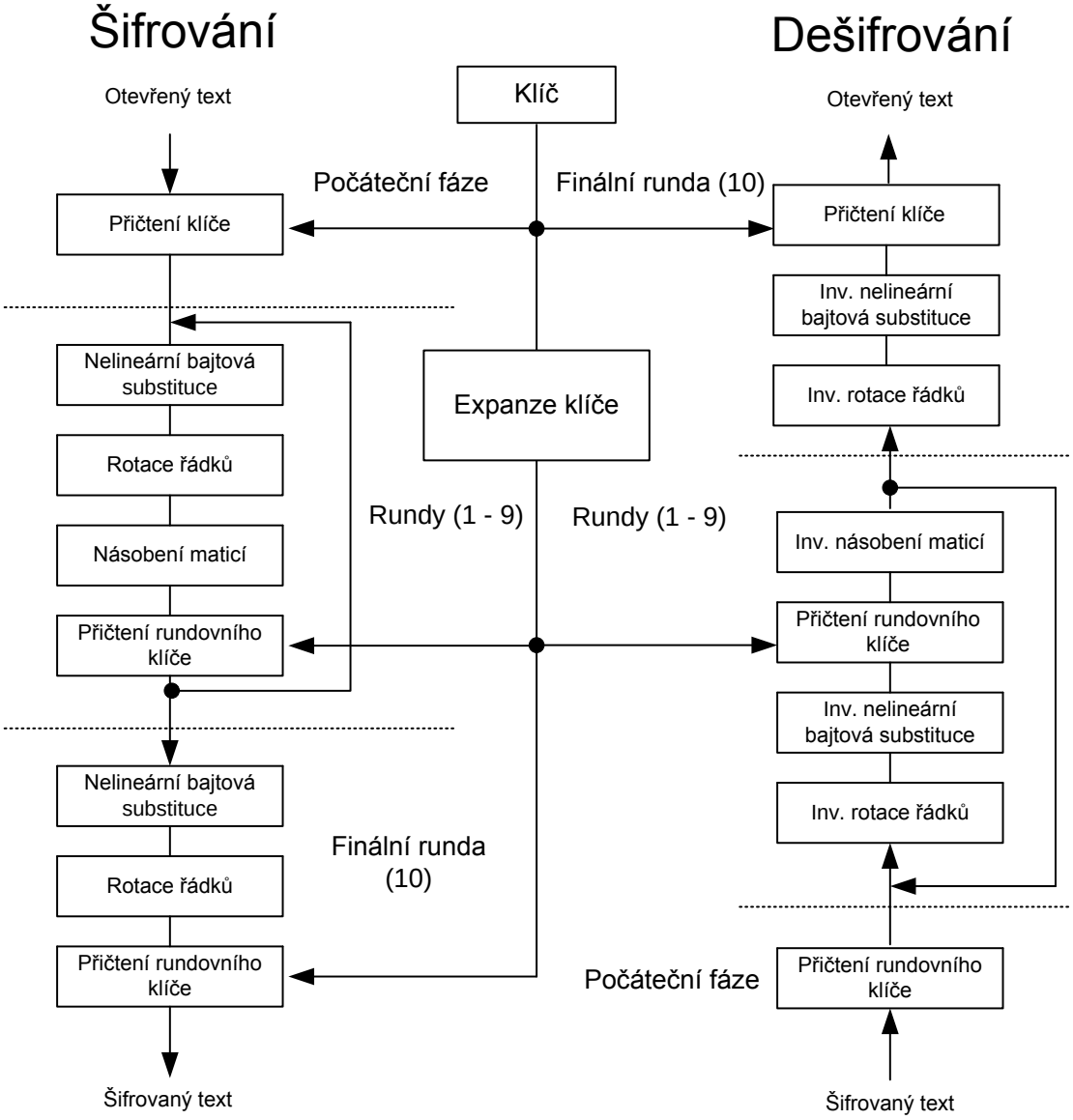
$$\begin{aligned} w_0 &= s_{0,0}s_{1,0}s_{2,0}s_{3,0}, \\ w_1 &= s_{0,1}s_{1,1}s_{2,1}s_{3,1}, \\ w_2 &= s_{0,2}s_{1,2}s_{2,2}s_{3,2}, \\ w_3 &= s_{0,3}s_{1,3}s_{2,3}s_{3,3}. \end{aligned} \tag{A.2}$$

