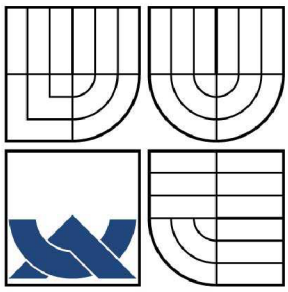
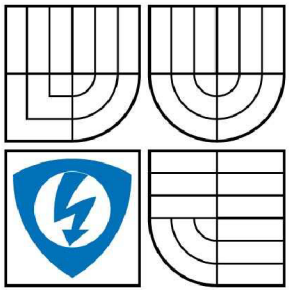


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKACNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ÚTOKY POSTRANNÍMI KANÁLY

SIDE-CHANNEL ATTACKS

DIPLOMOVÁ PRÁCE
MASTER´S THESIS

AUTOR PRÁCE — Bc. MICHAL POPOVSKÝ
AUTHOR

VEDOUCÍ PRÁCE — Ing. ZDENĚK MARTINÁSEK
SUPERVISOR

BRNO 2009

ANOTACE

V současné době je velmi obtížné v reálném čase prolomit používané kryptografické algoritmy. Klasická kryptoanalýza je založena pouze na hledání slabin v matematické struktuře algoritmu. Podstatná změna této vědy nastala objevením postranních kanálů. Útoky postranními kanály jsou založeny na nedokonalosti fyzické implementace kryptografických algoritmů. Tento nový způsob útoku, který je založený na vyzařování senzitivních informací přímo z kryptografických modulů, mění dosavadní představy o kryptografii a bezpečnosti systémů.

Tato diplomová práce obsahuje detailní popis celé problematiky postranních kanálů a zabývá se především útoky postranními kanály na asymetrický algoritmus RSA. Tato práce obsahuje návrh a realizaci laboratorní úlohy, která je zaměřená na časový útok na implementaci algoritmu RSA.

KLÍČOVÁ SLOVA

Postranní kanál, kryptografický modul, kryptoanalýza, algoritmus RSA, časový útok, laboratorní úloha.

ABSTRACT

It is very difficult in real-time breaking the cryptographic algorithms used at present. The Classical cryptoanalysis is based on finding weaknesses in the mathematical structure of the algorithm. Discovery of side channels caused a substantial change in this science. Side-channel attacks are based on incorrect physical implementation of cryptographic algorithms. This new way attack changes notions about cryptography and security of systems.

This master´s thesis contains a detailed description of the whole problem of side channels and deals with side-channel attacks on the RSA asymmetric algorithm. This thesis includes the design and realization of laboratory exercise, which is focused on the time attack on the RSA algorithm implementation.

KEYWORDS

Side channel, cryptographic module, cryptoanalysis, RSA algorithm, time attack, laboratory exercise.

BIBLIOGRAFICKÁ CITACE

POPOVSKÝ, M. *Útoky postranními kanály.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2009. 71 str.
Vedoucí diplomové práce Ing. Zdeněk Martinásek.

PROHLÁŠENÍ

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

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této 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.

V Brně dne …… ……..

(podpis autora)

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Ing. Zdeňkovi Martináskovi, za užitečnou odbornou pomoc a cenné rady při zpracování diplomové práce.

Dále chci poděkovat mé manželce Daniele Popovské, která měla se mnou trpělivost a podporovala mě při psaní diplomové práce.

V Brně dne …… ……..

(podpis autora)

OBSAH

ÚVOD

Přibližně před deseti lety nastala zásadní změna v myšlení a v pohledu na celou oblast kryptologie. Příčina této podstatné změny spočívá v objevení postranních kanálů a vědomí existence možnosti snadného útoku postranním kanálem na kryptografický modul. V dnešní době, kdy se stává velmi obtížné v rozumném čase prolomit daný algoritmus a získat soukromý klíč, se do popředí kryptoanalýzy dostávají útoky skrze postranní kanály. Tento zcela nový pohled na způsob úniku tajných informací z kryptografických systémů mění dosavadní představy o bezpečnosti i kryptografii a vyvolává vznik nových směrů především v kryptoanalýze.

Součástí této diplomové práce je rozbor problematiky postranních kanálů a jejich útoků na kryptografický modul. Na základě získaných informací bude vytvořen ucelený a detailní přehled o současném stavu tohoto relativně nového směru v oblasti kryptoanalýzy. V práci budou nejdříve definovány důležité základní pojmy a souvislosti, které jsou potřebné k pochopení celé problematiky postranních kanálů. Práce se bude podrobně zabývat jednotlivými základními útoky postranními kanály na kryptografické moduly.

Hlavním cílem diplomové práce je návrh a realizace laboratorní úlohy, která bude vhodně demonstrovat problematiku útoků časovým postranním kanálem na asymetrický algoritmus RSA.

1 ZÁKLADNÍ POJMY

Jak tomu bývá v každém vědním oboru, také zde je nezbytně nutné definovat několik základních pojmů, které umožní pochopení rozebírané problematiky. Terminologie v oblasti kryptologie často nebývá striktně dodržována, tzn. jeden pojem podle kontextu zastává i více významů. Nejen z tohoto důvodu je důležité v práci definovat následující základní pojmy.

1.1 Kryptologie

Pojem kryptologie pochází z řečtiny. Řecké slovo *kryptós* znamená „skrytý“ a slovo *logos* (*λόγος*) znamená „slovo“, „řeč“, ale také často „smysl“. Kryptologie je tedy obecný pojem pro označení vědy, která se zabývá šifrováním a dešifrováním informací.

Kryptologie se nezabývá utajením samotné existence určité informace, ale jak vyplývá i z řeckého slova *logos* (v překladu: slovo, smysl) soustředí se na utajení významu dané informace. Hlavními disciplínami kryptologie jsou kryptografie a kryptoanalýza.

1.2 Kryptografie

Slovo kryptografie také pochází z řečtiny. Jak bylo již uvedeno řecké slovo *kryptós* je českým ekvivalentem slova „skrytý“ a ekvivalentem pro řecké slovo *gráphein* je sloveso „psát“. Z toho plyne, že samotný pojem kryptografie označuje skrytí významu psané zprávy. Přesněji je kryptografie věda, která studuje způsoby a metody skrytí významu informace transformací do určité zdánlivě nesrozumitelné podoby (tzv. šifrovaný text), ze které lze původní význam informace získat pouze znalostí další speciální informace (např. znalostí soukromého klíče). Kryptografie se zabývá šifrovacími algoritmy, jejich fyzickou implementací, dalšími kryptografickými nástroji a protokoly.

1.3 Kryptoanalýza

Dalším základním pojmem je kryptoanalýza. Řecké slovo *analýein* je možné přeložit jako „rozvázat“. Ve své podstatě se jedná o opak kryptografie. Kryptoanalýza je věda, která se soustředí na vývoj metod k rozluštění šifrované informace bez znalosti soukromého klíče. Moderní kryptoanalýzu lze v širším významu definovat jako vědu o hledání slabin nebo prolamování matematických metod informační bezpečnosti.

1.4 Kryptografické služby

Kryptografické služby jsou metody, pomocí nichž jsou zajištěny konkrétní bezpečnostní požadavky (např. integrita dat). Kryptografické služby jsou realizovány pomocí kryptografických nástrojů, kterými jsou především kryptografické algoritmy a kryptografické protokoly.

1.4.1 Kryptografický algoritmus

Kryptografický algoritmus lze obecně definovat jako matematickou funkci, která se používá pro operace šifrování a dešifrování. Jedná se tedy o proces transformace, který převede vstupní informaci (otevřený text) do šifrované podoby (šifrovaný text) a naopak. Proces šifrování probíhá transformací pomocí klíče, který je určen k tomuto účelu. Potom zcela analogicky je tomu u procesu dešifrování. V moderní kryptografii nejsou algoritmy žádným způsobem utajeny, ale naopak jsou všeobecně známy. Tajemství šifrované informace je založeno na utajení čísla (šifrovacího klíče), prostřednictvím kterého vytváří algoritmus z původní informace šifrovanou informaci a naopak. V současné době se používají dvě základní třídy kryptografických algoritmů, tj. symetrické a asymetrické algoritmy.

Symetrické algoritmy se využívají zejména k zabezpečení rychlého přenosu většího objemu dat. U těchto algoritmů (šifer) slouží jeden jediný klíč k šifrování dat i k jejich dešifrování. Mezi nejznámější zástupce kryptografických algoritmů patří DES, AES, 3-DES, IDEA, Blowfish, atd.

Druhým základním typem kryptografických algoritmů jsou asymetrické algoritmy (šifry s veřejným klíčem). Základní charakteristikou, která je odlišuje od symetrických algoritmů, je existence dvou různých klíčů. Jeden klíč je používán při šifrování dat a druhý klíč je využíván při jejich dešifrování. Tyto algoritmy jsou díky pomalejšímu šifrování využívány převážně pro distribuci klíčů, digitální podpisy a autentizaci. Nejrozšířenějšími představiteli asymetrických algoritmů jsou RSA, Diffie – Hellman protokol, ECC, DSS, ElGamal, atd.

1.4.2 Kryptografický protokol

Kryptografický protokol je definovaná dohoda, která určuje způsob komunikace a druh sdílených informací. Každý kryptografický protokol je do značné míry principiálně založen na určitém kryptografickém algoritmu. Kryptografické protokoly jsou razantně využívány, protože se zabývají řešením velmi rozsáhlého okruhu problémů při zabezpečování přenosu dat. V praxi je účelem kryptografických protokolů autentizace

účastníků protokolu, utvoření dohody o kryptografickém klíči, výměna těchto klíčů a podobně. Dobrý kryptografický protokol neumožňuje zúčastněným osobám provést jiné akce nebo získat jiné informace, než které jsou v protokolu předem určeny. Nejrozšířenějším zástupcem kryptografických protokolů jsou bezpečnostní protokoly Secure Sockets Layer – SSL a Transport Layer Security – TLS, které poskytují možnost zabezpečené komunikace.

1.5 Bezpečný kryptografický systém

Kryptografický systém představuje celý proces zpracování zprávy a klíčů a všechny jeho okolnosti, zahrnující kryptografické algoritmy, operace a další pravidla, podle kterých je tento daný systém řízen. Bezpečnost nelze považovat za stálou vlastnost v oblasti kryptografických systémů, protože možnosti a metody v kryptoanalýze se s časem stále vyvíjejí.

V současné době každý obecný informační systém musí splňovat alespoň čtyři základní bezpečnostní požadavky (tzn. bezpečnostní cíle). Dobré kryptografické systémy by měli poskytovat zabezpečení těchto hlavních bezpečnostních požadavků. Mezi tyto základní bezpečnostní požadavky patří důvěrnost dat, autentizace, integrita dat a nepopiratelnost.

Důvěrnost dat – záruka, že informace jsou dostupné jen těm uživatelům, kteří mají ke konkrétním datům oprávnění. Je proto nutné utajit informace před neoprávněnými uživateli, respektive tito uživatelé nesmí být schopni datům porozumět. Důvěrnosti dat lze dosáhnout různými způsoby jako například kontrolou samotného fyzického přístupu k datům, ale převážně je důvěrnost dat zajištěna šifrováním.

Autentizace – jistota, že informace mají původní obsah, pochází od uvedeného zdroje, vznikla v uvedeném čase, apod. Na druhou stranu je ovšem bezpečnostním cílem autentizace také záruka, že odesílatel informace je tím, za koho se sám prohlašuje. Tento bezpečnostní požadavek je zajišťován identifikačními a autentizačními mechanismy (digitální podpisy, hašovací funkce s klíčem).

Integrita dat – zajištění, aby data nebyla úmyslně nebo náhodně modifikována neoprávněným uživatelem. V kryptologii všeobecně pojem integrita znamená platnost dat. Platnost dat může být porušena záměrným pozměněním významu informace (např. změna čísla účtu v bankovní transakci) nebo náhodnou změnou informace (např. chyby vzniklé při přenosu dat přenosovým kanálem). Integrita dat bývá v praxi zajišťována hašovacími funkcemi, kontrolními součty, samoopravnými kódy atd.

Nepopiratelnost – zajištění, aby daný zdroj dat nemohl s postupem času popřít nebo vyvrátit skutečnost odeslání těchto dat. Pokud je zajištěn tento bezpečnostní cíl, pak v případě sporu dvou stran je třetí nezávislá strana schopna s jistotou rozhodnout, zda se tento čin stal nebo vůbec neproběhl. Podstata a přesná definice nepopiratelnosti je souhrnně definována v normě [10]: „*cílem nepopiratelnosti je vytvářet, shromažďovat, udržovat, zajistit dostupnost a ověřovat důkazy týkající se údajné události nebo činnosti, aby bylo možné řešit spory o tom, zda se událost nebo činnost vyskytla či nikoliv*.“ Rozeznáváme hned několik typů nepopiratelnosti a to nepopiratelnost odeslání zprávy, nepopiratelnost jejího příjmu, nepopiratelnost jejího vytvoření, nepopiratelnost znalosti a přenosu zprávy. Nepopiratelnosti lze dosáhnout využitím kryptografického nástroje digitálních podpisů.

Celkový rozvoj informačních technologií a především vědomí potřebnosti zajistit nové bezpečnostní cíle vedou k hledání a vývoji nových matematických a kryptografických nástrojů.

1.6 Bezpečnost systému

Se stále zrychlujícím vývojem moderních komunikačních a počítačových systémů se objevila řada nových možností a typů útoku na kryptografické systémy. Díky tomuto trendu se požadavky na systémy ochrany dat stále zvyšují. Útočníci už nemusí pouze pasivně pozorovat přenosový kanál, ale mohou sami zasahovat do probíhající komunikace. Útočník má možnost jednoduše modifikovat šifrovaná data, přerušit i zakládat vlastní komunikaci, opětovně poslat zachycené šifrované data, má také možnost zprávu pozastavit, odstranit nebo nahradit svou vlastní zprávou. Žádaným požadavkem na kryptografické systémy je zajištění imunity vůči známým druhům útoků.

1.6.1 Nepodmíněná bezpečnost

Nepodmíněná bezpečnost je nejvyšší míra bezpečnosti kryptografického systému. Je to taková bezpečnost systému, která není podmíněna žádnými předpoklady na schopnosti a technické možnosti útočníka. Jinými slovy pokud útočník nemá k dispozici informace o klíčích, nemá žádnou možnost dostat se k utajeným informacím. Nepodmíněně bezpečné kryptografické systémy jsou také často označovány jako absolutně bezpečné. Samozřejmě drtivá většina kryptografických systémů je založena na určitých předpokladech o útočníkovi, což znamená, že se nejedná o systémy absolutně bezpečné. Přesto existují systémy, které splňují požadavky nepodmíněné bezpečnosti. Takovou míru bezpečnosti v současné době poskytuje kvantová kryptografie a Vernamova šifra.

Absolutní bezpečnost Vernamovi šifry byla matematicky dokázána, ovšem tato šifra je spojena s obtížnou distribucí samotných klíčů.

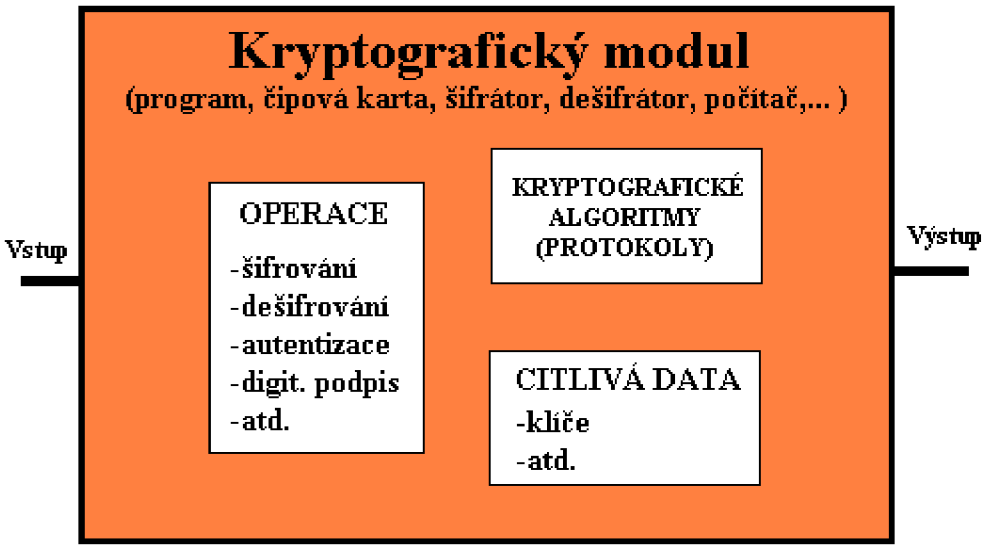
1.6.2 Dokazatelná a výpočetní bezpečnost

Dokazatelně bezpečné jsou takové kryptografické systémy, u kterých lze provést důkaz, že k jejich prolomení je nutné najít řešení výpočetně složitého problému. Typickými představiteli takového složitého problému je problém faktorizace velkých čísel, diskrétního logaritmu a eliptických křivek.

Mezi výpočetně bezpečné patří kryptografické systémy, u kterých je možnost prolomení, i při použití nejvýkonnějších výpočetních zdrojů, natolik složitá, že je prakticky nereálná. Při posuzování výpočetní bezpečnosti kryptografického systému musí být brán v úvahu velmi rychlý vývoj v oblasti výkonu výpočetních systémů. Samotné požadavky na bezpečnost se stále výrazně mění, a proto se vytváří nové normy a standardy.

1.7 Kryptografický modul

Pojem kryptografický modul je dalším pojmem, kterým je potřeba se zabývat ještě před samotnou definicí postranních kanálů. Kryptografický modul slouží k zajištění všech bezpečnostních cílů, jejichž detailní rozbor je uveden v podkapitole 1.5. Tento modul je v podstatě fyzickou implementací konkrétního kryptografického algoritmu (popř. kryptografického protokolu). Kryptografický modul představuje zařízení, které bývá realizováno do hardwarové nebo softwarové podoby. Uvnitř kryptografického modulu probíhají všechny procesy a citlivé úkony, které jsou spojené s šifrováním, dešifrováním, ověřením, podepisováním, autentizací, apod. Kryptografické moduly komunikují s okolím prostřednictvím vlastních vstupních a výstupních kanálů. V praxi jsou kryptografické moduly realizovány jako šifrátory, programy, počítače, servery, čipové karty, bankomaty, automaty, televizní karty, hardwarové moduly, SIM karty mobilních telefonů, atd. Obecné blokové schéma kryptografického modulu je uvedeno na obr. 1.1.

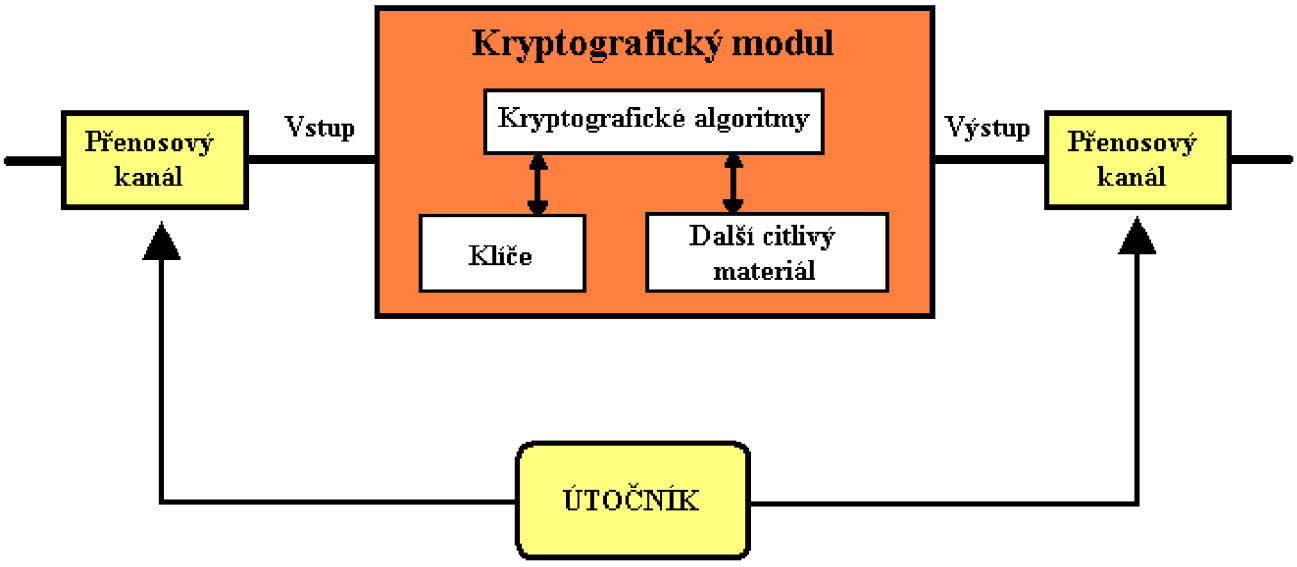


Obr. 1.1: Blokové schéma obecného kryptografického modulu

Praktická realizace kryptografického modulu, která v sobě obsahuje všechny pravidla, klíče a další senzitivní materiál, do značné míry ovlivňuje úroveň bezpečnosti celého systému. Při návrhu a konstrukci kryptografického modulu je třeba dbát na bezpečnostní požadavky.

