VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

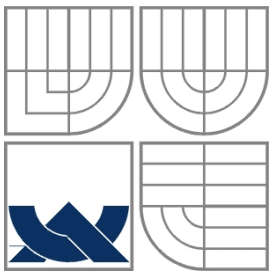
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ŠIFROVANÁ KOMUNIKACE MEZI FITKITEM A PC

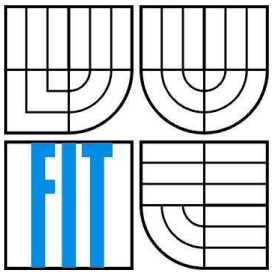
BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE MIROSLAV KOUŘIL
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

ŠIFROVANÁ KOMUNIKACE MEZI FITKITEM A PC

FITKIT – PC CRYPTED COMMUNICATION

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE — Miroslav Kouřil
AUTHOR

VEDOUCÍ PRÁCE — Ing. Richard Růžička, Ph.D.
SUPERVISOR

BRNO 2007

Abstrakt

Tato práce se zabývá problémem utajení citlivých dat při přenosu mezi dvěma systémy. Pro šifrování byl vybrán standard AES. Jedná se o blokovou symetrickou šifru. Pro praktickou realizaci bylo určeno spojení mezi platformou FITkit a osobním počítačem pomocí sériové komunikace. Na straně FITkitu je program realizován v jazyce C a na straně osobního počítače v jazyce C++. Pro ustavení spojení a výměnu informací o šifrování byl navržen jednoduchý protokol. Kvůli problémům se sériovou komunikací na straně kitu byly vytvořeny dvě aplikace. První čte šifrovaná data z kitu a překládá je pomocí přednastavených hodnot. Druhá komunikuje s emulátorem kitu na druhém počítači a pracuje v plném rozsahu, tzn. ustavení spojení, domluva na tvorbě průběžných klíčů, na počtu šifrovacích kol, bezpečná výměna klíčů a možnost čtení a zápisu dat do kitu.

Klíčová slova

Šifrovaná komunikace, symetrická šifra, bloková šifra, AES, Rijndael, FITkit, ECB, CBC, CFB, OFB, sériová komunikace, protokol pro výměnu klíčů

Abstract

This thesis deals with the issue of concealing confidential data being transmitted in between two systems. The coding standard AES as a block symmetric cipher has been selected. In practise, the connection between the FITkit platform and a PC was set via serial communication. The FITkit is programmed in language *C* and the PC in language *C++*. There has been designed a simple protocol for setting up the connection and for the information exchange about encoding. Due to the difficulties with serial communication on the kit side there have been created two applications. The first application reads encoded kit data and translates them with the assistance of preset values. The second one communicates with the kit emulator on the other computer and works at full range, what means - establishing the connection, generating keys modes and number of encoding rounds, safe key exchange and the possibility of data reading and writing to the kit.

Keywords

Crypted communication, symetric cipher, block cipher, AES, Rijndael, FITkit, ECB, CBC, CFB, OFB, serial communications, key exchange protocol

Citace

Miroslav Kouřil: Šifrovaná komunikace mezi FITkitem a PC, bakalářská práce, Brno, FIT VUT v Brně, 2007

Šifrovaná komunikace mezi FITkitem a PC

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Richarda Růžičky.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jméno Příjmení
Datum

Poděkování

Děkuji vedoucímu mé bakalářské práce panu Richardu Růžičkovi za pomoc při tvorbě této práce.

Obsah

Úvod

„Proč předpokládáš, že někdo zachová tvoje tajemství, když jsi je sám nedokázal zachovat?“

La Rochefoucauld

Toto přísloví jsem použil, abych hned na začátku mé práce zdůraznil její hlavní účel. Tím účelem je uchování tajemství. Každý člověk má několik tajemství a nikdo není rád, když je právě jeho tajemství odhaleno. Bohužel je spousta dalších, kteří po tajemství druhých pátrají z různých důvodů, od zlomyslnosti až po touhu tajemství zneužít ve svůj prospěch.

V této práci se zabývám utajováním informací přenášených po komunikačním kanále, který může potencionální útočník sledovat a přenášená data číst. V reálném životě si můžeme vhodnost šifrování přenosu ukázat na načítání dat z kamerového systému, který střeží libovolný objekt. Pokud by se na nešifrovanou linku připojil útočník, měl by okamžitě přístup k proudu dat z kamery, nehledě na skutečnost, že by mohl posílat příjemci svoje data a v prostorách hlídaných kamerou libovolně operovat. Pokud by byla data šifrována, mohl by útočník maximálně tok dat přerušit, čímž by na sebe okamžitě upozornil.

Existuje mnoho cest, kterými bychom se mohli vydat, ale ne všechny vedou k bezpečné komunikaci. Jako bezpečnou komunikaci považujeme skrytí dat takovým způsobem, který nedokáže útočník rozluštit v přijatelné době za použití dostupných nástrojů.

V této práci zkoumám aktuálně platný šifrovací standard AES, který byl vybrán ve veřejné soutěži na konci dvacátého století. Na internetu je dostupná řada implementací tohoto algoritmu, pro svoji práci použiji implementaci v jazyce C, kterou upravím pro použití na platformě FITkit.

Další částí práce bude realizace spojení mezi platformou FITkit a osobním počítačem. Toto spojení bude probíhat po sériové lince. V rámci tohoto spojení je nutné navrhnout protokol, kterým se komunikující zařízení dohodnou na parametrech šifrovacího procesu. Tato data nejsou pro potencionálního útočníka příliš důležitá, proto mohou přecházet po lince v otevřené podobě. To ovšem neplatí o výměně šifrovacího klíče, která bude v práci implementována jedním z protokolů, které jsou pro toto navrženy.

1 Základní pojmy

Pro lepší orientaci ve světě šifrování je nutné vymezit několik důležitých pojmů, které se budou dále v textu vyskytovat. Často se v textu bude vyskytovat pojem "Třetí strana", kterým označuji osobu či systém, který se snaží získat naše data za účelem jejich zneužití. Obecně si pod tímto pojmem můžeme představit nepřítele, který chce z naší komunikace získat údaje o přístupu například k bankovnímu kontu.

1.1 Kryptologie

Kryptologie zahrnuje kryptografii a kryptoanalýzu. Jedná se o souhrnné označení vědních disciplín, které se zabývají tvorbou a studiem šifer.

1.1.1 Kryptografie

Kryptografie neboli šifrování je nauka o metodách utajování smyslu zpráv převodem do podoby, která je čitelná jen se speciální znalostí. Slovo kryptografie pochází z řečtiny – kryptós je skrytý a gráphein znamená psát.

1.1.2 Kryptoanalýza

Kryptoanalýza je věda zabývající se metodami získávání obsahu šifrovaných informací bez přístupu k tajným informacím, které jsou za normálních okolností potřeba, například tajného klíče. Slovo kryptoanalýza pochází také z řečtiny - kryptós je skrytý a analýein znamená „uvolnit“ či „rozvázat“. V netechnickém kontextu je používán tento termín obecně pro prolamování kódu. Je vlastně opakem kryptografie, která šifry vytváří.

1.2 Způsoby ukrytí informací

Existují tři obecné způsoby, jak je možno informace ukrýt. Často jsou tyto způsoby chápány jako celek, proto je nutné je v úvodu jasně odlišit. Těmto způsobům se říká šifrování, kódování a steganografie.

1.2.1 Šifrování

Šifrování ukrývá informaci pomocí tajného klíče, který se může libovolně měnit. Bez tohoto klíče by jsme neměli v rozumném čase z šifrovaného textu získat původní otevřený text či nějaké zásadní informace o tomto textu. Při šifrování je nejdůležitější utajení šifrovacího klíče před třetí stranou.

Příkladem šifer jsou DES a nebo AES, kterému se věnuje tato práce.

1.2.2 Kódování

Je zvláštní druh šifrování, při kterém se vždy používá stejný šifrovací klíč. Oproti šifrování je u kódování důležité uchovat v tajnosti systém, kterým jsme zprávu zakódovali.

Příkladem kódování je například posun znaků dle ASCII tabulky – každému znaku předem odpovídá jiný znak posunutý o určité číslo vpřed v ASCII tabulce.

1.2.3 Steganografie

Steganografie nepřevádí zprávu do nečitelné podoby, ale snaží se utajit její vlastní existenci.

Příkladem steganografie je psaní vzkazů na vejce - roztokem z kamence a octu se napíše zpráva na vajíčko uvařené natvrdo. Tento roztok pronikne skrz skořápku a vzkaz je viditelný až po oloupání vejce (autor ital Giovanni Battista della Porta)

1.3 Způsoby šifrování

Obecně lze šifrování rozdělit do dvou tříd, které se zásadně liší způsobem zpracování šifrovacího klíče a do šifry, která kombinuje oba způsoby.