A.1.1 Šifrování algoritmem AES

Proces šifrování standardem AES je zobrazen na levé straně obr. A.2 a lze ho rozdělit do tří základních fází: počáteční fáze, fáze kde jsou prováděny rundy a fáze finální rundy. Před samotným procesem šifrování dojde k expanzi šifrovacího klíče z původní velikosti na velikost potřebnou pro celý proces šifrování (všechny rundy). Počáteční fáze šifrování obsahuje jen jednu operaci a to „Přičtení klíče“, kde je přičten původní šifrovací klíč k poli *Stav*. V jednotlivých rundách jsou prováděny operace postupně a ze základních čtyř operací realizovaných během algoritmu AES jsou pouze operace „Přičtení klíče“ parametrizovány vstupním šifrovacím klíčem. Všechny ostatní operace jsou při šifrování a dešifrování reverzibilní a nezaručují bezpečnost, pouze konfúzi a difúzi (zmatení a rozptýlení). Ve finální rundě je vynechána operace „Násobení maticí“ a to z důvodu snadné inverze procesu dešifrování. Následující text obsahuje podrobný popis jednotlivých kroků.

Proces šifrování lze shrnout do následujících bodů:

- Expanze klíče (`Key Expansion`)
- Počáteční fáze (`Initial Round`)
  - Přičtení šifrovacího klíče (`Add Round Key`)
- Rundy (`Rounds`)
  - Nelineární bajtová substituce (`Sub Byte`)
  - Rotace řádků (`Shift Row`)
  - Násobení maticí (`Mix Column`)
  - Přičtení rundovního klíče (`Add Round Key`)
- Finální runda (`Final Round`)

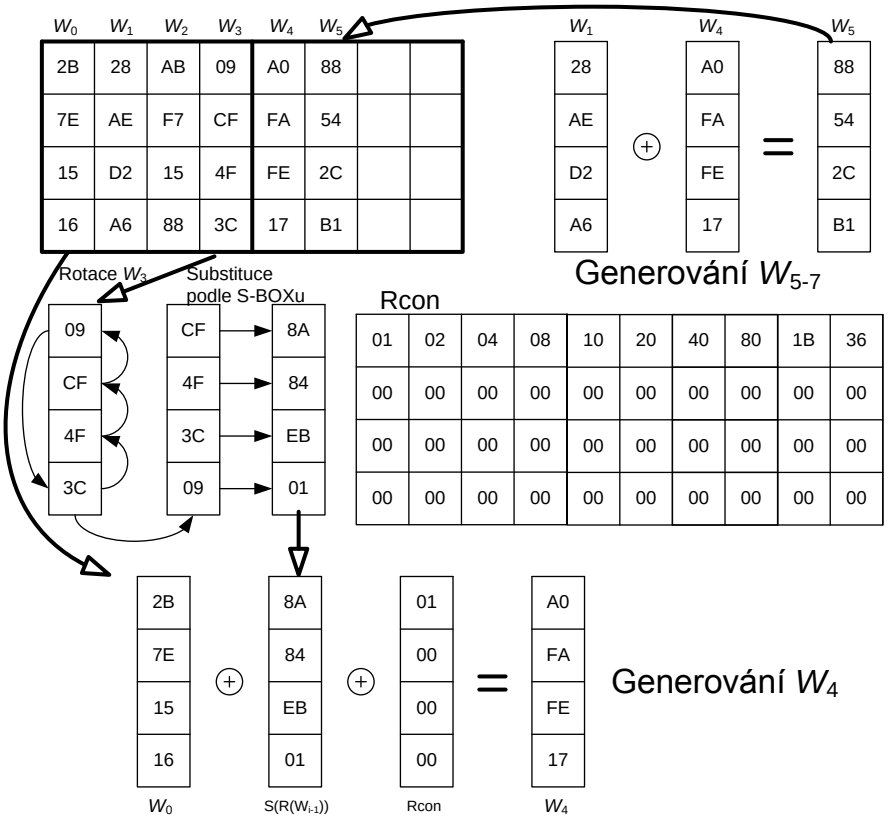


Obr. A.2: Struktura šifrování a dešifrování algoritmem AES.