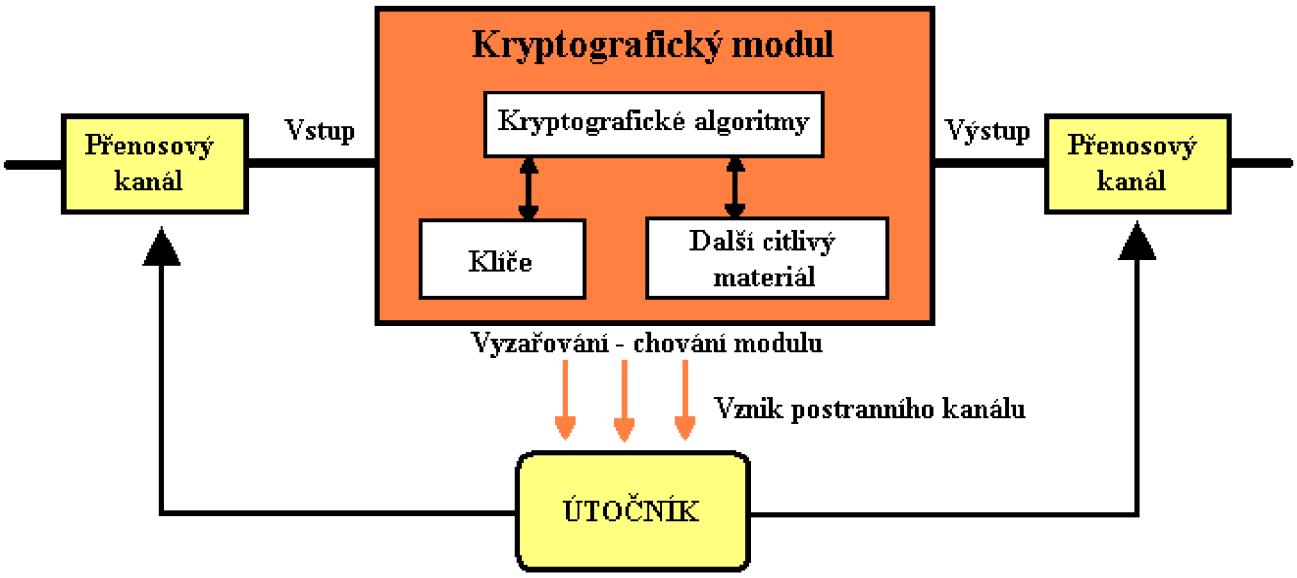
1.7.1 Možnosti útoku na kryptografický modul

Dosavadní konvenční způsob útoku se soustředil přímo na objevení slabiny v matematické podstatě kryptografických algoritmů a protokolů. Ještě před několika lety kryptoanalytici nepředpokládali jiný možný způsob útoku na kryptografický modul. V klasické kryptoanalýze a v jejím způsobu útoku se stal hlavním představitelem tzn. útok hrubou silou (*brute force attack*), který systematicky testuje všechny možné kombinace určitých znaků a stává se úspěšným pokud nalezne takovou kombinaci znaků, která určuje jednoznačně podobu soukromého klíče. Základním problémem těchto útoků je čas, který je potřeba pro vyzkoušení všech kombinací klíče. Výsledný potřebný čas závisí nejen na výpočetních možnostech a zdrojích útočníka, ale především na výběru množiny znaků a také na délce hledaného tajného hesla. Možnosti tohoto způsobu vedení útoku je dostatečně sledováno, a proto je tato metoda velmi neefektivní v boji proti současným kryptografickým algoritmům. Principiální schéma konvenční metody útoku na kryptografický modul je zobrazeno na obr. 1.2.



Obr. 1.2: Konvenční způsob vedení útoku na kryptografický modul

Každé zařízení v reálném světě určitým způsobem ovlivňuje své okolí a naopak toto prostředí působí na každé zařízení, které je v něm umístěno. Kryptografický algoritmus je sice v principu založen na matematické metodě, která sama o sobě nemusí obsahovat žádné slabé místo, ale aby tento algoritmus mohl být využit v reálném světě je nutné ho realizovat, a to implementací do kryptografického modulu. Při konstrukci modulu se předpokládá, že není možná žádná jiná výměna informací s okolím než skrze přesně vymezená pravidla. Ovšem kromě žádané a specifikované komunikace se svým okolím pomocí vstupních a výstupních kanálů vyzařuje každý kryptografický systém do svého okolí určité aspekty svého chování. Reálný kryptografický modul při své činnosti reaguje se svým okolím různými způsoby. Modul může vyzařovat do svého okolí různé informace o svých operacích. Mezi tyto projevy chování kryptografického modulu patří například tepelné, elektromagnetické nebo jiné záření. Každý reálný modul při své činnosti odebírá určitý proud, každá jeho operace způsobuje různé časové zpoždění, na konkrétní situace reaguje modul stavovými a chybovými hlášeními, klávesnice modulu může být mechanicky opotřebená nebo může vydávat různý akustický zvuk pro různé klávesy a podobně. Tyto projevy modulu jsou neodmyslitelně spojeny s činností jeho operací, a proto dochází k nežádoucí komunikaci s okolím, při které mohou být prozrazeny některé ze senzitivních informací. V případě, že uniklá informace je určitým způsobem závislá na dešifrovacím klíči použitého algoritmu, může tato informace útočníkovi ukázat nebo alespoň upřesnit podobu klíče. Tento nežádoucí únik informací je nazván postranním kanálem. Blokové schéma metody možného útoku s využitím postranních kanálů na reálný kryptografický modul je zobrazeno na obr. 1.3.



Obr. 1.3: Útok na kryptografický modul s využitím postranních kanálů

1.8 Postranní kanál

Postranní kanál označuje každý nežádoucí způsob výměny informací mezi okolím a kryptografickým modulem. Návrháři konstrukce kryptografického modulu často neví a ani nemohou vědět o existenci všech nežádoucích postranních kanálů. Existují ovšem některé postranní kanály, které neposkytují žádné důležité senzitivní informace, o které by mohl mít potencionální útočník zájem. V současné době neexistuje žádný konkrétní návod pro návrh zcela imunního kryptografického modulu vůči postranním kanálům. Celá kryptologie nyní stojí před problémem, jak prakticky realizovat třeba i velmi kvalitní abstraktní model, aby ve vzniklém reálném modulu nebyly nežádoucí postranní kanály.

1.8.1 Analýza postranního kanálu

Pro definici útoku postranním kanálem je nutné vymezit i tento pojem. Analýzou postranního kanálu je označován postup, při kterém je možné získat užitečné informace, které lze odvodit ze signálu přicházejícím po tomto kanálu.

1.8.2 Útok postranním kanálem

Útok vedený pomocí postranního kanálu je založen na využití analýzy konkrétního kanálu k napadení daného kryptografického modulu. Útoky postranními kanály jsou podrobně definovány a klasifikovány v následující kapitole 2.

S problematikou postranních kanálů je možné se setkat i v souvislosti s jinými obory. Přesto se tato práce bude zabývat postranními kanály pouze v souvislosti s kryptografickými moduly.

2 ÚTOKY POSTRANNÍMI KANÁLY

V oblasti kryptologie jsou útoky postranními kanály všechny takové útoky na kryptografický modul, které se oproti konvenčnímu způsobu vedení útoku nesoustředí pouze na hledání slabého místa v celkové struktuře kryptografického algoritmu, ale snaží se využít informací, které vyzařují z fyzické implementace systému (tj. z kryptografického modulu) při vykonávání operací algoritmu. Objevení možnosti vedení útoku na implementaci různých systémů pomocí postranních kanálů lze považovat za zcela nový směr v dalším uvažování kryptografů a kryptoanalytiků. Tato metoda vedení útoku na modul byla objevena jen před několika lety. Postranní kanály lze považovat za jisté nebezpečí v kryptografii, protože ještě nejsou provedeny na všech komunikačních, informačních a bezpečnostních systémech potřebné protiopatření vůči tomu způsobu útoků. Faktem je, že ani zavedení nových norem na návrh fyzických implementací systémů a vyvinutí nových a bezpečnějších způsobů ochrany dat ještě nemusí zaručovat, že v reálném modulu nebude vznikat žádný postranní kanál. Proto je nezbytné existenci postranních kanálů v kryptografických modulech brát v úvahu při konstrukci kryptografického modulu.

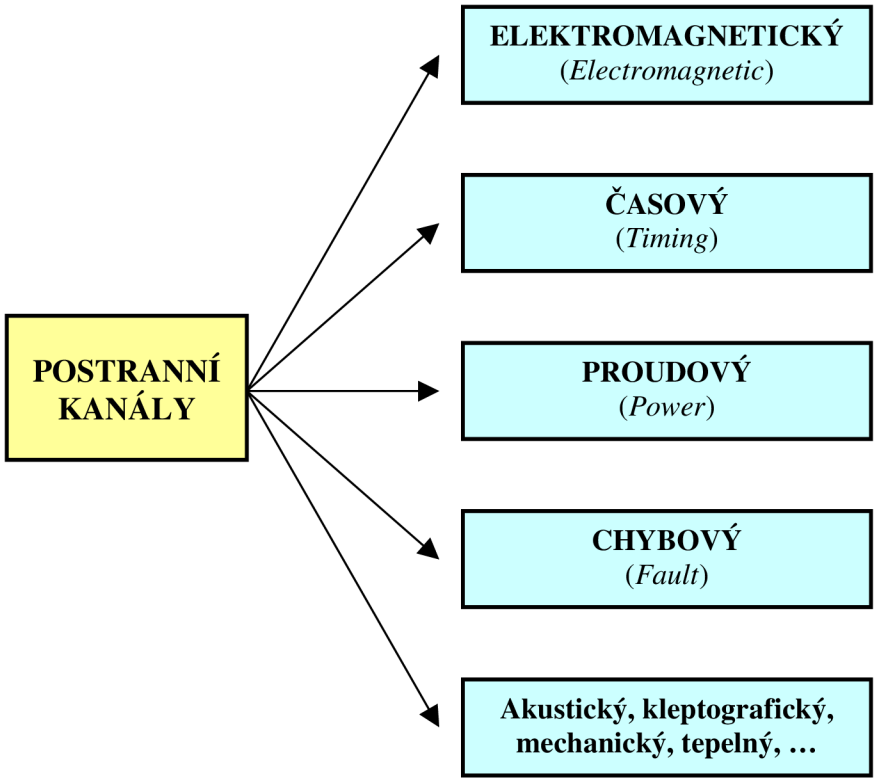
Cílem této kapitoly je nejprve klasifikovat útoky postranními kanály, poté popsat principy jednotlivých druhů postranních kanálů a dále se práce zabývá možnostmi jejich zneužití v různých informačních systémech. Záměrem kapitoly je vytvořit celkový přehled současného stavu problematiky útoků pomocí postranních kanálů.

Cílem diplomové práce je návrh a realizace laboratorní úlohy, která bude vhodně demonstrovat problematiku útoků postranními kanály na kryptografický algoritmus. Jako konkrétní kryptografický systém byl vybrán asymetrický algoritmus RSA. Proto se bude tato kapitola soustředit také na možné způsoby vedení útoků na implementace asymetrického algoritmu RSA.

2.1 Klasifikace útoků postranních kanálů

Tato kapitola se zaměřuje na základní rozdělení útoků podle typu analyzovaného postranního kanálu. Útok postranním kanálem lze chápat jako proces využití postranní informace k napadení kryptografického modulu. Každý typ postranního kanálu je založen na jedné konkrétní měřitelné informaci. Často mívají tyto informace podobu fyzikální veličiny, kterou potencionální útočník má možnost určitým způsobem změřit. Získané hodnoty této fyzikální veličiny mohou být do jisté míry závislé na průběhu

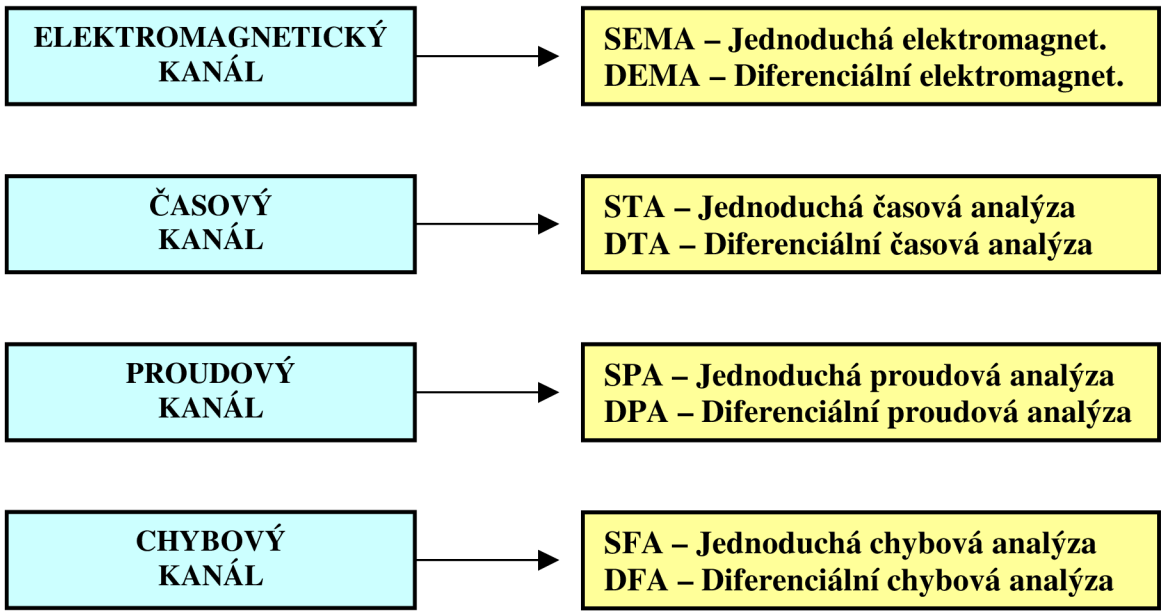
výpočtu jednotlivých operací kryptografického modulu. Potom zcela logicky je nazván daný postranní kanál podle druhu fyzikální veličiny nebo informace, kterou postranní kanál využívá. Poté lze hovořit o postranních kanálech elektromagnetických, akustických a podobně. Kryptoanalytici považují v současné době za hlavní druhy postranních kanálů, které lze využít s vysokou efektivitou při útoku na implementaci systému, především následující postranní kanály: elektromagnetický, časový, proudový (výkonový) a chybový kanál. Klasifikace postranních kanálů je přehledně znázorněna na obrázku obr. 2.1.



Obr. 2.1: Klasifikace postranních kanálů

Při útoku postranním kanálem se využívá zpracování a vyhodnocení získaných informací prostřednictvím analýzy kanálu. V oblasti kryptografie a postranních kanálů byly definovány dva druhy analýzy postranních kanálů. Jedná se o **jednoduchou** (*Simple*) a **diferenciální** (*Differential*) analýzu postranních kanálů. U každého druhu postranního kanálu je možné využít obou druhů analýzy. Toto rozdělení bylo zavedeno pro zpřesnění a zpřehlednění celé oblasti studia útoků postranních kanálů.

Klasifikace postranních kanálů a jejich analýz je zobrazena na obr. 2.2. Všechna terminologie a používané zkratky pochází z anglického jazyka.



Obr. 2.2: Klasifikace postranních kanálů a jejich analýzy

Pod pojmem jednoduchá analýza je označován snadný způsob zpracování informací a výsledků, které útočník získal při svém útoku na kryptografický modul. V praxi to znamená, že samotný útočník je schopen jednoduše vyhodnotit z postranního kanálu určité citlivé informace a to bez použití speciálních výpočetních metod.

Diferenciální analýza oproti tomu využívá statistických a matematických metod pro získání citlivých informací z postranního kanálu.

2.2 Elektromagnetický postranní kanál

Tento typ postranního kanálu je založen na skutečnosti, kdy průběh některých dějů ve fyzické implementaci algoritmu je doprovázen elektromagnetickým zářením. Tento fakt je způsoben tím, že všechny kryptografické moduly jsou složeny z elektronických částí, které pracují vždy s elektrickým proudem, a proto také vyzařují do svého blízkého okolí elektromagnetické vlny.

2.2.1 Historie

Elektromagnetický postranní kanál byl nejvíce ve své historii využíván v armádních složkách a tajných službách. Proto se také tyto organizace a úřady odborně zabývaly studiem problematiky parazitních emisí, která se označuje zkratkou TEMPEST [8]. Hlavním zájmem vojenských organizací bylo zabezpečení proti tomuto nežádoucímu

vyzařování a naopak také možnost jejího využití k monitorovací a 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. Cílem zmíněných standardů bylo omezit nebo zabránit možnosti vedení útoku s využitím elektromagnetického vyzařování na elektronické zařízení.

2.2.2 Princip útoku

Kryptografický modul složený z elektronických součástí při své činnosti vyzařuje elektromagnetické záření. Potencionální útočník, který disponuje potřebným vybavením, je potom schopen toto elektromagnetické záření zachytit a analýzou získat senzitivní informace. Zaznamenané elektromagnetické záření lze analyzovat přímo pomocí jednoduché elektromagnetické analýzy (SEMA) nebo lze využít speciálních matematických prostředků v rámci diferenciální elektromagnetické analýzy (DEMA). Často jsou útoky elektromagnetickým postranním kanálem zahrnuty do problematiky útoku proudovým (výkonovým) postranním kanálem.

Elektromagnetické záření je způsobeno změnou proudů při činnosti kryptografického modulu. Tato změna vyvolává vznik střídavého magnetického pole, jestliže je toto pole dostatečně silné, je útočník schopen zachytit a využít informace z tohoto záření. Základní elektronické součástky (např. tranzistory) nabývají při své činnosti stavu logické úrovně „0“ nebo úrovně „1“. Elektromagnetický postranní kanál využívá právě přechody mezi těmito dvěma stavy. Tento přechod se projeví změnou intenzity elekromagnetického pole v okolí sledovaného modulu. Podle druhu prováděné vnitřní operace s citlivými informacemi se určitým způsobem mění také intenzita pozorovaného elektromagnetického pole v okolí modulu.

Uvedená skutečnost je zjednodušeným principem, jak je možné získat informace týkající se vnitřních úkonů s citlivými daty v kryptografickém modulu.

2.2.3 Příklady možných útoků

Pravděpodobně nejstarším úspěšným útokem elektromagnetickým postranním kanálem a navíc velmi známým zástupcem je útok na CRT (cathode ray tube) monitory. Vychylovací cívka, která ovládá posun elektronového paprsku v obrazovce CRT monitoru, v principu pracuje jako anténa a její elektromagnetické pole je možné detekovat i ve vzdálenosti řádově několika metrů. S potřebným vybavením může útočník ze zachyceného signálu zrekonstruovat obraz na napadeném CRT monitoru, který je umístěn například ve vedlejší místnosti. Tento způsob vedení útoku postranním

kanálem na CRT monitor byl v minulosti často využíván armádními nebo tajnými složkami, kdy docházelo k odposlechu elektromagnetického signálu monitoru. V dnešní době je ovšem četnost použití tohoto typu monitoru s využitím vychylovací cívky velmi nízká.

