FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Útok postranním kanálem na šifru ChaCha
Student:	Martin Heinrich
Vedoucí:	Ing. Jiří Buček
Studijní program:	Informatika
Studijní obor:	Bezpečnost a informační technologie
Katedra:	Katedra počítačových systémů
Platnost zadání:	Do konce letního semestru 2018/19

Pokyny pro vypracování

Prostudujte metody diferenciální odběrové analýzy (DPA) a elektromagnetické analýzy (EMA) a šifru ChaCha. Implementujte šifru ChaCha na jednočipovém mikropočítači vybraném po konzultaci s vedoucím práce. Navrhněte a proveďte útok na implementaci šifry pomocí DPA, případně EMA. Vyhodnoťte míru úspěšnosti provedeného útoku.

Seznam odborné literatury

Dodá vedoucí práce.

prof. Ing. Róbert Lórencz, CSc.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 13. února 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Útok postranním kanálem na šifru ChaCha

Martin Heinrich

Katedra počítačových systémů

Vedoucí práce: Ing. Jiří Buček

15. května 2018

Poděkování

Rád bych poděkoval mému vedoucímu bakalářské práce Ing. Jiřímu Bučkovi za čas, ochotu a pomoc, které mi poskytl.

Prohlášení

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

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo"), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 15. května 2018 ……

České vysoké učení technické v Praze

Fakulta informačních technologií

Odkaz na tuto práci

Heinrich, Martin. *Útok postranním kanálem na šifru ChaCha*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Bakalářská práce teoreticky rozebírá dvě metody útoku vedlejším kanálem, a sice diferenciální odběrovou analýzu (DPA) a elektromagnetickou analýzu (EMA). Dále se zabývá aplikací DPA na proudovou šifru ChaCha, implementovanou na jednočipovém počítači, konkrétně implementovanou na AVR čipu na čipové kartě. V praktické části je nejprve proveden útok na značně zjednodušenou a zredukovanou verzi šifry ChaCha pouze s jednou rundou a následně je útok aplikován na celou šifru ChaCha.

Klíčová slova diferenciální odběrová analýza DPA, elektromagnetická analýza EMA, proudová šifra ChaCha, útok postranním kanálem

Abstract

Bachelor thesis theoretically analyzes two methods of side channel attack, namely differential power analysis (DPA) and electromagnetic analysis (EMA). It also deals with the application of DPA to a ChaCha stream cipher implemented on a single-chip computer, specifically implemented on an AVR chip on a chip card. In the practical part, the attack on the highly simplified and reduced version of the ChaCha cipher is initially executed with only one round and the attack is then applied to the entire ChaCha cipher.

Keywords differential power analysis DPA, electromagnetic analysis EMA, stream cipher ChaCha, side channel attack

Obsah

Seznam obrázků

Úvod

U šifrovacích algoritmů, které jsou považovány za kryptograficky bezpečné, je nepraktické pokoušet se takový algoritmus prolomit pomocí útoku na jeho matematický princip. Existují ovšem jiné kanály, které „vypouštějí“ do svého okolí informace o tajném klíči, používaném i v korektně implementovaných šifrovacích algoritmech. Tyto kanály jsou nazývány postranní kanály a umožňují získat tajný klíč u zranitelných implementací, ať už hardwarových nebo softwarových. Diferenciální odběrová analýza a elektromagnetická analýza patří mezi takové útoky.

V minulosti byl, ať už z hlediska kryptoanalýzy nebo z hlediska útoků, podrobně prozkoumán AES (Advanced Encryption Standard). I obecně blokové šifry jsou v publikacích podrobně a hojně zpracovány. Proudovými šiframi se však zabývá mnohem menší množství publikací.

Cílem této bakalářské práce je aplikovat DPA na šifru ChaCha, což je proudová šifra, která je založená pouze na ARX operacích (sčítání, rotace, xor). Šifra ChaCha získává na popularitě a v posledních letech se dostala do mnoha krypto systémů [2].

Práce je rozdělena na tři kapitoly. První kapitola je teoretická, představuje obě metody: diferenciální odběrovou analýzu (Differential power analysis, dále jen DPA) a elektromagnetickou analýzu (Electromagnetic analysis, dále jen EMA). Druhá kapitola kombinuje teoretickou i praktickou část, je zde popsána specifikace proudové šifry ChaCha, její implementace, realizovaná na zvolené platformě, a jsou zde rozebrány možnosti útoku. Třetí kapitola popisuje prostředí a nástroje, které byly při analýze použity. Dále se pak zabývá samotným útokem na šifru ChaCha a jeho vyhodnocením.

Kapitola 1

Útoky DPA a EMA

První kapitola představí problematiku metod DPA a EMA.

1.1 Princip DPA a EMA

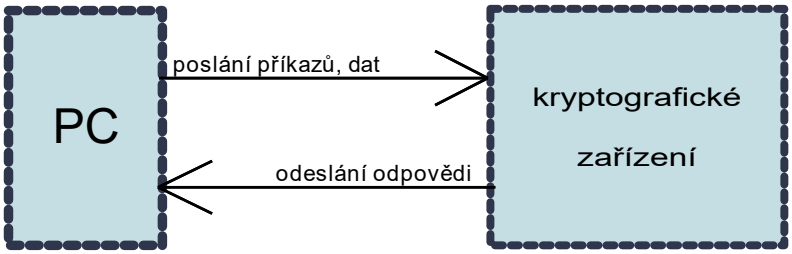
Ještě předtím, než bakalářská práce konkrétně představí principy metod DPA a EMA, je vhodné uvést čtenáře do problematiky.

V moderních informačních systémech je často třeba zajistit důvěrnost, integritu a autenticitu dat. Pro zajištění těchto aspektů se používají kryptografické algoritmy. Kryptografický algoritmus je matematická funkce, o které lze zjednodušeně říci, že bere na svém vstupu zprávu a klíč a na výstupu dává zašifrovanou zprávu. Důležitou vlastností kryptografických algoritmů je nutnost mít někde klíč uložený. Typické PC, tablety, telefony ad. jsou ovšem poměrně nebezpečná zařízení pro ukládání citlivých dat, jako je třeba právě zmiňovaný kryptografický klíč. Znalost cizího klíče může někomu umožnit např. přihlásit se do internetových služeb svázaných s tímto klíčem, a tedy vystupovat jako jiná osoba.

Pro ukládání kryptografických klíčů jsou mnohem vhodnější uzavřené platformy, jako např. smart karty. Smart karty jsou kryptografická zařízení, která nemají běžně přístup k internetu, nedovolují instalovat software, přesto lze jejich bezpečnost prolomit např. pomocí DPA.

> „*Kryptografická zařízení jsou elektronická zařízení, která implementují kryptografické algoritmy, a která ukládají tajný klíč.*“ [1]

Kryptografická zařízení, jako např. smart karty, jsou schopny provádět kryptografické operace a výsledek těchto operací poslat zpět jinému zařízení.



Obrázek 1.1: Komunikace mezi PC a kryptografickým zařízením

Jak může vypadat komunikace mezi kryptografickým zařízením a uživatelským počítačem, lze vidět na obrázku 1.1.

1.1.1 Útoky na kryptografická zařízení

V rámci této práce, pokud je popisován útok na kryptografická zařízení, jedná se vždy o útok postranním kanálem. Útoky postranním (vedlejším) kanálem jsou takové útoky, které se nepokoušejí prolomit matematický princip kryptografického algoritmu. Raději využívají fyzických vlastností daného kryptografického zařízení, a to ať už se jedná o spotřebu energie, emise elektromagnetického pole kolem zařízení nebo měření délky výpočtů.

Různé druhy útoků na kryptografická zařízení je možné rozdělit z hlediska způsobu na:

- **Pasivní:** Pasivní útok znamená, že zařízení není nijak ovlivňováno. Posílání standardního vstupu není považováno za ovlivňování zařízení. K odhalení tajného klíče dojde prostým pozorováním fyzických vlastností zařízení a analýzou takto získaných dat (např. čas potřebný pro výpočet kryptografické operace, spotřeba energie, které zařízení spotřebuje při výpočtu kryptografické operace, emise elektromagnetického pole).
- **Aktivní:** Při aktivním útoku je zařízením, jeho vstupními daty a prostředím, ve kterém se zařízení nachází, manipulováno. Cílem je dosáhnout toho, aby se zařízení chovalo abnormálně, tzn. vyvolat poruchu, která se projeví chybou ve výpočtu.

Z hlediska používání zařízení dělíme útoky na:

- **Invazivní:** Při invazivním útoku nejsou kladena žádná omezení na způsob, jakým je se zařízením nakládáno. Z tohoto důvodu se jedná o nejsilnější útok, jaký může být na zařízení aplikován.

Při invazivním útoku je možné celé zařízení rozebrat, a tedy jednotlivě přistupovat k jeho komponentám. Měřící sondy mohou být napojeny přímo na jednotlivé tranzistory, dochází k porušení pasivační vrstvy. Pokud jsou pouze pozorovány procesy v zařízení, jedná se o útok pasivní.

Invazivní útoky jsou sice nejmocnější, ale také nejdražší z hlediska potřebných prostředků pro jejich provedení.

- **Částečně invazivní:** V částečně invazivním útoku je zařízení také rozebráno, není však porušeno. To znamená, že po provedení útoku je toto zařízení možné znovu složit do původní podoby. Pasivační vrstva není porušena.
- **Neinvazivní:** V neinvazivním útoku je přistupováno pouze k přímo dostupným komponentám daného zařízení. Zařízení není nijak rozebráno ani nejsou zanechány žádné stopy po provedeném útoku.

  Neinvazivní útoky jsou také poměrně nenáročné na potřebné vybavení. Typicky postačuje např. osciloskop a pomocné zařízení, vložené mezi kryptografické zařízení a napájecí zdroj (např. čtečka karet).

  V kombinaci absence důkazů o provedení útoku a jeho finanční nenáročnosti je právě tento druh útoku praktickou a vážnou hrozbou pro bezpečnost kryptografických zařízení.

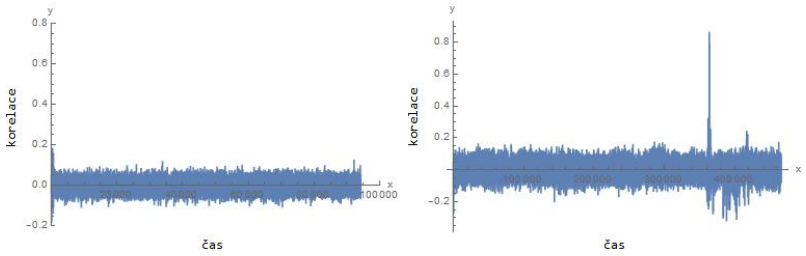
Pro vyhodnocení bezpečnosti kryptografických zařízení se předpokládá, že útočník má přístup nejen k zařízení samotnému, ale plně ovládá i vstupy, které do zařízení posílá. Zároveň má znalost o podrobnostech implementace kryptografických algoritmů, které jsou v daném zařízení použité.

1.2 Diferenciální odběrová analýza

Diferenciální odběrová analýza je pasivní neinvazivní útok postranním kanálem.

> „*Diferenciálni odběrová analýza využívá faktu, že okamžitá spotřeba energie kryptografického zařízení závisí na datech, které zařízení zpracovává, a na operacích, které právě provádí.*“ [1]

Základním principem DPA je změřit spotřebu energie, která je zapotřebí ke zpracování co největšího množství dat kryptografickými operacemi na daném kryptografickém zařízení. Následně se změřené průběhy podrobí statistické analýze a hledá se korelace mezi daty a tajným klíčem. To se provádí tak, že se vytvoří model spotřeby, kde se mapují hodnoty použitých dat ve výpočtu



(a) Hypotéza špatného bytu klíče (b) Hypotéza správného bytu klíče

Obrázek 1.2: Ukázka správné hypotézy o klíči (vpravo) a špatné (vlevo)

k hodnotám spotřeby. Podle použitého modelu se vytvoří hypotéza o klíči. Při vytváření hypotézy se nejdříve odhadne (dle modelu spotřeby), jaká hodnota spotřeby je očekávaná pro hodnotu klíče. Klíč není třeba hádat celý najednou, hypotéza se tvoří např. po bytech klíče. Tím dostaneme pouze 256 možných hodnot a pro všech 256 hodnot se vytvoří hypotéza. Následně se hypotézy korelují se skutečně naměřenými daty. Příklad, jak může vypadat hypotéza špatného a správného klíče, je na obrázku 1.2.