1.3.1 Symetrické šifrování

Symetrické šifry používají pro šifrování i dešifrování stejný šifrovací klíč. Tento způsob šifrování je velmi starý a pro svoji jednoduchost a zejména rychlost je neustále vyvíjen a aktualizován.

Zlomem v možnostech symetrických šifer bylo zavedení počítačů do šifrovacího procesu, což umožnilo mnohonásobné zesložitnění používaných algoritmů.

1.3.1.1 Historické a ruční šifry

Použití jednoduché transpozice

Tento styl byl založen na předpokladu nízké inteligence lidí, ke kterým se takto skrytý text dostal. Příkladem je dodnes používaná šifrovací mřížka, která pracuje na principu vkládání textu jen do

určitých políček mřížky, která se poté otočí o devadesát stupňů a proces se opakuje až do zapsání celé zprávy.

Využití substituce

Zde se používalo nahrazení znaků v otevřeném textu jinými znaky, které byly stanoveny podle předchozí dohody. Jako příklad substituce je možno uvést americký kód Navajo. Tento kód pracoval na principu překladu znaků do řeči indiánského kmene, který žil v Americe. V každé jednotce byl zástupce tohoto kmene, který skryté zprávy překládal.

1.3.1.2 Moderní šifry s využitím počítačů

Nástup počítačů do oblasti šifrování znamenal malou revoluci ve složitosti šifer. Obecně se šifry začaly dělit na dvě podskupiny: proudové a blokové.

Proudové šifry

Tyto šifry zpracovávají text po jednotlivých bitech. Příkladem tohoto zpracování je šifra RC4.

Blokové šifry

Blokové šifry vezmou otevřený text, který poté rozdělí na stejně velké bloky dle typu šifry. Poté se berou jednotlivé bloky a šifrují se algoritmem šifry.

Drobným omezením tohoto způsobu šifrování je fakt, že text musí být vždy v celém bloku. Může nastat situace, že text nevyplní celý blok dat. Potom záleží na domluvě mezi komunikujícími stranami, jak bude volné místo vyplněno. Obvykle je tento prostor vyplněn nulami nebo kombinací nul a jedniček. Příkladem blokové šifry je algoritmus AES.

1.3.2 Asymetrické šifrování

Hlavním rysem asymetrických šifer je fakt, že pro šifrování a pro dešifrování jsou použity odlišné klíče. Tento způsob je z hlediska bezpečnosti velmi efektivní, ale je složitější na implementaci a hardwarové nároky. Často je využíván pro elektronický podpis.

Strana, která chce komunikovat asymetrickou šifrou, si vygeneruje dva klíče – veřejný a privátní. Veřejný klíč vystaví veřejně a soukromý pečlivě ukryje. Pokud jí někdo chce poslat zprávu, tak ji zašifruje veřejným klíčem a odešle. Tuto zprávu lze dešifrovat pouze se znalostí privátního klíče. Představitelem asymetrické šifry je algoritmus RSA.

1.3.3 Kombinace obou metod šifrování

Obecně lze prohlásit, že symetrické šifrování vyniká v rychlosti, ale má možnou slabinu ve faktu, že je nutné nějak přenést společný šifrovací klíč. Na druhou stranu asymetrické šifry jsou velmi bezpečné, ale pomalejší a náročnější na implementaci. Proto se používá jejich kombinace. Nejprve si obě strany vymění šifrovací klíč pomocí asymetrické šifry a následné šifrování dat již probíhá symetrickou šifrou.

2 Šifrovací standardy

Prudký rozmach počítačů ve druhé polovině dvacátého století podnítil nutnost najít šifrovací systém, který by sloužil jako standard.

2.1 DES

Tento algoritmus byl dlouhá léta uznávaným šifrovacím standardem, proto se u něj trochu pozastavíme a podíváme se na něj podrobněji.

2.1.1 Historie algoritmu

V roce 1973 byla vyhlášena veřejná soutěž o návrh kryptosystému, který by mohl být prohlášen za standard. Roku 1976 byl jako standard přijat šifrovací systém DES firmy IBM. Bylo určeno, že se bude užívat po dobu 10 let a poté každých 5 let bude podroben novému testu bezpečnosti, podle kterého mu bude platnost prodloužena.

Velký rozruch vzbudil fakt, že americký Národní bezpečnostní úřad (dále jen NSA) vynutil změny v některých částech algoritmu, než byl oficiálně uznán jako standard. Přesněji to byly změny v S-boxech a také v délce šifrovacího klíče, který byl zkrácen ze 128 bitů na 56 bitů. NSA toto prezentovala s tím, že tato délka klíče je postačující. To způsobilo podezření, že tento úřad vlastní klíč, kterým se dá tento algoritmus prolomit, ovšem toto tvrzení nebylo nikdy potvrzeno.

Po přijetí jako standard se DES velmi rychle rozšířil mezi odbornou veřejnost. Ač měl velkou řadu kritiků, jeho platnost se z předpokládaných 10 let protáhla až na 25 let. Ke konci svého působení již nebyl považován za dostatečně účinný a v roce 1998 byl sestrojen přístroj, který text zašifrovaný algoritmem DES dokázal rozluštit za necelých 60 hodin. To způsobilo definitivní konec tohoto algoritmu. V roce 1997 byla vyhlášena veřejná soutěž o další standard.

2.1.2 Funkce algoritmu

DES je symetrická bloková šifra, které šifruje data ve 64 bitových blocích. 64 bitový blok prostého textu vstupuje do algoritmu a 64 bitový blok šifrovaného textu algoritmus opustí. Délka klíče je 56 bitů, ale obvykle se uvádí jako 64 bitové číslo, ve kterém poslední bit v každém bytu je pouze pro kontrolu parity a je ignorován.

Na své nejednodušší úrovni je algoritmus kombinací substituce a permutace založené na klíči, což tvoří základní blok konstrukce algoritmu, nazývaný kolo (anglicky round). DES se sestává ze 16 kol, tudíž aplikuje stejnou techniku kombinací na prostém textu 16krát za sebou.

2.2 AES

Tento standard je předmětem této práce, proto mu je věnován větší prostor. Bude zde popsána soutěž, která algoritmus jako standard vybrala a také probereme jeho funkci.

2.2.1 Průběh soutěže

Soutěž, která byla vyhlášena roku 1997, byla veřejná a každý se mohl zapojit se svým algoritmem a na ostatních algoritmech hledat možné slabiny. V podmínkách soutěže byla zakotvena i povinnost účastníků zřeknout se licenčních poplatků, autorských práv a vyřešit veškeré patentové ochrany tak, aby jejich algoritmus mohl být kýmkoliv bezplatně použit. Tím nabyla celá soutěž trochu rázu prestižního klání kryptologických es.

Požadavky na algoritmus byly následující: bloková symetrická šifra s délkou bloku 128 bitů, která musí podporovat délky klíčů 128, 192 a 256 bitů. Při výběru se posuzovala zejména bezpečnost, cena (v širokém smyslu slova - např. cena při hardwarové implementaci) a implementační vlastnosti. Z hlediska bezpečnosti se posuzovaly možné slabiny, cenila se čistota návrhu a již dříve osvědčené postupy. Několik šifer bylo vyřazeno hned v prvním kole, protože byla teoreticky odhalena slabina, která sice v dané době nebyla napadnutelná, ale k vyřazení ze soutěže to stačilo (algoritmus LOKI97 či Magenta). Z hlediska výkonnosti se posuzovala složitost zejména softwarové implementace, na hardwarovou se hledělo až v druhé řadě.

První kolo soutěže probíhalo na konferenci o novém šifrovacím standardu ve smyslu otázky "Měl by být tento algoritmus vybrán do druhého kola?". K této otázce se vyjádřili všichni účastníci konference u každého algoritmu. S přihlédnutím na výsledky hlasování a rychlostní testy bylo vybráno pět algoritmů, které postoupily do druhého kola. Byly to MARS, RC6, Rijndael, Serpent a Twofish.

Všech pět finalistů jsou iterovanými blokovými šiframi, tj. šifrující transformace nad blokem dat se několikrát iteruje. Taková iterace se nazývá runda (round). Rundy užívají k šifrování tzv. rundovní klíče (subkeys) odvozené z původního klíče zadaného uživatelem. Čtyři finalisti použili i tzv. S-boxy (substituční boxy), což jsou nelineární substituční funkce měnící jednotlivé bity. Tři finalisti použili Feistelovo schéma nebo nějakou variaci na něj - klasické Feistelovo schéma bylo použito např. v DES, při něm se v každé rundě jedna polovina vstupního slovo zašifruje, druhá se ponechá beze změny a obě poloviny se prohodí; transformace první poloviny slova je parametrizována druhou, neměněnou polovinou slova, po dvou rundách je tedy zašifrováno celé slovo. Druzí dva finalisté, kteří Feistelovo schéma nepoužili, zpracovávají paralelně vstupní slovo sérií substitucí a lineárních transformací, jsou tedy příkladem substitučně-lineární transformační sítě.