V současné době se řada kryptoanalytiků zaměřuje ve svých publikacích a výzkumech na popis a řešení problematiky útoku elektromagnetickým postranním kanálem na kreditní čipové karty (*smart card*). V současné době je známo několik typů útoků elektromagnetickým postranním kanálem na čipové nebo GSM karty. Jedním z příkladů je útok na GSM karty, u kterých byla dokázána rapidní časová úspora při prolomení ochranných prvků SIM karet právě využitím elektromagnetického postranního kanálu. Na tuto možnost útoku upozornila jedna z divizí společnosti IBM, která se dlouhodobě od roku 2002 věnuje bezpečnosti v oblasti mobilních technologií. Základem tohoto útoku je skutečnost, že pomocí sledovaných postranních kanálů (elektromagnetické vyzařování a spotřeba elektrické energie) lze získat tzv. COMP128 klíč ze SIM karty ve velmi krátkém čase ve srovnání s dosavadní možností útoku hrubou silou. Podle informací firmy IBM je pro úspěšný útok dostačující, aby SIM karta provedla alespoň sedmkrát pokus o vyhodnocení klíče s použitím neznámého klíče [26].

V poslední době se útoky elektromagnetickým postranním kanálem dostávají do popředí výzkumu, protože projevy elektromagnetického postranního kanálu není jednoduché plně odstranit z kryptografického modulu, a proto je zcela nevyhnutelné se zabývat novými bezpečnostními opatřeními před tímto druhem útoků.

2.3 Časový postranní kanál

Časový postranní kanál je prvním publikovaným a typickým příkladem postranních kanálů. Tento postranní kanál vzniká v takových kryptografických modulech, kde rychlost průběhu operace podstatným způsobem závisí na vstupních datech.

2.3.1 Historie

Základní myšlenka útoku časovým postranním kanálem byla poprvé publikována ve vědecké literatuře v roce 1996. S existencí možnosti vedení útoku na kryptografický modul zmíněným způsobem přišel známý americký kryptograf Paul Carl Kocher. Časový útok na fyzickou implementaci algoritmu bývá často nazýván podle svého objevitele jako Kocherův útok.

2.3.2 Princip útoku

Časový útok je založen na měření času, který je potřeba k vykonání určité operace ve sledovaném modulu. Pod pojmem operace uvnitř kryptografického modulu chápeme například výpočetní úkony potřebné k šifrování nebo dešifrování vstupních dat. Samozřejmě kryptografické moduly často provádějí různé úkony různou dobu v závislosti nejen na soukromém klíči, ale také v závislosti na vstupních datech. 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. V praxi útočník vysílá na vstup programu data a zároveň zaznamenává, jak dlouho trvá jejich zpracování kryptografickým modulem.

2.3.3 Příklady možných útoků

Každý konkrétní útok, založený na uvedeném principu využívá různé operace příslušného kryptografického modulu a různé sofistikované statistické nástroje pro vyhodnocování naměřených časových údajů. Jako první ukázal na možnou hrozbu časového útoku už zmiňovaný Paul Kocher, který ve svém publikovaném článku [7] uvedl konkrétní příklady využití časového postranního kanálu při útoku na fyzickou implementaci RSA, DSA, Diffie-Hellman a dalších algoritmů. V současné době je již ověřeno, že možnost útoku časovým postranním kanálem nastává i u symetrických algoritmů (RC5, AES, DES, IDEA). U kryptografického symetrického algoritmu RC5 se využívá časové závislosti bitových rotací se soukromým klíčem. Dále například časový útok u symetrického algoritmu AES využívá operace MixColumn a u algoritmu DES je založen na časově závislé přípravě klíče.

2.3.4 Časový útok na implementaci RSA

Tato práce se detailně zabývá studiem možností útoků postranními kanály na algoritmus RSA. Nejvíce sledovaným časovým útokem je využití postranního kanálu modulu s implementovaným RSA algoritmem. Tento asymetrický algoritmus je založen na matematické operaci modulární mocniny. Časový útok na soukromý klíč RSA algoritmu využívá samotné časově závislé operace modulární mocniny

$$y = (m^d) \bmod n \,, \tag{2.1}$$

kde m je reprezentováno vstupními daty, n označuje veřejný modul algoritmu a d je zastoupeno vždy jednotlivými bity soukromého klíče. Pro samotný výpočet modulární mocniny se používá tzv. algoritmus *square and multiply*, který je založen na postupném

zpracování samostatných bitů soukromého klíče *d* a jeho podobu můžeme vidět na obr. 2.3 [15].

```
1    R=m
2    for  i=1  to  (b-1)
3         {
4         R=(R*R)mod  n
5         if  (d(i)==1)
6              R=R*m)mod  n
7         }
8    return R
```

Obr. 2.3: Algoritmus square and multiply

Časový útok u implementace algoritmu RSA se soustředí na výpočet modulární mocniny využívající se při operaci dešifrování a podpisu. Jednotlivé bity soukromého klíče exponentu *d* lze popsat

$$d = d_{(0)}\, d_{(1)}\, d_{(2)}\, d_{(3)} \ldots d_{(b\text{-}1)}\,, \tag{2.2}$$

kde *b* udává počet všech platných bitů soukromého klíče *d*. Podstatou časového útoku je skutečnost, že doba trvání průchodu smyčkou zobrazené na obr. 2.3 je závislá na tom, jestli je hodnota daného bitu soukromého klíče $d_{(i)}$ rovna jedné nebo nule. V situaci, kdy má tento konkrétní bit klíče hodnotu nuly vůbec neproběhne výpočet na 6.řádku uvedeného algoritmu. Z časového hlediska bude doba trvání průchodu smyčkou velmi malá. Při opačné hodnotě bitu klíče bude průchod celou smyčkou razantně pomalejší. Z naměřených časových údajů je možné přímo odečíst bitovou podobu celého soukromého klíče. Výhodou útočníka je znalost vstupních dat *m*, které má možnost sám volit a posílat na vstup napadeného modulu.

Ve skutečnosti útok časovým postranním kanálem není jednoduchou záležitostí, protože nelze určit jednotlivé časové intervaly průchodu smyčkou, ale pouze celkovou dobu prováděné operace dešifrování. I přesto je útočník schopen zjistit alespoň Hammingovu váhu, tedy počet nenulových bitů tohoto soukromého klíče.

V dnešní době již ovšem existují speciální statistické metody, kterými lze výrazně ovlivnit úspěšnost i použitelnost uvedeného útoku časovým postranním

kanálem. Popsaný časový útok založený na modulární mocnině je možné využít podobným způsobem také u ostatních asymetrických algoritmů (DSA, D-H).

2.4 Proudový (výkonový) postranní kanál

Tato kapitola se věnuje popisu principu proudového nebo také výkonového postranního kanálu. V současné době se stává tento typ postranního kanálu velmi populární u kryptoanalytiků a odborníků, kteří se zabývají problematikou postranních kanálů v nejrůznějších informačních systémech. Proudový postranní kanál je možné najít u každého modulu, který obsahuje určitou elektronickou část.

2.4.1 Princip útoku

Základním principem tohoto typu postranního kanálu je skutečnost, že velikost energie spotřebované kryptografickým modulem je přímo závislá na právě prováděném druhu operace. Každé elektronické zařízení odebírá v klidovém režimu mnohonásobně menší proud, než při vykonávání složitých výpočtů. A právě této podstaty s výhodou využívá možný útočník, který sledováním spotřeby proudu při vykonávání různých operací uvnitř modulu má možnost získat pomocí jednoduché nebo diferenciální analýzy chtěné citlivé informace.

Oproti uvedenému časovému útoku, kdy je možné v praxi snadným způsobem zjistit pouze dobu trvání celé makroskopické operace modulu, je proudový útok postranním kanálem o poznání efektivnější, neboť může přinést informace o chování konkrétní části určité operace.

Proudovým útokem jsou často napadnutelné především čipové karty, protože nedisponují autonomním zdrojem a musí tak být externě napájeny. Z toho důvodu může útočník snadno měřit proudovou spotřebu. Hlavní výhoda proudového způsobu útoku u čipových karet spočívá ve snadné možnosti měření spotřeby při komunikaci s čtecím zařízením.

Samotná podstata vzniku proudového postranního kanálu je obdobná podstatě zmíněné v kapitole 2.2, která se zabývá problematikou elektromagnetických kanálů. Základním stavebním prvkem elektronických zařízení je tranzistor. V současné době jsou moderní elektronické moduly založeny na technologii CMOS, která využívá jako elementární elektronickou součástku invertor. Vnitřní zapojení invertoru je konstruováno ze dvou tranzistorů a kondenzátoru. Přičemž největší proudovou spotřebu má invertor při přechodu mezi svými stavy. Tato časově velmi krátká proudová spotřeba se ale projeví na sledovaném postranním kanále a může případnému útočníku zajistit informace o aktuálních dějích uvnitř modulu.

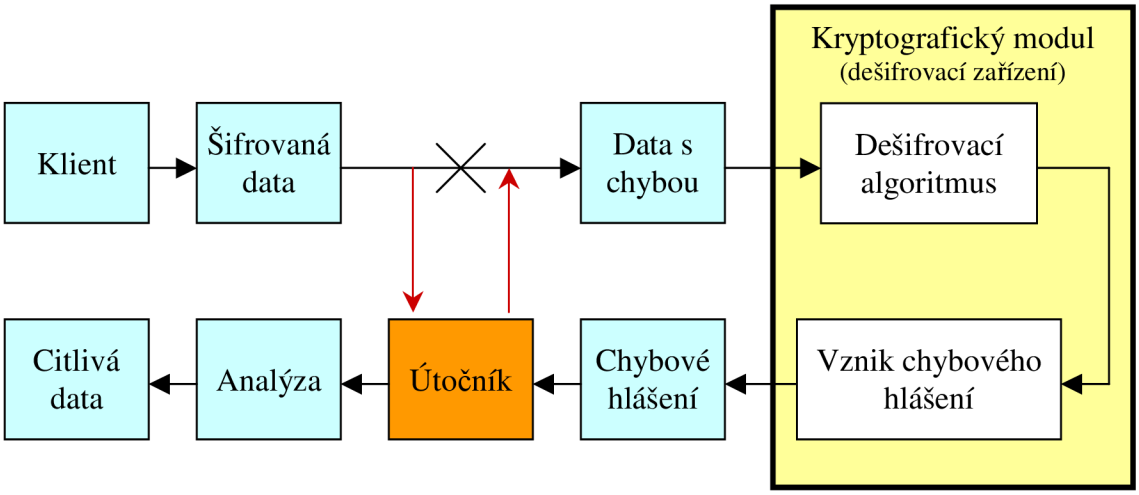
Při útoku vedeným tímto způsobem jsou naměřené signály proudového odběru silně ovlivňovány složkou šumu. Také z tohoto důvodu se využívá speciálních metod a analýz pro potlačení vzniklého šumu i zpracování získaných informací.

2.5 Chybový postranní kanál

Dosud se práce zabývala studiem principů postranních kanálů, které jsou založené na určité fyzikální vlastnosti kryptografického modulu. Tentokrát jsou podstatou vzniku postranního kanálu právě chybová hlášení. Několik let se všichni kryptologové plně věnovali bezpečnosti struktury výpočetních algoritmů a považovali chybová hlášení jako vedlejší činnost systému, která nemůže mít pro případného útočníka jakýkoliv význam. Přesto se chybová hlášení, které byly považovány za bezcenné, stávají další možností pro napadení kryptografických modulů.

2.5.1 Princip útoku

Útočník využívá chybového hlášení systému ke získání senzitivních informací uložených uvnitř modulu. Celková úspěšnost útoku chybovým postranním kanálem se značně odvíjí od možnosti útočníka modifikovat přicházející zašifrovaná data takovým způsobem, aby tato úprava způsobila odpovídající odezvu modulu, ve formě chybového hlášení o zamýšleném druhu selhání. Princip útoku uvedeným způsobem na kryptografický modul je možné pozorovat na obr. 2.4.



Obr. 2.4: Princip útoku chybovým postranním kanálem

Zachycený chybový útok začíná objevením šifrované komunikace mezi klientem a zařízením, následně útočník modifikuje zachycená data a způsobí v průběhu výpočtu dešifrovacího zařízení chybu, kterou zařízení rozpozná a odešle zpět klientovi informace ve formě chybového hlášení. Tyto informace se ovšem zpětně dostanou k útočníkovi, který vytváří a ukládá záznamy, které využívá pro následnou analýzu. Tento postup útoku je opakován do té doby, dokud útočník nemá dostatek zdrojů k odhalení citlivých informací uvnitř napadeného modulu.

2.5.2 Chybový útok na implementaci RSA

Nejčastěji se lze setkat s využitím útoku chybovým kanálem u implementace asymetrických algoritmů, mezi které patří zejména algoritmy RSA, DSA. V současné době je velké množství dokázaných příkladů útoků touto metodou, a proto jsou v této práci uvedeny pouze některé základní útoky.

Často zmiňovaným příkladem chybového útoku je útok na schéma digitálního podpisu RSA. Podstata útoku spočívá v napadení operace dešifrování, která využívá výpočetní metodu CRT (čínská věta o zbytcích) [19]. Základní princip útoku na schéma digitálního podpisu RSA vychází ze skutečnosti, že samotný výpočet podpisu se provádí ve dvou současně probíhajících krocích, jejichž výsledky se pomocí zmíněné metody CRT kombinují do konečné podoby digitálního podpisu. Správně provedenou modifikací lze potom vyvolat takovou reakci modulu ve formě chybového hlášení, ze kterého je možné získat cenné informace. V praxi ovšem není jednoduché najít a provést odpovídající úpravu zachycených dat. Bližšímu pohledu na tuto problematiku se věnuje například literatura [20, 13].

S další možností využití chybového postranního kanálu přišel v roce 1998 švédský kryptolog Daniel Bleichenbacher. Ten poukázal na možnost útoku, která vede k získání původní nešifrované zprávy. 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. Pro úspěšný útok tímto způsobem je potřeba řádově milióny dotazů na napadený modul. Další podrobné informace lze nalézt přímo v původní vědecké publikaci švédského kryptologa [17].

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. Metoda kódování

používá maskování zprávy, která využívá také vhodnou bezpečnou hašovací funkci. K výsledku kódování OAEP se přidává nulový bajt, který při nevhodné implementaci může vést k vyzařování senzitivních informací o zprávě. Na tuto potenciální chybu formátovací metody OAEP upozornil v roce 2001 kryptolog Manger. Manger dokázal existenci postranního kanálu využívající nedokonalosti implementace metody OAEP [9, 25].

Kromě uvedených typů využití chybových postranních kanálů existuje celá řada dalších druhů útoků (např. útok na blokovou šifru v CBC módu). V současné době se ukazuje, že útoky pomocí postranních kanálů patří mezi nejvíce efektivní a obecně se tak stávají nejúčinnějšími metodami v kryptoanalýze.

2.6 Útoky na implementaci RSA

Celá diplomová práce se zaměřuje na využití postranních kanálů při útoku na asymetrický algoritmus RSA. V současné době jsou známy tři hlavní druhy útoků využívající postranní kanály kryptografických modulů RSA.

- **Chybový útok**
  - Bleichenbacherův útok – 1998
  - Mangerův útok – 2001
- **Časový útok**
  - Kocherův útok – 1996

Z důvodů jednoduchosti a názornosti se navrhovaná laboratorní úloha bude soustředit na útok vedený pomocí časového postranního kanálu, tj. na Kocherův útok.

2.7 Možnosti zabezpečení

Záměr této kapitoly spočívá v naznačení principů možných zabezpečení v implementacích algoritmů, ve kterých je možné uskutečnit útok postranním kanálem. Techniky zabezpečení lze rozdělit stejným způsobem jako kryptografické moduly, tj. na softwarové a hardwarové. Preferovány jsou softwarová protiopatření, jejichž začlenění do implementace je ekonomicky nenáročné a výrazně jednodušší.

Cílem softwarových technik zabezpečení je určitým způsobem zajistit, aby každá z výpočetních operací v implementaci algoritmu vykazovala konstantní, například časový průběh. Tímto způsobem je možné eliminovat použitelnost časového nebo výkonového postranního kanálu.

Mezi základní používané techniky zabezpečení a ochrany před zneužitím postranních kanálů patří například:

- Zavedení nového doporučení – standardu,
- Využití techniky *blinding* – slepé podpisy,
- Využití techniky maskování dat,
- Zavedení náhodnosti,
- Odstranění pravidelností,
- Zavedení prázdných instrukcí.

3 MODULÁRNÍ ARITMETIKA

V souvislosti s kryptoanalýzou a studiem nových typů útoků na kryptografické algoritmy je nutné se zabývat podstatou bezpečnosti těchto algoritmů, která má své základy právě v modulární aritmetice. Modulární aritmetika je aritmetikou na množině celých čísel, v níž se čísla opakují po dosažení určité hodnoty *n*, která je nazvána modulem [19].

Bezpečnost používaných algoritmů je založena na určitém matematickém problému. Tato práce řeší především problematiku útoků časovými postranními kanály konkrétně na implementaci asymetrického algoritmu RSA. Bezpečnost tohoto algoritmu je založena na složitosti řešení úlohy faktorizace velkých čísel, tj. rozkladu čísel na prvočinitele. Míra bezpečnosti je potom přímo závislá na použité délce modulu. Tato část práce bude zaměřena pouze na některé oblasti modulární aritmetiky, jejichž znalost je zásadní pro studium časového útoku na asymetrické algoritmy.

3.1 Montgomeryho metoda

Montgomeryho metoda je velmi často používanou metodou, a to zejména v oblasti kryptografie. V praxi se Montgomeryho metoda objevuje v mnoha modifikacích pro softwarové i hardwarové implementace.

Jedná se o jeden z postupů, jak urychlit základní operace modulárního násobení, které ve značné míře používá i RSA algoritmus. Tato metoda se díky zmíněnému urychlení výpočtů velmi rozšířila do mnoha kryptografických modulů RSA. Praktickým využitím této metody je již uvedený algoritmus *square and multiply*, který se využívá právě při útoku časovým postranním kanálem na implementaci RSA.

3.2 Čínská věta o zbytcích

Čínská věta o zbytcích je matematická věta, která je často označována zkratkou CRT (Chinese Remainder Theorem). Pomocí tohoto matematického teorému se dosahuje velkého zkrácení časů při provádění operací s klíči algoritmu RSA (šifrování a dešifrování).

Samotná implementace Montgomeryho metody násobení i čínské věty o zbytcích do kryptografického modulu má za následek velké urychlení prováděných operací, avšak zároveň zapříčiňují vznik časového postranního kanálu a otevírají nové možnosti útoku na soukromý klíč algoritmu.

4 JEDNODUCHÝ ČASOVÝ ÚTOK

Existence hrozby možných útoků s využitím postranních kanálů, která je podrobně rozebrána ve 2. kapitole, bude ověřena vlastní praktickou realizací. Ve zvoleném integrovaném prostředí MATLAB bude realizován mírně zjednodušený případ časového útoku na asymetrický algoritmus RSA.

4.1 Algoritmus RSA

Algoritmus RSA (iniciály autorů – Rivest, Shamir, Adleman) byl vyvinut v roce 1977 a stále je hlavním představitelem asymetrických kryptosystémů. V současné době je tento asymetrický algoritmus považován za bezpečný při použití dostatečné délky klíče. Za bezpečnou délku klíče se považuje klíč i modul s minimální délkou 768 bitů. V současné době se používá modul o délce 1024 nebo 2048 bitů. Algoritmus RSA je vhodný jak k šifrování dat, tak i k jejich digitálnímu podpisu. Hlavním problémem hardwarových i softwarových implementací algoritmu RSA je jejich nízká rychlost oproti srovnatelným symetrickým systémům. K urychlení prováděných operací se využívá Montgomeryho metoda i čínská věta o zbytcích.

4.1.1 Princip

Podstata algoritmu RSA spočívá v obtížnosti řešení problému faktorizace velkých čísel, která není v současné době řešitelná v takovém čase, který by byl pro případného útočníka uplatnitelný. Z modulu n, který je určen součinem prvočísel p a q, tedy není možné v rozumném čase získat daná prvočísla p a q. Jak už bylo poznamenáno v podkapitole 2.3.4 je algoritmus RSA založen na matematické operaci modulární mocniny.