- Nelineární bajtová substituce (`Sub Byte`)
- Rotace řádků (`Shift Row`)
- Přičtení rundovního klíče (`Add Round Key`)

Expanze klíče

Algoritmus AES v rámci operace expanze klíče generuje všechny potřebné rundovní klíče na základě vstupního šifrovacího klíče. Schéma plánování klíče pro šifrovací standard AES-128 je zachyceno na obr. A.3. Z obrázku je patrné, že klíče w_5, w_6 a w_7, zapsané ve formě čtyř bajtových slov, jsou generovány jednoduše pomocí operace XOR, ale klíč $w4$ je



Obr. A.3: Expanze klíče.

vytvářen daleko složitěji za pomocí několika funkcí. Tyto funkce jsou aplikovány na slovo w_3 šifrovacího klíče. Provádění funkcí je následující:

- Cyklický posun bajtů slova w_3 o jednu pozici doprava. Operace je nazývána `Rot Word`.
- Substituce posunutých bajtů dle příslušné tabulky pro S-box. Operace je nazývána `Sub Word`.
- Výsledek předchozích dvou kroků je sečten operací XOR s konstantou `Rcon`. Tato konstanta je definována pro každou rundu. Každá hodnota konstanty `Rcon` je reprezentována čtyřmi bajty. Tři nejnižší bajty jsou vždy nulové. Na obr. A.3 jsou uvedeny hodnoty pro 10 rund.

Na tomto obrázku je znázorněna expanze pro první rundovní klíč. Další klíče jsou expandovány identicky. Obecně je potřebný celkový počet $N_b(N_r + 1)$ slov, který je generovaný z N_b slov šifrovacího klíče. Každá runda pak potřebuje N_b slov rundovního klíče. Výsledná tabulka klíčů je pak lineární jednorozměrné pole čtyř bajtových slov(w_i), kde i je z rozsahu $0 \leq i < N_b(N_r + 1)$.

Nelineární bajtová substituce

Nelineární bajtová substituce zpracovává nezávisle každý bajt vstupního *Stavu* dle substituční tabulky. Substituční tabulka je získána ze dvou následujících transformací:

- Výchozí je výpočet multiplikativního inverzního prvku pro každý bajt v konečném poli $GF(2^8)$ modulo $m(x) = x^8 + x^4 + x^3 + x + 1$. Prvek $\{00\}$ je transformován sám na sebe.
- Na prvek je aplikována afinní transformace (přes $GF(2)$) dle následujícího předpisu:

$$b_i^{'} = b_{(i+4) \bmod 8} \oplus b_{(i+5) \bmod 8} \oplus b_{(i+6) \bmod 8} \oplus b_{(i+7) \bmod 8} \oplus c_i, \tag{A.3}$$

  pro $0 \leq i \leq 8$, kde b_i je i-tý bit z bajtu a c_i je i-tý bit z bajtu c o hodnotě $c = \{63\} = \{01100011\}$.

Maticově lze afinní transformaci prvků S-boxu zapsat:

$$\begin{bmatrix} b_0^{'} \\ b_1^{'} \\ b_2^{'} \\ b_3^{'} \\ b_4^{'} \\ b_5^{'} \\ b_6^{'} \\ b_7^{'} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}. \tag{A.4}$$

S-box je reprezentován v hexadecimálním vyjádření substituční tabulkou viz. obr. A.2. Například $s_{1,1} = \{53\}$ potom substituční hodnota bude určena průnikem řádku s indexem 5 a sloupce s indexem 3. Výsledkem operace je tedy $s_{1,1}^{'} = \{\text{ed}\}$.