V následujícím období se začalo detailně zkoumat všech pět algoritmů. Softwarová implementace se posuzovala z hlediska využití paměti RAM a ROM, z hlediska poměru bezpečnost -

rychlost či podle rychlosti přípravy klíčů a rychlosti samotného šifrovacího (dešifrovacího) procesu. Podobně podrobně byla zkoumána i hardwarová implementace a odolnost vůči fyzickým útokům.

Soutěž skončila dne 2. října 2000, kdy byl jako vítěz vyhlášen algoritmus belgičanů Joana Deamena a Vincenta Rijmena jménem Rijndael. Ihned se začalo pracovat na sestavení standardu, aby se mohl protokol začít užívat globálně. Jako oficiální standard pro šifrovanou komunikaci byl přijat 26. listopadu 2001. Odhadovaná životnost tohoto algoritmu je 20 až 30 let.

2.2.2 Popis algoritmu

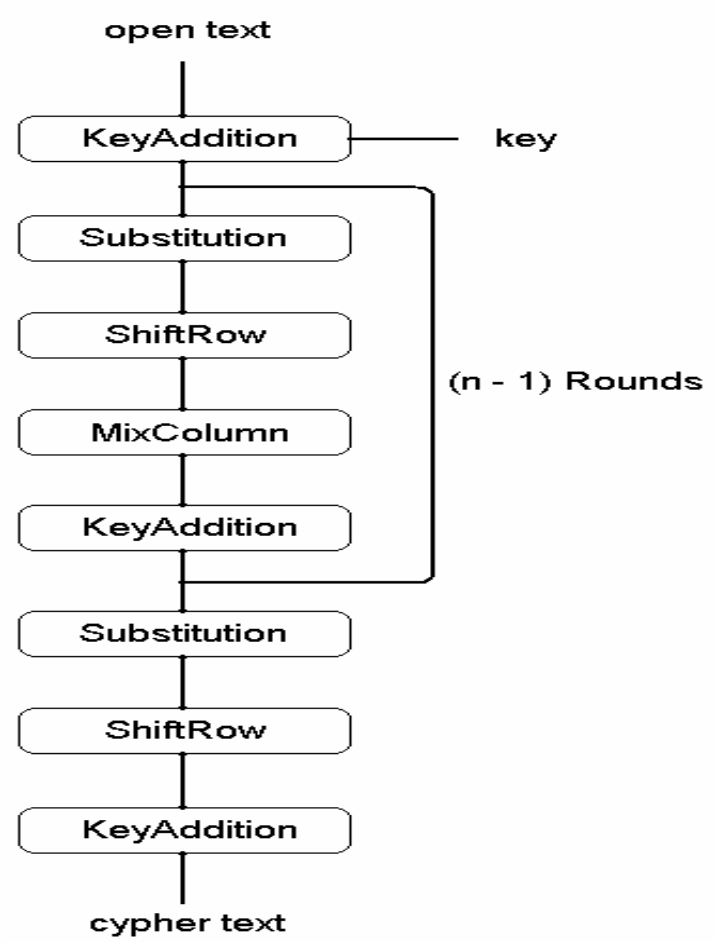
Délka vstupního bloku je 128 bitů, délka klíče je volitelná z hodnot 128, 196 a 256 bitů. Dle délky zvoleného klíče je doporučený počet rund 10, 12 nebo 14.
Za matematické základy algoritmu byly vzaty operace nad Galoisovým tělesem GF(2^8) , které mohou být realizovány buď za použití předpočítaných tabulek, což je výhodné pro softwarovou implementaci, nebo mohou být počítány až při samotném běhu algoritmu, což je naopak vhodné pro hardwarové implementace.

V následujícím textu vysvětlím funkci šifrovacího procesu, který se budu snažit vhodně doplnit komentáři s vysvětlením jednotlivých operací. Pro dešifrování se proces částečně liší, ale princip zůstává stejný, proto jeho vysvětlení vynechám. Případní zájemci o přesnou strukturu šifry nechť nahlédnou na konec práce do pramenů, kde najdou cestu na oficiální standard.

Na následujícím obrázku je uveden algoritmus šifrovacího procesu. Na začátku procesu je otevřený text naplněn do tabulky 4 x 4 byty, se kterou se celý šifrovací proces provádí. Také počítáme s faktem, že tabulky a klíče pro jednotlivá kola jsou předpočítány a uloženy v paměti. V obrázku jsou použité následující pojmy:

open text – otevřený text, který chceme zašifrovat
key – klíč, který bude použit pro šifrování
round – jedno kolo algoritmu
n – počet kol při šifrování textu
KeyAddition – vezme klíč pro aktuální kolo a přičte jej k šifrovanému textu
Substitution – funkce projde celý šifrovaný text a nahradí každý jeho byte příslušným bytem z tabulky S-Boxu.
ShiftRow – funkce provede rotaci řádků doprava podle toho, jaké má řádek číslo – první řádek o nula bytů, druhý o jeden byt atd.

MixColumn – tato funkce bere sloupce textu a násobí je s tabulkou tvořenou koeficienty s hodnotami 1, 2, 3 v různém pořadí.



Obr. 1: Průběh šifrování textu u šifry AES

Slovně lze obrázek vyjádřit následovně: Na začátek se vezme rundovní klíč a sečte se s otevřeným textem. Poté tento text projde řadou operací, při níž se nahradí všechny byty textu tabulkovými hodnotami, provede rotace řádků s posuny a vynásobí sloupce danými koeficienty. Pak se opět přičte ke klíči, který je vypočítán pro každé kolo jiný. Tento proces proběhne tolikrát, kolik je nastaven počet průchodů mínus jedna. Poslední průchod se provede bez operace s násobením sloupců.

3 Kryptografický mód šifry

Kryptografický mód šifry obvykle kombinuje její základní algoritmus se zpětnou vazbou. Tato kombinace je zajištěna pomocí nějaké jednoduché operace – jednoduché proto, že bezpečnost má být zajištěna samotnou šifrou a nikoliv módem jejího provozu. Přirozeně naopak nesmí platit, že kryptografický mód bude bezpečnost šifry snižovat.

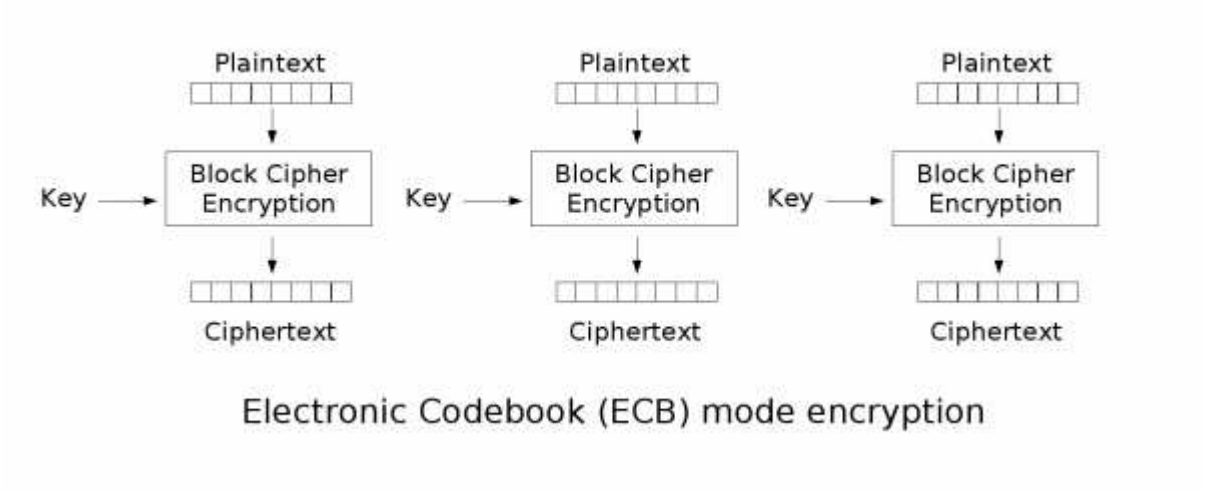
Důvodů pro použití jiného módu, než je základní (tj. originální podoba šifrovacího algoritmu), je několik. Jedním z nich je zvýšení bezpečnosti šifry – takovým příkladem může být snaha o zakrytí charakteristických rysů šifrového textu. Dalším důvodem může být to, že vlastnosti blokové šifry zcela neodpovídají požadavkům její aplikace (např. je třeba šifrovat text po menších částech než je velikost bloku potřebného pro šifrování). Důvodem může také být snaha o zabezpečení proti případným bitovým chybám v šifrovém textu (je třeba si uvědomit, že šifrová zpráva je přenášena od odesílatele k příjemci po komunikačním kanálu, který ne vždy je zcela bezporuchový). Jistě by se našla ještě celá řada smysluplných důvodů. Jak ale uvidíme, splňuje každý mód vždy pouze určitou podmnožinu všech těchto požadavků.