Celý algoritmus RSA je přesně určen parametry: $p, q, n, \varphi(n), e, d$. Samotná operace šifrování je uskutečněna vztahem

$$c = m^e \bmod n, \tag{4.1}$$

kde c je zašifrovaný text a m je vstupní zpráva. Zpětná operace (tj. dešifrování) je dána vztahem

$$m = c^d \bmod n. \tag{4.2}$$

4.1.2 Výpočet páru klíčů

Výpočet parametrů asymetrického algoritmu RSA lze rozdělit do uvedených šesti kroků:

1. Volba dvou náhodných velkých prvočísel p a q ($>10^{115}$),
2. Výpočet jejich součinu udává modul $n = pq$,
3. Výpočet hodnoty Eulerovy funkce $\varphi(n) = (p - 1)(q - 1)$,
4. Volba veřejného klíče e, musí být menší než $\varphi(n)$ a nesoudělný s $\varphi(n)$,
5. Výpočet soukromého klíče d z podmínky $(de) \bmod \varphi(n) = 1$,
6. Veřejné parametry jsou klíč e a modul n.

Po výpočtu a ustavení páru klíčů se mohou zveřejnit pouze dva parametry veřejný klíč e a modul n, ostatní parametry zůstávají utajeny.

4.2 Realizace časového útoku

Již dříve bylo vysvětleno, že při časovém útoku se využívá skutečnosti, kdy téměř všechny operace v kryptografickém modulu závislé na soukromém klíči trvají krátkou nebo dlouhou dobu v návaznosti na tom, jaké jsou hodnoty jednotlivých bitů soukromého klíče.

Při realizaci tohoto zjednodušeného časového útoku na RSA se využívá operace dešifrování, ve které je možné z časového postranního kanálu získat citlivé informace, tj. hodnoty jednotlivých bitů soukromého klíče d. V prostředí programu MATLAB byly vygenerovány všechny potřebné parametry algoritmu způsobem uvedeným v kapitole 4.1.2. Pomocí algoritmu *square and multiply* (obr. 2.3) je zajištěna závislost výpočtu původní zprávy m na jednotlivých bitech soukromého klíče d. S využitím příkazů programu MATLAB pro časového měření je možné zjistit relativně přesnou dobu trvání průchodu smyčkou *if*, jejíž výpočet je zcela závislý na hodnotě konkrétního bitu soukromého klíče. Při testování útoku časovým postraním kanálem je pro jednoduchost generován relativně krátký soukromý klíč d s délkou 31 bitů. Pro přesnější analýzu by bylo možné provést sadu opakovaných měření a pro výsledné časové intervaly jednotlivého bitu soukromého klíče určit aritmetický průměr.

```
for i = 1:length(deklic_bin)
    x = rem((x^2),n);
    if deklic_bin(i) == num2str(1)
        x= rem(x*c,n);
    end;
end;
```

Obr. 4.1: Konkrétní podoba algoritmu square and multiply v prostředí MATLAB

Časový útok se provádí právě na vyznačenou část uvedeného algoritmu *square and multiply* (obr. 4.1).

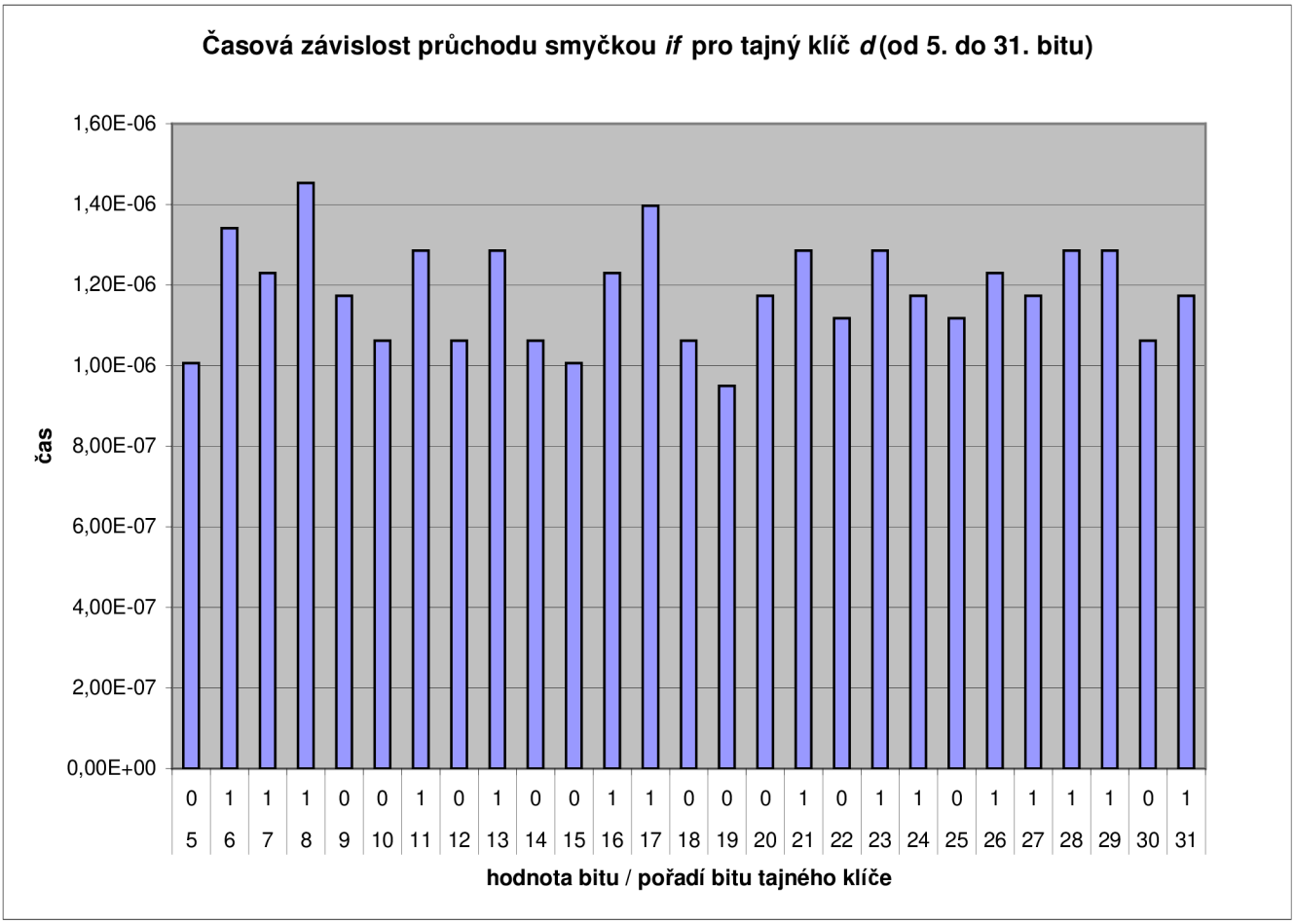
4.3 Dosažené výsledky

Naměřené časové hodnoty jsou získány při těchto náhodně vygenerovaných parametrech algoritmu RSA:

```
p=126127,
q=32159,
n=4056118193,
e=37189,
d=1133823421.
```

Měření času průchodu smyčkou u algoritmu *square and multiply* pro prvních několik bitů soukromého klíče je zatíženo relativně velkou nepřesností, která je pravděpodobně způsobena především počáteční inicializací parametrů. Proto je časová závislost na hodnotě bitu klíče uvedena až po tomto ustálení, tj. od 5.bitu soukromého klíče *d*. Z uvedené časové závislosti na obr. 4.2 je zřejmé, že útoky vedené časovým postranním kanálem je možné realizovat i v prostředí programu MATLAB. Z naměřených dat lze velmi dobře odhadnout zda má konkrétní bit soukromého klíče *d* nulovou nebo nenulovou hodnotu.

Naměřená data k zobrazené časové závislosti na hodnotě bitů klíče jsou získána aritmetickým průměrem z pěti samostatných měření. Z obr. 4.2 je zřetelné, že tímto způsobem útoku časovým postranním kanálem je možné získat určité senzitivní informace (v našem případě přímo soukromý klíč *d*).



Obr. 4.2: Časová závislost průchodu smyčkou algoritmu square and multiply (pro 5. až 31.bit soukromého klíče *d*)

V praxi ovšem není možné při útoku jednoduchým způsobem zjistit dobu trvání průchodu smyčkou při konkrétním bitu soukromého klíče, ale jsme schopni změřit pouze celkovou dobu dešifrování vstupní zprávy. I přesto lze pomocí tohoto typu útoku a pomocí jednoduché analýzy získat různé citlivé informace z dějů uvnitř kryptografického modulu (např. Hammingovu váhu klíče).

5 NÁVRH LABORATORNÍ ÚLOHY

Cílem diplomové práce je navrhnout a realizovat laboratorní úlohu, která by mohla jednoduchým způsobem studentům demonstrovat nové možnosti, které nabízí právě existence postranních kanálů v jednotlivých kryptografických modulech.

5.1 Výběr typu postranního kanálu

Historicky první objevený postranní kanál byl časový postranní kanál. Z důvodu velké názornosti byl pro účely laboratorní úlohy vybrán právě útok časovým postranním kanálem. Další podstatnou výhodou volby časového útoku je možnost demonstrovat tuto problematiku i jen s využitím softwarové implementace. Díky této skutečnosti jsou technické nároky na každé laboratorní pracoviště velmi nízké.

5.2 Cíle laboratorní úlohy

Před samotnou realizací laboratorní úlohy je důležité stanovit pevné cíle, které jsou od této úlohy očekávány. Hlavním požadavkem je, aby laboratorní úloha studentům přinesla základní informace o existenci postranních kanálů a zároveň připomněla i prohloubila souvislosti s celou kryptografií v informatice.

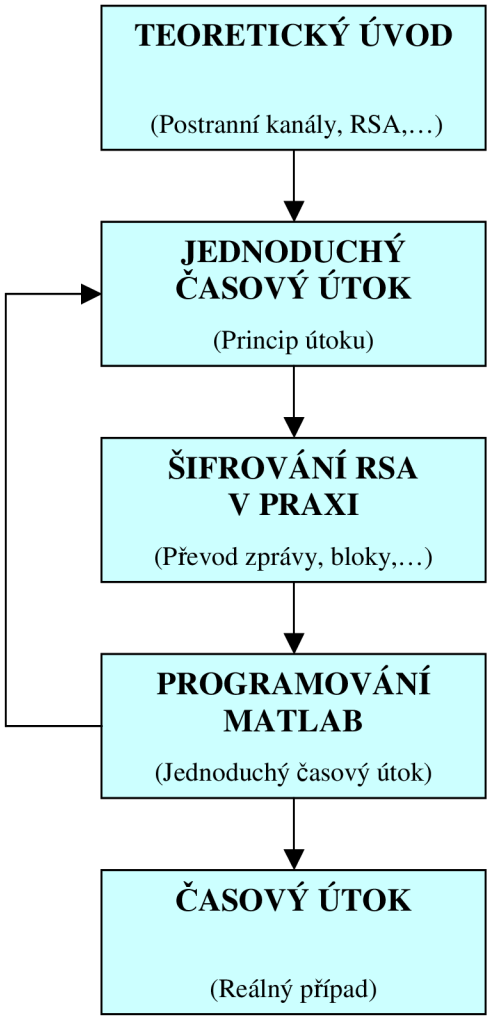
Absolvováním navrhované laboratorní úlohy by studenti měli získat nové zkušenosti a poznatky v následujících oblastech.

- Postranní kanály v kryptografii
  - Definice
  - Klasifikace
  - Nový způsob útoků
- Časový útok
  - Princip
  - Útok v praxi
- RSA algoritmus
  - Princip
  - Příprava parametrů
  - Operace šifrování v praxi
- MATLAB
  - Programování, GUI
  - Algoritmus *square and multiply*

Mimo uvedené body je důležitým cílem laboratorní úlohy, aby studenti měli možnost si prakticky vyzkoušet útok časovým postranním kanálem na algoritmus RSA a zároveň se zamyslet také nad možnostmi zabezpečení proti těmto druhům útoků.

5.3 Struktura laboratorní úlohy

Tato kapitola se zabývá návrhem obecné struktury laboratorní úlohy. Provedení laboratorní úlohy by mělo splňovat všechny požadavky, které jsou souhrnně uvedeny v kapitole 5.2.



Obr. 5.1: Navržená struktura laboratorní úlohy

Teoretický úvod musí obsahovat všechny potřebné informace pro pochopení a splnění všech jednotlivých úkolů laboratorní úlohy. Jeho náplní musí být definice postranních kanálů, klasifikace postranních kanálů, jejich přímá souvislost

s kryptografickými moduly, výpočet parametrů algoritmu RSA, objasnění potřebných funkcí programu MATLAB a další důležité informace potřebné při řešení této laboratorní úlohy.

Navržená struktura laboratorní úlohy plně zajišťuje uspokojení zmíněných požadavků. Uvedenou strukturou se bude řídit následná realizace laboratorní úlohy.

6 REALIZACE LABORATORNÍ ÚLOHY

Na základě návrhu, stanovených cílů a uvedené struktury laboratorní úlohy (obr. 5.1) bude úloha prakticky realizována a to formou, která by byla nejvhodnější pro podmínky běžné počítačové učebny.

Hlavní součástí celé laboratorní úlohy je uživatelská aplikace, která v sobě zahrnuje zjednodušený i reálný případ časového útoku na algoritmus RSA. Tato aplikace je vytvořena ve zvoleném vývojovém prostředí MATLAB.

6.1 MATLAB - systém handle graphics

Vzhledem k využití aplikace jako laboratorní úlohy bylo potřeba naprogramovat také vhodné interaktivní grafické rozhraní (GUI – Graphical User Interface). K tomuto účelu byl využit právě systém *handle graphics* prostředí MATLAB. Uvedený grafický systém je nástroj, s jehož pomocí lze efektivně pracovat s grafickými objekty v integrovaném prostředí MATLAB. Zahrnuje jednotlivé příkazy pro 2D a 3D vizualizaci dat, zpracování signálů, animaci a další grafické nástroje. Z pohledu uživatele přináší vyšší efektivitu a mnoho možností při práci s počítačovou grafikou [24].

Kromě popsaného systému nabízí prostředí MATLAB také druhou variantu a tím je způsob tvorby GUI pomocí nástroje GUIDE. Tento nástroj slouží k jednoduché, rychlé a interaktivní tvorbě grafického uživatelského rozhraní. Ovšem každé z těchto nástrojů má své výhody i nevýhody.

Pro vývoj uživatelské aplikace laboratorní úlohy bylo využito kombinace obou zmíněných nástrojů výpočetního prostředí MATLAB.

6.2 Implementace RSA

Pro realizaci časového útoku bylo potřeba vhodným způsobem implementovat algoritmus RSA do MATLABu. Což zahrnuje především generování všech parametrů potřebných pro správnou činnost operací RSA.

6.2.1 Generování prvočísel

Generace prvočísel je využita pro demonstrační část časového útoku, která má studentům poskytnout plné pochopení principu samotného vzniku a negativní činnosti časového postranního kanálu. Generování prvočísel je uzpůsobeno potřebám laboratorní úlohy, tj. délka výsledného soukromého klíče by neměla přesáhnout 35 bitů.

Tohoto omezení velikosti prvočísel je učiněno z důvodu přehlednosti a možnosti celkového grafického zobrazení v rámci vytvořené uživatelské aplikace.

```
39        %generace prvocisla p
40 -      max = 100000;
41 -      generace_prvoc = 0;
42
43 -      while (generace_prvoc == 0),
44 -          p = fix(rand * max);
45 -          prvocislo = 1;
46
47 -          for i = 2:fix(p^0.5)
48 -            if rem(p, i)==0
49 -                prvocislo = 0;
50 -            end;
51 -          end;
52
53            %vyjimky
54 -          if p==0
55 -              prvocislo = 0;
56 -          end;
57 -          if p==1
58 -              prvocislo = 0;
59 -          end;
60 -          generace_prvoc = prvocislo;
61 -      end;
```

Obr. 6.1: Generace prvočísla *p* – zdrojový kód MATLAB

Na obr. 6.1 je uvedena část zdrojového kódu aplikace laboratorní úlohy, který zajišťuje správnou generaci prvočísla *p*. Proměnnou *max* je zajištěno uvedené omezení maximální velikosti generovaného prvočísla na hodnotu 100000, což dává předpoklad malé délky soukromého klíče. Postup generování prvočísla je rozdělen do tří hlavních částí.

Nejprve je pomocí funkce `rand` náhodně vygenerováno reálné číslo mezi 0 a 1. Vynásobením se zmíněnou proměnnou *max* a následným zaokrouhlením je získáno náhodné číslo mezi 0 a 100000.

Ve druhé části se vygenerované náhodné číslo podrobuje testu, který určuje zda se jedná o prvočíslo. Prvočíslo je přirozené číslo, které je beze zbytku dělitelné pouze dvěma odlišnými čísly, a to jedničkou a sebou samým. V poslední části se musí ošetřit dvě výjimky, protože prvočíslem nemůže být nula ani jednička.

Stejným způsobem je vygenerováno také prvočíslo *q*. Pomocí těchto generovaných prvočísel se určují další potřebné parametry algoritmu RSA. Prvočísla *p* a *q* musí zůstat utajeny před případným útočníkem.

6.2.2 Výpočet klíčů

Pro správnou funkci operací šifrování a dešifrování je potřeba výpočtem stanovit veřejný i soukromý klíč RSA algoritmu. Zdrojový kód výpočtu klíčů je zobrazen na obr. 6.2.

```
100         %výpočet modulu n
101 -       n = p*q;
102         %výpočet Eulerovy funkce
103 -       euler = (p-1)*(q-1);
104
105         %volba šifrovacího klíče e a dešifrovacího klíče d
106 -       fi = 2*euler + 1;
107 -       e = 0;
108 -       d = 0;
109 -       while d==0,
110 -           for e=fix(fi^.5-1):-1:3;
111 -               if (rem(fi,e)==0)
112 -                   d = fi/e;
113 -                   break;
114 -               end;
115 -               if (rem(e,100)==0)
116
117 -               end;
118 -           end;
119 -           fi = fi + euler;
120 -           if (d==0)
121
122 -           end;
123 -       end;
```

Obr. 6.2: Výpočet klíčů RSA – zdrojový kód MATLAB

Prvním veřejným parametrem je modul *n*, který je určen součinem generovaných prvočísel. Pro určení klíčů je potřeba vypočítat ještě Eulerovu funkci. V další části kódu je zajištěn výpočet veřejného klíče *e*. Na základě stanovení tohoto veřejného parametru je určen také soukromý klíč *d*. Nyní jsou definovány všechny potřebné parametry pro realizaci základních operací algoritmu RSA.

6.2.3 Operace šifrování

Pro realizaci jednoduchého časového útoku, který se soustřeďuje při dešifrování na doby trvání jednotlivých operací se soukromým klíčem, je nutné implementovat také určitou metodou operaci šifrování zvolené zprávy.

```
152 -        for i = 1:length(m)
153 
154 -              result = 1;
155 -              for j = 1:e
156 -                  result = rem(result*m(i), n);
157 -              end;
158 -            c(i) = result;
159 -        end;
```

Obr. 6.3: Operace šifrování RSA – zdrojový kód MATLAB

Na uvedeném obrázku 6.3 je znázorněn zdrojový kód, který zajišťuje operaci šifrování vstupní zprávy *m* pomocí určeného veřejného klíče *e*. Výsledkem je šifrovaný text zprávy označovaný jako kryptogram *c*.

6.3 Demonstrativní časový útok

Demonstrativní časový útok na RSA je jedním z hlavních bodů laboratorní úlohy, který má studentům pomoci porozumět principu získání senzitivní informace při časovém útoku. V praxi tímto jednoduchým způsobem časový útok být veden nemůže, protože útočník nemá možnost měřit dobu výpočtu s jednotlivými bity, ale pouze je schopen získat informaci o celkovém čase dešifrování zprávy.

6.3.1 Operace dešifrování

Pro možnost vedení útoku časovým postranním kanálem je velmi podstatné, aby implementace algoritmu RSA využívala k dešifrování urychlovací Montgomeryho metodu násobení. Tato metoda je založena na algoritmu *square and multiply*, jehož obecná podoba je již zobrazena na obrázku 2.3. Časový útok lze uskutečnit pouze na tento typ implementací algoritmu RSA.