Pro vyhodnocení korelací je možné použít Pearsonův korelační koeficient. Pearsonův korelační koeficient je běžnou metrikou ve vyhodnocování lineárních korelací mezi daty a jeho použití je vhodné pro DPA. [1]

Formální definice Pearsonova korelačního koeficientu je:

$$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}, \tag{1.1}$$

kde

- X, Y jsou náhodné veličiny,
- $cov(X,Y)$ je kovariance náhodných veličin X, Y,
- σ_X je směrodatná odchylka veličiny X.

Pro účely DPA je využíván výběrový korelační koeficient, který je počítán z n měření a je definován jako:

$$\tau_{X,Y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_X s_Y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}, \tag{1.2}$$

kde

- x_i je i-té měření veličiny X,
- $\bar{x}$ je výběrový průměr veličiny X,
- s_X je výběrová směrodatná odchylka veličiny X.

Pearsonův korelační koeficient je definován na oboru hodnot $[-1; 1]$, kde 0 značí žádnou lineární závislost mezi hypotézou a daty.

Právě možnost statisticky analyzovat co největší množství dat o měření spotřeby dělá z DPA velmi silný útok, kdy ani není nutné znát implementační podrobnosti o kryptografickém zařízení. Postačuje informace o použitých algoritmech (např. ChaCha20 nebo AES). Nevýhodou ovšem je, že bez neomezeného přístupu k zařízení nemusí být možné naměřit dostatečné množství hodnot průběhu výpočtů a získat tak tajný klíč.

1.2.1 Základní modely DPA

V této podkapitole jsou představeny dva modely spotřeby. Jedná se o Hammingovu váhu a Hammingovu vzdálenost.

Oba modely jsou běžně používané pro DPA a lze je použít obecně pro jakékoliv zařízení. [1]

Hammingova váha

Hammingova váha (Hamming Weight, dále jen HW) je definována jako počet bitů nastavený na „1“. Je jednodušším modelem spotřeby, než je Hammingova vzdálenost (Hamming Distance, dále jen HD), která je popsána níže. HW je také o něco méně „mocná“. Tento model je obvykle používán tehdy, pokud není dostatek informací o po sobě jdoucích datových mezihodnotách ve výpočtu kryptografické operace. To znamená, pokud datové hodnoty, které jsou známé, na sobě nejsou závislé. Například známé datové hodnoty jsou jen data, která se posílají kryptografickému zařízení, a data, která se tímto zařízením odesílají v jeho odpovědi.

V případě HW modelu spotřeby se předpokládá, že spotřeba energie odpovídá počtu nastavených „1“ v příslušných právě zpracovávaných bitech. Vůbec nejsou brány v úvahu datové hodnoty, které se zpracovávají před nebo po šifrování. Tento model sice neodpovídá realitě v CMOS obvodech, přesto praxe ukazuje, že HW právě zpracovávané hodnoty alespoň částečně odpovídá reálné spotřebě energie. Nicméně pokud je to možné, pak je preferován HD model spotřeby.

„Útočníci používají HW model pouze v případech, kdy není možné použít HD model.“ [1]

Hammingova vzdálenost

Hammingova vzdálenost (Hamming Distance, HD) dvou hodnot h_0 a h_1 je definována jako:

$$HD(h_0, h_1) = HW(h_0 \oplus h_1). \tag{1.3}$$

Tedy pro použití HD modelu je třeba znát alespoň dvě po sobě jdoucí datové hodnoty.

Podstatou HD modelu je počítání bitových změn v určitém časovém intervalu. HD model spotřeby předpokládá, že bitové změny $0 \rightarrow 1$ a $1 \rightarrow 0$ spotřebovávají stejné množství energie. Stejně tak HD model nerozlišuje mezi $0 \rightarrow 0$ a $1 \rightarrow 1$, v případě žádné změny model předpokládá nulovou statickou spotřebu.

Podobnost mezi HD a HW

V některých speciálních případech se HW i HD modely rovnají. Pokud máme hodnoty h_0 a h_1 o stejném počtu n bitů, při kryptografické operaci h_0 přechází na h_1 a h_0 má všechny bity nastavené na „0“, potom platí:

$$HD(h_0, h_1) = HW(h_0 \oplus h_1) = HW(0 \oplus h_1) = HW(h_1). \tag{1.4}$$

Pokud všechny bity h_0 jsou nastavené na „1“, potom platí:

$$HD(h_0, h_1) = HW(h_0 \oplus h_1) = HW(11\ldots1_2 \oplus h_1) = n - HW(h_1). \tag{1.5}$$

Další modely

Kromě HW a HD modelu existují i jiné modely spotřeby. Na rozdíl od obou výše popsaných již ale typicky nebývají obecné. Např. pokud útočník má znalost, že bitová změna $0 \rightarrow 1$ má větší spotřebu energie než $1 \rightarrow 0$, může patřičně upravit HD model.

V souvislosti s dalšími modely spotřeby je třeba zmínit, že DPA tak, jak byla poprvé navržena v [3], nepracovala ani s HW, ani s HD modelem spotřeby. Nepoužívala také Pearsonův korelační koeficient. Tato původní metoda DPA

pracovala pouze s hypotetickými hodnotami jednotlivých bitů, nikoliv bytů. DPA, popsaná a definovaná v této bakalářské práci, se v jiných publikacích někdy označuje jako korelační odběrová analýza (Correlation Power Analysis, CPA), což lze chápat jako podmnožinu DPA.

1.2.2 Hypotézy o klíči

Pomocí výše popsaných modelů se tvoří hypotézy o tajném klíči. Hypotéza o klíči se vytvoří pro všech 256 možných hodnot jednoho bytu a vyjadřuje očekávanou spotřebu dle daného modelu šifrovacího algoritmu a pro daný byte klíče. Vytvoření hypotéz o klíči a jejich následnou korelaci se skutečně naměřenými hodnotami podrobně znázorňuje diagram 1.3.

1.3 Elektromagnetická analýza

Podobně jako DPA je i elektromagnetická analýza pasivním a neinvazivním útokem pomocí postranního kanálu.

Důvodem, umožňujícím úspěšně provádět EMA, je skutečnost, že každé elektronické zařízení vysílá do svého okolí elektromagnetické signály (pokud pracuje). Jedná se o triviální důsledek faktu, že každý vodič, kterým protéká proud, kolem sebe vytváří magnetické pole. A právě toto magnetické pole může odhalovat informace o právě probíhajících operacích v zařízení.

1.3.1 Rozdíly oproti DPA

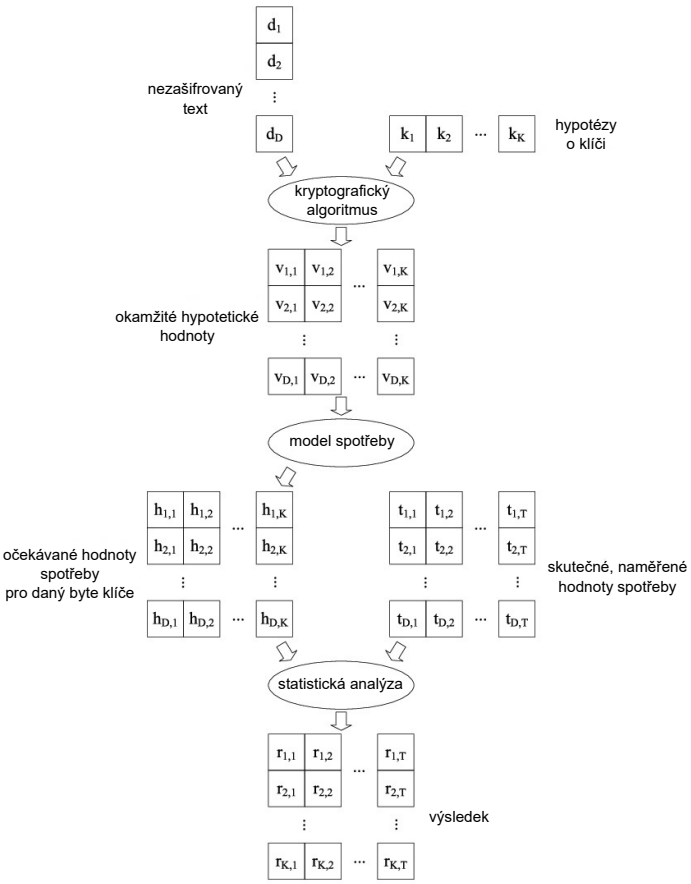
Stejně jako u DPA jsou i u EMA provedena měření a následně se na tato měření aplikuje statistická analýza. Lze použít shodné modely útoku jako u DPA. EMA se převážně liší pouze způsobem získávání dat. Na rozdíl od DPA není potřeba pomocné zařízení, na kterém se měří spotřeba, ale postačuje kryptografické zařízení samotné. Zato však potřebujeme měřit elektromagnetické pole pomocí antény (sondy blízkého pole).

Při použití EMA útoku postranním kanálem útočníka zajímají pouze frekvence, ve kterých probíhají kryptografické operace.

Frekvence, ve kterých kryptografické operace probíhají, nejsou pevně dané, ale závisejí na jednotlivém zařízení. K isolování určitých frekvencí lze použít např. BPF filtr (pásmová propust = Bandpass Filter). V jednoduchosti lze říci, že BPF filtr je elektronické zařízení, které propouští určitý pevný rozsah frekvencí a ostatní zamítá (v ideálním případě). Pokud ovšem není jasné, které frekvence patří ke kryptografickým operacím, lze si v takových případech zobrazit naměřená data jako spektrogram. Spektrogram je grafickým zobrazením spektra frekvencí elektromagnetických signálů, zvuku nebo jiných signálů v zá-

vislosti na čase. Grafická podoba dat může pomoci odlišit relevantní frekvence (tj. frekvence, v nichž se provádějí kryptografické operace) od tzv. šumu.

Dalším důležitým rozdílem od DPA je fakt, že pro EMA je zajímavá a dost podstatná i poloha měřící sondy. Pro efektivní EMA útok je třeba zachytit pokud možno co nejsilnější elektromagnetický signál, vyzařovaný daným kryptografickým zařízením. Zároveň signál nemusí být neměnný na daném místě umístění sondy, ale mění se v průběhu provádění kryptografických operací. Je proto vhodné mít k dispozici nějaký „polohovač“, který měřící sondu automaticky umísťuje na místo nejsilnějšího signálu.



Obrázek 1.3: Vytvoření hypotéz a jejich korelace, zdroj [1]

Kapitola 2

Šifra ChaCha a použitá implementace

ChaCha je proudová šifra, která je založená pouze na ARX operacích. Poprvé byla publikována jejím autorem D. Bernsteinem v roce 2008 [4]. Tato šifra je založená na šifře Salsa, jejímž autorem je také D. Bernstein. ChaCha (i Salsa) je 256bitová proudová šifra s typicky 20 rundami. ChaCha je definována také pro 12 a 8 rund. Většina implementací ovšem používá plnou 20 rundovou variantu [2]. V závislosti na počtu použitých rund se na šifru odkazuje jako na ChaCha20, ChaCha12 nebo ChaCha8. ChaCha byla navržena tak, aby vylepšila difuzi bitů v „proudovém klíči“ (keystreamu) a celkově tedy zvýšila svou rezistenci oproti kryptoanalýze a různým útokům postranním kanálem (narozdíl od své předchůdkyně Salsy) [4]. Vzhledem k tomu, že ChaCha používá pouze ARX operace, je tato šifra imunní proti útokům časovým kanálem. Pro srovnání AES implementovaný bez jakýchkoliv ochran není proti tomuto typu útoku imunní [5].

Zejména v posledních letech nabývá ChaCha na popularitě a dostala se od svého vzniku již do mnoha krypto systémů. Uplná historie vývoje ChaChy včetně výhledu do budoucna je v [2].

2.1 Princip

Úvodní stav šifry ChaCha zobrazuje následující matice:

$$\begin{matrix} c_0 & c_1 & c_2 & c_3 \\ k_0 & k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 & k_7 \\ b_0 & n_0 & n_1 & n_2 \end{matrix} \tag{2.1}$$

kde každá hodnota značí 32bitové slovo. Význam jednotlivých hodnot je:

- c – konstanty,
- k – hodnoty 256bitového klíče,
- b – čítač bloku,
- n – tzv. nonce (tj. hodnota, použitá striktně jednou pro dané šifrování).

Konstanty c jsou pevně definované hodnoty standardem ChaChy [6]:

$$\begin{aligned} c_1 &= 0\text{x}61707865, \\ c_2 &= 0\text{x}3320646e, \\ c_3 &= 0\text{x}79622d32, \\ c_4 &= 0\text{x}6b206574. \end{aligned} \tag{2.2}$$

Jednotlivé hodnoty b a n jsou v rámci této práce chápány jako hodnoty, které lze aktivně ovládat dle potřeby. Dle [6] se jedná o jedno slovo čítače bloku a tři slova tzv. „nonce“, tedy hodnoty, které jsou použité pouze jednou pro dané šifrování. ChaCha, použitá v různých krypto knihovnách (např. LibreSSL) nebo krypto systémech, se implementačně drží standardu [6]. Stejně tak se ho drží i implementace, vytvořená pro tuto práci. Od originalního návrhu [4] se [6] liší pouze v posledním řádku stavové matice, tedy hodnotách b a n, kde jako čítač bloku jsou použitá dvě 32bitová slova a pouze dvě slova jsou použitá pro „nonce“.