Některé módy jsou též standardizovány – konkrétně ECB, CBC, CFB a OFB jsou součástí norem FIPS 81 a ANSI X3.106-1983.

U popisu každého módu je připojen obrázek postupu při šifrovacím procesu.

3.1 ECB

Tento styl je ze všech nejjednodušší. Pracuje na principu, že každý blok dat je zašifrován stejným klíčem. Šifrování i dešifrování je prováděno stejnou cestou. Tento styl je tedy velmi jednoduchý z hlediska přípravy klíče, ale z hlediska kryptoanalýzy se naskytuje slabina šifrovaného textu - stejný blok dat je vždy zašifrován stejně, ať se nachází kdekoli v textu. Výhodou tohoto módu oproti ostatním módům je nezávislost šifrování každého bloku otevřeného textu na ostatních (ať už předcházejících nebo následujících) blocích. To umožňuje jednak s výhodou paralelizovat proces šifrování celé otevřené zprávy. Navíc šifrování můžeme provádět zcela libovolně (například můžeme začít šifrovat od středu zprávy, pak zašifrovat její začátek a jako poslední zpracovat konec).

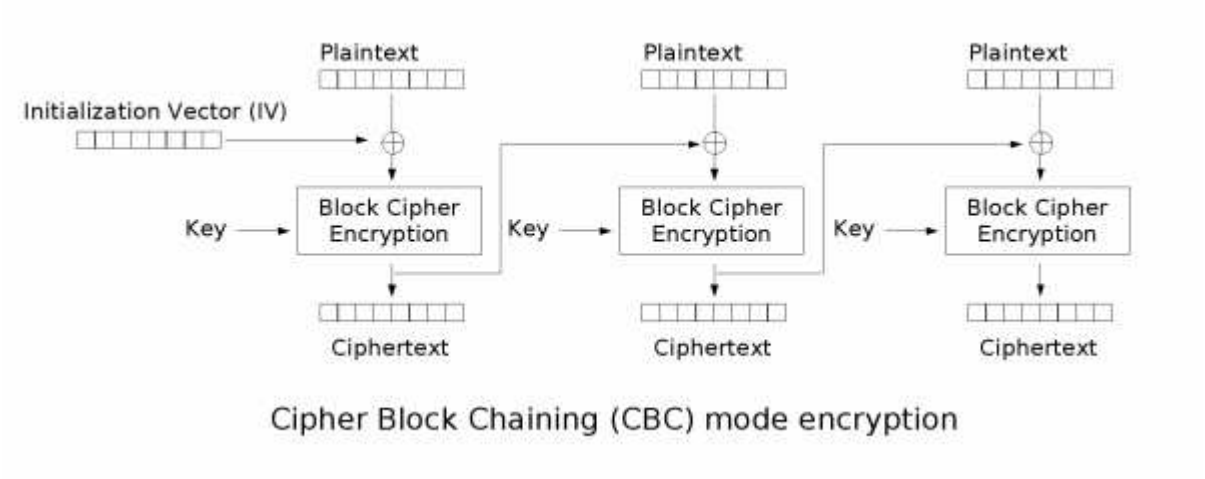


Obr. 2: Použití módu ECB při šifrovacím procesu

3.2 CBC

Tento styl používá zpětnou vazbu, při níž je výsledek šifrování předchozího bloku zařazen do šifrování současného bloku.Volný text je nejprve XORován se zašifrovaným textem předchozího bloku a teprve pak je zašifrován pro výstup. Zašifrovaný text je jednak odeslán a jednak zaznamenán pro použití na dalším bloku.

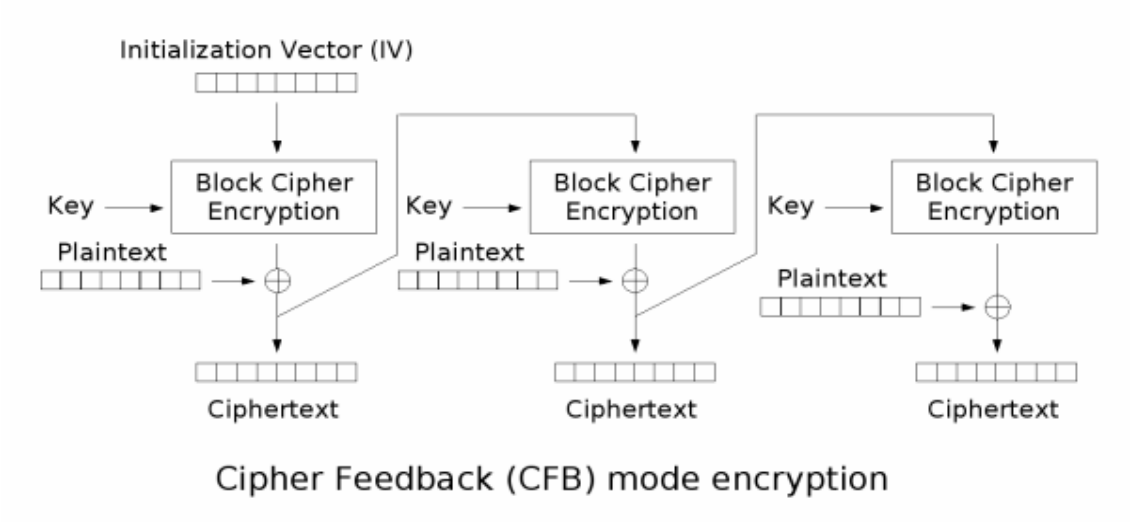
Dešifrování je stejně tak přímočaré. Zašifrovaný text je jednak dešifrován a jednak zaznamenán v registru. Poté, co je následující blok dešifrován, je na něm proveden XOR s uloženým blokem. CBC přiměje identický blok k zašifrování na jiné hodnoty pouze v případě, že část předchozího textu byla různá. Dvě stejné zprávy se zašifrují do stejného šifrovaného textu a ještě více různé zprávy, které stejně začínají se zašifrují do stejného šifrovaného textu až do prvního rozdílu. Způsob, jak tohle potlačit, je poslat v prvním bloku náhodná data nazývaná inicializační vektor (IV). Tento vektor není třeba držet v tajnosti, ale měl by být změněn s každou zprávou.



Obr. 3: Použití módu CBC při šifrovacím procesu

3.3 CFB

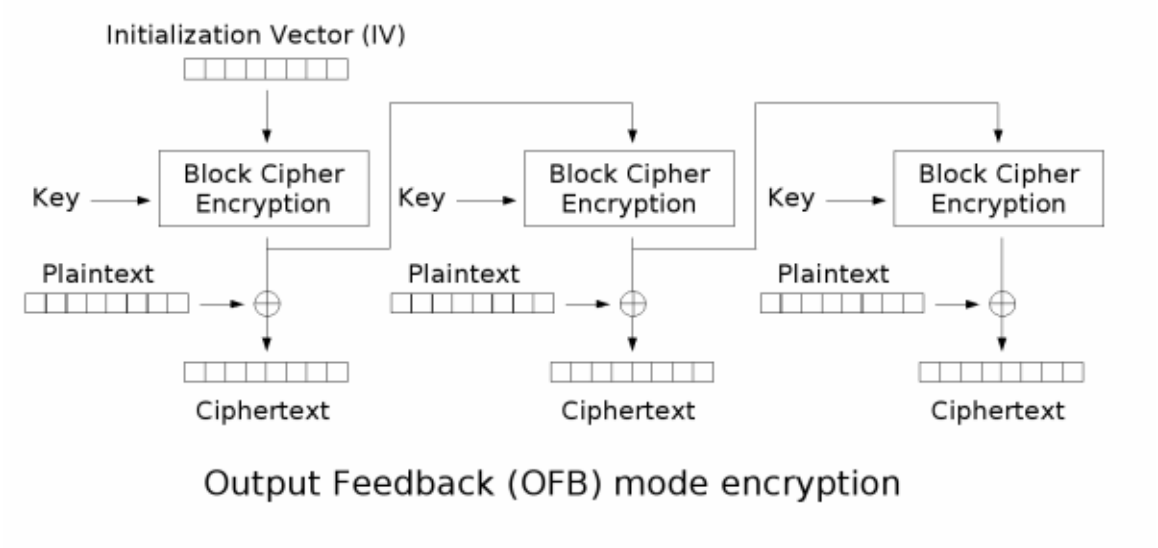
Pro CFB šifrování začíná, když jsou v zásobě data pro celý blok. Blok šifrovaného textu je získán zašifrováním minulého bloku šifrovaného textu (posledních 64 bitů) a přičtením části vzniklého šifrového textu (obvykle v délce 1 byte) modulo dva k stejně dlouhému bloku otevřeného textu.