Na následujícím obrázku 6.4 je uveden zdrojový kód operace dešifrování s využitím zmíněného algoritmu *square and multiply*. Tento algoritmus rozděluje zpracování kryptogramu do dvou rozdílných větví podle toho, zda má soukromý klíč hodnotu konkrétního bitu rovnou jedné nebo nule. Na této skutečnosti je založena podstata časového útoku.

```
177          for j = 1:pocet            %pocet meření
178              z = c;
179              for i = 1:length(d_bin)
180                   z = rem((z^2),n);
181                   tic
182                   if d_bin(i) == y
183                            z= rem(z*c,n);
184                   end;
185              p(i)= toc;
186              t(i)= t(i) + p(i);
187              end;
188          end;
189
190          for i=1:length(d_bin)
191              t(i)=t(i)/pocet;
192          end;
```

Obr. 6.4: Operace dešifrování a měření času – zdrojový kód MATLAB

I přesto, že se jedná o demonstrativní a zjednodušený časový útok je potřeba provést více nezávislých měření, aby bylo dosaženo kvalitního výsledku tohoto útoku. Z toho důvodu je měření času provedeno opakovaně a to podle uživatelem nastavené hodnoty počtu měření ve vytvořené aplikaci.

Důležitou částí pro vedení časového útoku na RSA je smyčka `if`, která je uvedena na 182. až 184. řádku zdrojového kódu (obr. 6.4). Právě čas průchodu touto smyčkou je přímo úměrný hodnotě konkrétního bitu soukromého klíče RSA.

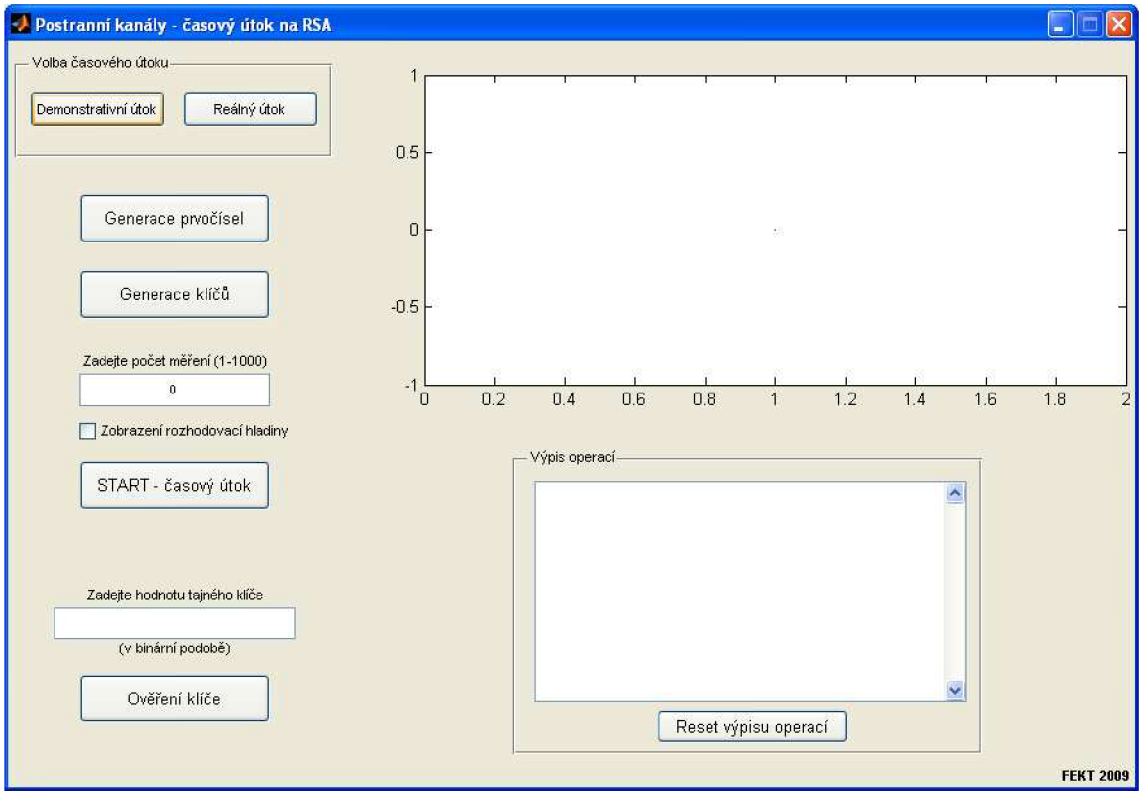
Pro měření času průchodu smyčkou jsou využito integrované funkce časovače (stopek). Funkci stopek zajišťují v MATLABu dva příkazy: `tic` a `toc`. Příkazem `tic` jsou stopky spuštěny a následným příkazem `toc` jsou stopky zastaveny a je změřen čas uplynutý od posledního použití příkazu `tic`.

Následně je vypočítána průměrná doba průchodu smyčkou `if` pro i-tý bit soukromého klíče. V případě, že i-tý bit soukromého klíče má nulovou hodnotu se pouze provede kontrola podmínky, ale běh programu už nevstoupí do těla smyčky. V opačném případě, kdy je hodnota i-tého bitu klíče rovna jedné, je podmínka splněna a provede se výpočet v těle dané smyčky. Z toho plyne, že doba trvání průchodu smyčkou pro bit klíče s hodnotou nuly bude o něco kratší než v situaci s bitem opačné hodnoty.

6.3.2 Vzhled a popis aplikace

Aplikace laboratorní úlohy by měla studentům svým grafickým rozhraním zajistit snadné interaktivní ovládání a vysokou přehlednost. Celá aplikace je rozdělena do dvou hlavních částí. První část je zaměřena na pochopení principu útoku časovým postranním kanálem, jehož detailním rozborem se zabývají kapitoly 6.2 a 6.3. Druhá část aplikace se zaměřuje na reálné postupy při operaci šifrování zprávy a při vedení časového útoku na kryptografický modul RSA.

Na následujícím obrázku 6.5 je znázorněno grafické uživatelské rozhraní, jehož podoba odpovídá stavu ihned po spuštění aplikace. V levé horní části je uživateli umožněno přepínání mezi demonstrativním a reálným časovým útokem. Po spuštění aplikace je automaticky již zobrazeno grafické rozhraní demonstrativního časového útoku.



Obr. 6.5: Grafické uživatelské rozhraní části aplikace – demonstrativní útok

Pro uskutečnění zjednodušeného typu časového útoku je potřeba generovat všechny parametry RSA algoritmu. K tomu účelu slouží tlačítka „Generace prvočísel“ a „Generace klíčů“.

Studentům je umožněno zvolit počet opakování časových útoků ve stanoveném rozmezí. Výsledné časové hodnoty útoku jsou potom dány průměrem ze všech uskutečněných měření.

Další možností je zapnutí pomocné rozhodovací hladiny, která může pomoci při stanovení hodnot bitů soukromého klíče. Rozhodovací hladina není dána jen prostým aritmetickým průměrem, ale potlačuje nepřesnosti způsobené nestálým výkonem při běhu programu. Je stanovena na základě hodnoty, která se v měřeném souboru vyskytuje nejčastěji, tj. hodnota s největší relativní četností.

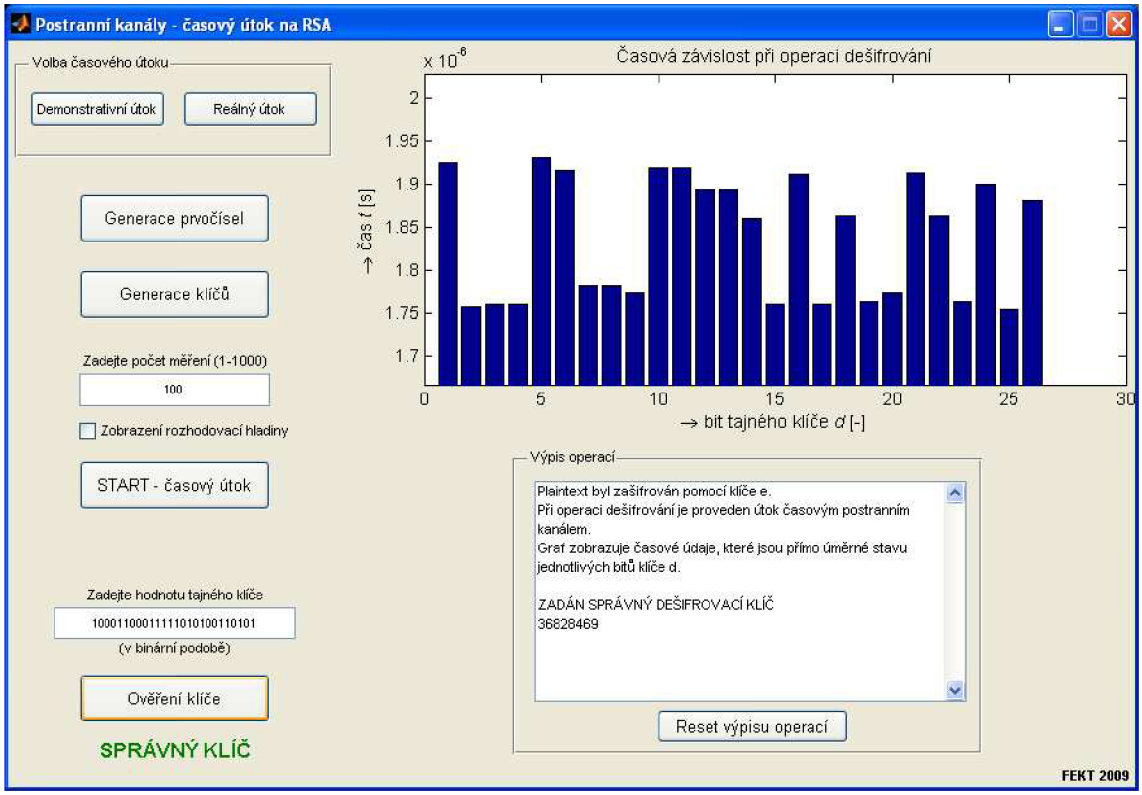
Samotný časový útok je realizován po stisku tlačítka „START – časový útok“. Reakcí je šifrování pevně zadané zprávy a následné dešifrování pomocí výše uvedeného algoritmu (obr. 6.4). Výsledkem časového útoku je přehledný sloupcový graf znázorňující doby průchodu smyčkou algoritmu *square and multiply* pro jednotlivé bity soukromého klíče.

Na základě graficky uvedených výsledků časového útoku si studenti mají možnost ověřit správnost a použitelnost tohoto typu útoku postranními kanály na implementovaný algoritmus RSA. Ověření je realizováno zadáním předpokládané binární hodnoty soukromého klíče do editačního pole aplikace.

Všechny důležité činnosti i hodnoty parametrů jsou souhrnně a přehledně zobrazeny v poli s názvem „výpis operací“.

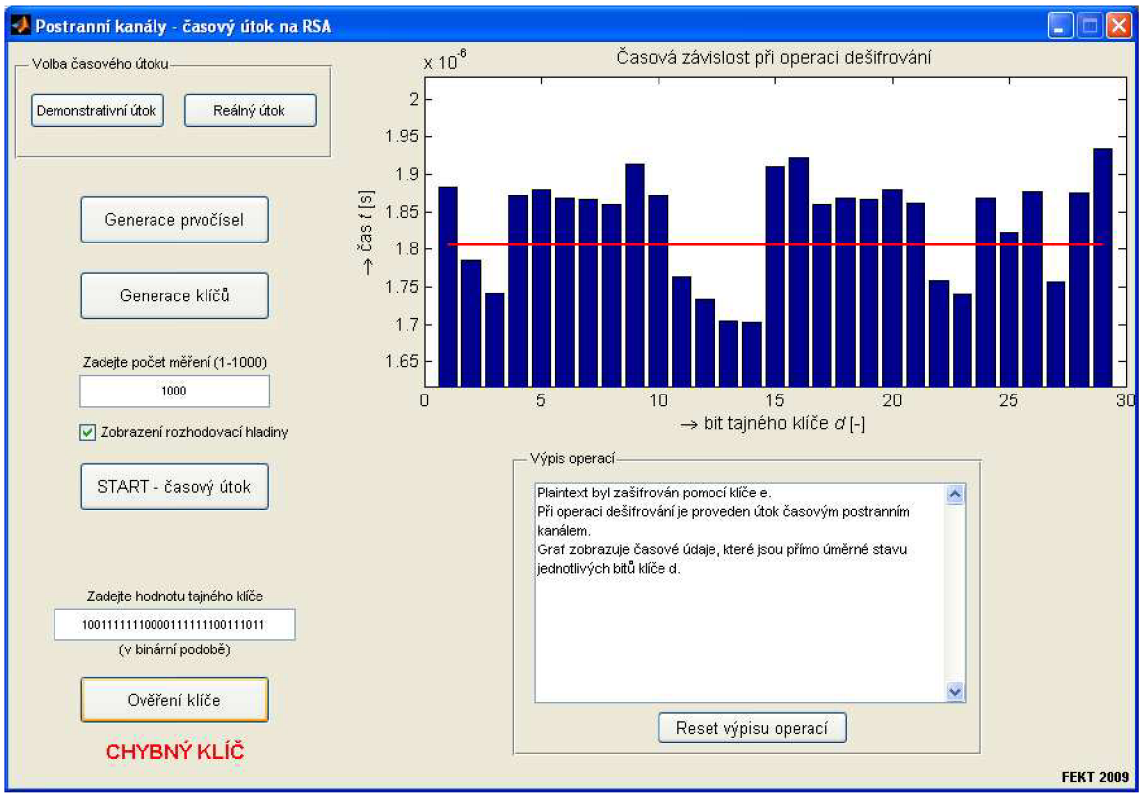
6.3.3 Dosažené výsledky

V této kapitole jsou zhodnoceny výsledky demonstrativního časového útoku na RSA.



Obr. 6.6: Časový útok na náhodně vygenerovaný soukromý klíč RSA – 100 měření

Na obrázku 6.6 jsou znázorněny výsledky časového útoku na implementaci RSA s vygenerovanými parametry. Počet měření je nastaven na stonásobné opakování útoku na soukromý klíč. Z obr. 6.6 je zřetelné, že z časové závislosti při operaci dešifrování lze snadným způsobem rozpoznat hodnotu jednotlivých bitů klíče. Po zapsání binární hodnoty klíče a následném ověření je uživatel ihned informován textovou indikací o správnosti předpokládané hodnoty. Při zadání správného soukromého klíče se v poli „výpis operací“ objeví také jeho dekadická hodnota.



Obr. 6.7: Časový útok na vygenerovaný soukromý klíč – rozhodovací hladina

Druhý příklad demonstrativního časového útoku je zobrazen na obr. 6.7. Útok je uskutečněn na nový vygenerovaný soukromý klíč s délkou 29 bitů. Měření je nastaveno na tisíc opakovaných útoků. Pro větší přehlednost je vyžádáno zobrazení červené rozhodovací hladiny. Obrázek 6.7 také ukazuje na 25. bitu klíče patrnou nepřesnost, která nastává i při měření zjednodušeného časového útoku a je způsobena nestálým a proměnlivým výkonem nebo chybou měřící funkce při běhu aplikace. Skutečná hodnota 25. bitu soukromého klíče je nula. Tento fakt ovšem neodpovídá naměřené a zobrazené časové hodnotě.

V případě chybného nebo nepřesného výsledku časového útoku je možné opakovaným stiskem tlačítka „START“ získat nový vzorek a odhalit tak pozici chybného bitu.

6.4 Operace šifrování RSA

Další částí laboratorní úlohy podle návrhu struktury na obr. 5.1 je šifrování RSA v praxi. Cílem tohoto tématu laboratorní úlohy je ukázat studentům postup, jakým je realizována operace šifrování obecné zprávy při reálném použití implementace algoritmu RSA.

6.4.1 Princip

Při reálném nasazení algoritmu RSA lze postup šifrování obecné zprávy rozdělit do několika jednotlivých úkonů.

Pro správnou činnost operace šifrování je nutná transformace všech znaků zprávy na číslo. Každý znak zadané zprávy je možné reprezentovat pomocí ASCII kódu. V původní sedmibitové definici ASCII tabulka obsahuje 128 platných znaků. Například zpráva „RSA“ je pomocí ASCII tabulky zakódována jako: `82 83 65`.

Poté je každé ASCII číslo převedeno do osmibitové binární podoby. V uvedeném příkladě šifrování zprávy „RSA“ se jedná o převedení následné kombinace `82 83 65` do binární podoby: `01010010 01010011 01000001`.

Dalším úkonem před samotným šifrováním zprávy je spojení získaných osmibitových slov do binárních bloků. V aplikaci k laboratorní úloze jsou pro jednoduchost spojeny každé dva znaky zprávy, tj. uskuteční se spojení dvou osmibitových slov do jednoho šestnáctibitového bloku. V případě lichého počtu znaků šifrované zprávy je poslední převod doplněn nulami do šestnáctibitového bloku. Reálné aplikace algoritmu RSA využívají spojení do bloků, které reprezentují 8 nebo 16 znaků zprávy. Délka těchto používaných bloků je potom 64 nebo 128 bitů. V případě zprávy „RSA“ se jedná o spojení do následujících bloků: `0101001001010011 0000000001000001`.

Každý blok zprávy vytvořený popsaným způsobem se převede zpět do dekadické podoby, ve které bude blok zprávy šifrován. Převedením šestnáctibitových bloků zprávy zpět do dekadické číselné soustavy získáme: 21075 65.

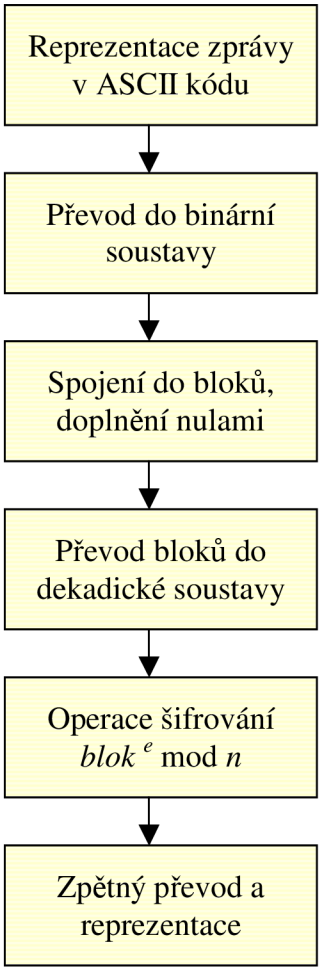
Každé získané číslo bude z definice algoritmu RSA šifrováno níže uvedeným způsobem:

$$blok^{e} \bmod n\,, \tag{6.1}$$

kde e je veřejný klíč a n je modul. Pomocí vztahu (6.1) jsou získány jednotlivé šifrované bloky.

Tyto zašifrované bloky se mohou zpětně převést do binární číselné soustavy, rozdělit do osmibitových částí, které reprezentují v ASCII kódu určité znaky a výsledkem je konečná podoba šifrované zprávy.

Celý princip činností, které jsou prováděny v souvislosti s operací šifrování, je přehledně blokově znázorněn na obrázku 6.8.