Po inicializaci úvodního stavu šifry jsou postupně aplikovány jednotlivé rundy na daný stav a výsledný stav se do úvodního jednoduše přičte operací plus. Operace plus je u ChaCha algoritmu definována pro 32bitovou aritmetiku, tedy veškeré výsledky operace plus jsou moduleny 2^{32}. Tím je vytvořen 512bitový „proudový klíč“ (anglický ekvivalent je keystream). Tento „proudový klíč“ je následně jednoduše vyxorován s nezašifrovanými daty o maximální velikosti 512 bitů. Další „proudový klíč“ je vytvořen opakovaným zavoláním šifry na úvodním stavu s jinou hodnotou b_0 (blokovým čítačem).

Princip ChaChy připomíná chovaní blokové šifry v „counter“ módu.

2.1.1 Runda

Každá jednotlivá runda mění stav šifry 2.1 střídavě po sloupcích a po diagonále. Tedy každá sudá runda je sloupcová, každá lichá diagonální. Dvě po sobě jdoucí rundy, které začínají sudou rundou, pak vypadají následovně:

```
// sloupce
quarterRound( 0, 4, 8, 12 );
quarterRound( 1, 5, 9, 13 );
quarterRound( 2, 6, 10, 14 );
quarterRound( 3, 7, 11, 15 );

// diagonály
quarterRound( 0, 5, 10, 15 );
quarterRound( 1, 6, 11, 12 );
quarterRound( 2, 7, 8, 13 );
quarterRound( 3, 4, 9, 14 );
```

Listing 1: Průběh dvou rund

2.1.2 Čtvrtrunda

Každá jednotlivá runda se skládá ze 4 čtvrtrund, které jsou definovány v 2.3.

$$\begin{gathered} a = a + b;\ d = d \text{ xor } a;\ d <<<= 16; \\ c = c + d;\ b = b \text{ xor } c;\ b <<<= 12; \\ a = a + b;\ d = d \text{ xor } a;\ d <<<= 8; \\ c = c + d;\ b = b \text{ xor } c;\ b <<<= 7; \end{gathered} \tag{2.3}$$

Operace „+“ je definována pro dvě 32bitová čísla, tedy modulo 2^{32}. Operace „$<<< n$“ je levá rotace o n bitů.

V použité implementaci je čtvrtrunda naprogramována následovně:

```
void quarterRound(uint32_t * a, uint32_t * b, uint32_t * c,
↪  uint32_t * d)
{
    *a = (*a + *b);
    *d ^= *a;
    *d = rotate(*d, 16);

    *c = (*c + *d);
    *b ^= *c;
    *b = rotate(*b, 12);

    *a = (*a + *b);
    *d ^= *a;
    *d = rotate(*d, 8);

    *c = (*c + *d);
    *b ^= *c;
    *b = rotate(*b, 7);
}
```

Listing 2: Implementace čtvrtrundy

Vzhledem ke specifikaci jazyka C, použitého pro implementaci algoritmu ChaCha, je zde modulo 2^{32} u operace „+“ docíleno prostým přetečením.

2.1.3 Endianita ChaChy

Veškeré operace nad jednotlivými stavy algoritmu jsou definovány jako „little–endian“ operace. Standardizace vstupních dat je v použité implementaci docílena převedením veškerých vstupů bez ohledu na endianitu systému, viz 3.

```
uint32_t convertString(unsigned char * s)
{
    uint32_t res;

    res = (uint32_t) s[0] | (uint32_t) ((uint32_t) s[1] << 8) |
    ↪  (uint32_t) ( (uint32_t) s[2] << 16) | (uint32_t)
    ↪  ((uint32_t) s[3] << 24);

    return res;
}
```

Listing 3: Konverze vstupních dat

Kapitola 3

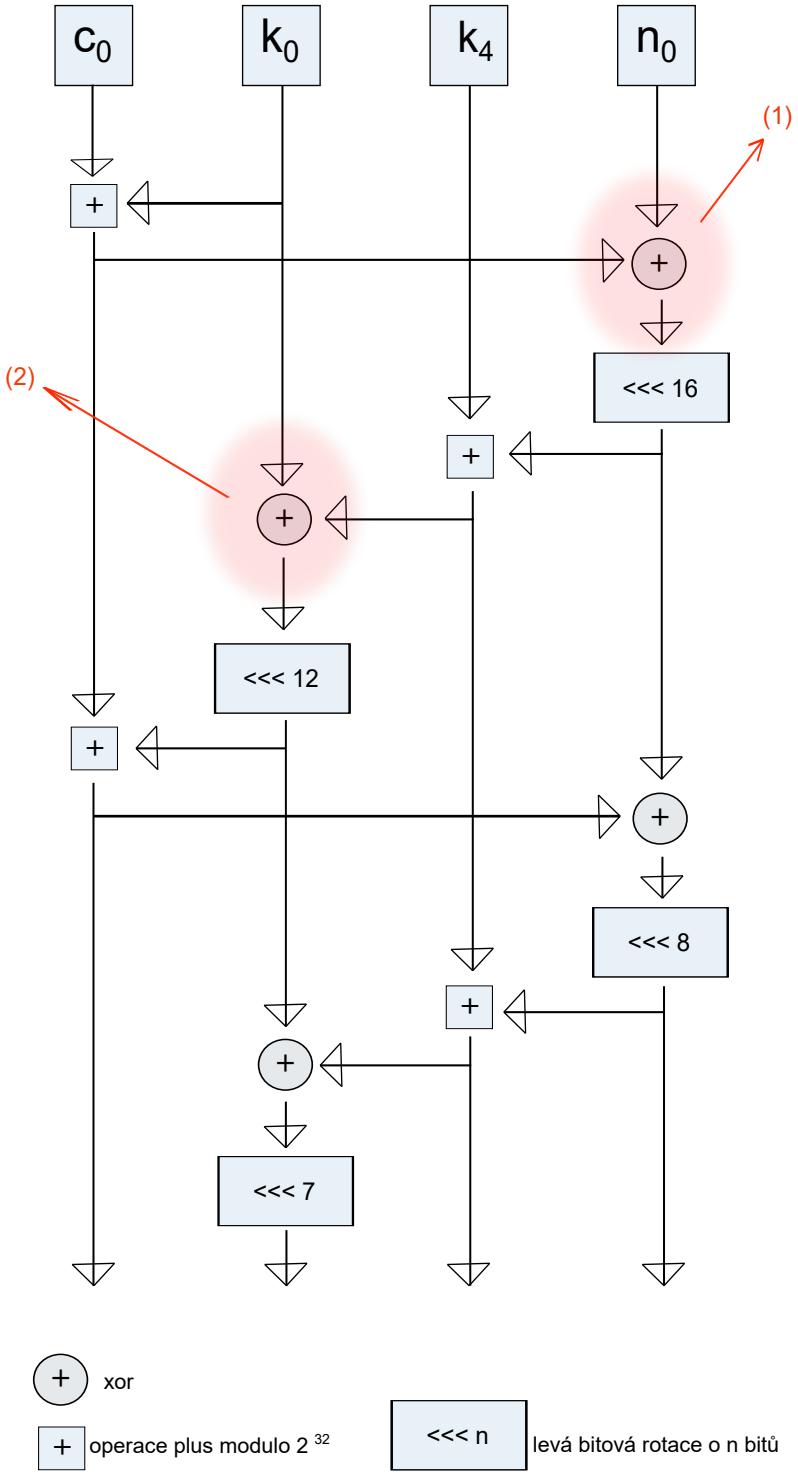
Návrh a realizace útoku

3.1 Návrh útoku

Šifra ChaCha, oproti své předchůdkyni Salse, vylepšuje difuzi jednotlivých bitů v „proudovém klíči“. Z tohoto důvodu je jako místo útoku zvolena první runda, kde se vyskytují původní nezměněné hodnoty tajného klíče.

Na obrázku 3.1 je vizualizace čtvrtrundy algoritmu ChaCha pro první probíhající čtvrtrundu nad počátečním stavem šifry, viz 2.1. Zvýraznění vyznačuje dvě potenciální místa pro útok. Na zvýrazněných místech ještě figurují původní hledané hodnoty tajného klíče. Později, díky velké difuzi bitů při jednotlivých rundách, je již získání tajného klíče nepravděpodobné.

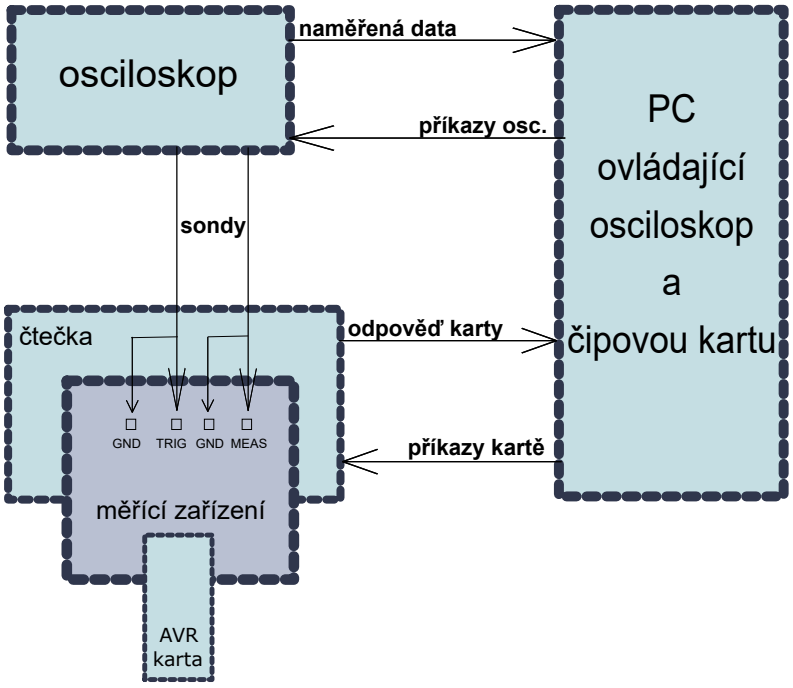
Další místo výskytu tajného klíče vznikne po proběhnutí všech rund, kdy se vytvořená stavová matice přičte do původní stavové matice. Operace plus je ovšem silně lineární matematická operace a vzhledem k tomu, že DPA spoléhá na hledání lineárních závislostí mezi daty, je toto místo nevhodné pro uskutečnění útoku pomocí DPA nebo EMA.



Obrázek 3.1: Grafické zobrazení první čvtrtrundy algoritmu ChaCha

3.2 Prostředí, ve kterém bylo uskutečněno měření

Měření dat, potřebných pro DPA, proběhlo pomocí vybavení RFID laboratoře ČVUT FIT. K veškerému měření byl použit osciloskop Agilent Technologies MSO6104A, viz obrázek 3.5. Postup měření lze charakterizovat diagramem 3.2.



Obrázek 3.2: Diagram měřícího prostředí

Jako čtečka čipové karty byla vybrána GemPC Twin, viz obrázek 3.3. Dále bylo použito zařízení pro měření spotřeby, které je zobrazeno na obrázku 3.4. Umístění tohoto zařízení mezi čtečku a čipovou kartu je uvedeno na diagramu 3.2. Jako samotné **kryptografické zařízení** byla zvolena čipová karta ATMega s procesorem ATMega163, viz obrázek 3.3. Jedná se o 8bitový procesor, a ačkoli je použitá implementace ChaChy optimalizovaná pro 32bitovou platformu, její použití na 8bitové platformě je pouze mírně pomalejší, nicméně stále korektní. Čipová karta ATMega byla vybrána zejména na základě dobrých zkušeností s algoritmem AES a jeho DPA analýzou na této platformě.

Nahrání implementovaného algoritmu proběhlo pomocí elektronického zařízení, které je zobrazeno na obrázku 3.4. Pro nahrání ChaChy a firmwaru do čipové karty byl použit volně dostupný program „avrdude“ a skripty, dostupné v rámci magisterského předmětu MI–HWB (Hardwarová bezpečnost), vyučo-

vaného na FIT ČVUT. Zároveň byl použit program pro ovládání osciloskopu a posílání dat čipové kartě, který je dostupný v rámci stejného předmětu rovněž na FIT ČVUT. Tento program tak, jak je dostupný v původní podobě, je připraven pro použití v kombinaci s algoritmem AES–128. Odesílá a přijímá pouze 16 bytů. To je pro šifru ChaCha nevhodné. ChaCha může v jedné iteraci poslat až 64 bytů. Pro potřeby této práce byl proto zmiňovaný program upraven tak, aby řídícímu PC posílal v odpovědi až 64 bytů. 16 bytové vstupy jsou pro inicializaci úvodní stavové matice dostatečné (více viz 3.1).