Obr. 4: Použití módu CFB při šifrovacím procesu

3.4 OFB

Tento mód je téměř stejný jako CFB, ale narozdíl od CFB se pro šifrování následujícího bloku používá předchozí blok, který byl připraven pro šifrování s otevřeným textem.



Obr. 5: Použití módu OFB při šifrovacím procesu

4 Propojení FITkitu a PC

Další součástí práce bylo navrhnout propojení osobního počítače s platformou FITkit, aby bylo možné přenášet větší objemy dat mezi těmito systémy. Pojem větší objemy beru jako desítky až stovky kB.

4.1 Komunikační rozhranní

Pro svou práci jsem zvolil sériové rozhranní RS232, protože rozhranní USB je využito pro nahrávání dat do pamětí kitu a rozhranní PS2 se obecně používá jen jako vstupní rozhranní.

Na straně kitu jsem použil již implementované rozhranní, které je uloženo v souborovém systému kitu. Toto rozhranní implementoval pan ing. Ján Markovič.

Na straně osobního počítače jsem měl na výběr z několika volně dostupných knihoven, kterými by se dalo sériovou komunikaci realizovat. Jako příklad uvádím knihovnu Varian Async32 či knihovnu ComPort. Pro svou práci jsem vybral knihovnu, kterou napsal pan ing. David Matoušek pro programovací jazyk C++ Builder 6.0, kterou jsem doplnil o výběr portů pro komunikaci.

4.1.1 Rozhranní RS232

Toto rozhranní slouží pro asynchronní sériovou komunikaci. Pořadí přenosu datových bitů je od nejméně významného bitu po bit nejvýznamnější. Počet datových bitů je volitelný, obvykle se používá 8 bitů, lze se také setkat se 7 nebo 9 bity. Základní tři vodiče rozhranní jsou pro příjem RxD, vysílání TxD a společná zem GND.

Synchronizace přenosu u tohoto rozhranní probíhá tak, že každé sekvenci datových bitů předchází jeden start bit, kterým se logická hodnota na lince přepne (původně v klidovém stavu) do opačného stavu. Po datových bitech následuje paritní bit a za ním jeden nebo více stop bitů, během kterých je linka opět v klidovém stavu.

Ve své práci jsem použil rychlost přenosu 9600 Baudů, 8 datových bitů s jedním stop bitem bez využití parity. Nastavení přenosu jde dle libosti měnit v obvyklých hodnotách pro toto rozhranní.

4.2 Komunikační protokol

Pro účely šifrované komunikace bylo nutné navrhnout protokol, kterým program běžící na straně počítače sdělí FITkitu, jaké budou parametry šifry, kterou chce komunikovat. Těmito parametry se rozumí mód šifry, počet šifrovacích kol a vlastní šifrovací klíč, aby bylo na straně počítače možné šifrovaná data dešifrovat.

Tento model jsem zvolil proto, že zachovat na FITkitu vždy stejné parametry šifrování včetně stejného klíče by nebylo vhodné z hlediska dlouhodobějšího využití. Proto jsem zvolil model, při kterém se zařízení, které chce s kitem komunikovat, nejprve domluví na stylu a parametrech šifrování a až poté čte či zapisuje šifrovaná data.

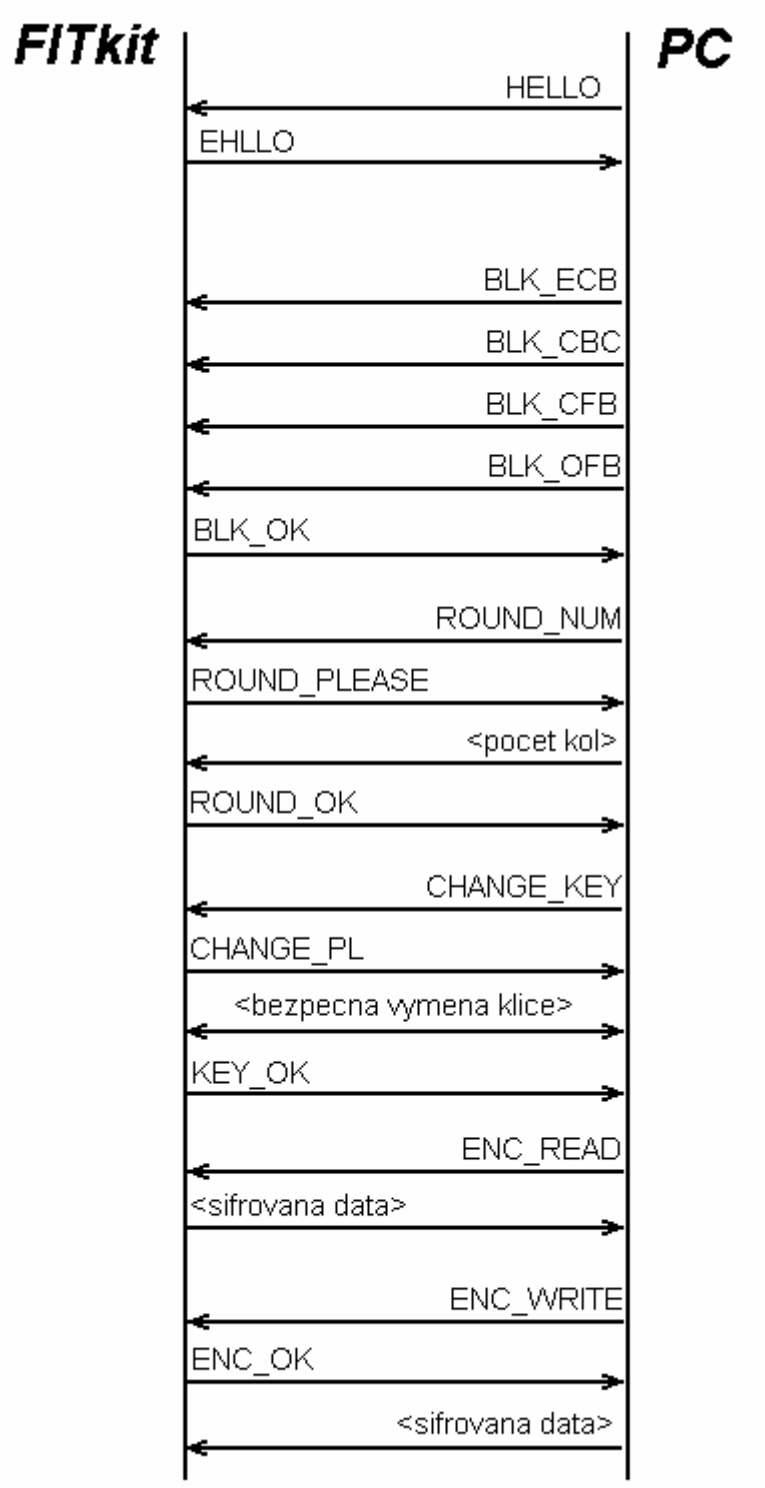
4.2.1 Popis protokolu

Protokol je na obou stranách zpracováván dle příkazu, který aktuálně přijde. Nezáleží na pořadí, ale je nutné projít všechna nastavení, aby mohlo být šifrování zahájeno. V případě neznámého příkazu nebo pokusu o zahájení šifrování s neúplným nastavením FITkit komunikaci končí.