Obr. 6.8: Blokové schéma reálného postupu při operaci šifrování zprávy

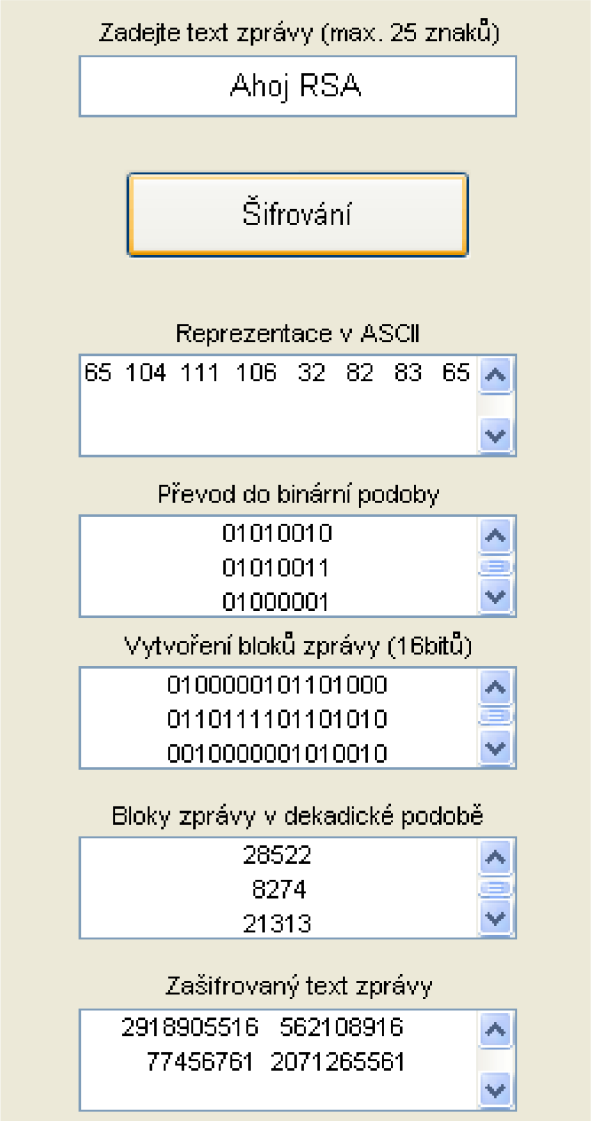
Záměrem této části laboratorní úlohy je osvětlit studentům princip, podle kterého pracuje v praktickém použití RSA operace šifrování. Pro ukázku šifrování zadané zprávy jsou použity tyto náhodně vygenerované parametry algoritmu:

```
p=27737,
q=111827,
n=3101745499,
d=13739370572477,
e=179.
```

V současných implementacích RSA se provádí před samotným šifrováním kromě uvedeného formátování také kódování zprávy, aby nebylo možné vést útok frekvenční analýzou. Některé formáty standardu RSA využívají hašovacích funkcí, které umožňují vytvořit krátké reprezentanty dané zprávy. Tyto metody výrazně zvyšují bezpečnost algoritmu RSA na základě maskování zprávy.

6.4.2 Vzhled a popis aplikace

Celá aplikace k laboratorní úloze je graficky členěna do dvou částí. První část je podrobně popsána především v kapitole 6.3. Obsahem druhé části aplikace je kromě reálného časového útoku také ukázka formátování zprávy při skutečném použití operace šifrování.



Obr. 6.9: Postup při reálném šifrování zprávy – část aplikace

Tato část laboratorní úlohy umožňuje studentům zadat vlastní zprávu určenou k šifrování. Zadání zprávy je podmíněno omezením na maximální délku 25 znaků.

Na obr. 6.9 je zobrazen příklad šifrování zprávy pomocí algoritmu RSA. Uvedený postup odpovídá teoretickým předpokladům, které jsou znázorněny na obr. 6.8.

6.5 Reálný časový útok

Záměrem této části laboratorní úlohy je alespoň do určité míry přiblížit studentům metody, kterými jsou realizovány skutečné útoky s využitím časových postranních kanálů.

6.5.1 Princip

Při vedení útoku pomocí časových postranních kanálů na fyzickou implementaci algoritmu RSA je hlavním cílem získat určitou senzitivní informaci. Zmíněnou citlivou informací je při reálném časovém útoku hodnota Hammingovy váhy klíče. Hammingova váha soukromého klíče je dána počtem bitů klíče, které mají hodnotu rovnou jedné. V rámci reálného časového útoku již není možnost získat dobu trvání dešifrování jednotlivého bitu klíče. Jedinou relevantní informací, kterou je útočník schopen využít je celkový čas dešifrování zprávy pomocí soukromého klíče. Útočník má také ve většině případů přibližnou informaci o době trvání dešifrování bitem klíče s hodnotou rovnou jedné. Informace jsou vztaženy vždy pro konkrétní kryptografický modul.

Aplikace laboratorní úlohy by měla z principu umožňovat instalaci a správnou činnost na více různých stanicích. Z toho důvodu není možné, aby byly potřebné informace o rychlosti a časech dané implementace určitým způsobem sjednoceny, protože každá stanice poskytuje vytvořené aplikaci jinou úroveň i stabilitu výkonu. Z uvažovaných poznatků je nutné před realizací časového útoku provést inicializaci, která stanovuje průměrnou dobu při dešifrování bitem s hodnotou rovnou nule a jedné. Před inicializací je potřeba provést výběr délky soukromého klíče, na který bude časový útok veden. Aplikace dává možnost volby ze dvou alternativ délky klíče, tj. 128 bitů a 256 bitů. Pro účely samotné laboratorní úlohy je nejvhodnější volbou soukromý klíč s délkou 128 bitů. Cílem inicializační funkce je stanovit průměrnou i nejčetnější dobu dešifrování pro bity klíče s hodnotou rovnou jedné a také pro bity s hodnotu rovnou nule. Z každých čtyřiceti měření jsou stanoveny uvedené parametry pro jednotlivé bity klíče. Tento postup je realizován opakovaně pro 3000 pokusů, ze kterých jsou průměrem určeny potřebné informace o dané implementaci algoritmu RSA.

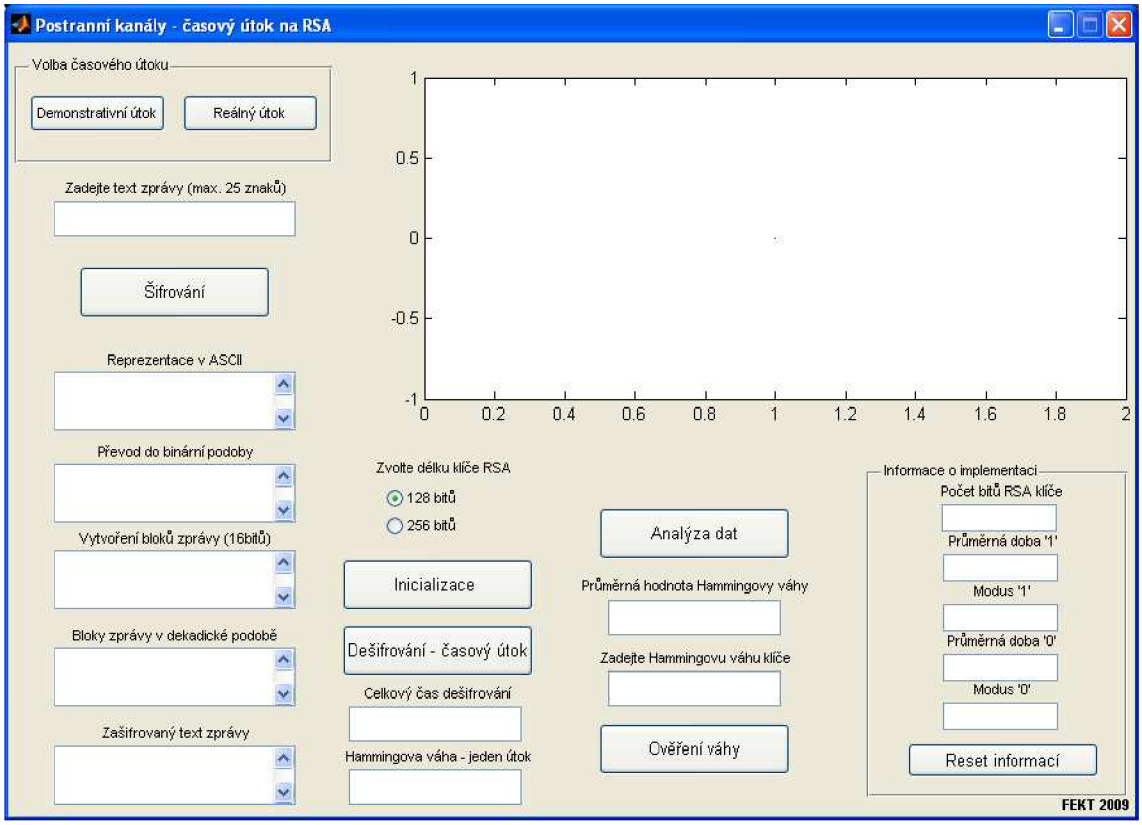
Nejpřesnějších výsledků je dosaženo v případě, kdy není inicializační funkce rušena jinými aplikacemi a v ideálním případě uživatel nepohybuje při běhu funkce s kurzorem myši.

Je patrné, že reálný časový útok postranním kanálem je veden měřením celkového času potřebného pro dešifrování kryptogramu zprávy. Měření je realizováno téměř shodným způsobem jako je uvedeno při demonstrativním časovém útoku (obr. 6.4). I při reálném útoku je měření celkového času opakováno pro 3000 pokusů, což je stanovený kompromis mezi rychlostí a kvalitou dosažených výsledků. Z každého změřeného celkového času při operaci dešifrování je pomocí inicializací stanovených průměrných hodnot dosaženo nejpravděpodobnější hodnoty Hammingovy váhy soukromého klíče. Principem určení počtu bitů klíče s hodnotou rovnou jedné je postupné porovnávání změřeného celkového času s časovými informacemi, které jsou definovány inicializací. Aplikace výpočtem stanovuje teoretický celkový čas, který odpovídá Hammingově váze rovné polovině celkové délky klíče. Následně tento údaj porovnává s naměřeným skutečným časem a snaží se zmíněný teoretický čas upravit tak, aby se co nejvíce blížil naměřené době dešifrování zprávy. Úprava spočívá v postupné změně Hammingovy váhy způsobené odečítání nebo přičítání průměrných inicializačních hodnot. Tímto postupem aplikace stanový 3000 možných výsledků hodnoty Hammingovy váhy. Nakonec určí četnost jednotlivých hodnot Hammingovy váhy a provede výpočet pravděpodobnosti jejich výskytu. Skutečná Hammingova váha soukromého klíče by se měla rovnat hodnotě s nejvyšší pravděpodobností.

6.5.2 Vzhled a popis aplikace

Při volbě reálného útoku se aplikace přepne do zobrazeného rozhraní uvedeného na obrázku 6.10. Pro časový útok na modul algoritmu RSA je nejdříve nezbytné zvolit délku klíče, na který bude aplikace vést útok. Dalším krokem je spuštění inicializační funkce, při které je nutné na dané stanici nevyvíjet žádnou další činnost. Konec inicializace je signalizováno naplněním zpočátku prázdných polí v části s názvem „informace o implementaci“. Tlačítko „dešifrování – časový útok“ slouží k ukázce principu tohoto typu útoku změřením celkového času dešifrování zprávy. Jedná se pouze o ukázku jediného měření, které nemá potřebnou statistickou váhu a proto nemůže pravděpodobně vést ke správné hodnotě Hammingovy váhy. Přesto je výpočet hodnoty Hammingovy váhy z tohoto jediného samostatného útoku uveden v příslušném poli aplikace. Hlavním krokem reálného útoku je spuštění analýzy dat, která uskuteční opakovaně 3000 útoků a z každého provede výpočet Hammingovy váhy. Pro dosažení korektních výsledků je nutné opět dodržet nečinnost kurzoru myši. Po ukončení časového útoku aplikace se zobrazí pravděpodobnost Hammingovy váhy klíče a také

průměrná hodnota této veličiny. Studentům je umožněno provést ověření správnosti určení hodnoty Hammingovy váhy soukromého klíče.

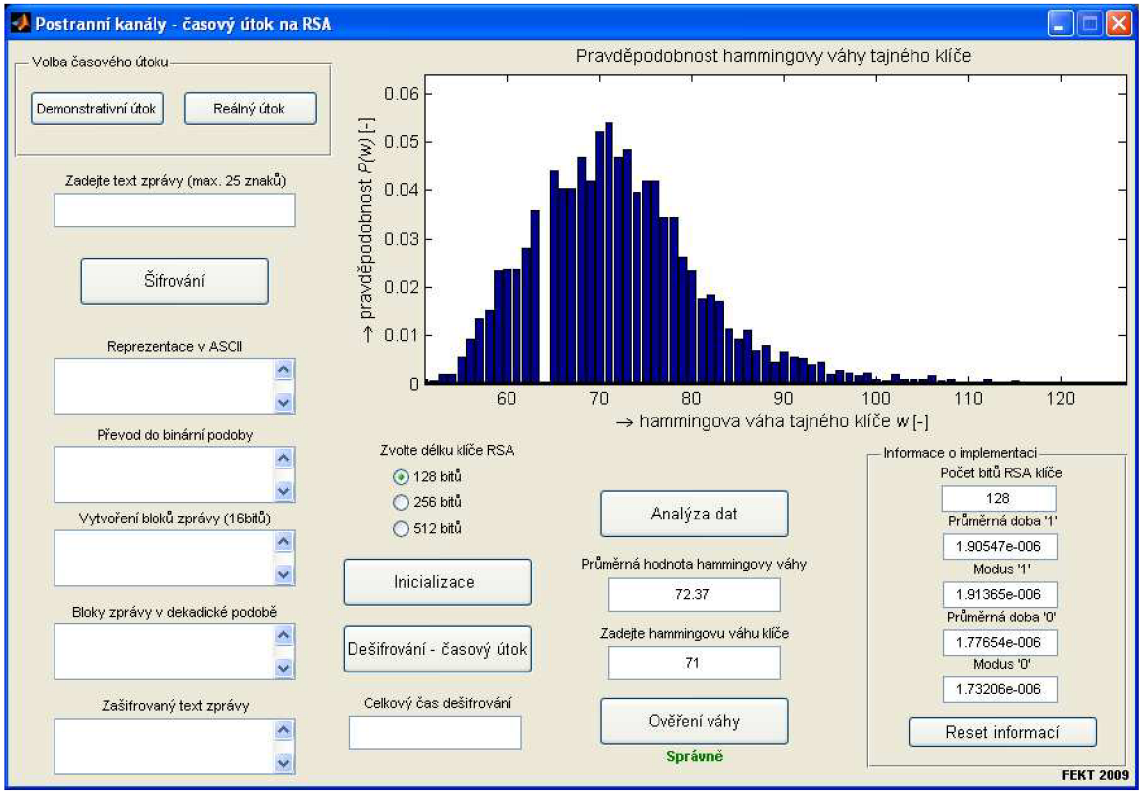


Obr. 6.10: Grafické uživatelské rozhraní části aplikace – reálný útok

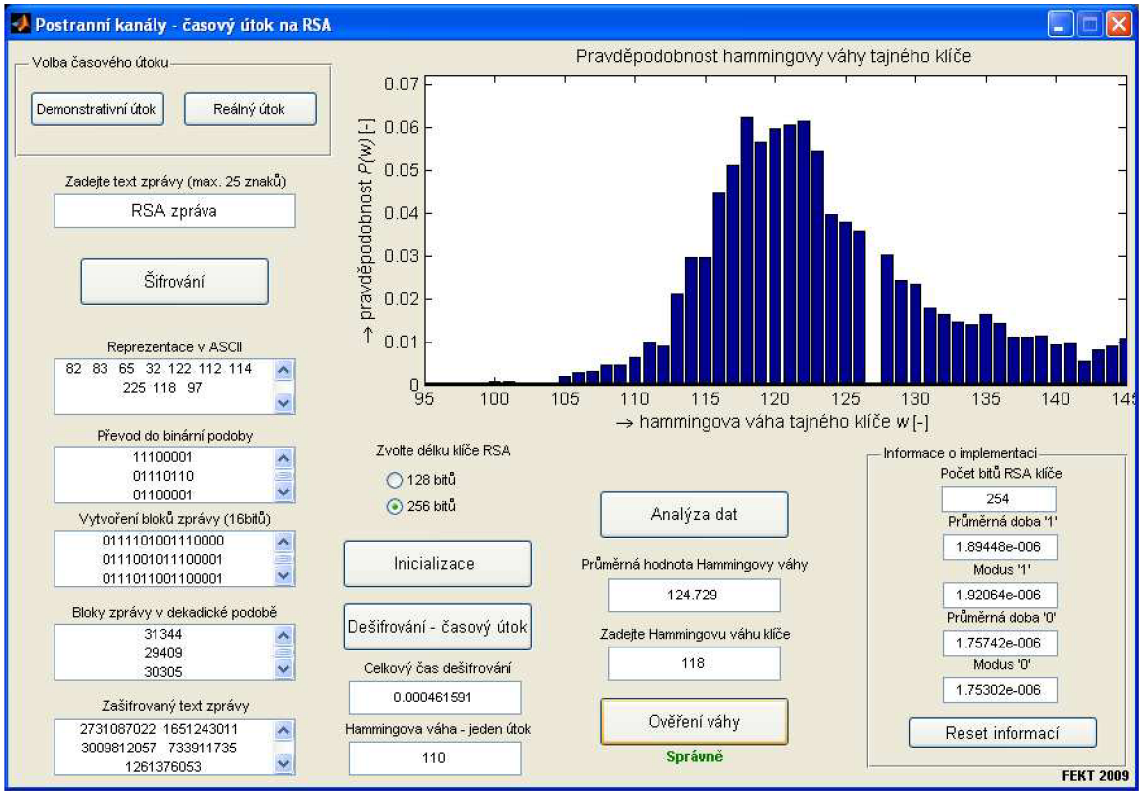
6.5.3 Dosažené výsledky

V této kapitole budou zobrazeny dosažené výsledky časových útoků pomocí aplikace laboratorní úlohy.

Na obrázku 6.11 je zachycen úspěšný pokus útoku pomocí časového postranního kanálu na soukromý klíč o délce 128 bitů. Inicializací jsou získány potřebné informace o konkrétním kryptografickém modulu RSA. Podle rozložení pravděpodobnosti hodnot Hammingovy váhy je zřejmé, že tento soukromý klíč obsahuje více bitů s hodnotou jedné než bitů nulové hodnoty. Rozhodujícím kritériem pro výběr a stanovení Hammingovy váhy by měla být hodnota s maximální četností ze všech časových útoků. Jedná se o hodnotu Hammingovy váhy, jejíž výskyt při realizaci časového útoku je nejvíce pravděpodobný. V případě útoku na obr. 6.11 je nejvyšší pravděpodobnost přidělena Hammingově váze s hodnotou 71. Při určení počtu bitů klíče rovných jedné je vedlejším kritériem také průměrná hodnota Hammingovy váhy. V tomto případě je skutečný počet jedničkových bitů soukromého klíče roven právě hodnotě 71, která byla nejčastější hodnotou při realizaci série časových útoků.



Obr. 6.11: Reálný časový útok na 128 bitový soukromý klíč RSA – 3000 měření



Obr. 6.12: Reálný časový útok na 256 bitový soukromý klíč RSA – 3000 měření

Další časový útok vedený na soukromý klíč RSA s délkou 256 bitů je zobrazen na obr. 6.12. Z výsledků časového útoku je snadno rozpoznatelné, že v tomto případě je počet jedničkových bitů klíče naopak menší než počet nulových bitů. Pravdivá Hammingova váha klíče se i v tomto případě rovná hodnotě, která měla při časovém útoku nejvyšší četnost.

6.5.4 Vlastnosti implementace

Útoky postranními kanály se vyznačují především vysokou efektivitou jejich snahy prolomit daný kryptografický modul. Vlastnosti útoku časovým kanálem lze stanovit pomocí výpočtu možných kombinací při získání hodnoty Hammingovy váhy klíče. K tomuto účelu lze využít vztahů pro výpočet variací a kombinací.

Počet variací *k*-té třídy z *n* prvků s opakováním, tzn. každý prvek se ve výběru může objevit vícekrát, je určen vztahem

$$V'_k(n) = n^k \ . \tag{6.2}$$

Kombinace *k*-té třídy z *n* prvků je skupina *k* prvků vybraných z celkového počtu *n* prvků a je definována vztahem

$$C_k(n) = \binom{n}{k} = \frac{n!}{k!(n-k)!} \ . \tag{6.3}$$

Pomocí vztahu (6.2) je možné určit počet všech možných klíčů o *k* bitech. S využitím vztahu (6.3) lze stanovit počet možných klíčů při znalosti hodnoty *k*, která udává Hammingovu váhu soukromého klíče.

Hlavní podnět vedoucí k vznikajícím nepřesnostem při realizaci reálného časového útoku je již zmíněná nestálost a úroveň výkonu při měření vykonávaných operací. Podmínkou správné činnosti části aplikace týkající se reálného útoku je dostatečně výkonná stanice, na které je aplikace spuštěna. Aplikace byla testována na několika různých stanicích. Všechny zobrazené výsledky reálného časového útoku jsou uskutečněny na stanici s technickými parametry: AMD Athlon 64 X2 Dual Core 5000+, 2,6 GHz, 2,0 GB.

Nutnou podmínkou pro kvalitní výsledky měřených časů je také zaručení nečinnosti kurzoru myši a běhu dalších paměťově náročných aplikací. Tato podmínka musí být splněna při činnosti inicializační funkce i při analýze dat, která realizuje sérii časových útoků. Důležité je zaručení nejlépe totožných podmínek při inicializaci i analýze dat.