Následná analýza dat byla provedena pomocí počítačového algebraického systému Mathematica (lze samozřejmě použít i jiné matematické programy, např. Matlab, SageMath atd.). Základ skriptů pro Mathematicu byl převzat ze sady skriptů, používaných ve výuce bakalářského předmětu BI–HWB (Hardwarová bezpečnost), vyučovaného též na FIT ČVUT.

3.2.1 Popis osciloskopu

Na obrázku 3.5 je osciloskop Agilent Technologies MSO6104A. Nejzajímavější části osciloskopu pro nastavení měření jsou dle obrázku:

(1) vertikální a horizontální pozicování kanálu č. 1, používá se pro detekci spouštění (Triggeru),

(2) vertikální a horizontální pozicování kanálu č. 2, používá se pro měření spotřeby,

(3) vertikální pozicování měřeného signálu,

(4) horizontální pozicování měřeného signálu.

3.2.2 Postup měření

Nejprve se do čipové karty pomocí programátoru 3.4 a ovládajícího PC (či jiné platformy) nahraje program, jehož průběh bude měřen a následně analyzován.

Po nahrání programu a zapojení všech zařízení dle schématu 3.2 jsou do čipové karty posílána pseudonáhodná data. Čipová karta posílá ve své odpovědi všech 64 bytů vytvořeného „proudového klíče“. Při celém procesu se měří spotřeba čipové karty. Zapojení sond na měřící zařízení je vidět na obrázku 3.6. Sonda odpovídající kanálu č. 1 je napojena na „TRIG a GND (uzemnění)“, sonda odpovídající kanálu č. 2 na „MEAS a GND“. „TRIG“ je kanál, který je v měřené implementaci ChaChy použit pro indikaci: „zde začíná zajímavá část výpočtu kryptografického zařízení“.

(a) Čtecka čipových karet GemPC Twin

(b) Čipová karta ATMega

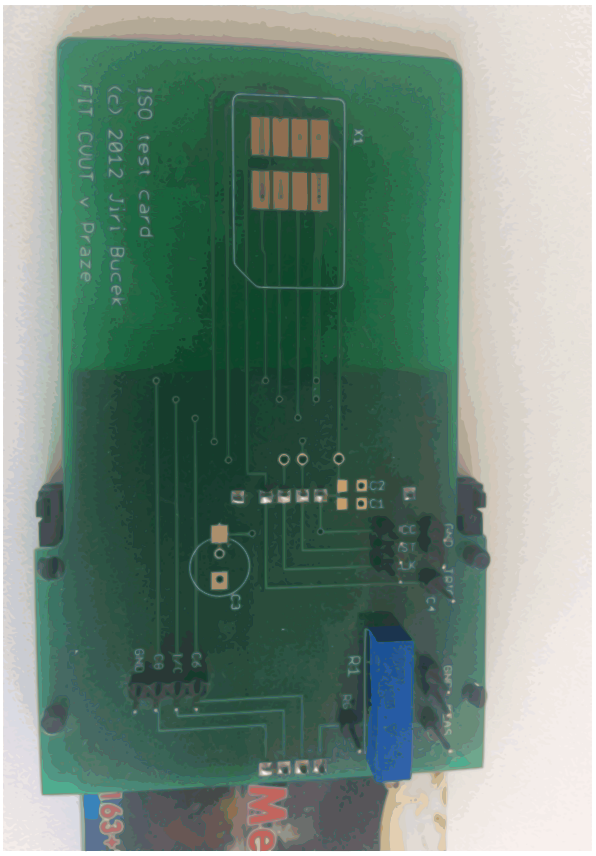
Obrázek 3.3: Komponenty laboratorního prostředí I.

Ukázka použití „TRIG“ v analyzovaném programu je např.:

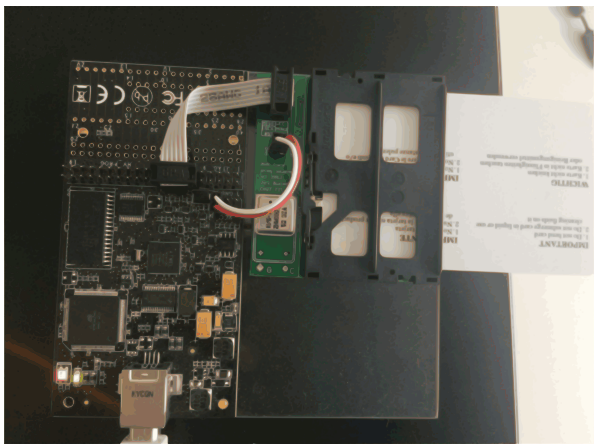
```
// nastavení trigr PINu
set_pin(DDRB, 0b10100000); // směr
set_pin(PORTB, 0b10100000); // hodnota
// jen čtvrtrunda
quarterRound( 0, 4, 8, 12 )
clear_pin(PORTB, 0b01011111); // konec zajímavého měření
```

Listing 4: Použití „TRIG“

PC posílá čipové kartě 16 bytů vstupních dat. Jedná se o hodnoty b_0, n_0, n_1, n_2, viz 2.1.

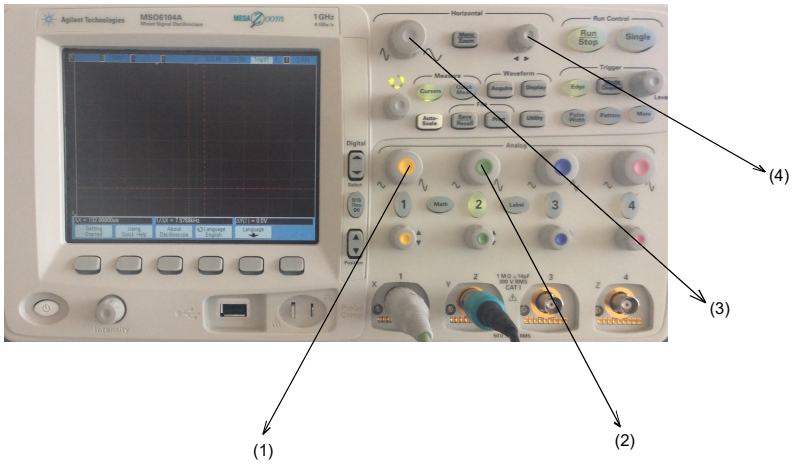


(a) Zařízení pro měření spotřeby



(b) Zařízení pro naprogramování čipové karty

Obrázek 3.4: Komponenty laboratorního prostředí II.



Obrázek 3.5: Osciloskop Agilent Technologies MSO6104A

Čipová karta vytvoří „proudový klíč“ šifry ChaCha, který odešle zpět PC, viz znázornění na diagramu 3.2. Počet rund, případně čtvrtrund, které čipová karta provede, závisí na měřeném experimentu, více viz sekce 3.3.

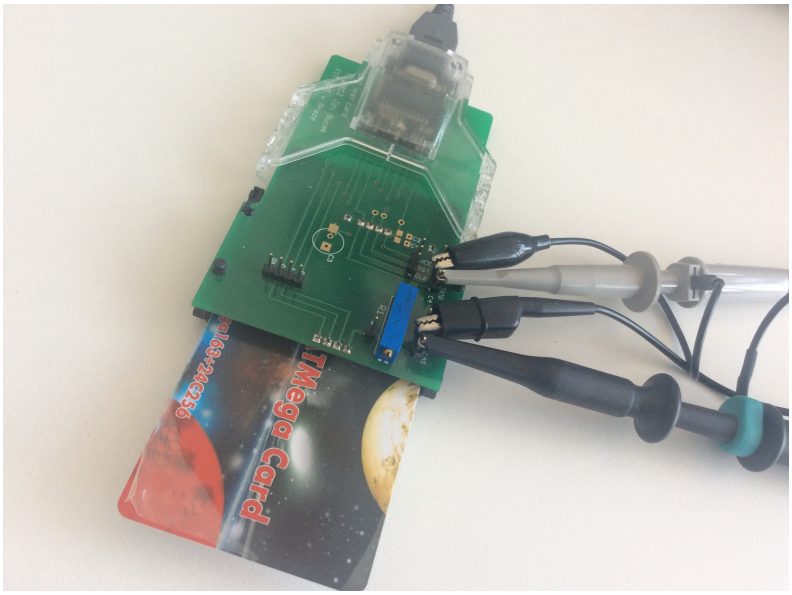
Posílaná data jsou ukládána do souboru „plaintext.txt“, odpovědi čipové karty do souboru „ciphertext.txt“ a naměřená spotřeba do souboru „traces.bin“. Viz příloha B.

3.2.3 Shrnutí

Pro kompletní naprogramování použitého kryptografického zařízení a pro naměření potřebných dat je třeba mít k dispozici:

- osciloskop,
- čtečku čipových karet,
- programovatelnou čipovou kartu,
- zařízení pro měření spotřeby, viz 3.4,
- programovač zařízení,
- PC nebo jiné zařízení s ovládacím programem osciloskopu a čipové karty, jak je popsáno v diagramu 3.2.

Pro samotnou analýzu dat je dále nutné mít k dispozici program, umožňující statistické výpočty (např. Mathematica, SageMath, Matlab atd.).



Obrázek 3.6: Měřící zařízení s napojenými sondami z osciloskopu

3.3 Jednotlivé experimenty

V této sekci jsou postupně provedeny série měření spotřeby pro různě upravené verze algoritmu ChaCha a následně je pro každý experiment provedena příslušná analýza naměřených dat. Ve všech níže uvedených experimentech jsou považovány za známé hodnoty konstanty šifry (první řádek úvodní stavové matice 2.1) a tzv. „nonce“ (tedy poslední, čtvrtý řádek úvodní stavové matice). Každý experiment je rozdělen na dvě části, v první části je pro útok použit HW model spotřeby, ve druhé HD model spotřeby.

3.3.1 Technické poznámky k experimentům

Pro DPA nebo EMA je mnohdy nutné analyzovat velký počet průběhů měření o spotřebě najednou. V rámci této bakalářské práce bylo naměřeno až 8000 průběhů pro jednotlivé experimenty. Z důvodu omezené výpočetní kapacity PC, na kterém byla data analyzována, ovšem není možné brát libovolné množství naměřených dat. Proto v provedených experimentech jsou data často komprimována definovaným koeficientem, a to za účelem použití a analyzování co největšího množství dat. Pro představu jeden průběh datasetu2 (viz B) má 960000 B, což pro načtení všech 8000 průběhů dělá více než 7,5 GB dat. PC, na kterém byla data analyzována, mělo k dispozici 8 GB RAM. Po načtení všech nijak nezpracovaných (nekomprimovaných) dat by byla veškerá dostupná paměť spotřebována a nezbyla by již potřebná rezerva na výpočty.

Komprese dat s koeficientem 2 probíhá tak, že se vypočítá průměr dvou sousedních hodnot a do výsledné analýzy se dostane jen nově vypočítaná hodnota. To umožňuje při analýze použít dvojnásobné množství naměřených průběhů. Zároveň komprese s koeficientem 2 ještě zachovává přesnost měření a je v analýze použita jako standardní předzpracování dat.

Každý jednotlivý experiment je proto rozdělen na 3 fáze, kdy se postupně bere omezené množství dat s koeficientem komprese 2, a poté větší množství dat s vyšším (definovaným) koeficientem komprese. Následně se bere všech 8000 naměřených průměrů s již poměrně vysokým koeficientem komprese 16.

3.3.2 Vytvoření hypotézy o klíči

Pro každý model spotřeby (HW i HD) byly vytvořeny různé hypotézy o klíči. Každý model totiž pracoval s různými informacemi, které byly dostupné. HW model neměl kromě vstupů, jež byly posílány algortimu ChaCha, žádnou dodatečnou informaci o vnitřním stavu šifry. Oproti tomu, pokud byl použit HD model spotřeby, byla předpokládána znalost poloviny klíče.

Způsob tvorby hypotéz pro HW model spotřeby

Pro HW model se hypotézy o klíči vytvoří nejprve přičtením příslušného bytu konstanty pro všechny možné hodnoty klíče:

$$k_i = k_i + \text{příslušný byte konstanty } c. \tag{3.1}$$

Následně se všechny možné hodnoty klíče vyxorují se vstupními hodnotami (nezašifrovaným textem), čímž se vytvoří matice $xmat$, udávající očekávané hodnoty spotřeby pro daný byte klíče:

$$xmat = k \oplus d, \tag{3.2}$$

kde d označuje příslušné vstupní hodnoty a k všechny možné hodnoty klíče.