Zde je vysvětlení jednotlivých příkazů při komunikaci:

Příkaz protokolu	Popis funkce	Vysílající strana
HELLO	Žádost o začátek komunikace	PC
EHLLO	Potvrzení komunikace	FITKIT
BLK_ECB	Mód šifry ECB	PC
BLK_CBC	Mód šifry CBC	PC
BLK_CFB	Mód šifry CFB	PC
BLK_OFB	Mód šifry OFB	PC
BLK_OK	Přijat mód šifry	FITkit
ROUND_NUM	Zahájení domluvy na počtu kol	PC
ROUND_PLEASE	Čekám na počet kol	FITkit
ROUND_OK	Počet kol přijat	FITkit
CHANGE_KEY	Zahájení výměny klíče	PC
CHANGE_PL	Čekám na výměnu klíče	FITkit
KEY_OK	Výměna klíče proběhla v pořádku	FITkit
ENC_READ	Čekám na šifrovaná data	PC
ENC_WRITE	Chci posílat šifrovaná data	PC
ENC_OK	Pošli data pro zápis	FITkit

Tabulka 1: Přehled příkazů navrženého protokolu



Obr. 6: Obraz komunikace mezi zařízeními navrženým protokolem

Jediným závažnějším problémem tohoto modelu je nutnost výměny šifrovacího klíče. Tomuto problému bude věnována následující podkapitola.

4.2.2 Výměna šifrovacího klíče

Tato výměna je obecným problémem všech algoritmů, které pracují na principu symetrické šifry. Říká se, že každý řetěz je tak pevný, jako jeho nejslabší článek. Pokud se nám nepodaří dostatečně ukrýt klíč, kterým jsou data zašifrována, tak je nám k ničemu sebelepší šifrovací algoritmus.

4.2.2.1 Shamirův algoritmus

Tento algoritmus umožňuje bezpečně vyměnit klíč mezi komunikujícími stranami bez toho, aniž by byl přenášen v otevřené podobě. Problémem tohoto algoritmu je fakt, že se musí použít komutativní šifra. To znamená, že pokud je zpráva dvakrát zašifrovaná, tak nezáleží na pořadí, ve kterém se zpráva dešifruje. Postup výměny klíče je následující:

1. Vezmu navrhovaný šifrovací klíč a zašifruji jej pomocí svého privátního šifrovacího klíče. Takto šifrovanou zprávu odešlu druhé straně.
2. Druhá strana přijme zprávu, zašifruje ji svým privátním klíčem a odešle ji zpět.
3. Přijatou zprávu odšifruji svým privátním klíčem. Nyní je již zpráva zašifrována jen klíčem druhé strany. Tuto zprávu odešlu druhé straně.
4. Druhá strana přijatou zprávu dešifruje svým privátním klíčem a tím získá klíč, který pro komunikaci navrhuji.

4.2.2.2 Diffie-Hellman protokol

Další algoritmus, umožňující bezpečně ustanovit šifrovací klíč pro symetrickou kryptografii, navrhli pánové Whitfield Diffie a Martin Hellman. Pokud by třetí strana odposlouchávala tuto výměnu, musela by umět pro získání klíče vypočítat diskrétní logaritmus. Má se však zato, že v současné době se jedná o neřešitelný problém. Krok po kroku by výměna klíče proběhla tímto způsobem:

1. Obě strany se dohodnou na velkém prvočíslu p a číslu q.
2. První strana si zvolí libovolné číslo x takové, že $1 \leq x < (p-1)$, a vypočítá $X = q^x \bmod p$, které pošle druhé straně.
3. Druhá strana si také zvolí libovolné číslo y takové, že $1 \leq y < (p-1)$, a vypočítá $Y = q^y \bmod p$, které pošle zpět první straně.
4. První strana nyní vypočítá $Y^x \bmod p$ a druhá strana $X^y \bmod p$, čímž obě strany získají šifrovací klíč.

4.2.2.3 RSA

V roce 1975 byl zveřejněn návrh způsobu výměny šifrovacích klíčů, který následně položil základy pro asymetrickou kryptografii. Navržený způsob probíhal následujícím způsobem:

1. První strana si vygeneruje dvojici klíčů - soukromý a veřejný klíč.
2. Soukromý klíč, kterým bude dešifrovat zprávy, musí držet v naprosté tajnosti. Naproti tomu veřejný klíč, jimž budou odesílatelé šifrovat jí určené zprávy, zpřístupní každému.

3. Pokud druhá strana bude chtít poslat první straně šifrovanou zprávu, nalezne si její veřejný klíč a s jeho pomocí zprávu zašifruje.
4. Jedině první strana, která vlastní odpovídající soukromý klíč, může zprávu správně dešifrovat.

5 Programová implementace

V této části práce bude popsána implementace šifrovacího procesu mezi počítačem a FITkitem. Kvůli omezené funkčnosti knihovny pro sériové rozhranní na straně kitu byly vytvořeny dvě sady programů, které budou popsány v následujících podkapitolách. Postup při spouštění programů a jejich přesný popis je uveden v příloze číslo 1 jako Manuál.

5.1 Komunikace FITkit – PC

Během prací na této bakalářské práci jsem zjistil, že stávající sériové rozhranní pro platformu FITkit je jen omezeně funkční. Průběžně jsme se snažili rozhranní opravit, ale ukázalo se, že rozhranní není optimálně navrženo a je naplánována jeho kompletní přestavba.

Omezení se týkají příjmu znaků do kitu, odesílání pracuje správně. Proto jsem vytvořil program, který vezme blok dat, který je uložený v paměti kitu, zašifruje jej dle přednastavených hodnot a odešle po sériové lince. Přednastavená hodnota, kterou může uživatel změnit, je blokový mód šifry. Tato hodnota je nastavena konstantou v hlavičce programu. Nedoporučuji měnit šifrovací klíč, protože se jedná o 32 bytovou hodnotu a jeho rozdílné nastavení by způsobilo špatné dekódování přijatých dat na straně počítače. Samozřejmě při správném (shodném) vložení klíče do obou programů funguje přenos správně. Dále nedoporučuji měnit počet šifrovacích kol, jsou nastavena podle standardu, který určuje jejich počet podle délky klíče.

Na druhé straně linky čeká program, který tato data přijme a dekóduje. Opět lze určit, jaký je blokový mód šifry a počet kol. Program má dvě textová pole. Do horního se vloží přímo text, který přišel po sériové lince, do spodního se přijatý text zobrazuje po dešifrování. Zobrazil jsem obě formy textu záměrně, aby byl zřejmý význam jednotlivých módů při šifrovacím procesu. Zejména doporučuji zkusit mód ECB, kde je pěkně vidět shoda výstupu šifrovaného textu při stejných blocích dat.

5.2 Komunikace PC – emulátor kitu

Kvůli omezení na straně kitu jsem vytvořil druhou sadu programů s emulátorem kitu, který běží na osobním počítači. V této sadě je provedena komunikace mezi počítačem a emulátorem v plném rozsahu, v jakém jsem chtěl provést komunikaci s FITkitem. Tato sada využívá protokol, který je popsán v předchozích kapitolách.

Program v počítači umožňuje plné nastavení šifrovacího procesu a rozhodnutí, zda se budou data z emulátoru kitu číst nebo zapisovat, samozřejmě v šifrované podobě. Dále je implementována bezpečná dohoda na šifrovacím klíči. Program po spuštění pracuje následovně:

- Zjistí, zda je dostupný zvolený sériový port. Pokud není, upozorní uživatele na chybu.
- Jakmile je spuštěna tlačítkem komunikace s kitem:
    - Naváže spojení s emulátorem.
    - Pošle emulátoru blokový mód šifry a počet šifrovacích kol.
    - Vymění si s emulátorem svůj šifrovací klíč.
    - Dle nastavení:
        - Odešle zašifrovaná data z formuláře „Data pro odeslání“.
        - Přijme a dešifruje data do formuláře „Přijatá data“.
    - Ukončí spojení.

6 Závěr

V mé bakalářské práci jsem se seznámil s různými šifrovacími metodami, zejména s aktuálním standardem AES. Prozkoumal jsem jeho funkci a z dostupných implementací vybral tu, která se nejvíce hodila pro prostředí vestavěných systémů. Tuto metodu jsem doplnil o proměnné nastavení rundovních klíčů a pro nastavení počtu kol, kterými otevřený text při šifrovacím procesu prochází. Pro realizaci programového vybavení na straně počítače jsem zvolil prostředí C++ Builder 6.0 od firmy Borland.

Jako komunikační linku jsem zvolil rozhranní RS232. Na straně FITkitu se ukázalo, že sériové rozhranní je sice implementováno, ale není plně funkční. Omezení se týkají zejména příjmu znaků do kitu, odesílání funguje korektně. V průběhu tvorby práce jsem se snažil s týmem podporujícím FITkit tento problém opravit, ale ukázalo se, že stávající model sériové komunikace není vhodně navržen a plánuje se jeho kompletní reimplementace. Nalezené chyby jsem předal osobě, která se o sériové rozhranní stará.

Díky nemožnosti implementace plného zadání mé práce jsem si rozšířil zadání o tvorbu emulátoru, který v komunikaci zastoupí FITkit. Tento emulátor běží na jiném osobním počítači. Jeho struktura je navržena tak, aby byl snadný přenos na platformu FITkit.