Rotace řádků

V této transformaci jsou bajty posledních tří řádků pole *Stav* cyklicky rotovány o definovaný počet bajtů. Tímto způsobem je zajištěno, že každý výstupní sloupec obsahuje bajt ze všech vstupních sloupců. Matematicky lze celou transformaci zapsat:

$$s_{r,c}^{'} = s_{r,(c+\text{posun}(r,N_b)) \bmod N_b}, \tag{A.5}$$

pro $0 \leq r \leq 4$ a $0 \leq c \leq N_b$, kde hodnota funkce $\text{posun}(r, N_b)$ závisí na počtu řádků r. Pro $N_b = 4$ pak tedy:

$$\text{posun}(1,4) = 1,\ \text{posun}(2,4) = 2,\ \text{posun}(3,4) = 3. \tag{A.6}$$

U AES je tedy první řádek bez rotace, druhý je rotován o jeden bajt doleva, třetí o dva bajty doleva a čtvrtý o tři bajty doleva.

Násobení maticí

Tato transformace je prováděna na poli *Stav* sloupec po sloupci a pracuje s každým sloupcem jako s polynomem s koeficienty v konečném poli $GF(2^8)$. Jednotlivé sloupce jsou násobeny modulo $x^4 + 1$ s konstantním polynomem $a(s)$ definovaným takto:

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + 02. \tag{A.7}$$

Tuto operaci můžeme zapsat maticově $s^{'} = a(x) \otimes s(x)$, pro $0 \leq c \leq N_b$.

$$\begin{bmatrix} s^{'}_{0,c} \\ s^{'}_{1,c} \\ s^{'}_{2,c} \\ s^{'}_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}. \tag{A.8}$$

Výsledkem tohoto násobení jsou čtyři bajty ve sloupci nahrazeny následujícím způsobem:

$$\begin{aligned} s^{'}_{0,c} &= (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}, \\ s^{'}_{1,c} &= s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}, \\ s^{'}_{2,c} &= s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c}), \\ s^{'}_{3,c} &= (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}). \end{aligned} \tag{A.9}$$

Přičtení rundovního klíče

Během této operace je rudnovní klíč (nebo šifrovací klíč v počáteční fázi procesu šifrování) přičten operací `XOR` k poli *Stav*. Každý rundovní klíč se skládá z N_b slov z tabulky klíčů popsané v kapitole A.1.1. Tato slova jsou přičtena ke sloupcům pole *Stav* následujícím způsobem:

$$[s^{'}_{0,c}, s^{'}_{1,c}, s^{'}_{2,c}, s^{'}_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{3,c}] \oplus [w_{runda \cdot N_{b+c}}], \tag{A.10}$$

pro $0 \leq c \leq N_b$, kde $[w_i]$ je klíč vytvořený expanzí klíče a *runda* je v rozmezí $0 \leq runda \leq N_r$. Pro případ přičítání klíče v počáteční rundě je proměnná $runda = 0$. V průběhu zpracování jednotlivých rund je tedy $1 \leq runda \leq N_r$.

A.1.2 Dešifrování algoritmem AES

Postup dešifrování je odvozen od šifrování a je zobrazen na pravé polovině obr. A.2. Základním rozdílem je inverze některých operací a změna pořadí provádění jednotlivých operací. Použit je dešifrovací klíč, který je totožný se šifrovacím, ale je vyčítán v opačném pořadí. Inverzní schéma dešifrování je následující.

- Expanze klíče (`Key Expansion`)
- Počáteční fáze (`Initial Round`)
    - Přičtení rundovního klíče (`Add Round Key`)

- Rundy (`Rounds`)
    - Inverzní rotace řádků (`Inv Shift Row`)
    - Inverzní nelineární bajtová substituce (`Inv Sub Byte`)
    - Přičtení rundovního klíče (`Add Round Key`)
    - Inverzní násobení maticí (`Inv Mix Column`)
- Finální runda (`Final Round`)
    - Inverzní rotace řádků (`Inv Shift Row`)
    - Inverzní nelineární bajtová substituce (`Inv Sub Byte`)
    - Přičtení šifrovacího klíče (`Add Round Key`)

Popis jednotlivých funkcí je v podstatě identický s popisem funkcí během šifrování s tím rozdílem, že operace pracují inverzně.