Způsob tvorby hypotéz pro HD model spotřeby

Při použití HD modelu je útok navržen na místo (2) ve schématu 3.1. Vždy je v této části experimentu předpokládána znalost poloviny klíče (hodnoty k_4, k_5, k_6, k_7 stavové matice 2.1). Stejně jako pro HW model se nejprve přičte příslušný byte konstanty c:

$$k_i = k_i + \text{příslušný byte konstanty } c. \tag{3.3}$$

Při vytváření matice očekávané spotřeby již hraje roli přidaná informace navíc (znalost druhé poloviny klíče). Nejprve se přixorují příslušné vstupy:

$$xmat = k \oplus d, \tag{3.4}$$

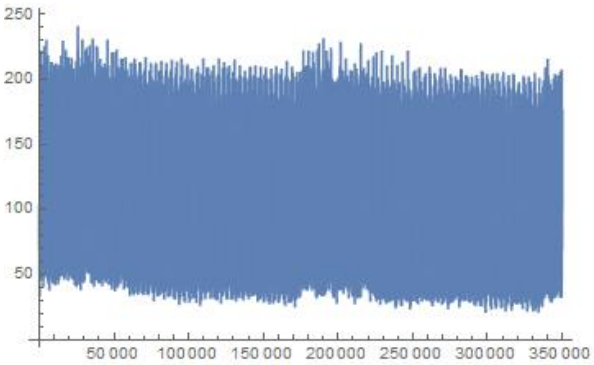
následně se přičte známá hodnota klíče:

$$xmat = xmat + \text{známé } k,\ xmat = xmat \oplus \text{známé } k. \tag{3.5}$$

3.3.3 Experiment: měření pouze výpočtu čtvrtrundy

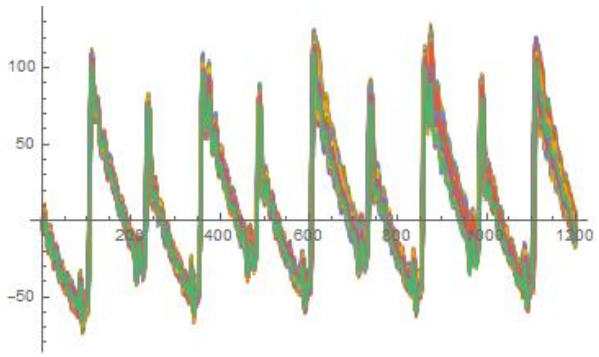
V tomto experimentu je algoritmus ChaCha modifikován tak, aby vracel „proudový klíč“ po provedení pouze jedné čtvrtrundy. Tedy pokud se vezme v úvahu úvodní stavová matice 2.1, je modifikován pouze první sloupec této matice, což jsou v našem případě hodnoty c_0, k_0, k_1, n_0. Útok tedy probíhá na 8 bytů tajného klíče (hodnoty k_0 a k_4) v místě (1) obrázku 3.1. Pro tento experiment byl použit dataset1 z přílohy B.

Celkový průběh spotřeby v čase zobrazuje následující obrázek:



Obrázek 3.7: Průběh spotřeby

Analýza naměřených dat ukazuje, že data jsou dobře zarovnána (dle očekávání, protože měřená implementace není nijak chráněna proti útokům postranním kanálem), viz obrázek 3.8. Proto není nutné data nijak dále upravovat.



Obrázek 3.8: „ZOOM“ na průběh spotřeby při výpočtu

HW model spotřeby, žádná informace navíc

Tato část experimentu (stejně tak, jako každá další jednotlivá část experimentu) byla rozdělena do tří fází.

V první fázi této části experimentu se pracovalo s 500 průběhy měření s koeficientem komprese 2.

Výsledky jsou nepříznivé a nevedou k odhalení tajného klíče. První byte hledané hodnoty k_0 se ze všech 256 hypotéz o klíči umístil až na 28. místě s korelací 0.653867. Druhý byte se umístil podstatně hůře, na 120. místě. Třetí byte potom ještě relativně dobře - na 4. místě, a čtvrtý na 102. místě. Jednotlivé korelace pro hledané byty a byty s nejvyšší korelací jsou na obrázku 3.9.

Ve druhé fázi se pracovalo s 1200 průběhy měření s koeficientem komprese 4.

Žádný ze čtyř hledaných bytů tajného klíče se sice neumístil na 1. pozici, nicméně došlo k velmi mírnému posunu směrem k prvním pozicím u prvního a druhého hledaného bytu. První se umístil na 22. místě, druhý na 110. místě, třetí na 4. místě. Pozice čtvrtého bytu se překvapivě zhoršila, dostala se na 108. místo. Na obrázku 3.10 jsou zobrazená porovnání korelací pro všechny čtyři hledané byty a pro byty s nejvyšší korelací.

Ve třetí fázi se uvažovalo všech 8000 průběhů měření s koeficientem komprese 16.

V této fázi došlo nečekaně jen k malé změně v umístění jednotlivých bytů oproti druhé fázi. První a druhý se umístily o několik málo pozic hůře, než ve fázi 2. Na třetí byte nemělo zvětšení počtu analyzovaných dat vliv. Po-

zice čtvrtého bytu byla stejná jako v první fázi. Grafy korelací jednotlivých bytů pro 8000 průběhů jsou zobrazeny níže (3.11). Je zde vidět, že došlo ke zvýraznění špiček, k tomuto zvýraznění však došlo i u ostatních „špatných" bytů.

Hypotézy o bytech hodnoty k_4 nebyly vytvořeny, protože k jejich vytvoření je zapotřebí znát byty hodnoty k_0, které ovšem nekorelovaly příliš dobře ani v jedné fázi experimentu.

HD model spotřeby, znalost poloviny klíče

V této části experimentu byl útok výrazně usnadněn předpokladem o znalosti dolní poloviny tajného klíče (hodnoty k_4, k_5, k_6, k_7 úvodní stavové matice 2.1). Zmíněná znalost umožňuje použití HD modelu spotřeby při zkoumání úniku informací postranním kanálem kryptografického zařízení.

Útok je opět proveden na úvodní stavovou matici a pouze první čtvrtrundu. Na rozdíl od experimentu výše se zde útočí na pozici (2) v obrázku 3.1. Hypotézy o klíči jsou sestaveny s touto znalostí a jsou tedy upraveny příslušné hodnoty očekávané spotřeby.

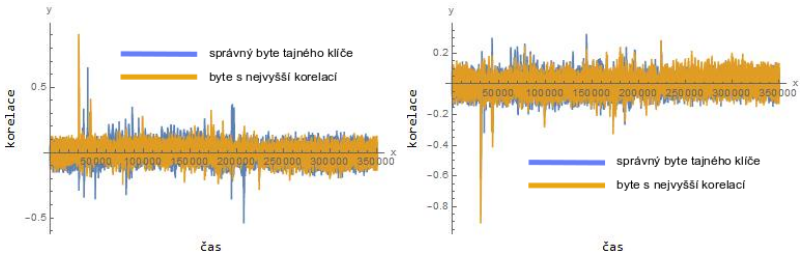
V první fázi se vzalo v úvahu standardně 500 průběhů měření.

Výsledky ukazují výrazné posunutí správných hodnot klíče k prvním pozicím. První a druhý byte hodnoty klíče k_0 mají dokonce nejvyšší korelační hodnoty ze všech hypotéz. Třetí byte se umístil na 58. místě a čtvrtý na 69. místě. Grafy korelací jednotlivých bytů pro 500 průběhů k_0 jsou na obrázcích 3.12. V grafech je jasně vidět, že hodnota prvního bytu výrazně koreluje.

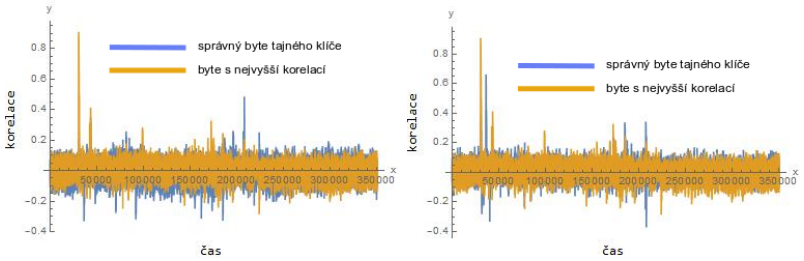
Ve druhé fázi se uvažovalo a zpracovalo 1200 průběhů měření s kompresním koeficientem 4.

V této fázi se pozice prvního bytu překvapivě zhoršila. První byte se nalézal na 2. místě s korelačním koeficientem 0.574133 oproti prvnímu bytu 0xec s korelačním koeficientem 0.575375. Umístění na 2. pozici je překvapivé, protože se vzrůstajícím množstvím dat je očekáváno zlepšení umístění, případně stagnace. Druhý byte se umístil na 1. místě, třetí na 30. místě, čtvrtý na 53. místě. U třetího a čtvrtého bytu došlo tedy ke zlepšení oproti 1. fázi. Grafy hledaných bytů jsou na obrázku 3.13.

V závěrečné fázi se vzaly všechny naměřené průběhy, tedy 8000 průběhů. Oproti druhé fázi se analyzování většího množství dat nijak výrazně neprojevilo. Vyšší koeficient komprese pouze zdůraznil naměřené špičky spotřeby, jak je vidět na obrázku 3.14.

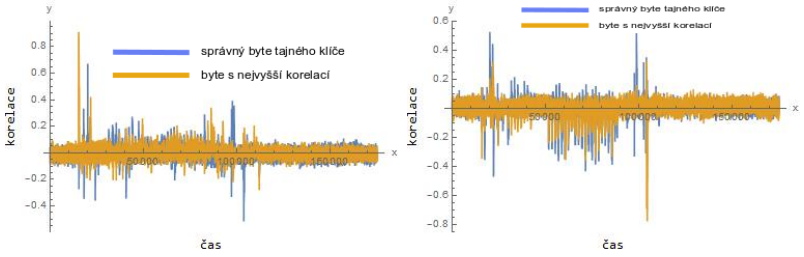


(a) První byte k_0 (b) Druhý byte k_0

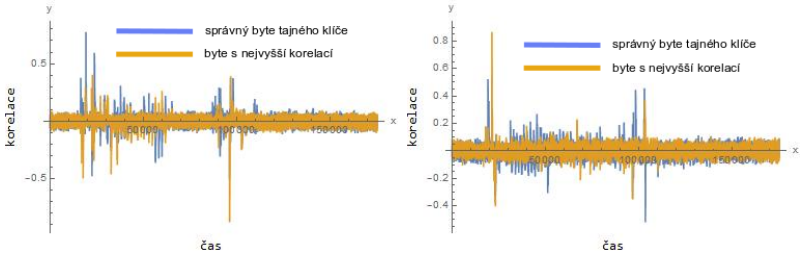


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.9: HW model spotřeby, korelace, čtvrtrunda, 500 průběhů

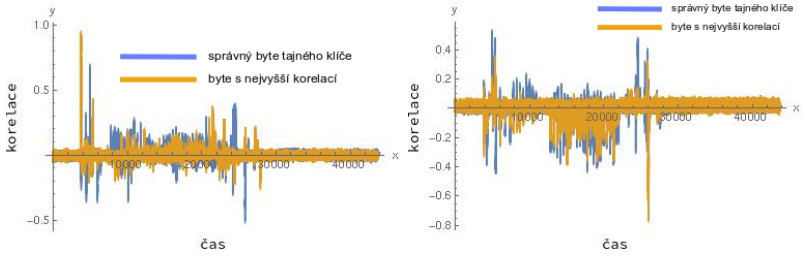


(a) První byte k_0 (b) Druhý byte k_0

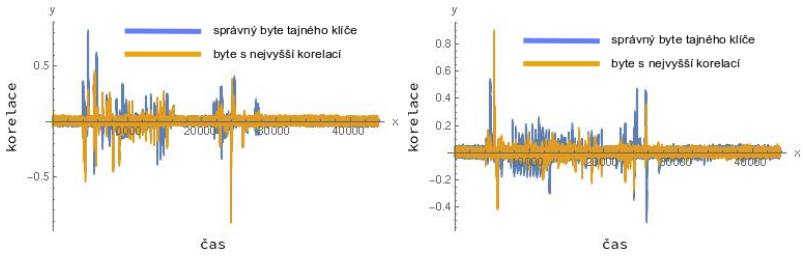


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.10: HW model spotřeby, korelace, čtvrtrunda, 1200 průběhů