Kvůli omezeným možnostem komunikace je mezi FITkitem a počítačem implementován šifrovací program jen v jednom směru bez možnosti nastavení parametrů šifrování uživatelem. Po spuštění programu na kitu jsou zašifrována data a odeslána po sériové lince, kde je čte program na počítači. Ten má dvě okna, do horního napíše přijatý šifrovaný text a do spodního jeho dešifrovanou variantu.

V programu komunikujícím s emulátorem jsem implementoval všechny vlastnosti, které vycházejí ze zadání práce. Zejména je kladen důraz na možnost zvolit, zda se budou data z kitu číst, nebo zapisovat.

Pro komunikaci mezi zařízeními jsem navrhl jednoduchý protokol, který je na obou stranách zpracováván stavovým automatem. Protokol umožňuje ustavení spojení, určení stylu a počtu rundovních klíčů a možnost výběru, zda se bude z kitu číst nebo do něj zapisovat. Dále byla implementována bezpečná výměna klíčů Shamirovým protokolem, která je založená na operaci XOR.

Výsledná aplikace umožňuje komplexní nastavení parametrů sériového přenosu, nastavení parametrů šifrovacího (či dešifrovacího) procesu.

V aplikaci jsou dvě okna, horní slouží jako vstup dat pro odeslání na KIT a do spodního jsou přijímána data při čtení z kitu. Horní okno lze naplnit ze souboru a spodní okno lze do souboru uložit. Emulátor kitu je jen jednoduché rozhranní s nastavením parametrů sériové linky a spuštěním komunikace, nastavení šifrovacího procesu je dohodnuto již zmiňovaným protokolem.

Možných cest v pokračování na tomto projektu je několik, například přesun šifrovacího algoritmu do hardwarové části nebo implementovat lepší metodu pro dohodu na šifrovacím klíči – RSA. Z globálního hlediska by bylo možno aplikaci rozšířit o další šifrovací algoritmy, zejména proudovou a asymetrickou šifru či přesun komunikace na další rozhranní, která jsou na kitu implementována.

Literatura

[1] Bitto, O., Šifrování a biometrika. Kralice na Hané, ComputerMedia, 2005

[2] Pipper, F., Kryptografie, Praha, Dokořán, 2006

[3] Vondruška, P., Kryptologie, šifrování a tajná písma, Praha, Albatros, 2006

[4] Farana, R., Šifrování pro radost a poučení, Brno, Mravenec, 2006

[5] Virius, M., C++ Builder verze 5, Praha, Grada, 2000

[6] Matoušek, D., C++ Builder vývojové prostředí 1. díl, Praha, BEN, 2004

[7] Matoušek, D., C++ Builder vývojové prostředí 2. díl, Praha, BEN, 2003

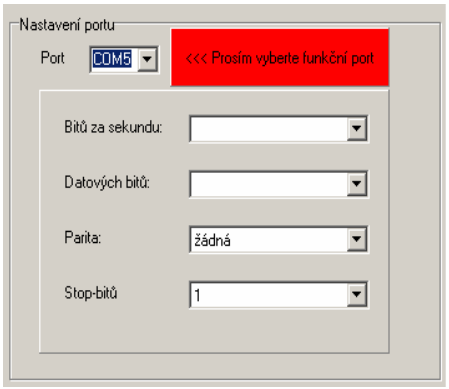
[8] Sedgewick, R., Algoritmy v C 1. díl, Praha, SoftPress, 2003

Seznam příloh

Příloha 1 – Manuál

Spouštění a ovládání aplikace mezi FITkitem a PC

1. Aplikace na osobním počítači
    1. Program je napsaný v prostředí C++ Builder 6.0. Buď zdrojové kódy znovu přeložíme nebo použijeme spustitelný soubor ze složky „EXE“.
    2. Zkontrolujeme sériové spojení kitu s osobním počítačem, tj. zda je v programu nastaveno správné číslo sériového portu. KIT komunikuje v rámci kabelu USB, který slouží k nahrávání programů do kitu. **Pro účely tohoto programu se připojte na port sloužící FPGA.** Pokud je nastaven špatný port, tak vpravo nahoře je zobrazeno červené upozornění. Jakmile zvolíme správný port, upozornění zmizí. Pozor, je testována pouze fyzická přítomnost portu, nikoli zda určený port slouží právě FPGA na kitu.



Obr. 7: Chybové hlášení při špatném nastavení portu

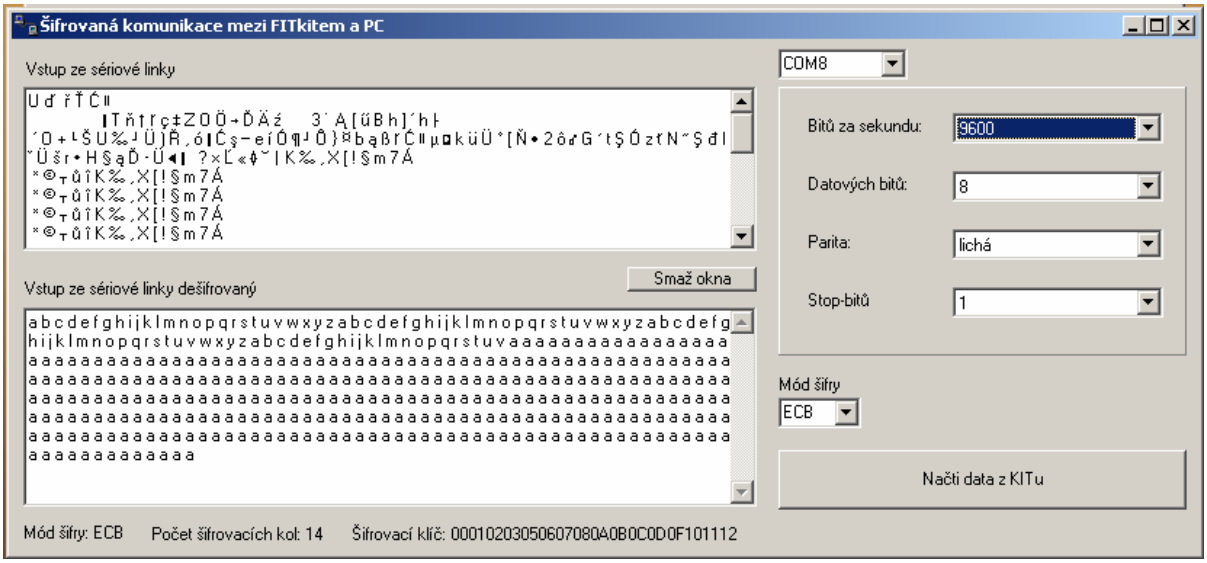
    3. Kliknutím na tlačítko „Načti data z kitu“ spustíme aplikaci.

2. Aplikace na FITkitu
    1. Složku se zdrojovými kódy nahrajeme do složky „APP“ v souborovém systému FITkitu.
    2. Obě složky nahrajeme do kitu standardním způsobem.
    3. **Velmi důležité je propojení pinu JP10(5) z FPGA s pinem JP9(26) do MCU.**
    4. Na terminálu, pomocí kterého jsme nahrávali kód z VHDL, si vypíšeme nápovědu pomocí příkazu „HELP“.
    5. Pomocí příkazu „TEST“ spustíme aplikaci.

Souhrnné pokyny

- Aplikace na FITkitu se chová jako server, aplikace na osobním počítači jako klient. Proto nejprve spusťte aplikaci na PC (tlačítko Načti data z kitu) a poté aplikaci na FITkitu (v terminálu příkaz TEST).
- Neměňte vlastní nastavení přenosu portu, měňte jen čísla portu.

Lze změnit blokový mód šifry, na straně PC změna nabídce „Mód šifry“, na straně kitu nutná změna ve zdrojovém kódu v sekci #define. Možné hodnoty jsou ECB, CBC, CFB, OFB.