Výrazný vliv na výsledky reálného útoku má také měřící časová funkce prostředí MATLAB, která může i při zajištění shodných podmínek stanovit měřením různé časové intervaly.

Z uvedených důvodů nelze plně zaručit, že hodnota Hammingovy váhy s nejvyšší pravděpodobností se vždy bude rovnat skutečnému počtu nenulových bitů soukromého klíče. Přesnost dosažených výsledků se zvyšuje s rostoucím počtem uskutečněných časových útoků. Ovšem je nutné učinit kompromis mezi přesností výsledků a dobou potřebnou k jejich dosažení. Při realizaci časového útoku je provedeno 3000 samostatných pokusů, což odpovídá časovému intervalu v řádu desítek sekund, které jsou potřebné jednak pro inicializaci, ale i pro samotnou analýzu dat. Při vývoji aplikace bylo testováno několik verzí, které se lišily v počtu opakovaných pokusů. Byly vyzkoušeny reálné útoky s následujícími počty opakovaných pokusů: 20000, 10000, 5000, 3000, 1000, 80. Jako optimální kompromis mezi rychlostí měření a kvalitou výsledků bylo stanoveno měření s třemi tisíci časovými útoky.

6.6 Programování GUI

Posledním dosud nepopsaným bodem návrhu laboratorní úlohy je programování grafického uživatelského rozhraní a operace dešifrování pro realizaci časového útoku. V této části laboratorní úlohy si mohou studenti vyzkoušet programování s využitím systému *handle graphics*, který umožňuje tvorbu interaktivního rozhraní v MATLABu.

Hlavní úkol je naprogramování potřebných částí zdrojového kódu pro správnou funkci demonstrativního časového útoku. Cílem bude korektně doplnit přiložené soubory, tj. m-soubor a fig-soubor.

Doplnění fig-souboru obsahujícího již předpřipravené rozhraní GUI spočívá v přidání takových prvků, které umožní uživateli snadným způsobem zvolit počet opakování časového útoku.

V přiloženém m-souboru je úkolem studentů vytvořit kód, který bude zajišťovat zadaný počet časových útoků při operaci dešifrování. Pro správnou funkci je nutné, aby studenti při programování dodrželi doporučené značení důležitých proměnných.

Na obrázku 6.13 je zobrazena část kódu přiloženého m-souboru, který má být doplněn o samotnou realizaci časového útoku. Pro usnadnění je studentům zobrazena nápověda podoby potřebných částí kódu. Hlavním úkolem je vytvořit algoritmus *square and multiply* a pomocí časových měřících funkcí MATLABu uskutečnit časový útok. Nakonec podle počtu zadaných měření získat a zobrazit průměrné časové hodnoty pro jednotlivé bity soukromého klíče.

```
154
155  % ************************************************************************
156
157  %------DOPLŇTE KÓD PRO REALIZACI ČASOVÉHO ÚTOKU - OPERACE DEŠIFROVÁNÍ---------
158
159
160      % binární podoba klíče d označte d_bin
161
162      % pomocné proměnné
163
164      % naplnění časového pole t nulami
165
166      % načtení počtu měření
167
168      % dešifrování pomocí algoritmu square and multiply
169
170      % měření času pomocí funkce stopek
171
172      % získání průměrných časových hodnot pro jednotlivé bity klíče
173
174
175  %----------------------KONEC REALIZACE ČASOVÉHO ÚTOKU-------------------------
176
177  % ************************************************************************
178
```

Obr. 6.13: Zadání úkolů – programování operace dešifrování

Na níže uvedeném obrázku 6.14 je příklad možného řešení zadaného úkolu.

```
155  % ************************************************************************
156  %------DOPLŇTE KÓD PRO REALIZACI ČASOVÉHO ÚTOKU - OPERACE DEŠIFROVÁNÍ---------
157 -      d_bin = dec2bin(d); % binární podoba klíče d
158 -      y = num2str(1);  %pomocná proměnná
159 -      z = c;
160 -      for i=1:length(d_bin) % naplnění časového pole nulami
161 -          t(i)=0;
162 -      end;
163
164 -      pocet = str2num(get(findobj('Tag','edit1'),'String')); %načtení počtu měření
165 -      if (pocet == 0) || (pocet > 1000)
166 -          set(findobj('Tag','pozor'),'Visible','on');
167 -      else
168 -          set(findobj('Tag','pozor'),'Visible','off');
169
170 -          for j = 1:pocet     %pocet meření
171 -              for i = 1:length(d_bin)
172 -                    z = rem((z^2),n);
173 -                    tic
174 -                    if d_bin(i) == y
175 -                            z= rem(z*c,n);
176 -                    end;
177 -              p(i)= toc;
178 -              t(i)= t(i) + p(i);
179 -              end;
180 -          end;
181
182 -          for i=1:length(d_bin)
183 -              t(i)=t(i)/pocet;
184 -          end;
185  %----------------------KONEC REALIZACE ČASOVÉHO ÚTOKU-------------------------
186  % ************************************************************************
```

Obr. 6.14: Možný způsob řešení zadaného úkolu

Součástí zadání laboratorní úlohy zaměřené na útoky časovým postraním kanálem bude také obecná ukázka potřebného algoritmu *square and multiply*. V zadání úlohy budou uvedeny všechny potřebné informace a funkce, které jsou potřeba pro splnění zadaných úkolů.

Absolvování této části laboratorní úlohy pomůže hlouběji pochopit princip nasazení urychlujícího algoritmu do kryptografických modulů RSA a možnost vedení útoků postranním kanálem. Dalším záměrem je poukázat na možnosti interaktivních grafických aplikací prostředí MATLAB.

Podmínkou realizace této části laboratorní úlohy je vybavení stanice integrovaným prostředím MATLAB. Základem pro řešení této části úlohy jsou přiložené dva soubory `casovy.m` a `casovy.fig`, které jsou uloženy na přiloženém paměťovém médiu této diplomové práce.

6.7 Instalace aplikace

Podmínkou pro realizaci navržené laboratorní úlohy je vybavenost konkrétní stanice vývojovým prostředím MATLAB. Aplikace k laboratorní úloze byla vyvinuta v prostředí programu MATLAB verze 7.1 (R14) Service pack 3. Pro úspěšné spuštění aplikace souborem `laboratorni_uloha.exe` je nutné na každé stanici provést instalaci kompileru MATLAB Compiler Runtime. Tuto instalaci lze uskutečnit pomocí souboru `MCRInstaller.exe`, který je také součástí přílohy na DVD.

7 ZADÁNÍ LABORATORNÍ ÚLOHY

ÚTOKY POSTRANNÍMI KANÁLY NA RSA – *laboratorní úloha*

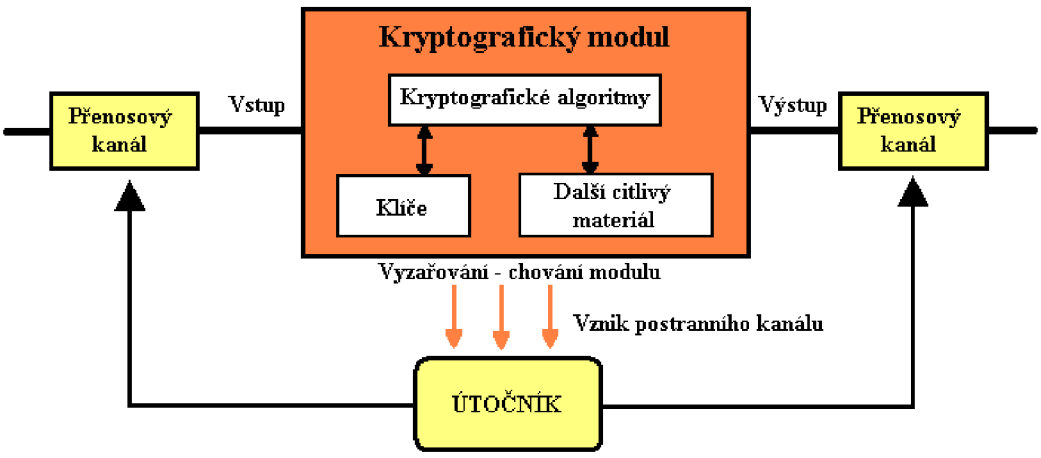
CÍLE

Hlavním cílem laboratorní úlohy je ukázat na nové možnosti kryptoanalýzy, které nabízí existence postranních kanálů v jednotlivých kryptografických modulech. Absolvováním laboratorní úlohy získají studenti nové poznatky i praktické zkušenosti v následujících oblastech.

- **Postranní kanály v kryptografii**
  - Definice
  - Klasifikace
  - Nový způsob útoků
- **Útok časovým postranním kanálem**
  - Princip
  - Útok v praxi
- **RSA algoritmus**
  - Princip
  - Operace šifrování v praxi
- **MATLAB**
  - Programování, algoritmus *square and multiply*
  - Tvorba GUI (grafické uživatelské rozhraní)

TEORETICKÝ ÚVOD

Dosavadní způsob útoků se soustředil přímo na objevení slabiny v matematické podstatě kryptografických algoritmů a protokolů. Ještě před několika lety nepředpokládali odborníci na kryptoanalýzu jiný možný způsob útoku na kryptografický modul.



Obr. 1: Nový typ útoku na kryptografický modul s využitím postranních kanálů

Kryptografický modul

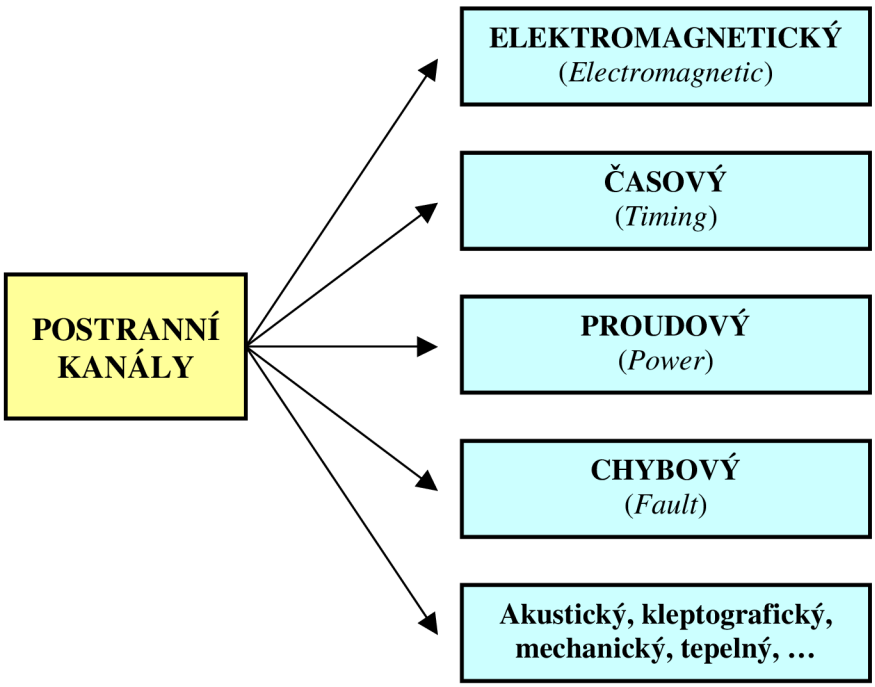
Kryptografický modul je v podstatě fyzickou implementací konkrétního kryptografického algoritmu (popř. kryptografického protokolu). Tento modul představuje zařízení, které bývá realizováno do hardwarové nebo softwarové podoby. Uvnitř modulu probíhají všechny procesy a citlivé úkony, které jsou spojené s šifrováním, dešifrováním, ověřením, podepisováním, autentizací, apod.

Reálný kryptografický modul při své činnosti komunikuje se svým okolím a může vyzařovat různé informace o svých operacích. Každý reálný modul při své činnosti odebírá určitý proud, každá jeho operace způsobuje různé časové zpoždění, na konkrétní situace reaguje modul stavovými a chybovými hlášeními, klávesnice modulu může být mechanicky opotřebená nebo může vydávat různý akustický zvuk pro různé klávesy a podobně. Tyto projevy modulu jsou neodmyslitelně spojeny s činností jeho operací, a proto dochází k nechtěné komunikaci s okolím, při které mohou být prozrazeny některé ze senzitivních informací. Tento nežádoucí únik informací je nazýván postranním kanálem.

Postranní kanály (PK)

Postranní kanál označuje každý nežádoucí způsob výměny informací mezi okolím a kryptografickým modulem. Celá kryptografie nyní stojí před problémem, jak prakticky realizovat třeba i velmi kvalitní abstraktní model, aby ve vzniklém reálném modulu nebyly nežádoucí postranní kanály.

V současné době jsou považovány za hlavní druhy postranních kanálů, které lze využít s vysokou efektivitou při útoku na implementaci systému, především následující postranní kanály: elektromagnetický, časový, proudový (výkonový) a chybový kanál. Klasifikace postranních kanálů je přehledně znázorněna na následujícím obrázku.



Obr. 2: Klasifikace postranních kanálů

Algoritmus RSA

Algoritmus RSA (iniciály autorů – Rivest, Shamir, Adleman) byl vyvinut v roce 1977 a stále je hlavním představitelem asymetrických kryptosystémů. V současné době je tento asymetrický algoritmus považován za bezpečný při použití dostatečné délky klíče. Algoritmus RSA je vhodný jak k šifrování dat, tak i jejich digitálnímu podpisu. Hlavním problémem hardwarových i softwarových implementací algoritmu RSA je jejich nízká rychlost oproti srovnatelným symetrickým systémům. K urychlení prováděných operací se využívají matematické metody (Montgomeryho metoda, čínská věta o zbytcích,…).

Postup výpočtu potřebných parametrů RSA lze rozdělit do těchto šesti kroků:

1. Volba dvou náhodných velkých prvočísel p a q ($>10^{115}$),
2. Výpočet jejich součinu udává modul $n = pq$,
3. Výpočet hodnoty Eulerovy funkce $\varphi(n) = (p - 1)(q - 1)$,
4. Volba veřejného klíče e, musí být menší než $\varphi(n)$ a nesoudělný s $\varphi(n)$,
5. Výpočet soukromého klíče d z podmínky $(de) \bmod \varphi(n) = 1$,
6. Veřejné parametry jsou pouze klíč e a modul n.

Celý algoritmus RSA je přesně určen parametry: $p, q, n, \varphi(n), e, d$. Samotná operace šifrování je uskutečněna pomocí vztahu $c = m^e \bmod n$, kde c je zašifrovaný text a m je vstupní zpráva. Zpětná operace (tj. dešifrování) je dána vztahem $m = c^d \bmod n$.

Možné útoky PK na algoritmus RSA

Útok vedený pomocí postranního kanálu je založen na využití analýzy konkrétního kanálu k napadení daného kryptografického modulu., tj. získání nějaké citlivé informace. V současné době jsou známy tři hlavní druhy útoků využívající postranní kanály kryptografických modulů RSA.

- **Chybový útok**
  - Bleichenbacherův útok – 1998
  - Mangerův útok – 2001
- **Časový útok**
  - Kocherův útok – 1996

Z důvodů jednoduchosti a názornosti se tato laboratorní úloha věnuje útoku vedeného pomocí časového postranního kanálu, tj. Kocherovu útoku.

Časový postranní kanál je prvním publikovaným a typickým příkladem postranních kanálů. Tento postranní kanál vzniká v takových kryptografických modulech, kde rychlost průběhu operace podstatným způsobem závisí na vstupních datech a klíči. V případě mnoha kryptografických modulů je tato podmínka zajištěna použitím zmíněné urychlovací Montgomeryho metody, která je založena na algoritmu *square and multiply*. Nasazení tohoto algoritmu do implementací RSA je základní podmínkou pro realizaci útoku časovým postranním kanálem.

ZADÁNÍ ÚKOLŮ

1. Pomocí přiložené aplikace „Postranní kanály – časový útok na RSA“ realizujte jednoduchý časový útok na náhodně vygenerované parametry asymetrického algoritmu RSA. Pokuste se ze získaných výsledků stanovit přesnou podobu soukromého klíče. Prostudujte princip a možnosti tohoto typu útoku v aplikaci.

Princip

Časový útok na soukromý klíč algoritmu RSA využívá časově závislé operace modulární mocniny, která se využívá se při operaci dešifrování. Pro samotný výpočet modulární mocniny se používá algoritmus *square and multiply*, který je založen na postupném zpracování samostatných bitů soukromého klíče *d* a jeho obecná podoba je uvedena na obrázku 3.

```
1   R=m
2   for i=1 to b
3       {
4       R=(R*R)mod n
5       if d(i)= 1
6           R=(R*m)mod n
7       }
8   return R
```

Obr. 3: Obecná podoba algoritmu square and multiply

Kde *m* jsou vstupní data (tzn. ciphertext), *n* je modul a *d* soukromý klíč RSA. Podstatou časového útoku je skutečnost, že doba trvání průchodu smyčkou zobrazené na obr. 3 je závislá na tom, jestli je hodnota daného bitu soukromého klíče $d_{(x)}$ rovna jedné nebo nule. V situaci, kdy má tento konkrétní bit klíče hodnotu nuly vůbec neproběhne výpočet na 6.řádku uvedeného algoritmu. Z časového hlediska bude doba trvání průchodu smyčkou velmi malá. Při opačné hodnotě bitu klíče bude průchod celou smyčkou razantně pomalejší.

S využitím příkazů programu MATLAB pro časového měření je možné zjistit relativně přesnou dobu trvání průchodu smyčkou `if`.

Postup

Proveďte spuštění aplikace „Postranní kanály – časový útok na RSA“ přiloženým souborem `laboratorni_uloha.exe`. Aplikace je rozdělena na dvě hlavní grafická okna, které lze přepínat v levé horní části pomocí tlačítek. Tento úkol se zabývá demonstrativním časovým útokem. Před realizací časového útoku je nutné vygenerovat všechny potřebné parametry RSA algoritmu, zadat počet měření (opakování časového útoku) a určit zda je žádoucí zobrazení rozhodovací hladiny, která může pomoci při stanovení hodnot bitů soukromého klíče. Samotný časový útok je uskutečněn stiskem tlačítka „START – časový útok“. Výsledkem je časová závislost při operaci dešifrování pro jednotlivé bity klíče. Výsledné časové hodnoty útoku jsou potom dány průměrem ze všech uskutečněných měření. Ze získaných informací lze stanovit přesnou podobu soukromého klíče RSA. Všechny důležité činnosti i hodnoty parametrů jsou souhrnně zobrazeny v poli s názvem „výpis operací“. V případě chybného nebo nepřesného výsledku časového útoku je možné opakovaným stiskem tlačítka „START“ získat nový vzorek a odhalit tak pozici chybného bitu.

2. Naprogramujte jednoduchou grafickou aplikaci pomocí prostředí MATLAB (GUI), která bude realizovat podobným způsobem časový útok jako hlavní aplikace „Postranní kanály – časový útok na RSA“. Zajistěte, aby si uživatel mohl snadným způsobem volit počet měření.

Postup

V této části laboratorní úlohy se budete zabývat programováním s využitím systému *handle graphics*, který umožňuje v MATLABu tvorbu interaktivního rozhraní. Vaším úkolem bude korektně doplnit potřebný kód do přiložených souborů, tj. do m-souboru a fig-souboru.

Doplnění souboru `casovy.fig` obsahujícího již předpřipravené rozhraní GUI spočívá v přidání takových prvků, které umožní uživateli snadným způsobem zvolit počet opakování časového útoku. Pro tuto činnost využijte editace přiloženého souboru v prostředí GUIDE. V okně MATLABu zvolte *File / New / GUI*. Dále zvolte záložku *Open Existing GUI* a vyberte cestu k vašemu souboru `casovy.fig`. Nyní máte možnost volit různé grafické objekty typu *uicontrol*. Pro splnění zadaného úkolu je důležité okno pro řízení vlastností – tzv. *Property Inspector*, který lze vyvolat dvojitým kliknutím na konkrétní grafický objekt.

V přiloženém souboru `casovy.m` je úkolem vytvořit kód, který bude zajišťovat operaci dešifrování se zadaným počtem časových útoků. Pro správnou funkci je nutné, aby byly při programování dodrženy doporučené značení důležitých proměnných. Na obrázku 4 je zobrazena jediná část kódu přiloženého m-souboru, která má být doplněna o samotnou realizaci časového útoku. Hlavním úkolem je vytvořit algoritmus *square and multiply* a pomocí časových měřících funkcí MATLABu uskutečnit časový útok (`tic,toc`). Nakonec podle počtu zadaných měření získat a zobrazit průměrné časové hodnoty pro jednotlivé bity soukromého klíče.

```
155   % ********************************************************************************
156
157   %------DOPLŇTE KÓD PRO REALIZACI ČASOVÉHO ÚTOKU - OPERACE DEŠIFROVÁNÍ---------
158
159
160       % binární podoba klíče d (označte d_bin)
161
162       % pomocné proměnné
163
164       % naplnění časového pole nulami (označte t)
165
166       % načtení počtu měření (využití příkazu set/get - propojení s GUI)
167
168       % dešifrování pomocí algoritmu square and multiply
169
170       % měření času pomocí funkce stopek
171
172       % získání průměrných časových hodnot pro jednotlivé bity klíče
173
174
175   %----------------------KONEC REALIZACE ČASOVÉHO ÚTOKU-------------------------
176
177   % ********************************************************************************
```

Obr. 4: Zadání úkolů – programování operace dešifrování

Při porovnávání dvou hodnot je nutné, aby byly stejného typu (string – string)! Další potřebné informace můžete získat z interaktivního *helpu* MATLABu. Dostupný z přímo hlavní nabídky MATLABu volbou *View / Help*.

3. Zaměřte se na průběh zpracování zprávy, která má být podrobena operaci šifrování. Pomocí aplikace si vyzkoušejte zadat zprávu a pozorujte její zformátování, které souvisí s operací šifrování v reálných implementacích algoritmu RSA. Prostudujte princip formátování zprávy v současných standardech RSA. Jakým druhem útoku by mohl být tento demonstrativní způsob šifrování napadnutelný?

Princip

Před šifrováním algoritmem RSA se v reálných implementacích provádí zpracování textové zprávy. Pro správnou činnost operace šifrování je nutná transformace všech znaků zprávy na číslo. Tento proces zpracování zprávy je možné rozdělit do několika kroků:

a) Každý znak zadané zprávy je nejprve reprezentován pomocí ASCII kódu.

b) Každé ASCII číslo je převedeno do osmibitové binární podoby.

c) Vzniklá osmibitová slova jsou spojovány do binárních bloků. V aplikaci k laboratorní úloze jsou pro jednoduchost spojeny každé dva znaky zprávy, tj. uskuteční se spojení dvou osmibitových slov do jednoho šestnáctibitového bloku. V případě lichého počtu znaků šifrované zprávy je poslední převod doplněn nulami do šestnáctibitového bloku. Reálné aplikace algoritmu RSA využívají spojení do bloků, které reprezentují 8 nebo 16 znaků zprávy. Délka těchto používaných bloků je potom 64 nebo 128 bitů.

d) Každý blok zprávy se převede do dekadické podoby, ve které bude blok zprávy šifrován. Tyto bloky musí být menší než modul n, v opačném případě by data nebylo možné jednoznačně dešifrovat.

e) Každé získané číslo bude z definice algoritmu RSA šifrováno uvedeným vztahem: $blok^{e} \bmod n$, kde e je veřejný klíč a n je modul. Pomocí tohoto vztahu jsou získány jednotlivé zašifrované bloky zprávy.

f) Tyto zašifrované bloky se mohou zpětně převést do binární číselné soustavy, rozdělit do osmibitových částí, které reprezentují v ASCII kódu určité znaky a výsledkem je konečná podoba šifrované zprávy.

V současných standardech implementací RSA se provádí při operaci šifrování také kódování zprávy. Některé formáty standardu RSA využívají hašovacích funkcí, které umožňují vytvořit krátké reprezentanty dané zprávy. Tyto metody výrazně zvyšují bezpečnost algoritmu RSA na základě maskování zprávy.

Postup

V aplikaci „Postranní kanály – časový útok na RSA“ se přepněte do druhé části, tj. reálný útok. Zaměřte se na levou část aplikace, která se věnuje právě operaci šifrování a ukázce zjednodušeného principu zpracování zprávy. Zadání zprávy je podmíněno omezením na maximální délku 25 znaků.

4. Prostudujte princip a postup realizace skutečného časového útoku na modul RSA. S využitím aplikace „Postranní kanály – časový útok na RSA“ se pokuste realizovat reálný časový útok. Ze získaných informací stanovte Hammingovu váhu soukromého klíče RSA. Vyzkoušejte reálný časový útok pro obě varianty délky soukromého klíče. Čím mohou být způsobeny možné nepřesnosti výsledku útoku?

Princip

Při vedení útoku pomocí časových postranních kanálů na fyzickou implementaci algoritmu RSA je hlavním cílem získat určitou senzitivní informaci. Zmíněnou citlivou informací je při reálném časovém útoku hodnota Hammingovy váhy klíče. Hammingova váha soukromého klíče je dána počtem bitů klíče, které mají hodnotu rovnou jedné. V rámci reálného časového útoku již není možnost získat dobu trvání dešifrování jednotlivého bitu klíče jako tomu bylo u demonstrativního útoku. Jedinou relevantní informací, kterou je útočník schopen využít je celkový čas dešifrování zprávy pomocí soukromého klíče. Útočník má také ve většině případů přibližnou informaci o době trvání dešifrování bitem klíče s hodnotou rovnou jedné. Informace jsou vztaženy vždy pro konkrétní kryptografický modul.

Před realizací časového útoku je nutné provést inicializaci, která stanovuje průměrnou dobu při dešifrování bitem s hodnotou rovnou nule a jedné. Cílem inicializační funkce je stanovit průměrnou i nejčetnější dobu dešifrování pro bity klíče s hodnotou rovnou jedné a také pro bity s hodnotu rovnou nule. Z každých čtyřiceti měření jsou stanoveny uvedené parametry pro jednotlivé bity klíče. Tento postup je realizován opakovaně pro 3000 pokusů, ze kterých jsou průměrem určeny potřebné informace o dané implementaci algoritmu RSA. Nejpřesnějších výsledků je dosaženo v případě, kdy není inicializační funkce rušena jinými aplikacemi a v ideálním případě uživatel nepohybuje při běhu funkce s kurzorem myši.

Obdobným způsobem je proveden samotný časový útok, kdy měření celkového času je opakováno pro 3000 pokusů, což je stanovený kompromis mezi rychlostí a kvalitou dosažených výsledků. Z každého změřeného celkového času při operaci dešifrování je pomocí inicializací stanovených průměrných hodnot dosaženo nejpravděpodobnější hodnoty Hammingovy váhy soukromého klíče.

Postup

V aplikaci „Postranní kanály – časový útok na RSA“ se přepněte opět do druhé části. Pro časový útok na modul algoritmu RSA je nejdříve nezbytné zvolit délku klíče, na který bude aplikace vést útok. Dalším krokem je spuštění inicializační funkce, při které je nutné na dané stanici nevyvíjet žádnou další činnost. Konec inicializace je signalizováno naplněním zpočátku prázdných polí v části s názvem „informace o implementaci“. Tlačítko „dešifrování – časový útok“ slouží k ukázce principu tohoto typu útoku změřením celkového času dešifrování zprávy. Jedná se pouze o ukázku jediného měření, které nemá potřebnou statistickou váhu a proto nemůže pravděpodobně vést ke správné hodnotě Hammingovy váhy. Přesto je výpočet hodnoty Hammingovy váhy z tohoto jediného samostatného útoku uveden v příslušném poli aplikace.

Hlavním krokem reálného útoku je spuštění analýzy dat, která uskuteční opakovaně 3000 útoků a z každého provede výpočet Hammingovy váhy. Pro dosažení korektních výsledků je nutné opět dodržet nečinnost kurzoru myši. Po ukončení reálného útoku aplikace zobrazí pravděpodobnost Hammingovy váhy klíče a také průměrnou hodnotu této veličiny. Poté máte možnost provést ověření správnosti určení hodnoty Hammingovy váhy soukromého klíče.

5. Určete počet všech možných kombinací obou soukromých klíčů. Srovnejte tento počet s počtem kombinací při znalosti Hammingovy váhy soukromého klíče. O kolik se sníží počet možných kombinací klíče po realizaci časového útoku? Uveďte pro každý ze zadaných soukromých klíčů *d*.

Postup

Počet **variací** *k*-té třídy z *n* prvků s opakováním, tzn. každý prvek se ve výběru může objevit vícekrát, je určen vztahem

$$V'_k(n) = n^k \ .$$

Kombinace *k*-té třídy z *n* prvků je skupina *k* prvků vybraných z celkového počtu *n* prvků a je definována vztahem

$$C_k(n) = \binom{n}{k} = \frac{n!}{k!(n-k)!} \ .$$

6. Zamyslete se na možnostmi zabezpečení proti realizovanému skutečnému časovému útoku. S využitím internetu zjistěte jakým protiopatřením by mohli být a jsou modifikovány reálné implementace RSA (*square and multiply*)?

Po absolvování laboratorní úlohy byste měli být schopni zodpovědět následující otázky:

- Co jsou to postranní kanály?
- Jakým způsobem lze klasifikovat postranní kanály?
- Jaký byl historicky první objevený PK a kdo ho publikoval?
- Jaké hlavní tři útoky PK lze realizovat na implementace RSA?
- Na jakém principu je založen jednoduchý časový útok na RSA?
- Jak se používá GUIDE prostředí MATLAB?
- Jak je realizována operace dešifrování v rychlých implementacích RSA?
- Jaké funkce MATLABu lze použít pro měření času?
- Jakým je postup v reálných implementacích RSA při operaci šifrování?
- Jaké operace jsou navíc oproti aplikaci použity před šifrováním a proč?
- Jaký je princip skutečného časového útoku na implementace RSA?
- Jakou citlivou informaci je možné získat analýzou časového PK RSA?
- Co vyjadřuje a jakou má značku tato informace?

8 ZÁVĚR

Cílem této diplomové práce bylo poukázat na možné nebezpečí útoků postranními kanály na kryptografické moduly a vytvořit podrobný přehled z dostupných informací o současném stavu této nově se rozvíjející problematiky v oblasti kryptoanalýzy. Práce se podrobně zabývá studiem jednotlivých postranních kanálů, historií a podstatou jejich vzniku, základními principy jejich využití při útoku na kryptografický modul a také jsou zmíněny příklady i dnes hrozících útoků využívající postranní kanály.

Hlavní prioritou diplomové práce bylo na základě zpracování uceleného přehledu současné situace této problematiky vytvořit návrh a realizaci laboratorní úlohy, která by vhodným způsobem mohla studentům demonstrovat novou oblast kryptologie. Teoretická část práce se zaměřuje na využití postranních kanálů při útoku na implementace asymetrického algoritmu RSA.

Z důvodů jednoduchosti a názornosti byl pro účely laboratorní úlohy zvolen útok časovým postranním kanálem na implementaci algoritmu RSA. Hlavní součástí celé laboratorní úlohy je funkční uživatelská aplikace, která v sobě zahrnuje zjednodušený i reálný případ časového útoku na algoritmus RSA. Aplikace je vytvořena ve zvoleném vývojovém prostředí MATLAB s využitím systému *handle graphics*. Laboratorní úloha je sestavena z následujících témat, jejichž účelem je poskytnout studentům nové poznatky i praktické zkušenosti z problematiky útoků časovými postranními kanály. Jedná se o demonstrativní časový útok, tvorbu grafického uživatelského rozhraní, programování časového útoku v prostředí MATLAB, operaci šifrování v reálných implementacích RSA a reálný časový útok.

Dosažené výsledky reálného časového útoku v laboratorní úloze podtrhují závažnost postranních kanálů, které se v současnosti řadí mezi nejvíce úspěšnou a efektivní metodu při útoku na různé kryptografické systémy.

9 SEZNAM POUŽITÉ LITERATURY

[1] BITTO, O.: *Historie kryptologie*, článek MU, 2003, dostupné z www: <http://www.fi.muni.cz/usr/jkucera/pv109/2003/xbitto.htm>

[2] KLÍMA, V., ROSA, T.: *Further Results and Considerations on Side Channel Attacks on RSA,* Technical Report, May 2002, dostupné z www: <http://eprint.iacr.org/2002/071.pdf>

[3] KLÍMA, V.: *Symetrická kryptografie,* soubor přednášek na MFFUK, 2003-2007, dostupné z www: <http://cryptography.hyperlink.cz/index_mffuk.htm>

[4] SINGH, S.: *Kniha kódů a šifer*, nakladatelství Dokořán, 2003

[5] KUNDEROVÁ, L.: *Bezpečnost IS/IT* , soubor přednášek na MZLU v Brně, dostupné z www: <https://akela.mendelu.cz/~lidak/bis/>

[6] PINKAVA, J.: *Úvod do kryptologie*, odborný článek, květen 1998, dostupný z www: <http://crypto-world.info/pinkava/uvod/uvod98.pdf>

[7] KOCHER, P.: *Timing attacks on implementations of Diffie-Hellmann, RSA, DSS and other systems*. Proc. of CRYPTO'97, Springer LNCS vol. 1109, pp.104-113, 1997

[8] BURDA, K.: *Bezpečnost informačních systémů*, skripta VUT, 2005, dostupné z www: <https://www.feec.vutbr.cz/et/skripta/utko/Bezpecnost_informacnich_systemu_S.pdf>

[9] KLÍMA, V., ROSA, T.: *Vybrané aspekty moderní kryptoanalýzy,* odborný článek, 2003, dostupné z www: <http://crypto.hyperlink.cz/files/ST_2003_03_str_03_07.pdf>

[10] ČSN ISO/IEC 13888-1, 2, 3, 4: *Nepopiratelnost*, ČSNI, květen 2001

[11] PIPER, F., MURPHY, S.: *Kryptografie - průvodce pro každého*, nakladatelství Dokořán, 2006

[12] KŘÍŽ, J.: *Postranní kanály v kryptografii*, bakalářská práce VUT Brno, 2007

[13] ROSA, T.: *Kryptografie v klidu a bezpečí 1-6*, soubor odborných článků, časopis CHIP 2001

[14] DANĚČEK, P.: *Útoky na kryptografické moduly,* doktorská práce VUT, 2007

[15] DHEM, J., KOEUNE, F., LEROUX, P., QUISQUATER , J., WILLEMS, J.: *A Practical Implementation of the Timing Attack,* Technical Report, 1998

[16] KLÍMA, V, ROSA, T.: *Na kanálu se pracuje aneb o revolučním objevu v kryptoanalýze*, odborný článek, součástí Open weekend 2003

[17] BLEICHENBACHER, D.: *Chosen Ciphertexts Attacks Against Protocols Based on the RSA Encryption Standard PKCS#1,* Technical Report CRYPTO '98

[18] QUISQUATER, J.-J.: *Side Channel Attacks*, Technical Report, October 2002

[19] PŘIBYL, J.: *Matematické algoritmy - čínská věta o zbytcích*, soubor přednášek na ČVUT, 2007 , dostupné z www: <http://euler.fd.cvut.cz/predmety/ma/files/ma-05-2007.pdf>

[20] JOYE, M., QUISQUATER, J.: *Faulty RSA Encryption*, Technical Report, 1998, dostupné z www: <www.ussrback.com/cryptopapers/1997/www.dice.ucl.ac.be/crypto/tech_reports/CG1997_8.ps.gz>

[21] MENEYES A. J.,OORSCHOT, P. C., VANSTONE, S. A., Handbook of Applied Cryptography, CRC Press, 1996

[22] ROSA, T.: *Modern Cryptology –Standards Are Not Enough*, doktorská práce, ČVUT Praha, 2004

[23] AGRAWAL, D., ARCHAMBEAULT, B., RAO, J. R., ROHATKI, P.: *The EM-side channel(s)*, 2002, dostupné z www: <www.research.ibm.com/intsec/emf-paper.ps>

[24] ZAPLATÍLEK, K., DOŇAR, B.: *MATLAB-tvorba uživatelských aplikací*, nakladatelství BEN, Praha 2008, 216 stran

[25] KLÍMA, V., ROSA, T.: *RSA v novém světle,* série odborných článků – časopis CHIP, 2001

[26] IBM RESEARCH: *Partitioning Attacks on GSM Cards,* odborné články – internetová bezpečnostní skupina, 2003

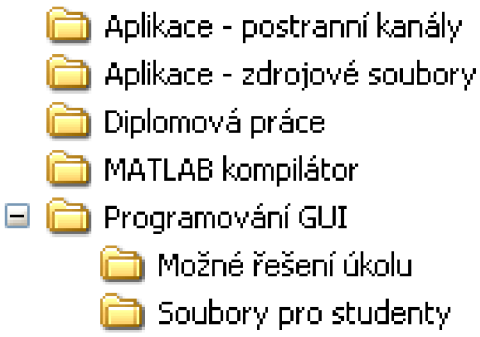
10 SEZNAM POUŽITÝCH ZKRATEK

3-DES	Symetrický blokový algoritmus (*Triple Data Encryption Standard*)
AES	Symetrický blokový algoritmus (*Advanced Encryption Standard*)
CBC	Mód symetrického algoritmu (*Cipher-Block Chaining mode*)
CMOS	Technologie integrovaných obvodů (*Complementary metal–oxide–semiconductor*)
COMP128	Algoritmus využívaný v GSM
CRT	Monitor založený na katodové trubici (*cathode ray tube*)
CRT věta	Čínská věta o zbytcích (*Chinese Reminder Theorem*)
DEMA	Diferenciální elektromagnetická analýza
DFA	Diferenciální chybová analýza
DPA	Diferenciální proudová analýza
DTA	Diferenciální časová analýza
DES	Symetrický blokový algoritmus (*Data Encryption Standard*)
DSA	Algoritmus digitálního podpisu (*Digital Signature Algorithm*)
DSS	Standard digitálního podpisu (*Digital Signature Standard*)
D-H	Kryptografický asymetrický protokol (*Diffie-Hellman*)
ECC	Asymetrický algoritmus (*Eliptic Curve Cryptography*)
GSM	Globální Systém pro Mobilní komunikaci (*Groupe Spécial Mobile*)
GUI	Grafické uživatelské prostředí (*Graphical User Interface*)
GUIDE	Nástroj pro interaktivní tvorbu GUI (*Development Environment*)
IDEA	Symetrický blokový algoritmus (*International Data Encryption Algorithm*)
OAEP	Způsob formátování – standard RSA (*Optimal Asymmetric Encryption Padding*)
RC5	Symetrický blokový algoritmus (*Rivest Cipher*)
RSA	Asymetrický algoritmus (*Rivest, Shamir, Adleman*)
SEMA	Jednoduchá elektromagnetická analýza
SFA	Jednoduchá chybová analýza
SPA	Jednoduchá proudová analýza
STA	Jednoduchá časová analýza
SSL	Kryptografický bezpečnostní protokol (*Secure Sockets Layer*)
SIM	Účastnická identifikační karta (*subscriber identity module*)
TEMPEST	Skupina standardů (*Transient Elec.Magnetic Pulse Emanation Standard*)
TLS	Kryptografický bezpečnostní protokol (*Transport Layer Security*)

A PŘÍLOHY

A.1 Obsah DVD

Přiložené paměťové médium obsahuje tyto uvedené složky.



Obr. 11.1: Složky obsahu přiloženého DVD

Aplikace – postranní kanály

V této složce je uložena hlavní zkompilovaná aplikace „Postranní kanály – časový útok na RSA“, která je základem laboratorní úlohy. Obsahuje šest souborů, z nichž je podstatný spustitelný soubor `laboratorni_uloha.exe`.

Aplikace – zdrojové soubory

Složka obsahuje zdrojové soubory zkompilované aplikace „Postranní kanály – časový útok na RSA“, tj. `casovy.m` , `casovy.fig`.

Diplomová práce

Složka obsahuje celý text diplomové práce ve formátu pdf, tj. soubor `útoky postrannímí kanály.pdf`.

MATLAB kompilátor

Složka obsahuje potřebný instalační soubor kompilátoru `MCRInstaller.exe`, jehož instalace je vyžadována pro funkci aplikace laboratorní úlohy. Popis instalace je uveden v kapitole 6.7.

Programování GUI

Složka obsahuje základní soubory pro druhý úkol laboratorní úlohy, tj. `casovy.m`, `casovy.fig`. Obsahuje také možné řešení zadaného úkolu.