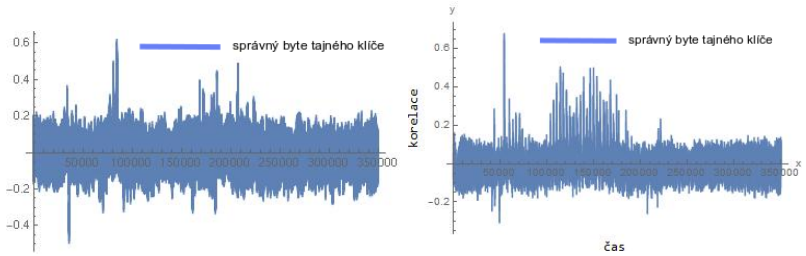


(a) První byte k_0 (b) Druhý byte k_0

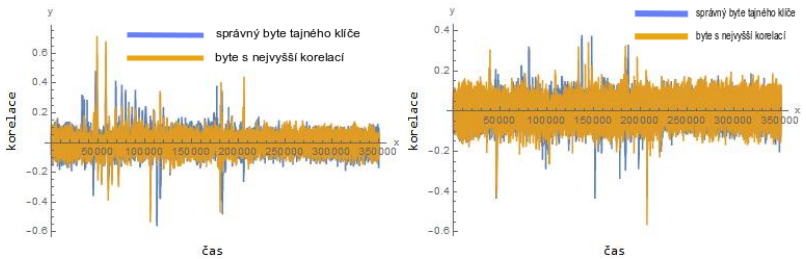


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.11: HW model spotřeby, korelace, čtvrtrunda, 8000 průběhů

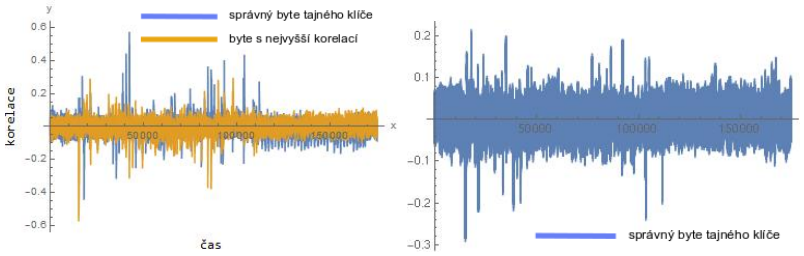


(a) První byte k_0 (b) Druhý byte k_0

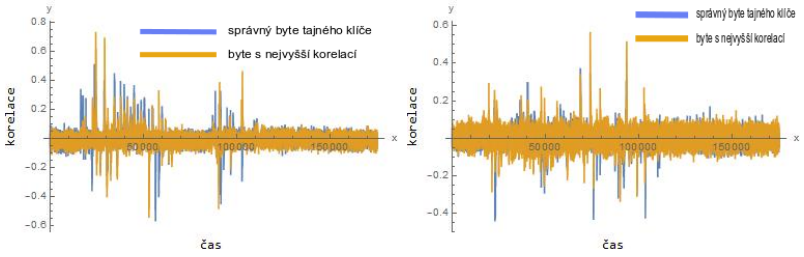


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.12: HD model spotřeby, korelace, čtvrtrunda, 500 průběhů

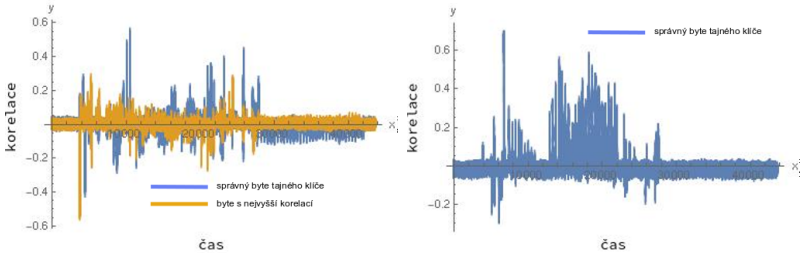


(a) První byte k_0 (b) Druhý byte k_0

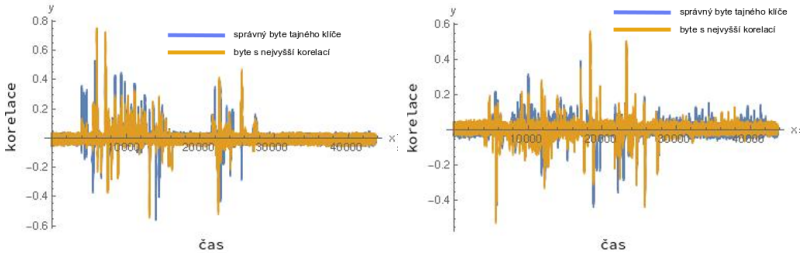


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.13: HD model spotřeby, korelace, čtvrtrunda, 1200 průběhů



(a) První byte k_0 (b) Druhý byte k_0



(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.14: HD model spotřeby, korelace, čtvrtrunda, 8000 průběhů

Shrnutí výsledku experimentu

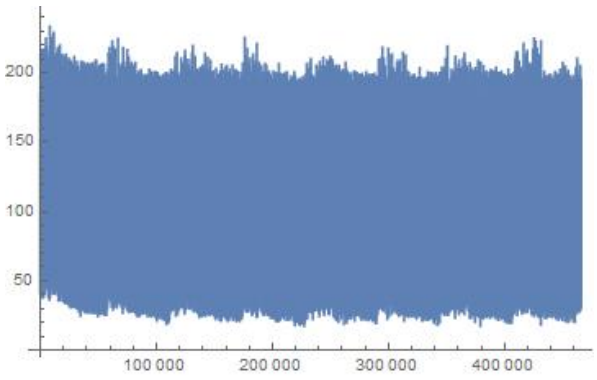
Výsledky experimentu, kdy se měřila spotřeba pouze při výpočtu čtvrtrundy, jsou rozporuplné. HW model spotřeby nedokázal odhalit byty správného klíče. Ani HD model spotřeby nesplnil očekávání, přestože dokázal bezpečně odhalit první a druhý byte. Znalost poloviny tajného klíče je ovšem velmi silná informace, proto se očekávalo odhalení klíče celého.

3.3.4 Experiment: měření celé první rundy algoritmu ChaCha

Pro tento experiment byla měřená implementace ChaChy upravena tak, aby proběhla celá jedna kompletní runda. Byla tak získána data o spotřebě pro všechny byty tajného klíče. Vzhledem k definici čtvrtrundy algoritmu ChaCha, viz 2.3, na dolní polovinu klíče (hodnoty k_4, k_5, k_6, k_7 úvodní stavové matice 2.1) lze zaútočit teprve po odhalení předchozích hodnot klíče.

V první fázi bylo standardně zpracováno 500 průběhů měření s kompresním koeficientem 2. Vzhledem k náročnosti výpočtu, který na analyzujícím PC způsoboval pády programu, bylo nutné naměřená data pro fázi 2 a 3 předzpracovat s vyšším kompresním koeficientem. Počet uvažovaných průběhů se již nezměnil. Tedy ve druhé fázi bylo zpracováno 1200 průběhů s kompresním koeficientem 5, ve třetí všech 8000 s kompresním koeficientem 25.

Obrázek 3.15 znázorňuje průběh spotřeby v čase. Se znalostí, že se jedná pouze o první rundu, lze na daném obrázku lehce odlišit jednotlivé čtvrtrundy.



Obrázek 3.15: Průběh spotřeby

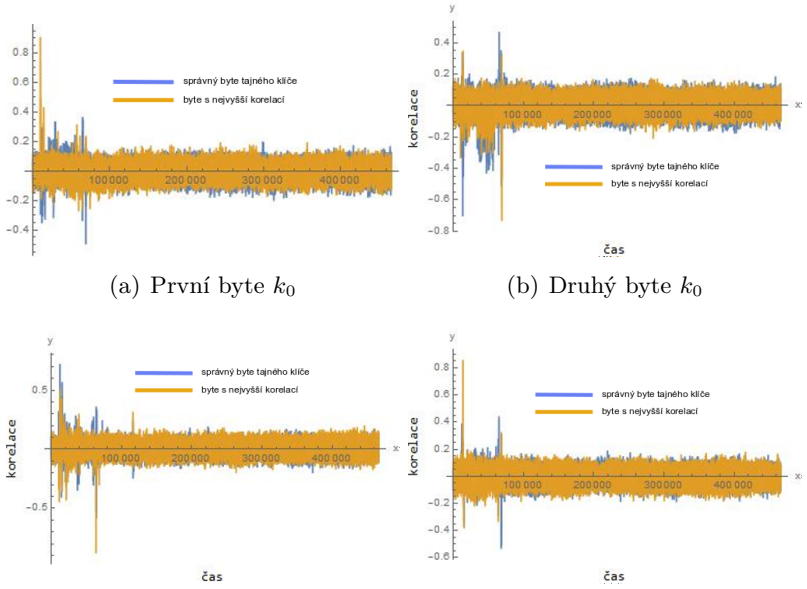
HW model spotřeby, počáteční runda, žádná informace navíc

Stejně tak jako v experimentu 3.3.3, ani zde nedokázal HW model spotřeby odhalit žádný z bytů tajného klíče. Některé byty, konkrétně se jedná o 2. a 3. byte k_0, 1., 2. a 3. byte k_1, 3. byte k_3, se pohybovaly v prvních dvaceti pozicích. Naopak byty slova k_2 se pohybovaly až od 58. pozice a hůře. Pro porovnání jsou uvedeny grafy korelací bytu k_0 pro jednotlivé fáze. Pro fázi 1 3.16, pro fázi 2 3.17, pro fázi 3 3.18. Na grafech je vidět, že výskyt „správné“ hledané hodnoty jednotlivých bytů se výrazně posunul směrem k začátku měření.

Hypotézy o bytech hodnoty k_4 nebyly vytvořeny, protože k jejich vytvoření je zapotřebí znát byty hodnoty k_0, které ovšem nekorelovaly příliš dobře ani v jedné fázi experimentu.

HD model spotřeby, počáteční runda, znalost poloviny klíče

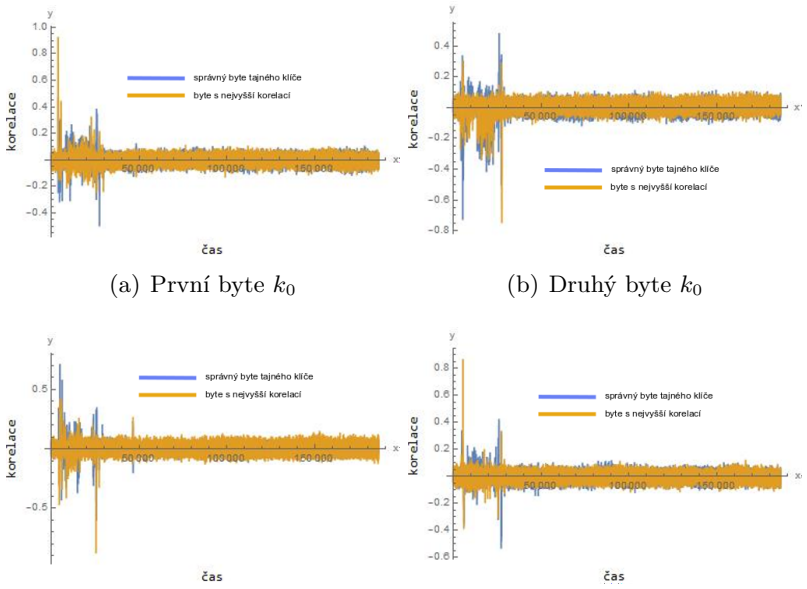
Oproti HW modelu se zde použitý HD model osvědčil lépe. A to dokonce i oproti HD modelu, který analyzoval pouze čtvrtrundu, viz 3.3.3. Znalost poloviny klíče umožnila odhalit několik bytů napříč hodnotami k_0, k_1, k_2, k_3, zdaleka ne však všechny. Konkrétně se bezpečně podařilo odhalit 4 byty, což z 16 celkových není mnoho. Také je překvapivé, že pozice jiných bytů, např. třetího bytu k_0, se zhoršila oproti použití HW modelu. Pro porovnání jsou opět uvedeny grafy korelací bytu k_0 pro jednotlivé fáze. Pro fázi 1 3.16, pro fázi 2 3.17, pro fázi 3 3.18.