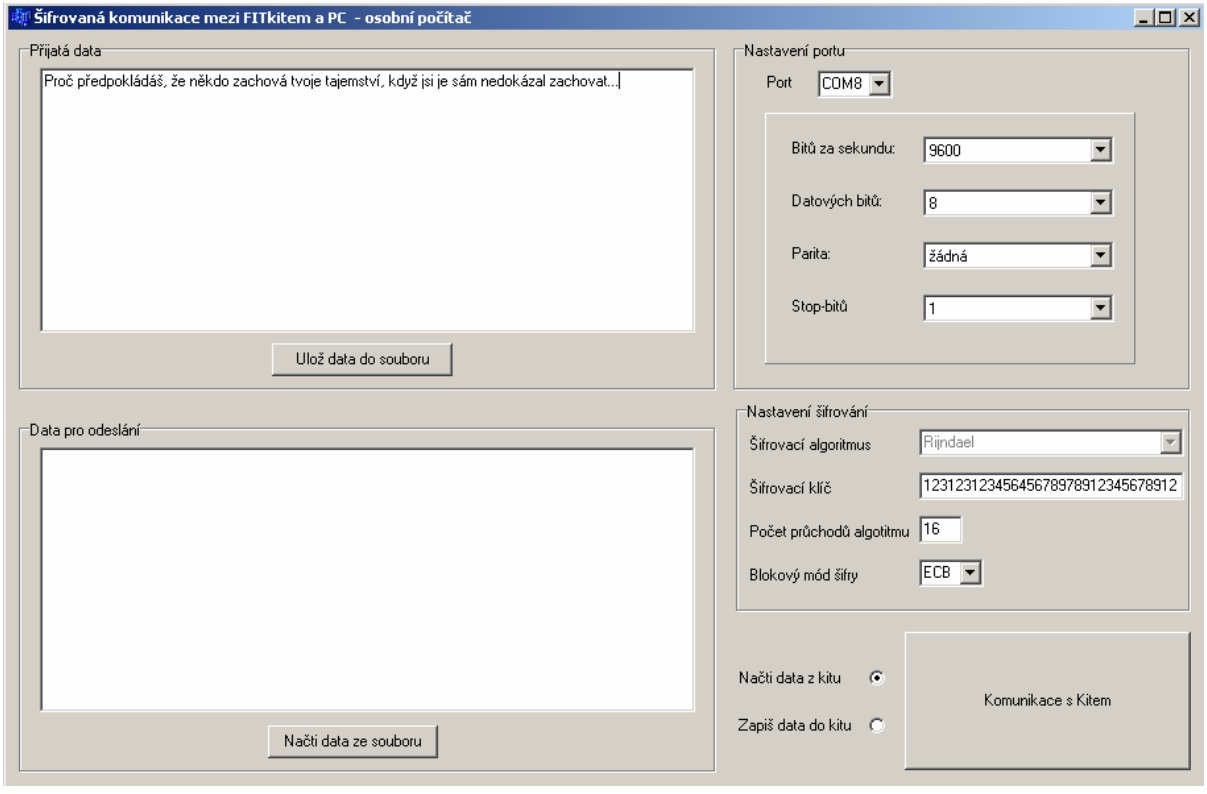
Obr. 8: Počítačová aplikace pro čtení a dešifrování dat přijatých z FITkitu

Postup při nesprávné funkci

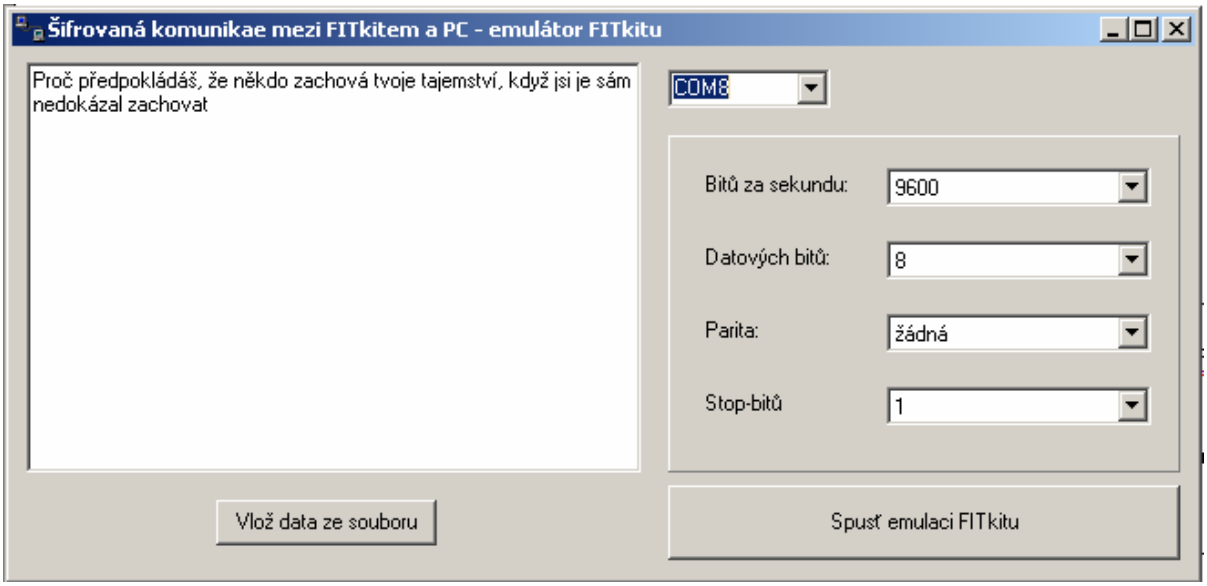
- Jsou správně propojeny piny na KITu?
- Je na straně počítače vybrán správný port?
- Dodrželi jste pořadí spouštění přenosu nejprve na počítači a až potom na kitu?
- Jsou na obou stranách nastaveny stejné módy šifry, počty šifrovacích kol, šifrovací klíče?

Spouštění a ovládání aplikace mezi PC a emulátorem kitu běžícím na PC

1. Obě aplikace jsou napsány v prostředí C++ Builder 6.0. Buď zdrojové kódy znovu přeložíme, nebo použijeme spustitelný soubor ze složky „EXE“.
2. Počítače, na kterých jsme aplikace spustili, propojíme sériovým kabelem (LapLink).
3. Hlavní aplikace se chová jako server, emulátor je klient. Při komunikaci vždy nejprve spouštíme emulátor (tlačítko „Spusť emulaci FITkitu), až poté spouštíme komunikaci u hlavní aplikace (tlačítko „Komunikuj s kitem“).
4. Veškerá nastavení šifrování se provádějí na straně hlavní aplikace.
5. Pokud si přejeme číst data z kitu, zaškrkneme volbu „Načti data z kitu“. Data budou načtena do horního formuláře pro přijatá data. Tato data lze uložit do souboru tlačítkem „Ulož data do souboru“. Pokud jsme v programu, který emuluje kit, vložili libovolná data do formuláře, budou do hlavní aplikace přijata právě tato data. Pokud jsme tak neučinili, budou načtena přednastavená data.
6. Pokud si přejeme do kitu zapisovat, zaškrkneme volbu „Zapiš data do kitu“. Do kitu budou odeslána data z formuláře „Data pro odeslání“. Tento formulář jde naplnit ze souboru tlačítkem „Načti data ze souboru“. Pokud je formulář prázdný, odešleme přednastavená data.



Obr. 9: Hlavní program aplikace



Obr. 10: Program emulující FITkit

Příloha 2

Typické užití knihovny CRYPT v jazyce C

```
#include <stdio.h>
#include "crypt.h"        //hlavičkový soubor knihovny
#include "crypt.c"        //knihovna CRYPT

Cryp1_TCryptParam cp;          //proměnná pro předvýpočet parametrů pro šifrování
TData128 iv = {0, 0, 0, 0};       //inicializační vektor

int main(void) {
   byte pt1[16] = {0x50, 0x68, 0x12, 0xA4, 0x5F, 0x08, 0xC8, 0x89, 0xB9, 0x7F, 0x59, 0x80, 0x03,
   0x8B, 0x83, 0x59 };             //data pro zašifrování
   byte pt2[16], ct[16];             //ct-šifrovaná data, pt2-dešifrovaná data
   char key[33] = "000102030506070 80A0B0C0D0F101112\x0";          //šifrovací klíč

   Cryp1_SetKeyHex(key, &cp);          //zpracování klíče do formátu pro šifrování
   Cryp1_InitKey(iv, &cp);                //předvýpočet klíčů pro každé kolo šifrování

   Cryp1_Encrypt(pt1, ct, &cp, 16, OFB);    //šifrování (co, kam, šifrovací parametry, počet kol, mód)
   Cryp1_Decrypt(ct, pt2, &cp, 16, OFB);    //dešifrování

   printf("\n********Rijndeal Algoritmus*******\n");  //výpis vstupů a výstupů šifrování
   printf("\npt1: ");         //původní text
   for (i=0;i<16;i++) printf(" %d",pt1[i]);
   printf("\nct : ");         //zašifrovaný text
   for (i=0;i<16;i++) printf(" %d",ct[i]);
   printf("\npt2: ");         //dešifrovaný text
   for (i=0;i<16;i++) printf(" %d",pt2[i]);

   printf("\n\nSifrovaci klic: %s",key);

   return 0;
}
```