A.2 Substituční tabulka S-box

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

A.3 Implementovaný program

```
;-------------------------------------------------------------
;Operace AddRoundKey
;-------------------------------------------------------------
Main
bsf LED0
movf k0,w
xorwf s0,f
movf k1,w
xorwf s1,f
movf k2,w
xorwf s2,f
movf k3,w
xorwf s3,f
movf k4,w
xorwf s4,f
movf k5,w
xorwf s5,f
movf k6,w
xorwf s6,f
movf k7,w
xorwf s7,f
;-------------------------------------------------------------
;Operace Substitution
;-------------------------------------------------------------
shiftr movlf pclath,sbox
movfcf sx,sx
movfcf s1,s1
movfcf s2,s2
movfcf s3,s3
movfcf s4,s4
movfcf s5,s5
movfcf s6,s6
movfcf s7,s7
  bcf LED0
;-------------------------------------------------------------
;S-Box
;-------------------------------------------------------------
lookup movwf pcl
        org     ($+0xff)&~0xff  ; force page alignment
```

```
sbox equ ($>>8)
dt 0x63,0x7c,0x77,0x7b,0xf2,0x6b,0x6f,0xc5,0x30,0x01,...
dt 0xca,0x82,0xc9,0x7d,0xfa,0x59,0x47,0xf0,0xad,0xd4,...
dt 0xb7,0xfd,0x93,0x26,0x36,0x3f,0xf7,0xcc,0x34,0xa5,...
dt 0x04,0xc7,0x23,0xc3,0x18,0x96,0x05,0x9a,0x07,0x12,...
dt 0x09,0x83,0x2c,0x1a,0x1b,0x6e,0x5a,0xa0,0x52,0x3b,...
dt 0x53,0xd1,0x00,0xed,0x20,0xfc,0xb1,0x5b,0x6a,0xcb,...
dt 0xd0,0xef,0xaa,0xfb,0x43,0x4d,0x33,0x85,0x45,0xf9,...
dt 0x51,0xa3,0x40,0x8f,0x92,0x9d,0x38,0xf5,0xbc,0xb6,...
dt 0xcd,0x0c,0x13,0xec,0x5f,0x97,0x44,0x17,0xc4,0xa7,...
dt 0x60,0x81,0x4f,0xdc,0x22,0x2a,0x90,0x88,0x46,0xee,...
dt 0xe0,0x32,0x3a,0x0a,0x49,0x06,0x24,0x5c,0xc2,0xd3,...
dt 0xe7,0xc8,0x37,0x6d,0x8d,0xd5,0x4e,0xa9,0x6c,0x56,...
dt 0xba,0x78,0x25,0x2e,0x1c,0xa6,0xb4,0xc6,0xe8,0xdd,...
dt 0x70,0x3e,0xb5,0x66,0x48,0x03,0xf6,0x0e,0x61,0x35,...
dt 0xe1,0xf8,0x98,0x11,0x69,0xd9,0x8e,0x94,0x9b,0x1e,...
dt 0x8c,0xa1,0x89,0x0d,0xbf,0xe6,0x42,0x68,0x41,0x99,...
;------------------------------------------------------------
Konec

goto Main
end
```

A.4 Implementace neuronové sítě

```
%*********************************************************************
% Inicializace skriptu
%*********************************************************************
close all
clear all
clc                                 % Smazání příkazového okna

disp('Inicializace skriptu');
addpath('netlab')                   % Přidání cesty k toolboxu

pocet_neuronu = 256;                % použít 128 - 256 neuronů
pocet_iteraci = 500;                % k natrénování stačí 500 cyklů
vypis = 1;  % Výpis průběhu při učení sítě (1 - zapnuto, 0 - vypnuto)

%*********************************************************************
% Vzorová data
%*********************************************************************
disp('Načtení dat');
load hlavni;
vzor = vzor(:,3000:6000-1);
pocet_mereni = length(vzor(:,1));
pocet_vzorku = length(vzor(1,:));

%*********************************************************************
% Příprava matice
%*********************************************************************
disp('Vytváření klasifikační matice')
clas = eye(256);

%*********************************************************************
% Vytvoření a trénink neuronové sítě
%*********************************************************************
 disp('Vytváření neuronové sítě');
%
nn = mlp(pocet_vzorku, pocet_neuronu, pocet_mereni,'logistic');
%Vytvoření neuronové sítě
disp('Nastavení parametrů neuronové sítě');

options = zeros(1,18);                  % Reset konfiguračního pole
```

```
options(1) = vypis;                     % Výpis chyby během učení
options(14) = pocet_iteraci;            % Počet trénovacích cyklů

disp('Trénování neuronové sítě');

tic;
[nn, options] = netopt(nn, options, vzor, clas, 'scg');
toc;
%Nastavení parametrů sítě
disp('Ukládání sítě');
naucena_sit_nn = nn;
disp('Hotovo');
```

Curriculum Vitae

Zdeněk Martinásek

Kontaktní údaje:

Bydliště:	Školní 349, Otnice, 683 54
E-mail:	martinasek@feec.vutbr.cz
GSM:	+420 774 303 173

Vzdělání:

od 2008	VUT FEKT v Brně, doktorské studium Fakulta Elektrotechniky a komunikačních technologií Obor: Teleinformatika Dizertační práce: Kryptoanalýza postranními kanály
9.2011–1.2012	Technische Universität Wien, odborná stáž Department of Teleinformatik Práce na dizertační práci (Power analysis)
2006–2008	VUT FEKT v Brně, magisterské studium Fakulta Elektrotechniky a komunikačních technologií Obor: Telekomunikační a informační technika Diplomová práce: Tenký měřič plošného teplotního rozdělení s maticí negastorů
2003–2006	VUT FEKT v Brně, bakalářské studium Fakulta Elektrotechniky a komunikačních technologií Obor: Teleinformatika Bakalářská práce: Vybrané obvody zpracování senzorových signálů

Současná pozice:

od 2008 odborný asistent, Ústav telekomunikací, VUT v Brně

Účast na projektech

2012-2014	TA02011260: Systém pro kryptografickou ochranu elektronické identity. Řešitel prof. Ing. Kamil Vrba, CSc.
2012-2014	FR-TI4/647: Integrační server s kryptografickým zabezpečením. Řešitel prof. Ing. Kamil Vrba, CSc.
2010-2013	FR-TI2/220: Výzkum modulárního systému pro komunikační technologie a ověření na 2N communication serveru. Řešitel prof. Ing. Kamil Vrba, CSc.

2011-2013	TA01031072, Inteligentní telematický informační systém veřejné dopravy. Řešitel doc. Ing. Václav Zeman, Ph.D.
2008-2010	FT-TA5/012: Decentralizované čistění odpadních vod s telemetrickým řídícím systémem pro malé obce. Řešitel prof. Ing. Kamil Vrba, CSc.
2011	3046/2011/G1: Zavedení problematiky postranních kanálů laboratorních cvičení předmětu Kryptografie v informatice. Řešitel Ing. Peter Stančík
2010	2383/2011/F1a: Inovace laboratorních úloh v kurzech zaměřených na datovou komunikaci. Řešitel Ing. Martin Koutný
2010	2829/2010/G1: Autentizační kryptografické moduly Řešitel Ing. Jiří Sobotka
2010	2534/2009/F1: Modernizace výuky předmětu Kryptografie v informatice. Řešitel doc. Ing. Otto Dostál, CSc.

Vyžádané recenze pro vědecké časopisy a konference:

- Conference on Telecommunications and Signal Processing (TSP)
- Conference on Student Electrical Engineering, Information and Communication Tech-nologies (EEICT)
- Conference on Research in Telecommunication Technologies (RTT)
- Elektrorevue - Internet Journal
- International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems - Internet Journal

Výsledky vědecké činnosti:

Počet článků v impaktovaných časopisech: **1**
Počet příspěvků na konferencích indexovaných ve Web of Science: **4**
Ostatní odborné časopisy a konference: **18**
H-index podle Web of Science: **1**

Poslední aktualizace: 11. října 2012