(a) První byte k_0 (b) Druhý byte k_0

(c) Třetí byte k_0 (d) Čtvrtý byte k_0

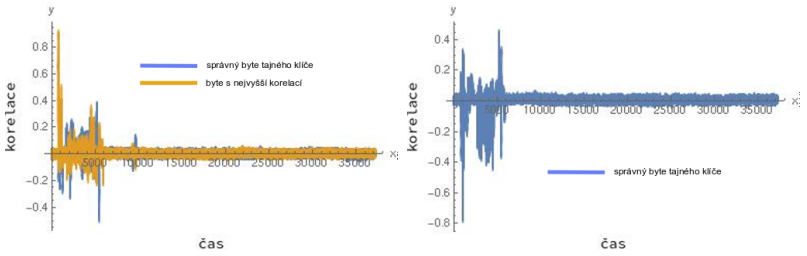
Obrázek 3.16: HW model spotřeby, korelace, první runda, 500 průběhů



(a) První byte k_0 (b) Druhý byte k_0

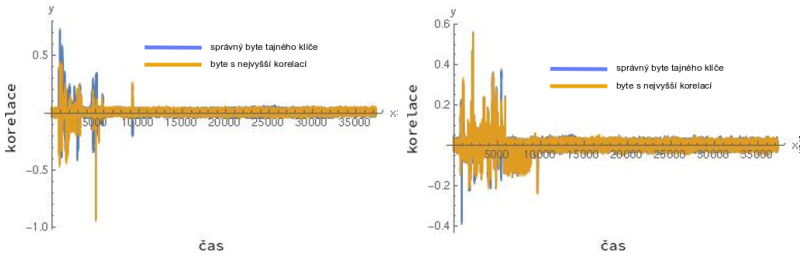
(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.17: HW model spotřeby, korelace, první runda, 1200 průběhů



(a) První byte k_0

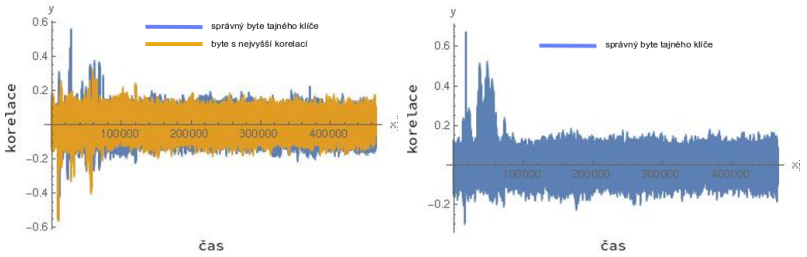
(b) Druhý byte k_0



(c) Třetí byte k_0

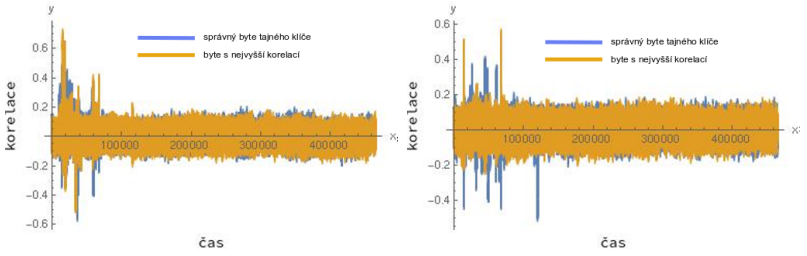
(d) Čtvrtý byte k_0

Obrázek 3.18: HW model spotřeby, korelace, první runda, 8000 průběhů



(a) První byte k_0

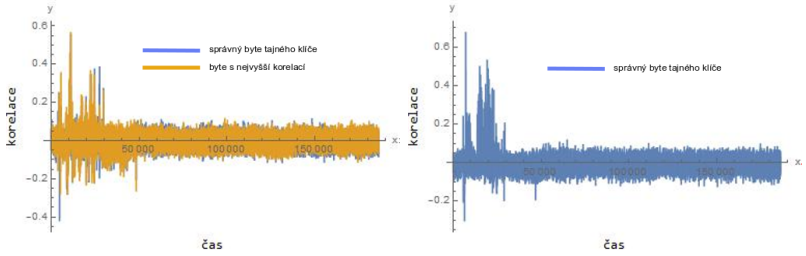
(b) Druhý byte k_0



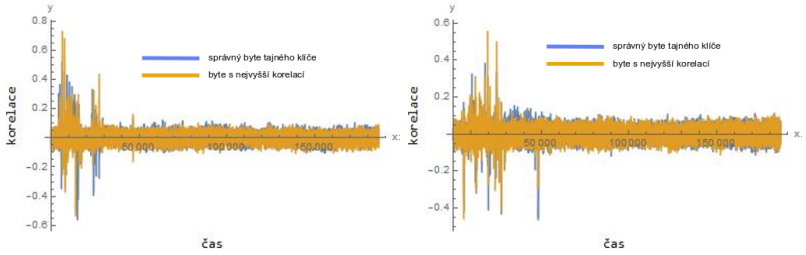
(c) Třetí byte k_0

(d) Čtvrtý byte k_0

Obrázek 3.19: HD model spotřeby, korelace, první runda, 500 průběhů

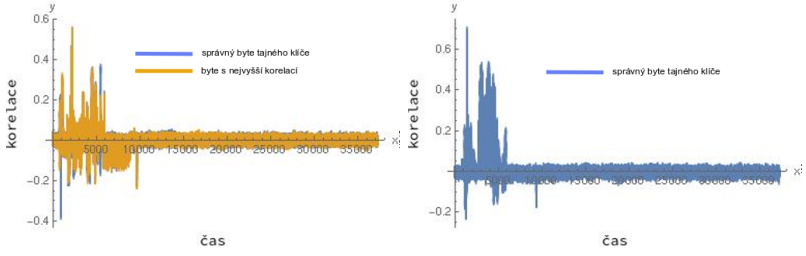


(a) První byte k_0 (b) Druhý byte k_0

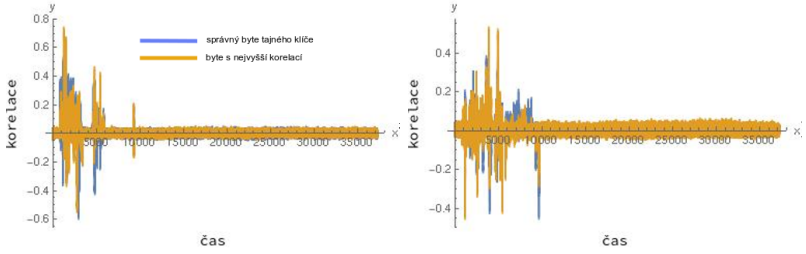


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.20: HD model spotřeby, korelace, první runda, 1200 průběhů



(a) První byte k_0 (b) Druhý byte k_0



(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.21: HD model spotřeby, korelace, první runda, 8000 průběhů

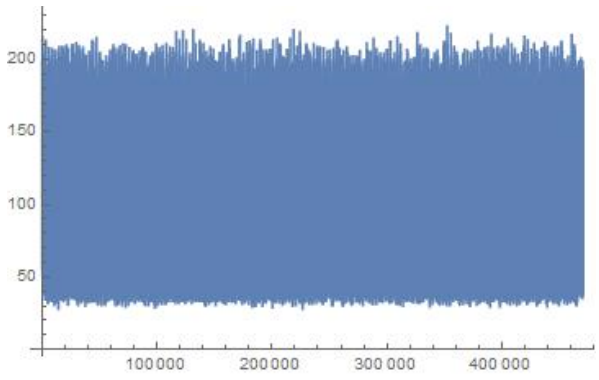
Shrnutí výsledku experimentu

V tomto experimentu, který analyzuje průběh pouze jedné rundy šifry ChaCha, bylo útočeno na prvních 16 bytů tajného klíče. Vzhledem k tomu, že správné hodnoty se nepodařilo uspokojivě odhalit, nebylo již útočeno na zbylých 16 bytů z druhé poloviny klíče. Byty druhé poloviny klíče značně závisí na bytech první poloviny, jejichž bezpečné odhalení je nezbytné pro pokračování útoku.

Umístění jednotlivých 16 „správných“ bytů se dost lišilo napříč experimentem. To budí dojem, že algoritmus ChaCha vypouští rozdílnou informaci o jednotlivých bytech tajného klíče. V grafech je také často vidět několik korelačních špiček, což může být důsledek výskytu bytu sice shodného s tajným klíčem, ale vzniklého v rámci provádění rundy, a tedy defacto zkreslující výsledek.

3.3.5 Experiment: měření celého průběhu algoritmu ChaCha

V posledním experimentu bylo provedeno měření spotřeby celého průběhu šifry ChaCha. Tedy byl změřen průběh všech dvaceti rund. Vzhledem k náročnějším výpočtům u měření více než 500 průběhů byly opět navýšeny kompresní koeficienty pro fázi 2 a 3, a to na stejné hodnoty, jako v experimentu 3.3.5.



Obrázek 3.22: Průběh spotřeby

HW model spotřeby, celá šifra, žádná informace navíc

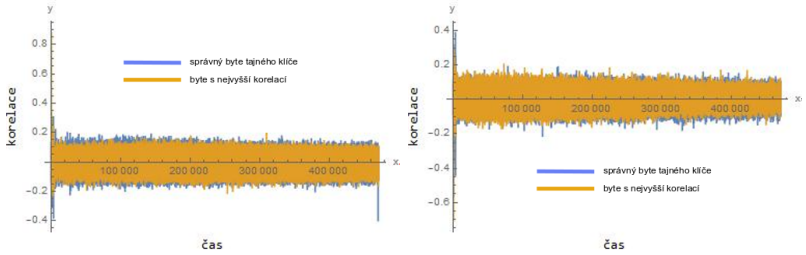
Při měření celého průběhu algoritmu ChaCha se pozice „správných“ bytů dle očekávání celkově zhoršily. Při průběhu všech 20 rund jsou již „správné“ hodnoty značně zastíněné. Na jednotlivých grafech 3.23, 3.24 a 3.25 je ovšem vidět, že korelační špičky jsou převážně na začátku algoritmu, tedy při průběhu první rundy, což nahrává správnosti návrhu útoku na první rundu. Slabší

korelační špičky se objevují také ke konci průběhu, kdy z hlediska implementace probíhá závěrečné přičtení úvodní stavové matice, změněné 20 rundami, a nezměněné úvodní stavové matice 2.1. Nutno dodat, že v rámci této práce bylo měření usnadněno nastavením signálů, které byly poslány osciloskopu na vhodném místě algoritmu. Obrázek 3.22 znázorňuje průběh jednoho měření spotřeby v čase, kde průběh jednotlivých rund je těžko odlišitelný.

Hypotézy o bytech hodnoty k_4 nebyly vytvořeny, protože k jejich vytvoření je zapotřebí znát byty hodnoty k_0, které ovšem nekorelovaly příliš dobře ani v jedné fázi experimentu.

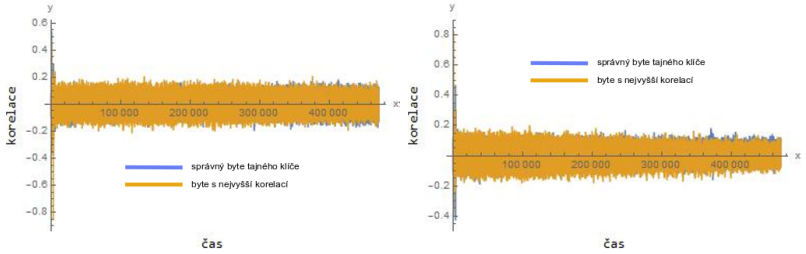
HD model spotřeby, celá šifra, znalost poloviny klíče

Zde HD model byty správného klíče odhaloval lépe, než v experimentu 3.3.4, nicméně celý klíč se ani zde odhalit nepodařilo. Vzhledem k tomu, že ani znalost poloviny klíče výrazně nepomohla v obou výše provedených experimentech, nebylo ani zde získání celého klíče již očekáváno. Pro srovnání se dvěma dalšími provedenými experimenty jsou uvedeny grafy korelací bytů klíče k_0: 3.26, 3.27 a 3.28.



(a) První byte k_0

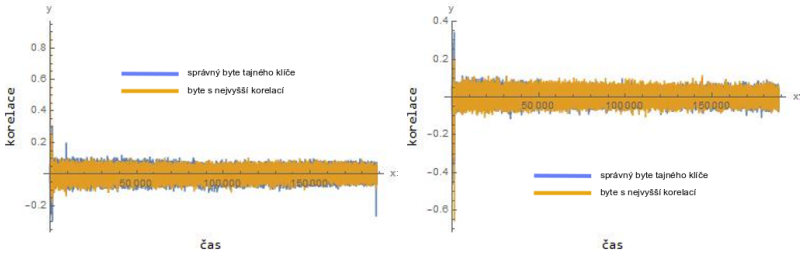
(b) Druhý byte k_0



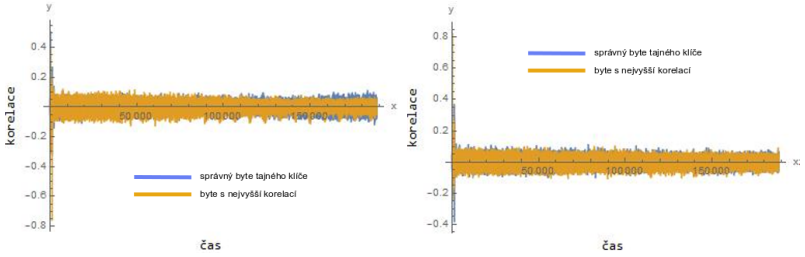
(c) Třetí byte k_0

(d) Čtvrtý byte k_0

Obrázek 3.23: HW model spotřeby, korelace, 20 rund, 500 průběhů

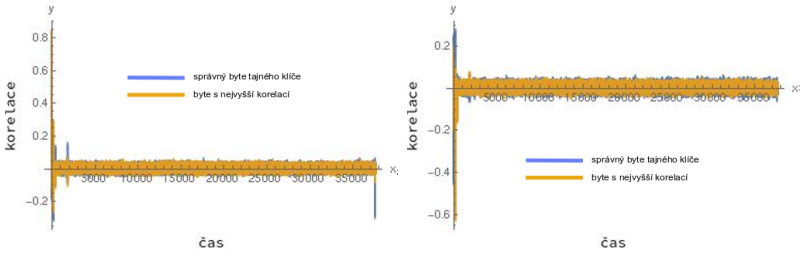


(a) První byte k_0 (b) Druhý byte k_0

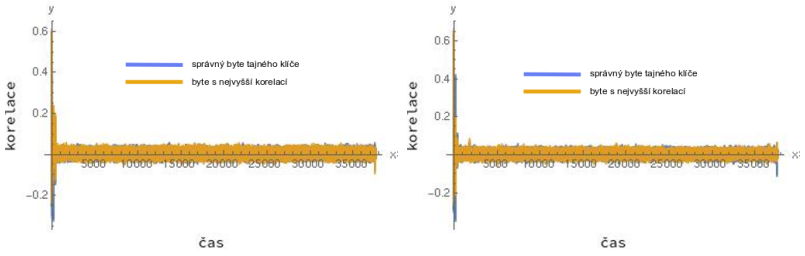


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.24: HW model spotřeby, korelace, 20 rund, 1200 průběhů

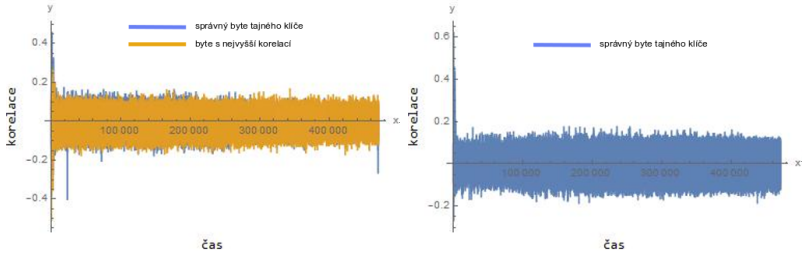


(a) První byte k_0 (b) Druhý byte k_0

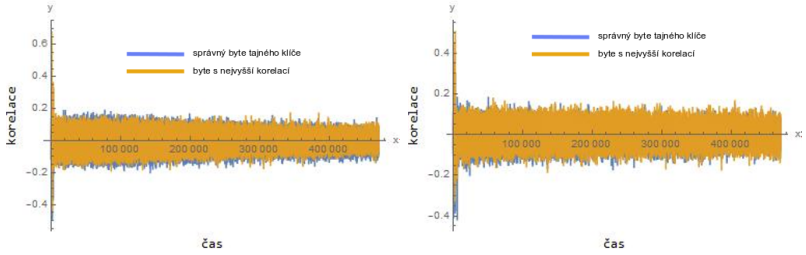


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.25: HW model spotřeby, korelace, 20 rund, 8000 průběhů

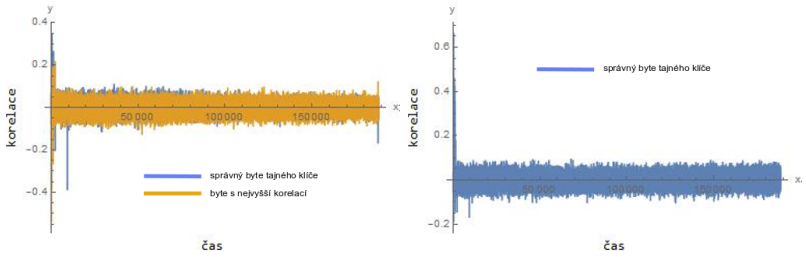


(a) První byte k_0 (b) Druhý byte k_0

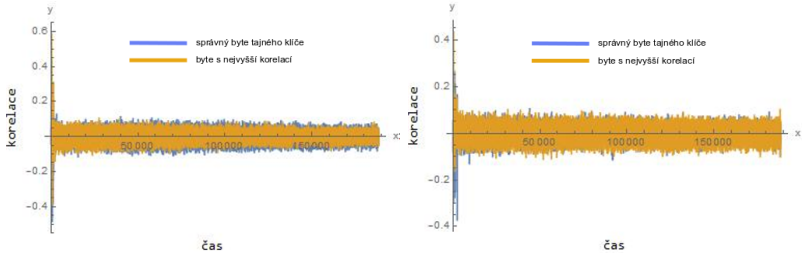


(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.26: HD model spotřeby, korelace, 20 rund, 500 průběhů

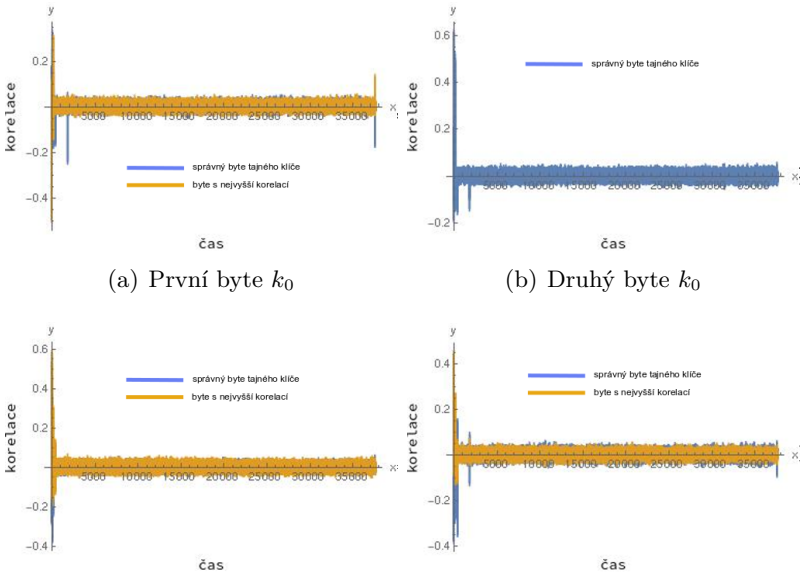


(a) První byte k_0 (b) Druhý byte k_0



(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.27: HD model spotřeby, korelace, 20 rund, 1200 průběhů



(a) První byte k_0 (b) Druhý byte k_0

(c) Třetí byte k_0 (d) Čtvrtý byte k_0

Obrázek 3.28: HD model spotřeby, korelace, 20 rund, 8000 průběhů

Shrnutí výsledku experimentu

Při analýze datasetu3 (viz B) bylo očekáváno horší odhalení bytů tajného klíče, než u zjednodušených měření rundy a čtvrtrundy. Tento předpoklad byl naplněn u HW modelu, HD model naopak dokázal najít správné byty lépe, než experiment 3.3.4. Korelační špičky hledaných bytů se také nalézají blíže k začátku měření, kdy probíhá první runda. Celý klíč odhalen nebyl.

3.3.6 Celkové vyhodnocení výsledku experimentů a srovnání s AES

Přes veškerou snahu tajný klíč nebyl celý odhalen v žádném z provedených experimentů. Přesto analýza ukazuje, že při kryptografickém výpočtu šifry ChaCha uniká postranním kanálem spotřeby určité množství informací o použitém klíči. Důvodem obtížnosti odhalení klíče je pravděpodobně silná lineárnost operací ARX. Neboli malá změna na vstupu vyvolá jen malou změnu na výstupu. Jediná „trochu“ nelineární operace je operace plus, a to jen za podmínek, kdy vznikne tzv. přenos. Např. pro dvě 4bitová čísla $1011_2 + 0101_2 = 0000_2$, pokud se ve sčítanci vhodně změní jeden bit $1011_2 + 0100_2 = 1111_2$, dojde k maximální změně. Oproti tomu algoritmus AES má silně nelineární operaci „Sbox“, neboli SubBytes (více viz standard [7]). Právě operace SubBytes, kdy jen 1bitová změna na vstupu vyvolá velkou

změnu na výstupu, dělá AES tak zranitelným oproti diferenciální odběrové analýze nebo elektromagnetické analýze.

Závěr

Cílem bakalářské práce bylo prozkoumat šifru ChaCha a metody útoku postranním kanálem - diferenciální odběrovou analýzu a elektromagnetickou analýzu. Šifru ChaCha dále implementovat na vybrané platformě a navrhnout, uskutečnit a vyhodnotit útok na tuto šifru a vybranou platformu pomocí jedné ze zkoumaných metod.

Jako platforma pro implementaci šifry se použila čipová karta ATMega s 8bitovým procesorem ATMega163. Za metodu útoku byla vybrána diferenciální odběrová analýza (DPA) a po prozkoumání specifikace šifry ChaCha byla zvolena první runda jako vhodné místo pro útok.

I přes pečlivou analýzu nasbíraných dat byl tajný klíč odhalen jen částečně. Šifra ChaCha se ukazuje poměrně odolná proti DPA a při šifrování neuniká tímto postranním kanálem dostatek informací pro odhalení celého tajného klíče. Alespoň co se týče navrženého místa útoku. Analýza dat naznačuje, že dalším vhodným místem útoku by mohl být závěr kryptografického výpočtu, tedy útočit odzadu. Tato možnost nebyla v této práci zkoumána.

Literatura

[1] MANGARD Stefan, Elisabeth Oswald, Thomas Popp: *Power Analysis Attacks: Revealing the Secrets of Smart Cards*. Springer Science & Business Media, 2007, ISBN 978-0-387-30857-9.

[2] IANIX: *ChaCha Usage & Deployment [online]*. [cit. 2018-01-05]. Dostupné z: https://ianix.com/pub/chacha-deployment.html

[3] KOCHER Paul, Joshua Jaffe, Benjamin Jun: *Differential power analysis. In Advances in Cryptology—CRYPTO'99*. Springer, Berlin, Heidelberg, 1999, ISBN 978-3-540-66347-8, s. 388–397.

[4] BERNSTEIN J. Daniel: *ChaCha, a variant of Salsa20 [online]*. 2008, [cit. 2018-01-05]. Dostupné z: http://cr.yp.to/chacha/chacha-20080128.pdf

[5] BERNSTEIN J. Daniel: *Cache-timing attacks on AES [online]*. The University of Illinois at Chicago, [cit. 2018-01-05]. Dostupné z: https://cr.yp.to/antiforgery/cachetiming-20050414.pdf

[6] NIR Y., A. Langley: *ChaCha20 and Poly1305 for IETF Protocols [online]*. květen 2015. Dostupné z: https://tools.ietf.org/html/rfc7539

[7] National Institute of Standards and Technology: *FIPS 197: ADVANCED ENCRYPTION STANDARD (AES)*. NIST, 2001, [cit. 2018-01-05]. Dostupné z: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf

Dodatek A

Seznam použitých zkratek

DPA Diffrential Power Analysis (diferenciální odběrová analýza)

EMA Electromagnetic analysis (elektromagnetická analýza)

ARX Add, Rotate, Xor (sčítání, rotace, xor)

HW Hamming Weight (Hammingova váha)

HD Hamming Distance (Hammingova vzdálenost)

AES Advanced Encryption Standard

Dodatek B

Obsah přiložené SD karty

- `README.txt` ..... stručný popis obsahu SD karty
- `src`
  - `analysis.nb` ..... zdrojový kód notebooku pro analýyu dat
  - `bpr` ..... zdrojová forma práce ve formátu XeLaTeX
  - `soft` ..... programy pro interakci s osciloskopem a čipovou kartou
- `data` ..... jednotlivé datasety
  - `dataset1` ..... dataset1 a zdrojový kód měřeného programu
  - `dataset2` ..... dataset2 a zdrojový kód měřeného programu
  - `dataset3` ..... dataset3 a zdrojový kód měřeného programu
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF