Přírodovědecká fakulta Masarykovy univerzity

Bakalářská práce

Kryptografie v digitálním televizním vysílání

Miroslav Šemora

Vedoucí práce: doc. RNDr. Jan Paseka, CSc.

Prohlášení:
Prohlašuji, že jsem tuto bakalářskou práci vytvořil samostatně, pouze s použitím zdrojů uvedených v referencích.

V Brně dne 25. 5. 2005, Miroslav Šemora

Poděkování:
Na tomto místě bych rád poděkoval doc. RNDr. Janu Pasekovi, CSc. za cenné připomínky a rady během psaní, ing. Zdeňku Sehnalovi za zajímavé informace a postřehy z praxe a MgrA. Blance K. Špičákové za ruční vazbu této práce.

Obsah

Kapitola 1

Televizní vysílání a kryptografie

1.1 Účel kryptografie v televizním vysílání

Kryptografie má mnoho funkcí. Především však slouží k *utajení* informace při přenosu k příjemci, k zabezpečení před *nežádoucí modifikací* během přenosu a konečně k ověření původu – *autentizaci*.

Televizní vysílání spočívá v přenosu obrazu a zvuku k divákovi. Zde je primárním úkolem šifrování *ochrana duševního vlastnictví*. Pomocí systémů jako je *Subscription-TV* nebo *(Impulse)-Pay-per-View* je zajištěno, že vysílání může sledovat jen divák, který si jej předplatil, popřípadě umožňuje platbu za jednotlivé shlédnuté pořady.

1.2 Druhy televizního vysílání

Podle způsobu přenosu můžeme televizní vysílání rozdělit na *pozemní*, *kabelové* a *satelitní*. Ve všech třech případech dochází na straně vysílače k namodulování obrazu a zvuku na nosný elektromagnetický kmitočet a u koncového uživatele k jeho demodulaci a přehrání. Způsob přenosu bude pro nás nadále nepodstatný, jen poznamenejme, že v případě kabelového vysílání se nepoužívají šifrovací metody, nýbrž se provádí jen nějaká forma filtrování signálu podle toho, co si uživatel předplatí. Naopak v satelitním vysílání by takovéto filtrování bylo neúčinné.

Podle formátu dat rozlišujeme vysílání na *analogové* a *digitální*. V pozemním analogovém se šifrování prakticky nikdy nevyskytovalo, v satelitním analogovém se jednalo (a stále ještě někde jedná) o různé úpravy signálu, které zmíníme jen pro dokreslení situace.

1.3 Šifrování analogového vysílání

Televizní signál sestává ze zvukových a obrazových dat. Obraz je rozdělen do řádků a řádky pak do jednotlivých bodů. Body nesou informaci o barvě a jejich sestavením vznikne výsledný obraz. V analogovém případě na straně vysílače dochází k *modulaci*, kdy na tzv. *nosný* elektromagnetický signál vysoké frekvence jsou v pravidelných intervalech naneseny informace o jednotlivých obrazových bodech. Provádí se to úpravou síly (tzv. *amplitudová* modulace, AM) vlnění nebo jeho frekvence (*frekvenční* modulace, FM).

Obdobně zvuk, což je chvění vzduchu v určitých frekvencích. Elektronicky je zaznamenán jako střídavý nepravidelný signál, vzniklý – zjednodušeně řečeno – „sečtením“ mnoha sinusoid. Tvar signálu je namodulován do vysokofrekvenčního elektromagnetického vlnění, avšak jiné vlnové délky než jemu příslušný obraz.

Na straně přijímače pak dochází k demodulaci, tedy odečtení signálu z nosného vysokofrekvenčního vlnění.

Zde byl prostor pro „kryptografii“. Každý půlsnímek obrazu analogové televize obsahuje synchronizační signál, řídicí signály a vlastní obrazová data. Provádělo se například odfiltrování synchronizačního signálu nebo inverze řídicího i obrazového signálu a jejich nejrůznější kombinace. Přijímač o těchto změnách věděl a dokázal je napravit.

Obrázek 1.1: Analogový televizní signál – jeden půlsnímek

K odhalení takovéto úpravy signálu však stačil osciloskop a docházelo k němu často. Provozovatelé se bránili různým obměňováním způsobu zakódování signálu a jeho častějším střídáním. Pro nás je takové šifrování zajímavé možná právě jen jako demonstrace neúčinnosti *security by obscurity*, tedy systému, kdy dešifrování je založeno jen na znalosti kryptovací metody (a nanejvýš na vyzkoušení několika jednoduchých „klíčů“).

Trochu dokonalejší pak byly systémy *hybridní*, představované systémem *VideoCrypt* [VC-97]. Obraz se pořád přenáší analogově, avšak zašifrovaně a informace o šifře se přenáší digitálně na nevyužité frekvenci původně určené pro zvukový kanál. Zde se již využívá pseudonáhodných posloupností, které říkají, jak jsou obrazové půlsnímky jsou zakódovány.

Odkazy: [DRM-02, strana 57-68]

1.4 Digitální televizní vysílání

Mnohem zajímavější pak je šifrování digitálního vysílání, neboť, jak v následujícím uvidíme, umožňuje šifrování pro široké spektrum účelů.

Vývojem standardů pro digitální vysílání v Evropě se zabývá konsorcium *Digital Video Broadcasting Project (DVB)*, který sdružuje výrobce a poskytovatele služeb. Jednotlivé větve vývoje nesou označení DVB-T (terrestrial) pro pozemní, DVB-S pro satelitní vysílání a podobně. DVB definuje hlavně *technické* standardy, mezi nimi formát vysílaných dat a způsob jejich zašifrování před nežádoucím příjemcem, tzv. *Conditional Access* [WWW-CA].

V následujícím textu naznačím, v jakém formátu takové vysílání probíhá. Zařízení pro digitální snímání obrazu obecně nejdříve produkují *nekomprimovaný* záznam složeny z jednotlivých snímků *(frames)*. Snímky pak obsahují údaje o barvě a světelnosti jednotlivých bodů – *pixelů*. Například formátem, který produkuje webová kamera, je YV12, viz [Web-YV].
Ten lze uložit na disk nebo poslat po síti *(streamovat)* na jiný počítač.

Obraz v tomto formátu obsahuje ale obrovské množství dat a proto nastupuje *komprese*, kdy dochází zjednodušeně řečeno k

- vyhledávání a odstranění redundantních informací,
- záměrnému vynechávání některých dat tak, aby lidské oko zaznamenalo co nejmenší odchylku od původního nekomprimovaného obrazu.

Existuje nepřeberné množství kompresních algoritmů, řada z nich je pak chráněna kontroverzními, a proto dnes tolik diskutovanými *patenty*. Stejně jako u obrazu, i zvuk je komprimován, byť potřeba zde není tak akutní jako u obrazu. Rozdíl velikostí je patrný u známého formátu *MP3* oproti nekomprimovanému *wav*.

Obraz i zvuk jsou nakonec rozděleny do *paketů* a pakety střídavě spojeny do *transportního proudu (transport stream)*, který je možno přenášet k divákovi. Specifikace DVB určují jako formát komprese obrazu a zvuku *MPEG-2*. *MPEG (Moving Picture Experts Group)* zahrnuje množství standardů pro kódování audiovizuálních informací v digitálním komprimovaném formátu [Web-MPEG]. MPEG-2 je pak jednou z konkrétních norem, která se uplatní v řadě aplikací [FAQ-MPEG2].

Transportní proud kromě paketů s audiovizuálními daty však ještě obsahuje řídicí data, tzv. *Control Messages*. V nich se přenášejí údaje pro *Conditional Access (CA)*. Conditional Access specifikuje, jak jsou data zašifrována a který ze zákazníků má právě možnost ten který kanál sledovat. Formát transportního proudu, který používají systémy DVB se nazývá *MPEG Transport Stream* [MPEG-TS].

1.5 Conditional Access

Conditional Access je označení pro proces šifrování audiovizuálních dat pro digitální televize za účelem autorizace příjmu vysílání. Šifrování probíhá v několika rovinách, jež budou popsány v tomto oddíle.

1.5.1 Scrambling

Vlastní zašifrování audiovizuálních dat se nazývá *scrambling*. Provede se ještě před konečným spojením dat do transportního proudu. Dešifrování na straně příjemce se označuje jako *descrambling*.

Scrambling a descrambling se musí v reálném čase provádět pro velké množství audiovizuálních dat a algoritmus musí být proto velmi rychlý. Scrambling je příklad *symetrické* šifry, kdy se na šifrování i dešifrování použije stejný klíč.

Aby nedocházelo k častému prolamování šifry, klíč pro scrambling a descrambling se obměňuje. To je v praxi realizováno tak, že v přístroji příjemce je současně více klíčů a ty se v pravidelných intervalech (řádově desítky sekund) mění. Jakmile divák vytáhne kartu z přístroje, do deseti sekund mu obraz zmizí, neboť nemá klíč pro jeho dešifrování.

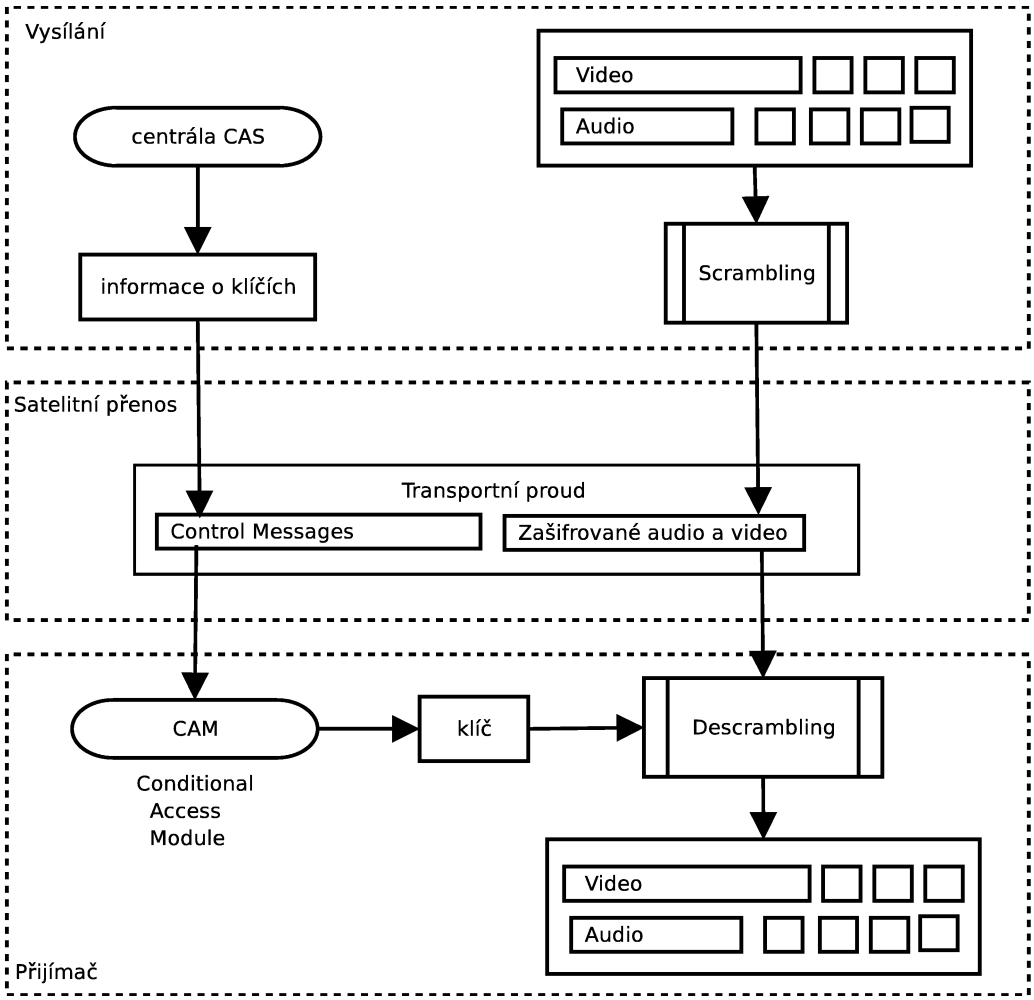
Standard pro scrambling používaný prakticky všude v Evropě je definován normou *ETSI – European Telecommunications Standards Institute* [ETSI] a nazývá se *DVB Common Scrambling Algorithm – CSA*.

1.5.2 Control Messages

Klíč pro scrambling/descrambling se nazývá *control word*. Vysílání pak probíhá tak, že v přijímači je současně uloženo více klíčů *(control word)* pro descrambling a v krátkých intervalech je vybírán vždy jeden z nich. Výběr provede vysílací centrála a pošle spolu s audiovizuálními daty transportním proudem přijímači v podobě tzv. *řídicích zpráv (control messages)*. Zprávy, kterými se přenášejí informace o právě použitém klíči, se nazývají *EMM – Entitlement management message*. Data v EMM zprávách jsou již šifrovány „doopravdy“, algoritmem s nízkým rizikem prolomení za cenu delšího šifrování.

Přijímající strana potom v závislosti na kanále může a nemusí být oprávněna přijímat daná audiovizuální data a v tom případě dojde nebo nedojde k dešifrování informace, který klíč *(control word)* má být použit.

Klíče pro descrambling jsou navíc v pravidelných delších časových intervalech obměňovány přímo v přístroji. Příkaz k takové obměně – tentokrát spolu



Obrázek 1.2: Conditional Acceses System – Přenos šifrovaných dat

s klíčem – pošle centrála transportním proudem opět v podobě zašifrované kontrolní zprávy. Ta se nazývá *ECM – Entitlement Control Message*.

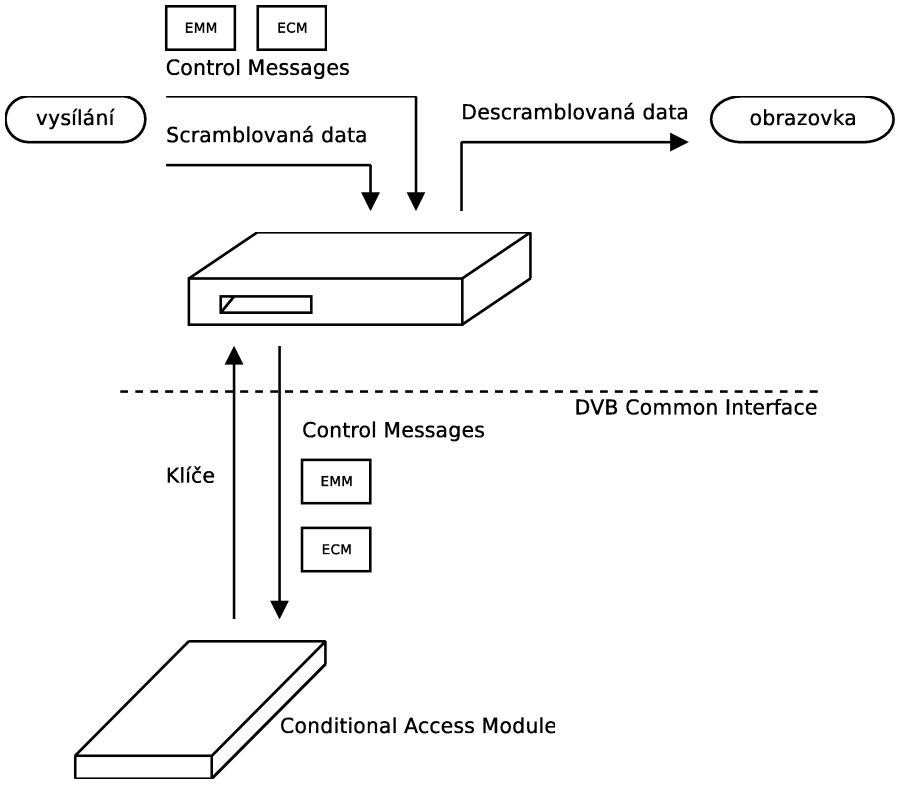
1.5.3 DVB CA Common Interface

Konkrétní implementace Conditional Access se označuje jako *Conditional Access System*. Existuje mnoho Conditional Access systémů od různých výrobců. Aby byla zajištěna interoperabilita, tj. aby poskytovatelé televizního vysílání mohli používat různé CA systémy, byly i zde vyvinuty (a samozřejmě stále se vyvíjejí) standardy.

DVB definuje jednotné rozhraní, tzv. DVB CA Common Interface. To

umožňuje aby uživatel nebyl při koupi přístroje vázán na konkrétní implementaci Conditional Access, a tedy aby měl možnost sledování vysílání různých poskytovatelů. Je-li zařízení pro příjem signálu vybaveno slotem kompatibilním s DVB CA Common Interface, může uživatel pouhou výměnou kryptografického modulu využívat služby chráněné jiným systémem CA. Kryptografický modul kompatibilní s DVB CA Common Interface se nazývá *Conditional Access Module – CAM.*

Samotné descramblování se provádí uvnitř přístroje a je pro veškeré digitální vysílání rámci normy DVB univerzální. Přístroj po kryptografickém modulu (Conditional Access Module) nechce nic jiného, než klíč, kterým má audiovizuální data descramblovat. Přitom mu předává Control Messages z transportního proudu, aniž by ho jejich obsah nějak zajímal. Rozhraní DVB Common Interface tedy specifikuje, jakým způsobem přístroj předá Control Messages a dostane klíč. Samotný obsah Control Messages a správa klíčů je plně v režii dodavatele systému pro Conditional Access.



Obrázek 1.3: Výměna informací mezi přístrojem a Conditional Access Module

Tento fakt však s sebou nese velké potenciální riziko. Čas od času dojde

k prolomení některého systému pro Conditional Access. To s sebou může nést nutnost výměny modulů. Pokud by došlo k prolomení samotného CSA, bylo by nutné obměnit všechny přístroje, neboť všichni broadcasteři v Evropě používají stejnou šifru – Common Scrambling Algorithm. CSA samotný dosud prolomen nebyl, vždy selhaly jeho implementace nebo implementace Conditional Access systémů.

Standardu DVB CA Common Interface vyhovuje i systém *Cryptoworks*, na který se v následujícím zaměříme.

Odkazy: [DRM-02, strana 69-78]

1.5.4 Cryptoworks DVB CA

Systém *Cryptoworks DVB CA* vyvinutý firmou *Royal Philips Electronics* zatím jako jediný dosud odolal všem snahám o prolomení. Díky své flexibilitě se kromě televizního vysílání (tedy Conditional Access) dobře uplatní i v dalších oblastech, jako jsou bankovní transakce nebo autorizace v počítačových sítích. Cryptoworks u nás využívá například vysílání *UPC Direct*.

Kryptografický modul systému Cryptoworks je vybaven terminálem pro smartkartu. **Smartkarta** *(Smartcard)* je karta velikosti klasické bankovní karty, která obsahuje samostatný čip s procesorem [FAQ-SC, ALT-SC]. Karta Cryptoworks může současně držet sady klíčů pro různé služby různých poskytovatelů a různé kryptografické techniky či šifrovací algoritmy.

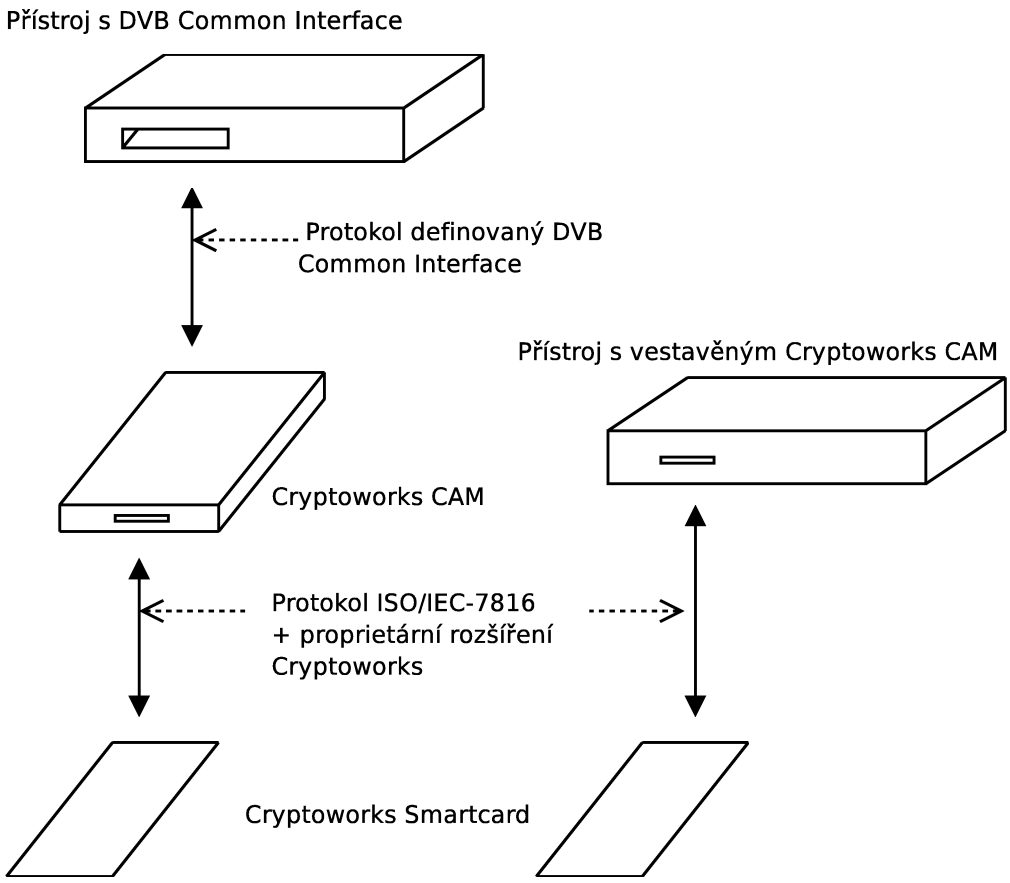
V případě digitálního vysílání jde o klíče pro descrambling a implementaci v podobě Conditional Access Module s rozhraním DVB Common Interface. Philips dodává ovšem i levné přístroje bez DVB Common Interface pouze s vestavěným dekódovacím systémem Cryptoworks – příkladem je *Philips DSX 6010* [Philips], který se standardně dodává se službou UPC Direct. Ten má vstup jen pro smartkartu Cryptoworks. Smartkarta komunikuje s CAM protokolem podle normy ISO/IEC-7816 [Web-CAM], avšak jsou zde některé proprietární (specifické pro výrobce a nezveřejněné) odchylky.

Na smartkartě jsou uloženy klíče pro dekódování (descrambling) digitálního obrazového a zvukového signálu. Centrála je čas od času vyměňuje zasláním ECM. Intervaly výměn jsou typicky co měsíc. Pro digitální televizní vysílání bývá na kartě v jednom okamžiku uloženo více klíčů. V pravidelných intervalech během vysílání se střídá klíč, kterým je obraz a zvuk zašifrován a centrála pošle informaci o změně prostřednictvím EMM. Proto pokud uživatel vytáhne kartu z přístroje, příjem do několika desítek vteřin ustane.

Systém je velmi propracovaný. Disponuje škálovatelným adresovacím schématem a control messages lze adresovat všem kartám, jednotlivé kartě i různým sdíleným prostorům karet včetně geografických, pro služby typu pay-per-view a video-on-demand nabízí preview funkce apod. [Web-CAM,

LET-CAM] Je nezávislý na šifrovací technologii, kterou přenáší. Nás nyní bude zajímat především Common Scrambling Algorithm.

Co bude pro nás podstatné, je způsob komunikace kryptografického modulu s centrálou. Veškeré citlivé informace jsou při přenosu chráněny šifrovacím algoritmem RSA – zejména klíče pro descrambling. Pro realizaci výše zmíněných funkcí a vlastností systému je na každé kartě uložena sada RSA klíčů. Klíče nikdy neopustí kartu. Karta dostává a vydává zprávy pro centrálu v podobě instrukcí zašifrovaných RSA – tento protokol však není veřejně zcela znám. Například pošle-li vysílací centrála ECM s novým klíčem, CAM ji předá kartě jako instrukci uvozenou kódem `A4 48 00 00` (přidání/smazání záznamu na skupině karet) a následovanu blokem dat šifrovaných RSA. Control word uložený v datech je stále ještě zašifrovaný RSA. Při descramblingu si přístroj přes DVB Common Interface vyžádá control word od CAM. CAM si jej vyžádá od karty, která odpoví instrukcí s klíčem, pokud má karta příslušné oprávnění. Veškeré RSA šifrování a dešifrování probíhá na kartě, tedy kartu v tomto okamžiku už opouští control word nechráněný RSA – viz [CWFaq]



Obrázek 1.4: Znázornění funkce systému Cryptoworks DVB CA

Kapitola 2

Základní pojmy kryptografie

2.1 Šifrovací algoritmus, zpráva, klíč, kryptogram

Základní metoda utajení dat před neautorizovaným příjemcem se nazývá **šifrování**. Původní data nazýváme v kryptografii **zpráva**, data v utajené podobě **kryptogram**. Převedení kryptogramu zpět na zprávu se nazývá **dešifrování**. Kromě utajení před neautorizovaným příjmem se kryptografie zabývá též *ochranou před nežádoucí modifikací kryptogramu*, *autentizací odesilatele* apod.

Postup, jakým se zpráva převede na kryptogram označujeme jako **šifrovací algoritmus** nebo jen **šifra (cipher)**. Aby byla data utajena před neautorizovaným příjemcem, používá šifrovací algoritmus datový údaj, který je známý jen chtěnému příjemci. Ten se nazývá **klíč**. Jedno ze základních pravidel kryptografie je, aby utajení záviselo jen na klíči a nikoli na znalosti šifrovacího algoritmu.

2.2 Symetrická a asymetrická šifra

Symetrická šifra používá stejný klíč pro šifrování i dešifrování a matematicky ji lze popsat jako funkci E *(encryption)*

$$C = E(K, M),$$

kde E je šifrovací funkce *(encryption)*, K je klíč *(key)*, M zpráva *(message)* a C kryptogram *(ciphertext)*. Dešifrování je pak funkce D *(decryption)*

$$M = D(C, K),$$

kde D je inverzní funkce k E a převede data do původní podoby.

Zmínili jsme šifrovací a dešifrovací funkce E a D. Aby mělo šifrování smysl, musíme dešifrováním kryptogramu obdržet původní zprávu, tedy

$$D(E(K, M)) = M.$$

Tomuto požadavku říkáme **korektnost** algoritmu.

Asymetrická šifra používá jiný klíč pro zašifrování a jiný klíč pro dešifrování, mluvíme o *dvojici klíčů (key pair)*. K zašifrování se použije **veřejný (public)** klíč, k dešifrování **soukromý (private)** neboli **tajný (secret)** klíč. Někdy též mluvíme o jednom klíči a jeho *veřejné* a *soukromé* části. Veřejný klíč se použije pro zašifrování a z jeho znalosti nelze odvodit soukromý klíč a tedy ani dešifrovat zprávu z kryptogramu. Může být veřejný v plném významu slova a jej může mít každý, aniž by šifrou chráněná data byla nějak ohrožena. Šifrování a dešifrování probíhá takto:

$$C = E(K_{pub}, M),$$

$$M = D(K_{priv}, C).$$

Korektnost asymetrické šifry zapíšeme jako

$$D(K_{priv}, E(K_{pub}, M)) = M.$$

Nejčastějším případem dvojice klíčů je ten, kdy soukromým klíčem jsou dvě velká prvočísla a veřejným klíčem je součin těchto prvočísel. Problém nalezení privátního klíče z veřejného je tedy problém nalezení rozkladu celého nezáporného čísla na prvočinitele, **faktorizace**. Faktorizace velkého součinu je časově tak náročná, že přesahuje možnosti současné výpočetní techniky.

Výhody asymetrického šifrování proti symetrickému lze shrnout do dvou bodů:

- Přenos klíče při symetrickém šifrování je vždy spojen s určitým nepohodlím a rizikem. V případě asymetrické šifry příjemce jednoduše vygeneruje dvojici klíčů a její veřejnou část pošle bez obav a bez potřeby utajení.

- Množství klíčů potřebných ke komunikaci se rapidně snižuje. Při symetrické komunkaci mezi n stranami „každý s každým“ je třeba $n(n-1)/2$ klíčů, kdežto pro symetrickou jen $2n$ klíčů, přesněji n párů (veřejný klíč, tajný klíč).

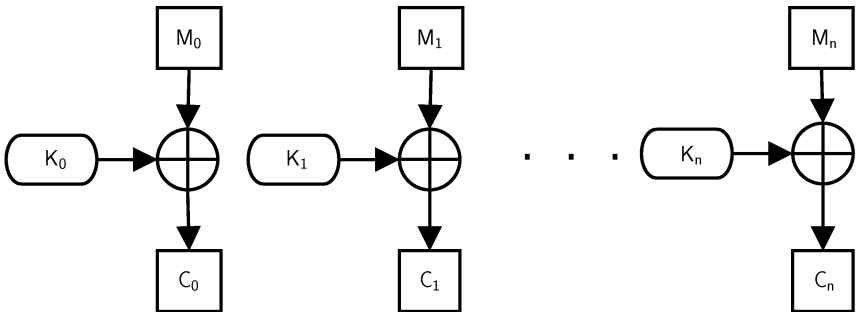
Podstatnou nevýhodou asymetrických šifer je však *časová náročnost* šifrování a dešifrování, proto se často používá kombinace asymetrické a symetrické šifry. U protokolu *SSH* počítače nejdříve naváží asymetricky šifrované spojení (*RSA* nebo *DSA*) a jím si pošlou ad hoc vygenerované klíče pro symetrickou šifru *DES*, jíž je chráněn zbytek spojení. Stejně tak u DVB se asymetrickou šifrou RSA přenáší control word pro symetrickou šifru *Common Scrambling Algorithm*.

2.3 Operace xor a one-time pad

Použijeme-li v šifře jako šifrovací funkci *xor* zprávy s klíčem, vznikne tzv. **one-time pad**. **xor** je logická operace „exkluzivní nebo *(exclusive or)*“, tedy

$$xor(0,0) = 0,$$
$$xor(1,0) = 1,$$
$$xor(0,1) = 1,$$
$$xor(0,0) = 0.$$

V souladu s konvencí budeme xor na schématech i v textu značit $\oplus$:



Obrázek 2.1: One time pad xoruje každý bit zprávy jedním bitem klíče

Řekněme, že máme zprávu $M = (0,1,1,0,1,0,0,1)$ a klíč $K = (K_0, \ldots, K_7) = (1,1,0,1,1,0,1,1)$, potom

$$C = M \oplus K = (1,0,1,1,0,0,1,0).$$

Opětovným xorováním kryptogramu a klíče vznikne původní zpráva:

$$M = C \oplus K = (0,1,1,0,1,0,0,1).$$

Tedy algoritmus je *korektní*, což můžeme zapsat jako

$$K \oplus (K \oplus M) = M.$$

Operace $\oplus$ je komutativní, neboť $0 \oplus 1 = 1 \oplus 0$, tedy $a \oplus b = b \oplus a$, asociativní, neboť

$$\begin{aligned}
&(0 \oplus 0) \oplus 0 = 0 \oplus 0 = 0 \oplus (0 \oplus 0)\\
&(1 \oplus 0) \oplus 0 = 1 \oplus 0 = 1 = 0 \oplus 1 = 1 \oplus 0 = 1 \oplus (0 \oplus 0)\\
&(1 \oplus 1) \oplus 0 = 0 \oplus 0 = 0 = 1 \oplus 1 = 1 \oplus (1 \oplus 0)\\
&(0 \oplus 0) \oplus 1 = 0 \oplus 1 = 1 = 0 \oplus 1 = 0 \oplus (0 \oplus 1)\\
&(1 \oplus 0) \oplus 1 = 1 \oplus 1 = 1 \oplus (0 \oplus 1)\\
&(1 \oplus 1) \oplus 1 = 0 \oplus 1 = 1 \oplus 0 = 1 \oplus (1 \oplus 1).
\end{aligned}$$

Je-li délka klíče při one-time pad větší nebo rovna délce zprávy, poskytuje systém tzv. *perfektní bezpečnost*, avšak za cenu nutnosti výměny velkých klíčů. V praxi se proto k zašifrování používají *pseudonáhodné posloupnosti* vygenerované z klíčů [Laws, Paseka]. Navíc klíč je možné použít jen jednou, neboť pokud se útočník zmocní dvojice (zpráva, kryptogram), prostým xorováním dostane klíč ($K = M \oplus C$).

2.4 Bloková šifra

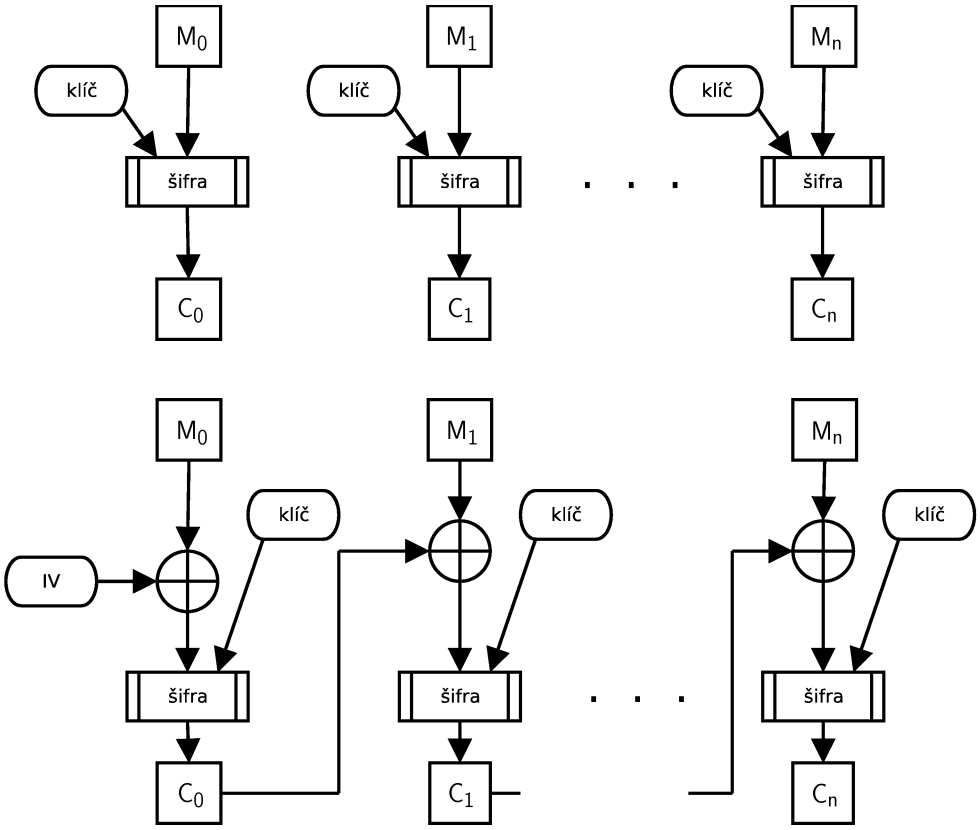
Šifru označíme jako **blokovou**, jestliže zprávu rozdělí na stejně velké úseky a ty postupně zpracovává do bloků téže velikosti [Klíma-03]. Mějme zprávu $M = (M_0, ..., M_N)$ o n blocích. Pak šifrujeme

$$C_0 = E(M_0), \ldots, C_n = E(M_n)$$

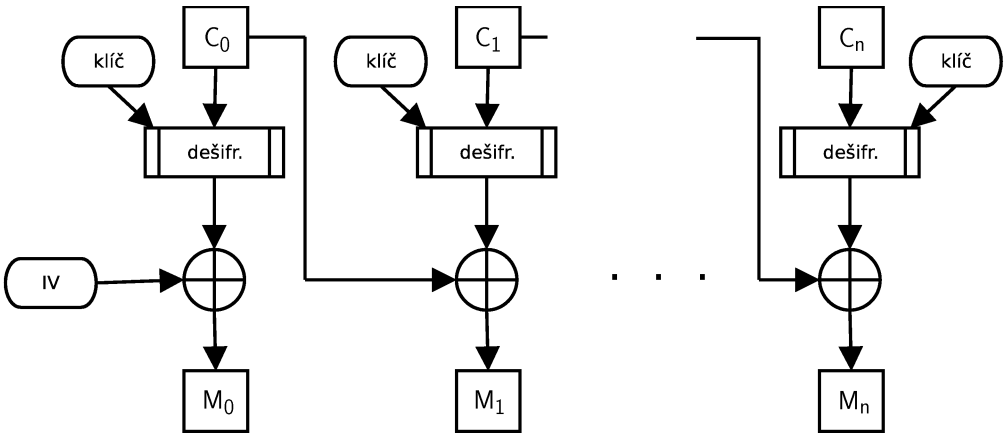
a kryptogram dešifrujeme

$$M = (M_0, \ldots, M_n) = (D(C_0), \ldots, D(C_n)).$$

Tento způsob kódování se nazývá *ECB mód – elektronická kódová kniha (Electronic Codebook Mode)*. Má ovšem tu nevýhodu, že stejné bloky kryptogramu znamenají i stejné úseky původní zprávy. Útočník pak může zprávu kromě dešifrování i nežádoucím způsobem modifikovat, například výměnou nebo vynecháním bloků. Proto se častěji používá *zřetězení šifrovaného textu – CBC (cipher block chaining)*. Zde se v každém kroku šifruje původní text



Obrázek 2.2: Bloková šifra v ECB a CBC módu



Obrázek 2.3: Dešifrování blokové šifry v CBC módu

xorovaný s předchozím zašifrovaným textem, přičemž první blok je xorován s tzv. *inicializační hodnotou (IV – initial value)*:

$$\begin{aligned} C_0 &= E(K, IV \oplus M_0)), \\ C_1 &= E(K, C_0 \oplus M_1), \\ &\dots \\ C_n &= E(K, C_{n-1} \oplus M_n)). \end{aligned}$$

Inicializační hodnota může být součástí klíče, častěji je ovšem přiložena ke kryptogramu. Dešifrování potom vypadá takto:

$$\begin{aligned} M_0 &= IV \oplus D(K, C_0)), \\ M_1 &= C_0 \oplus D(K, C_1), \\ &\dots \\ M_n &= C_{n-1} \oplus D(K, C_n)). \end{aligned}$$

2.5 Proudová šifra

Proudovou označíme šifru tehdy, jestliže zprávu šifruje postupně bit po bitu a k zašifrování používá pseudonáhodnou posloupnost vytvořenou z klíče. Takovouto pseudonáhodnou posloupnost označujeme jako *heslo*. Jednoduchým příkladem je *Vigenerova šifra*. Zde se však jedná o nejtriviálnější případ –

klíč je prostým způsobem opakován a kryptogram je poměrně snadno rozluštitelný hledáním délky klíče *Kasiského testem* [Paseka, strana 28]. Další velmi známá proudová šifra je například algoritmus *RC4*.

Samotné šifrování nejčastěji spočívá v prostém *xorování* původní zprávy s heslem.

Skutečnost, že stejná zpráva se stejným klíčem zašifruje do stejného kryptogramu, se stejně jako v případě blokové šifry eliminuje použitím *inicializační hodnoty (nonce – number used once)*, která potom je součástí klíče nebo kryptogramu.

Při generování pseudonáhodné posloupnosti se velmi často používají *posouvací registry*.

2.6 Posouvací registry

Posouvací registr je posloupnost bitů konečné délky, která se v jednotlivých krocích výpočtu mění posunutím doprava a dosazením nových hodnot na začátek. [Wiki-FSR]

Stav registru v čase t budeme označovat

$$\mathbf{X}(t) = (X_1(t), \ldots, X_n(t)).$$

Na začátku má registr stav

$$\mathbf{X}(0) = (R_0, \ldots, R_n).$$

V každém kroku se všechny bity posunou o jednu pozici doprava, tj.

$$X_k(t) = X_k(t-1), k = 2, \ldots, n.$$

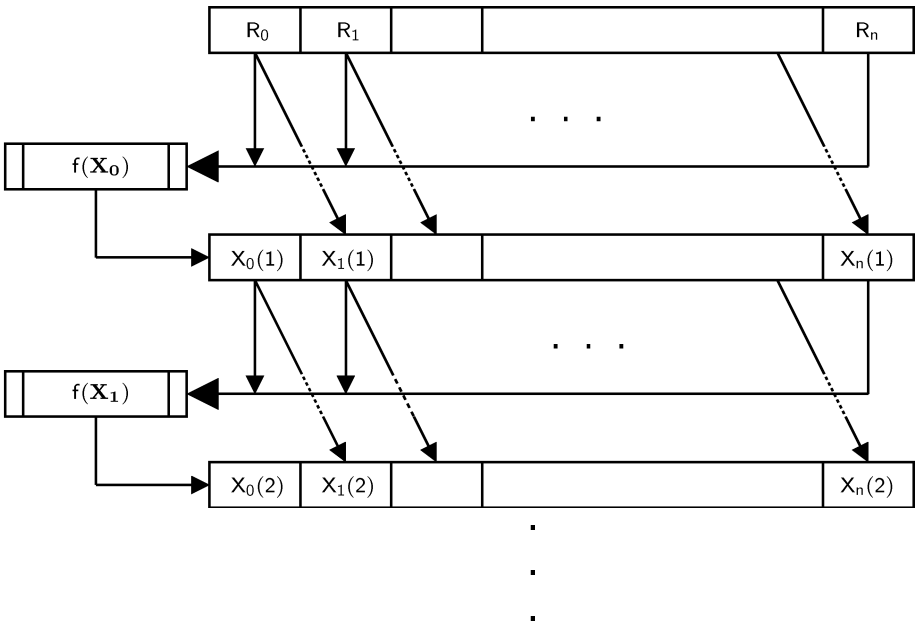
Zároveň se na první pozici dosadí hodnota závislá na předchozím stavu

$$X_1(t) = f(X_0(t-1), \ldots, X_n(t-1)),$$

tedy

$$\mathbf{X}_n(t) = (f(X_0(t-1), \ldots, X_n(t-1)), X_1(t-1), \ldots, X_{n-1}(t-1)).$$

Stav registru je tedy úplně popsán funkcí f a počátečním stavem. Tím je obvykle právě n-bitový klíč nebo nějaká jiná hodnota závislá na klíči.



Obrázek 2.4: Posouvací registry

2.7 Kryptoanalýza a napadání šifer

Útokem na šifru rozumíme činnost, kdy se útočník snaží

- *jednorázově* získat klíč, kterým byla nějaká zpráva zašifrována,
- najít *obecný postup* jak získávat klíč,
- najít postup jak z kryptogramu odvodit původní zprávu

apod. – mluvíme o *prolomení* šifry. Prolomením šifry získává útočník možnost zprávu dešifrovat, případně ji po cestě k původnímu příjemci modifikovat nebo zcela podstrčit a falešně autorizovat apod. **Kryptoanalýza** označuje proces zkoumání napadnutelnosti šifry, odhalování jejích potenciálních slabin.

Podle druhu informace, kterou má útočník k dispozici, lze útoky rozdělit na

- **Ciphertext only** – útočník má k dispozici jen zašifrovaný text;
- **Known plaintext** – útočník má k dispozici dvojici (zpráva, kryptogram);

- **Chosen plaintext** – útočník má přístup k šifrovacímu modulu a k libovolné zprávě je schopen vygenerovat kryptogram;
- **Chosen ciphertext** – útočník má přístup k dešifrovacímu modulu a k libovolnému kryptogramu je schopen získat původní zprávu.

Podle způsobu provádění můžeme rozpoznat útoky:

- **Algebraické**. Kryptoanalytické metody, které se snaží přímo najít chybu v algoritmu. Do této skupiny zařadíme Kasiského test [Paseka, strana 28]. Algebraické útoky mohou končit prolomením šifer, ale mnohem častěji pouze odhalením potenciálních slabin algoritmů. Z nich lze dále vycházet při dalších útocích.
- Hledání chyb v *implementaci* šifrovací metody. Může se jednat o implementaci *hardwarovou* – sem lze zařadit situaci, kdy z bankovní karty nějakým způsobem dostaneme PIN nebo ze smartkarty přečteme RSA klíč. O příklad nalezení chyby v *softwarové* implementaci půjde tehdy, když SSH serveru podstrčíme svůj klíč a takto se neautorizovaně dostaneme do cizího účtu. Mimořádný význam měl objev *postranních kanálů*, kterými se budeme ještě zabývat.
- **Útok hrubou silou** spočívá v ověření nějakého předem definovaného souboru dat, například vyzkoušení všech možných klíčů nebo nějaké velké skupiny klíčů.
- Důležitou skupinou, kterou se však zde nebudeme zabývat, je *sociální inženýrství*. Ta využívá převážně lidský faktor – pohodlnost, neznalost a naivitu. Zavoláme sekretářce, aby nám prozradila heslo do systému kvůli údržbě. Nebo do diskusní skupiny na WWW vložíme příspěvek se skriptem a ten využije chybu v prohlížeči jiného návštěvníka. Do této skupiny také můžeme zařadit útoky typu *Man-in-the-middle* (např. [MITM]) apod.

Všechny postupy lze úspěšně kombinovat.

2.7.1 Algebraické metody u proudových šifer

U proudové šifry lze zkoumat *předperiodu (preperiod)* a *periodu* generované pseudonáhodné posloupnosti, tedy počet kroků výpočtu, po kterém se posloupnost začne opakovat a počet kroků, pro které se hodnoty donekonečna opakují. Na zjištění periody je založen Kasiského test [Paseka, strana 28] v případě Vigenerovy šifry. Perioda je obecně závislá na dvojici (nonce,klíč).

předperioda				perioda 1				perioda 2				...	...
p_1	p_2	p_3	...	a_1	a_2	a_3	...	a_1	a_2	a_3	...	...	...

Dále si všímáme periody stavů na různých místech logických obvodů realizujících algoritmus – posuvných registrů, výstupů S-Boxů apod. a tyto hodnoty algebraicky zkoumáme. Lze vysledovat závislosti mezi těmito místy a popsat je *soustavami rovnic*. Pokud budeme provádět dostatečný počet pokusů s generovanými dvojicemi (inicializační hodnota, klíč), můžeme výsledky zpracovávat *statisticky*.

Tím výrazně omezíme počet možných stavů na pozorovaných místech oproti všem možným hodnotám, které by se zde mohly vyskytnout. Množina klíčů se potom redukuje na množinu nalezených přípustných stavů[1] a další kryptoanalýza nebo útok hrubou silou je potom již mnohem jednodušší.

2.7.2 Algebraické metody u blokových šifer

Lineární kryptoanalýza [DEF-LK] je metoda která využívá lineární závislosti kryptogramu, zprávy a klíče. Jedná se o known plaintext attack ale předpokládá se, že útočník má k dispozici větší množství dvojic (zpráva, kryptogram). Šifrovací funkci nahrazuje soustavami lineárních rovnic, např.

$$M_{i_1} \oplus \cdots \oplus M_{i_k} \oplus C_{j_1} \oplus \cdots \oplus C_{j_l} = 0,$$

kde $M_{i_1}, \ldots, M_{i_k}$ je část zprávy a $C_{j_1}, \ldots, C_{j_l}$ je část kryptogramu. Poté sleduje, s jakou pravděpodobností rovnice budou platit. Soustavou tedy aproximuje šifrovací funkci.

Odolnost vůči lineární kryptoanalýze – *nelinearita* – je důležitým kriteriem kvality šifry. Napadnutelnost lineární kryptoanalýzou můžeme vyjádřit například počtem dvojic (zpráva, kryptogram) které musíme znát, abychom aproximovali šifrovací funkci s daným klíčem. Například algoritmus DES vyžaduje 2^{43} takových dvojic, abychom jej mohli prolomit lineární kryptoanalýzou.

Dalším častým útokem je *diferenciální kryptoanalýza*. Ta zkoumá, jak se změní kryptogram, jestliže změníme zprávu. Diferenciální kryptoanalýza je chosen plaintext útok. Útočník totiž musí zkoušet vybrané dvojice zpráv s malými odchylkami, aby zjistil jejich rozdíl na výstupu.

Diferenciální a lineární kryptoanalýza se často používá ve spojení, mluvíme o *lineárně-diferenciální kryptoanalýze*. Úvod do lineární a diferenciální

[1] Množství informace o klíči, kterou poskytuje kryptogram, je jedním ze základních kritérií kvality šifry, viz teorie perfektní bezpečnosti [Paseka, strana 38]. Redukcí počtu klíčů se toto množství informace zvyšuje.

kryptoanalýzy lze najít například v [DL-tut]. Kryptoanalýza však používá spoustu dalších algebraických, statistických a jiných matematických metod [Wiki-CA] ke zkoumání blokových šifer.

2.7.3 Metoda postranních kanálů

Postranními kanály *(side channels)* rozumíme každou nežádoucí výměnu informací mezi kryptografickým modulem a jeho okolím [Klíma-VA]. Typickým příkladem je *časový postranní kanál*, kde se z časové náročnosti realizace algoritmu odvodí informace o klíči. Tento typ útoků se nazývá *timing*. Pozorovat lze prakticky všechny fyzikální veličiny, například spotřebu energie šifrovacího zařízení. [Web-ICZ]

Dalším typem jsou *chybové útoky (fault attacks)* [Wirt-04], prováděné hardwarovou modifikací kryptografických modulů. Útočník se může například pokoušet vyřadit nebo pozměnit některé části kryptografického modulu. Následně se pozoruje rozdíl mezi původním a modifikovaným výstupem. Kryptografický modul lze osvětlovat, ovlivňovat laserovým paprskem, vystavovat různým teplotám apod. – známé je zmrazování generátoru náhodných čísel u hracích automatů.

Objev postranních kanálů je relativně nedávný a a ukázalo se, že představují poměrně silnou hrozbu. Současná kryptografie se jimi proto velmi intenzivně zabývá [Klíma-02].

Kapitola 3

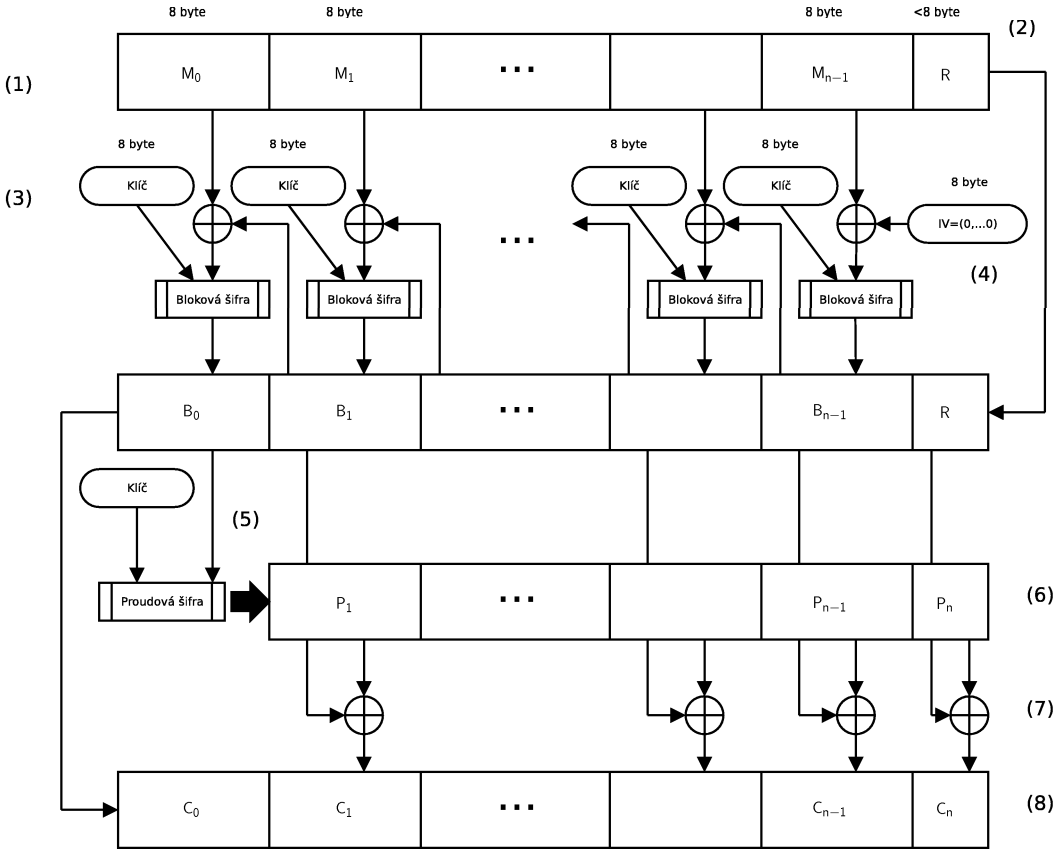
DVB Common Scrambling Algorithm

Common Scrambling Algorithm byl zpočátku neveřejný a byl k dispozici pouze na základě tzv. *non-disclosure agreement – dohody o utajení*. Posléze se ale objevil program pro operační systém Windows *FreeDec* neznámého autora, ze kterého byl reverzním inženýrstvím algoritmus ihned odvozen a vystaven na WWW stránkách [Web-irde.to]. To opět dokazuje neúčinnost „security by obscurity" – ačkoli důvodem bylo spíše ztížit kryptoanalýzu a tím riziko prolomení, než samotné skrytí algoritmu. Přes odtajnění však CSA dosud nijak vážně ohrožen nebyl.

CSA je kombinací 64bitové *blokové šifry* a *proudové šifry*. Šifra je *symetrická* – k šifrování i dešifrování se použije stejný klíč. Klíč je shodný pro blokovou i proudovou část šifry a nazývá se *control word*. Má délku 64 bitů, tedy $K = (k_0, ..., k_{63})$.

3.1 Scrambling

Na začátku je zpráva rozdělena na bloky $M_0, .., M_{n-1}, R$ po osmi bytech (1). Poslední blok R může být menší než osm bytů a v tom případě jej nazýváme *reziduum (residue)* (2). Sekvence bloků je postupně zašifrována blokovou šifrou (3) v opačném pořadí. Tedy začíná se s M_{n-1} a končí s M_0. Případné reziduum se nezašifrovává a putuje na vstup proudové šifry nezměněno. Blok M_{n-1} se před šifrováním xoruje s *inicializační hodnotou IV* v tomto případě složenou z nul (4). Každý další blok M_i se pak před šifrováním xoruje s předchozím kryptogramem B_{i+1}. Poslední hodnota B_0 se použije jako inicializační hodnota (nonce) pro proudovou šifru (5). Proudová šifra vygeneruje z klíče a nonce pseudonáhodnou posloupnost (6), která se xoruje (7) s vý-



Obrázek 3.1: Scrambling

stupem blokové šifry. Na konci dostaneme kryptogram – *scramblovaná data* $C_0, \ldots, C_{n-1}, C_n$ (8).

3.1.1 Bloková šifra

Označíme-li funkci blokového zašifrování E_B, můžeme psát

$$B_i = E_B(B_{i+1} \oplus M_i, K) \quad \text{pro} \quad i = n-1, \ldots, 0,$$

přičemž $B_n = (0, \ldots, 0)$ je inicializační hodnota.

Osmibytový úsek zprávy M_i se xoruje s přechozím výsledkem B_{i+1} a nastupuje vlastní bloková šifra E_B. Ta sestává z 56 stejných kroků – *cyklů*. Funkci, která realizuje každý jednotlivý cyklus, budeme označovat φ.

Klíč, kterým se v jednotlivých cyklech zašifrovává, je odvozen z původního klíče K – mluvíme o tzv. *proměnném klíči (running key)* nebo o *rozšířeném*

klíči (expanded key). Ten má 448 bitů. Označme jej

$$K^E = (k_0^E, \ldots, k_{447}^E).$$

V každém cyklu budeme potřebovat 8 bitů rozšířeného klíče. Jelikož $448 = 8 \times 56$, celý rozšířený klíč se postupně vyčerpá. Označme velkým písmenem bloky rozšířeného klíče pro jednotlivé cykly:

$$K_0^E = (k_0^E, \ldots, k_7^E)$$
$$K_i^E = (k_{8i}^E, \ldots, k_{8i+7}^E) \quad \text{pro} \quad i = 1, \ldots, 55.$$

Potom použijeme posloupnost bloků v pořadí

$$K_0^E, \ldots, K_{55}^E.$$

Mějme nyní permutaci ρ definovanou tabulkou 3.1.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
17	35	8	6	41	48	28	20	27	53	61	49	18	32	58	63
16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
23	19	36	38	1	52	26	0	33	3	12	13	56	39	25	40
32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
50	34	51	11	21	47	29	57	44	30	7	24	22	46	60	16
48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
59	4	55	42	10	5	9	43	31	62	45	14	2	37	15	54

Tabulka 3.1: Scrambling – permutace ρ pro expanzi klíče blokové části

448bitový rozšířený klíč $K^E = (k_0^E, ..., k_{447}^E)$ získáme takto:

$$(k_0^E, \ldots, k_{63}^E) = (k_0, \ldots, k_{63})$$
$$(k_{64j}^E, \ldots, k_{64j+63}^E) = \rho((k_{64(j-1)}^E, \ldots, k_{64(j-1)+63}^E)) \oplus p \quad \text{pro} \quad j = 1, \ldots, 6,$$

kde

$$p = (0, 0, 0, 0, [j], 0, 0, 0, 0, [j], 0, 0, 0, 0, [j], 0, 0, 0, 0, [j]),$$

přičemž $[j]$ jsou čtyři bity čísla j, tj. p je 64bitová konstanta.

Nechť $S = (S_0, \ldots, S_7)$ je osm byte výsledku nějakého cyklu $j-1$. V následujícím cyklu aplikujeme funkci φ s klíčem K_j a obdržíme další mezivýsledek $\varphi(S, K_j)$. Označíme-li $\varphi_j(S) = \varphi(S, K_j)$, lze všech 56 cyklů psát

$$E_B(B_{i+1} \oplus M_i, K) = \varphi_{55}(\varphi_{54}(\ldots \varphi_0(B_{i+1} \oplus M_i) \ldots))$$

pro nějakou část zprávy $M_i, \quad i = 1, \ldots, n-1$

Funkce $\varphi(S, K_j)$ pro 64bitový mezivýsledek $S = (S_0, \ldots, S_7)$ a pro 64bitový blok K_j proměnného klíče je definována takto:

$$\varphi(S, K_j) = (S_1, S_2 \oplus S_0, S_3 \oplus S_0, S_4 \oplus S_0, S_5, S_6 \oplus \tau'(K_j \oplus S_7), S_7, S_0 \oplus \tau(K_j \oplus S_7)),$$

kde τ' a τ jsou funkce operující na jednom byte, svázané spolu vztahem

$$\tau'(s_0, \ldots, s_7) = \tau(s_{\sigma(0)}, \ldots, s_{\sigma(7)}),$$

kde σ je permutace s cyklem plné délky 8:

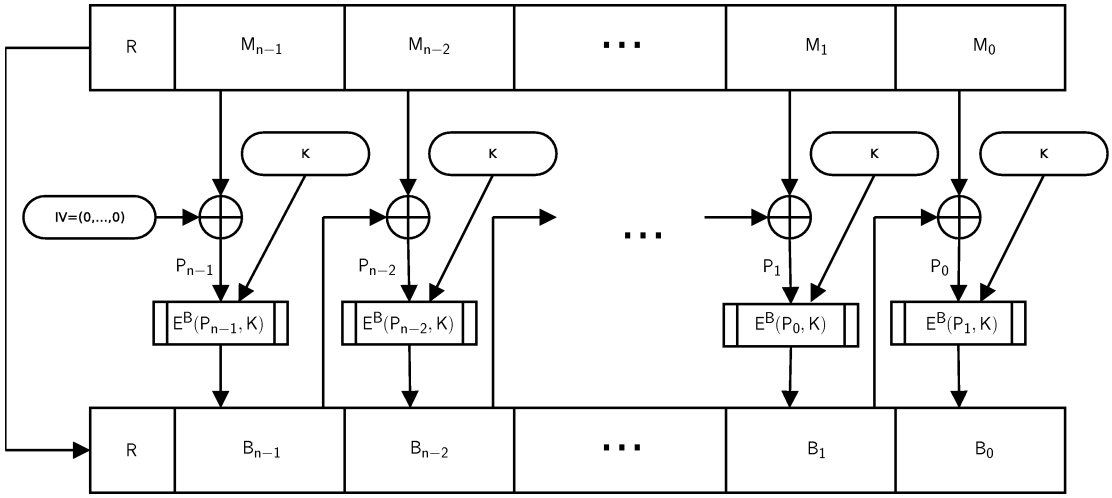
$$\sigma = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 7 & 5 & 4 & 2 & 6 & 0 & 3 \end{pmatrix}.$$

Funkce τ a τ' se tudíž liší jen pořadím bitů na výstupu. Obě lze chápat jako permutace 256prvkové množiny svých argumentů. Pokud vyjádříme jednobytové argumenty v hexadecimální podobě, pak τ je dána tabulkou 3.2.

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	0x3A	0xE8	0xBE	0xFF	0x16	0x71	0x6C	0x7E	0x29	0xEB	0x99	0x8E	0xF8	0xD1	0xFB	0xAD
0x01	0xEA	0x27	0x7A	0x69	0xAB	0xA1	0x19	0x0A	0x3D	0xC3	0x9E	0x5D	0xFC	0xF7	0xD3	0x46
0x02	0x68	0x61	0x6D	0xEF	0xAC	0xF9	0x42	0x9A	0xE7	0x34	0x4D	0x37	0x81	0xE0	0xB6	0x0B
0x03	0xFE	0x95	0x47	0x03	0x4C	0x4F	0x79	0x13	0x92	0x2D	0xD9	0x11	0x09	0xB5	0xCA	0xAF
0x04	0x33	0x0C	0xC1	0x4E	0xF1	0x2E	0xDD	0x2A	0x87	0xB8	0xDA	0xD2	0xB1	0x98	0x43	0x80
0x05	0xE9	0x36	0x51	0x48	0x6A	0xAA	0xEE	0x9D	0x1B	0x44	0x8D	0x28	0x01	0x22	0x72	0x52
0x06	0x88	0xE5	0x8F	0x4A	0x2F	0xC5	0x96	0xC2	0x2B	0x25	0x6F	0x75	0x76	0xB3	0x07	0x2C
0x07	0x1A	0x70	0xF3	0x84	0x3C	0x56	0xF6	0x5E	0x4B	0xA4	0x5F	0xD6	0x91	0x20	0xF4	0xFA
0x08	0x83	0xA2	0xCC	0x3F	0x3B	0xE3	0x8A	0x5A	0xA5	0x1C	0x3E	0xA7	0x7D	0x1D	0xD8	0x8C
0x09	0xCF	0x06	0x5B	0xB4	0xD4	0x39	0xEC	0x1F	0x57	0xC7	0xD7	0x77	0x0F	0xA6	0x41	0x89
0x0A	0xE1	0x82	0x67	0x10	0xD5	0x93	0x1E	0x32	0x97	0x23	0x21	0x24	0xC8	0xDB	0x14	0x66
0x0B	0x7F	0x7C	0xBD	0x04	0x94	0xCE	0x85	0x35	0x40	0xED	0x74	0xBF	0xA0	0x7B	0x55	0xFD
0x0C	0xBA	0x17	0xCD	0xDC	0xD0	0x65	0x53	0x9C	0x15	0x90	0x86	0xF0	0xF2	0x59	0x0D	0xB2
0x0D	0xE2	0xA3	0x18	0xF5	0xC4	0x64	0x45	0xA8	0xE6	0x6E	0xDF	0xB0	0xCB	0x9F	0x54	0xA9
0x0E	0x38	0x26	0x08	0x5C	0x63	0xE4	0xDE	0x73	0xBC	0x50	0x6B	0x02	0x78	0xAE	0x8B	0x9B
0x0F	0x12	0x49	0xC9	0xC6	0x62	0x58	0xBB	0x30	0x0E	0x00	0x05	0xB7	0x60	0x31	0xB9	0xC0

Tabulka 3.2: Funkce τ pro cyklus blokové šifry

Nechť $S_i = (s_i^0, s_i^1, s_i^2, s_i^3, s_i^4, s_i^5, s_i^6, s_i^7)$ je byte, na který chceme aplikovat funkci τ. V řádku tabulky odpovídajícímu $(s_i^0, s_i^1, s_i^2, s_i^3)$ a sloupci odpovídajícímu $(s_i^4, s_i^5, s_i^6, s_i^7)$ najdeme výsledek operace.



Obrázek 3.2: Scrambling – bloková část šifry

Celkem lze podobu blokové části E_B šifry převádějící zprávu

$$(M_0, \ldots, M_{n-1}, R)$$

na vstup proudové šifry

$$(B_0, \ldots, B_{n-1}, R)$$

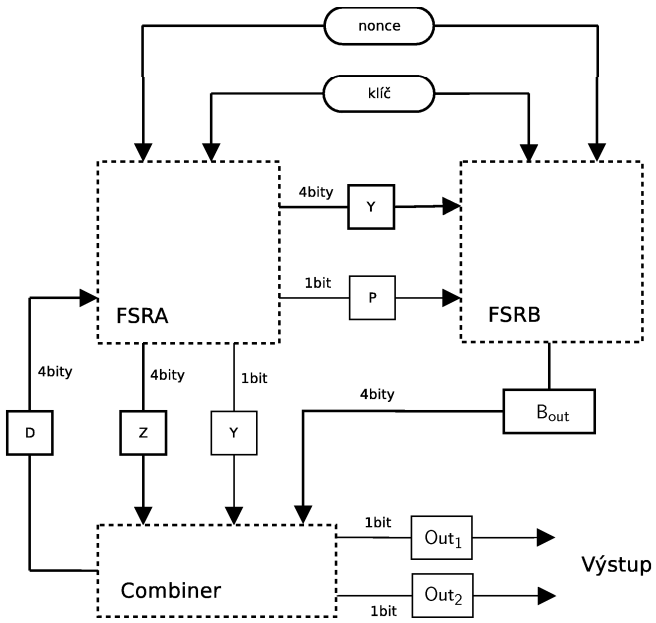
shrnout takto:

$$\begin{aligned} R &= R, \\ B_n &= (0, \ldots, 0), \\ B_i &= \varphi_{56}(\varphi_{55}(\ldots \varphi_0(M_i \oplus B_{i+1}) \ldots)), \quad 0 \leq i \leq n-1. \end{aligned}$$

3.1.2 Proudová šifra

Proudová šifra vytvoří z klíče (control word) a posledního výsledku B_0 blokové šifry pseudonáhodnou posloupnost $P_1, \ldots, P_n$, se kterou se xoruje zbytek výsledku $B_1, \ldots, B_{n-1}, R$ blokové šifry, abychom dostali kryptogram – scramblovaná data.

Proudová šifra využívá ke generování dva posouvací registry $FSRA$ a $FSRB$ *Feedback Shift Registers* a logický obvod s pamětí, tzv. *combiner (combiner with memory)*. Combiner s pamětí realizuje logickou funkci, přičemž výsledek této funkce je závislý na předchozích hodnotách (proto s pamětí), viz obrázek 3.3.



Obrázek 3.3: Scrambling – přehled proudové části

Oba posouvací registry jsou čtyřicetimístné. Označme první registr

$$A = (a_{i,j})_{i=0,\ldots,9}^{j=0,\ldots,3},$$

druhý registr

$$B = (b_{i,j})_{i=0,\ldots,9}^{j=0,\ldots,3}$$

a klíč $K = (k_0, \ldots, k_{63})$.

Pak počáteční stav registrů je dán následujícími pravidly – viz tabulka 3.3:

$$\begin{array}{lll} a_{i,j} = k_{4i+j} & \text{pro} & i = 0, \ldots, 7, \\ a_{i,j} = 0 & \text{pro} & i = 8, 9, \\ b_{i,j} = k_{32+4i+j} & \text{pro} & i = 0, \ldots, 7, \\ b_{i,j} = 0 & \text{pro} & i = 8, 9. \end{array}$$

Prvních 32 kroků je inicializačních a není odečítán výstup proudové šifry. Proto číslujeme kroky od -31, tak, aby v kroku označeném $t = 1$ byly odečteny z výstupů první hodnoty.

Zpětnovazební funkci a výstupy registru $FSRA$ (obrázek 3.4) lze popsat *S-Boxy*. **S-Box** *(substitution box)* je tabulka popisující nějakou logickou funkci, tj. pro každý stav na vstupech definuje stav výstupů. Podle počtu vstupů a výstupů mluvíme o *rozměrech* S-Boxu.

Registr A			
k_0	k_1	k_2	k_3
k_4	k_5	k_6	k_7
k_8	k_9	k_{10}	k_{11}
k_{12}	k_{13}	k_{14}	k_{15}
k_{16}	k_{17}	k_{18}	k_{19}
k_{20}	k_{21}	k_{22}	k_{23}
k_{24}	k_{25}	k_{26}	k_{27}
k_{28}	k_{29}	k_{30}	k_{31}
0	0	0	0
0	0	0	0

Registr B			
k_{32}	k_{33}	k_{34}	k_{35}
k_{36}	k_{37}	k_{38}	k_{39}
k_{40}	k_{41}	k_{42}	k_{43}
k_{44}	k_{45}	k_{46}	k_{47}
k_{48}	k_{49}	k_{50}	k_{51}
k_{52}	k_{53}	k_{54}	k_{55}
k_{56}	k_{57}	k_{58}	k_{59}
k_{60}	k_{61}	k_{62}	k_{63}
0	0	0	0
0	0	0	0

Tabulka 3.3: Počáteční stav ($t = -31$) posouvacích registrů proudové šifry

Označme $a_i = (a_{i,0}, \ldots, a_{i,3})$ i-tý řádek matice popisující stav registru A. Výstup S-Boxu je v každém kroku xorován s posledními čtyřmi bity $a_9 = (a_{9,0}, \ldots, a_{9,3})$ registru a tato hodnota vložena na první čtyři bity registru A. Poté, co je na začátku do registru nahrán klíč, proběhne nejprve 32 inicializačních cyklů bez odečítání výstupů. Na vstupu $FSRA$ je kromě klíče a SB_0 ještě výstupní registr D combineru.

V inicializačním módu se v každém z cyklu pro $i = -31, \ldots, 0$ vypočítá další zpětnovazební hodnota

$$a_0(i+1) = D_i \oplus I_i^A(B_0) \oplus S(A) \oplus a_9(i),$$

kde $S(A) = S(a_0, \ldots, a_9) = S(a_{0,0}, \ldots, a_{9,3})$ je výstup S-Boxu, a_9 poslední čtyři bity registru A, D_i je stav registru D combineru a I^A následující funkce inicializační hodnoty:

$$I_i^A = \begin{cases} SB_0 \text{ div } 2^4 & \text{pro } i \text{ lichá}, \\ SB_0 \text{ mod } 2^4 & \text{pro } i \text{ sudá}. \end{cases}$$

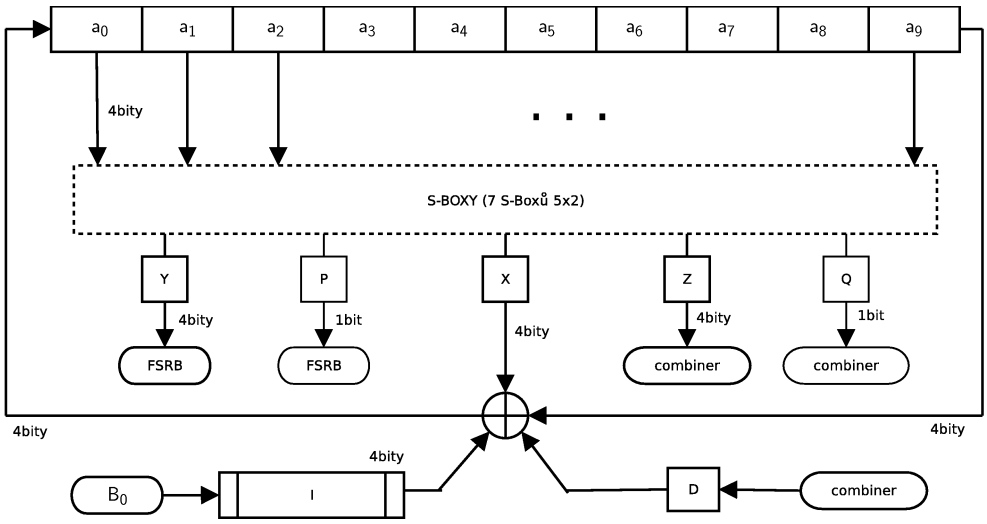
Nový stav registru A je po posunutí

$$A(i+1) = (a_0(i+1), a_0(i), \ldots, a_8(i)),$$

jde tedy o rotaci 4 bity doprava.

První posouvací registr má čtyři výstupy. Vždy jeden čtyřbitový a jeden jednobitový jsou pak vstupem druhého posouvacího registru a combineru.

Vstupy a výstupy S-Boxů $FSRA$ jsou popsány tabulkami 3.4.



Obrázek 3.4: Scrambling – posouvací registr A proudové části

S1	$a_{3,0}$	$a_{0,2}$	$a_{5,1}$	$a_{6,3}$	$a_{8,0}$
S2	$a_{1,1}$	$a_{2,2}$	$a_{5,3}$	$a_{6,0}$	$a_{8,1}$
S3	$a_{0,3}$	$a_{1,0}$	$a_{4,1}$	$a_{4,3}$	$a_{5,2}$
S4	$a_{2,3}$	$a_{0,1}$	$a_{1,3}$	$a_{3,2}$	$a_{7,0}$
S5	$a_{4,2}$	$a_{3,2}$	$a_{5,0}$	$a_{7,1}$	$a_{8,2}$
S6	$a_{2,1}$	$a_{3,1}$	$a_{4,0}$	$a_{6,2}$	$a_{8,3}$
S7	$a_{1,2}$	$a_{2,0}$	$a_{6,1}$	$a_{7,2}$	$a_{7,3}$

X	$S_{4,0}$	$S_{3,0}$	$S_{2,1}$	$S_{1,1}$
Y	$S_{6,0}$	$S_{5,0}$	$S_{4,1}$	$S_{3,1}$
Z	$S_{2,0}$	$S_{1,0}$	$S_{6,1}$	$S_{5,1}$
p	$S_{7,1}$			
q	$S_{7,0}$			

Tabulka 3.4: Scrambling – tabulka vstupů a tabulka výstupů S-Boxů posouvacího registru A proudové části

Každý S-Box má pět bitů vstupu. První tabulka definuje pro 7 S-Boxů $S_1, \ldots, S_7$, které bity registru A jdou do jejich vstupů, tj. například vstup S_6 tvoří pětice

$$(a_{2,1}; a_{3,1}; a_{4,0}; a_{6,2}; a_{8,3}).$$

Každý registr má dva bity výstupní, označené $S_{i,0}$ a $S_{i,1}$ pro $i = 1, \ldots, 7$. Druhá tabulka říká pro registry proudové šifry, kterými výstupy S-Boxů jsou v každém cyklu výpočtu vytvářeny. Například jednobitový registr p je tvořen výstupem 1 registru 7. Čtyřbitový registr Y je tvořen čtveřicí výstupů $(S_{6,0}, S_{5,0}, S_{4,1}, S_{3,1})$ ze šestého, pátého, čtvrtého a třetího registru.

Vlastní funkce S-Boxů je popsána tabulkou 3.5.

V tabulce jsou ve sloupcích jednotlivé S-Boxy. K pětici bitů vstupu na řádku je uvedena dvojice bitů na výstupu S-Boxu. S-Boxy lze kromě tabulky

$Vstup$	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$	$S7$
00000	00	11	01	01	10	10	01
00001	11	01	11	00	11	11	00
00010	11	00	00	11	10	00	11
00011	00	11	01	01	00	10	11
00100	10	11	11	10	00	11	00
00101	10	10	00	11	11	00	01
00110	01	00	10	00	01	01	01
00111	01	10	10	11	01	01	10
01000	10	00	10	00	01	10	10
01001	10	00	00	11	00	01	11
01010	00	01	01	10	11	01	01
01011	11	10	10	00	10	10	00
01100	01	10	00	01	11	00	10
01101	01	01	11	10	01	11	11
01110	11	11	11	10	00	11	00
01111	00	01	01	01	10	00	10

$Vstup$	$S1$	$S2$	$S3$	$S4$	$S5$	$S6$	$S7$
10000	10	11	10	11	10	00	00
10001	00	01	00	01	00	01	11
10010	01	00	01	10	00	10	10
10011	01	10	10	11	01	11	10
10100	10	10	10	00	11	01	11
10101	11	11	11	10	10	10	00
10110	11	11	11	01	11	10	00
10111	00	00	01	10	10	00	01
11000	11	01	01	01	00	00	11
11001	10	11	01	10	01	01	00
11010	10	10	00	00	11	11	01
11011	00	01	11	01	11	00	11
11100	01	00	11	11	01	10	01
11101	01	00	00	00	00	11	10
11110	00	01	10	00	10	01	10
11111	11	10	00	11	01	11	01

Tabulka 3.5: Scrambling – definice S-Boxů posouvacího registru A proudové části

$$
\begin{aligned}
S_{1,0} &= abce + abc + abd + bde + ab + ae + be + ce + b + d \\
S_{1,1} &= abcd + abde + abc + abd + acd + ade + bcd + bce + ab + ac \\
&\quad + bc + bd + be + cd + ce + de + a + d + e + 1 \\
S_{2,0} &= abce + abde + ade + bce + bde + ab + ac + ce + c + d + 1 \\
S_{2,1} &= abde + abc + abd + abe + acd + cde + cd + ce + b + d + e + 1 \\
S_{3,0} &= ce + de + a + b + d \\
S_{3,1} &= abcd + acde + abe + ac + abc + acd + ace + ade + bcd + bde + cde \\
&\quad + ad + bc + bd + be + cd + ce + a + b + d + e + 1 \\
S_{4,0} &= abcd + abde + acde + abc + abe + bde + ab + ad + ae + bc + be + de + c + d + 1 \\
S_{4,1} &= abcd + abde + acde + abc + abe + bcd + cde + ad \\
&\quad + ab + ae + de + a + b + c + e + 1 \\
S_{5,0} &= abde + acde + acd + abe + abd + ace + bce + cde \\
&\quad + ab + ac + ae + bd + be + ce + de + c \\
S_{5,1} &= abcd + abce + acde + abd + abe + acd + bcd + bce + bde + cde + ac \\
&\quad + ad + ae + be + cd + ce + de + b + d + e + 1 \\
S_{6,0} &= abcd + abde + acde + acd + ade + bcd + cde + bc + bd + cd + c + e \\
S_{6,1} &= abe + ade + bce + bde + bc + ce + a + d \\
S_{7,0} &= abde + abd + cde + bc + cd + de + a + b + c + e \\
S_{7,1} &= abcd + abdebc + acde + acd + ade + bde + ac + ae + de + b + c + d + e
\end{aligned}
$$

Tabulka 3.6: Algebraická definice funkce S-Boxů $FSRA$

popsat též algebraicky. V tabulce 3.6 je každý výstup popsán jako funkce $S_{i,j}(a, b, c, d, e)$.

Po opuštění inicializačního módu, tj. pro $i = 1, \ldots$ se zpětnovazební funkce zjednoduší na

$$a_0(i) = S(a_0(i-1), \ldots, a_{39}(i-1)) \oplus a_9(i-1).$$

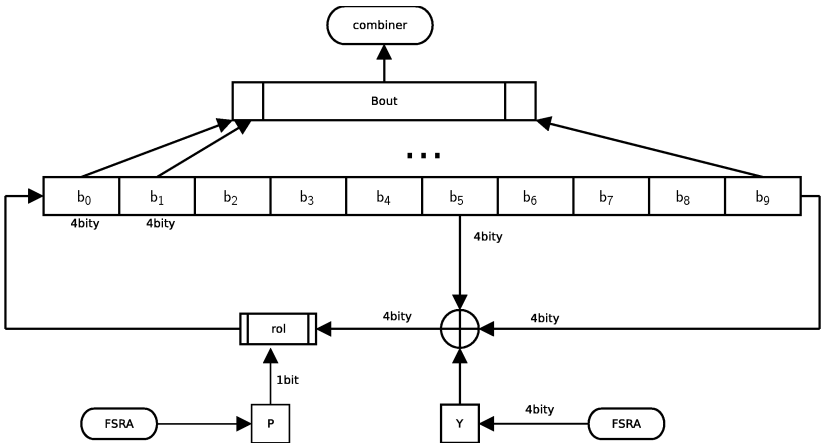
Tímto je $FSRA$ úplně popsán.

Zpětnovazební funkce $FSRB$ (obrázek 3.5) spočívá v xorování 4bitových hodnot b_6, b_9 a registru Y prvního registru $FSRA$. Pokud je na jednobitovém výstupu p registru $FSRA$ logická 1, tyto čtyři bity ještě rotují doleva. V inicializačním módu se dále ještě přixoruje funkce $I^B(B_0)$ definovaná takto:

$$I_i^B = \begin{cases} SB_0 \bmod 2^4 & \text{pro } i \text{ lichá}, \\ SB_0 \operatorname{div} 2^4 & \text{pro } i \text{ sudá}. \end{cases}$$

Tedy celkem nový stav registru bude

$$B(i+1) = \begin{cases} (b_0(i+1), b_0(i), \ldots, b_8(i)) & \text{pro } p(i) = 0 \\ (rol(b_0(i+1)), b_0(i), \ldots, b_8(i)) & \text{pro } p(i) = 1, \end{cases}$$



Obrázek 3.5: Scrambling – posouvací registr B proudové šifry

kde *rol* je permutace

$$\begin{pmatrix} 0 & 1 & 2 & 3 \\ 1 & 2 & 3 & 0 \end{pmatrix},$$

a hodnota na začátku bude

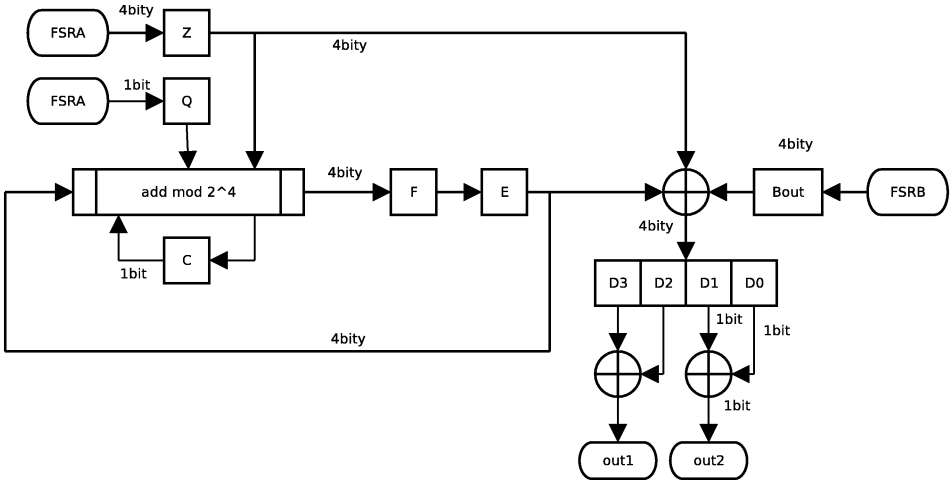
$$\begin{aligned} b_0(i+1) &= b_6(i) \oplus b_9(i) \oplus Y(i) \oplus I^B(i) && \text{pro} \quad i = -31, \ldots, 0 \\ b_0(i+1) &= b_6(i) \oplus b_9(i) \oplus Y(i) && \text{pro} \quad i = 1, \ldots. \end{aligned}$$

Tímto je stav registru $FSRB$ úplně popsán a zbývá nadefinovat jeho výstup B^{out}:

$$\begin{aligned} B_0^{out} &= b_{2,0} \oplus b_{5,1} \oplus b_{6,2} \oplus b_{8,3}, \\ B_1^{out} &= b_{5,0} \oplus b_{7,1} \oplus b_{2,3} \oplus b_{3,2}, \\ B_2^{out} &= b_{4,3} \oplus b_{7,2} \oplus b_{3,0} \oplus b_{4,1}, \\ B_3^{out} &= b_{8,2} \oplus b_{5,3} \oplus b_{2,1} \oplus b_{7,0}. \end{aligned}$$

Combiner (obrázek 3.6) proudové šifry produkuje dva bity výstupu v každém cyklu. Paměť combineru tvoří tři registry E, F a c. Jejich stav v každém kroku je určen takto:

$$\begin{aligned} E(i+1) &= F(i), \\ F(i+1) &= \begin{cases} E(i) & \text{pro} \quad q(i) = 0 \\ E(i) + Z(i) + c(i) \bmod 2^4 & \text{jinak,} \end{cases} \\ c(i+1) &= \begin{cases} c(i) & \text{pro} \quad q(i) = 0 \\ E(i) + Z(i) + c(i) \bmod 2^4 & \text{jinak.} \end{cases} \end{aligned}$$



Obrázek 3.6: Scrambling – combiner proudové části

Nakonec výstupní hodnoty out_1 a out_2 dostaneme jako:

$$out_1 = d_3 \oplus d_2$$
$$out_2 = d_1 \oplus d_0,$$

kde $D = (d_0, \ldots, d_3)$ a

$$D(i+1) = E(i) \oplus B_{out}(i) \oplus Z(i).$$

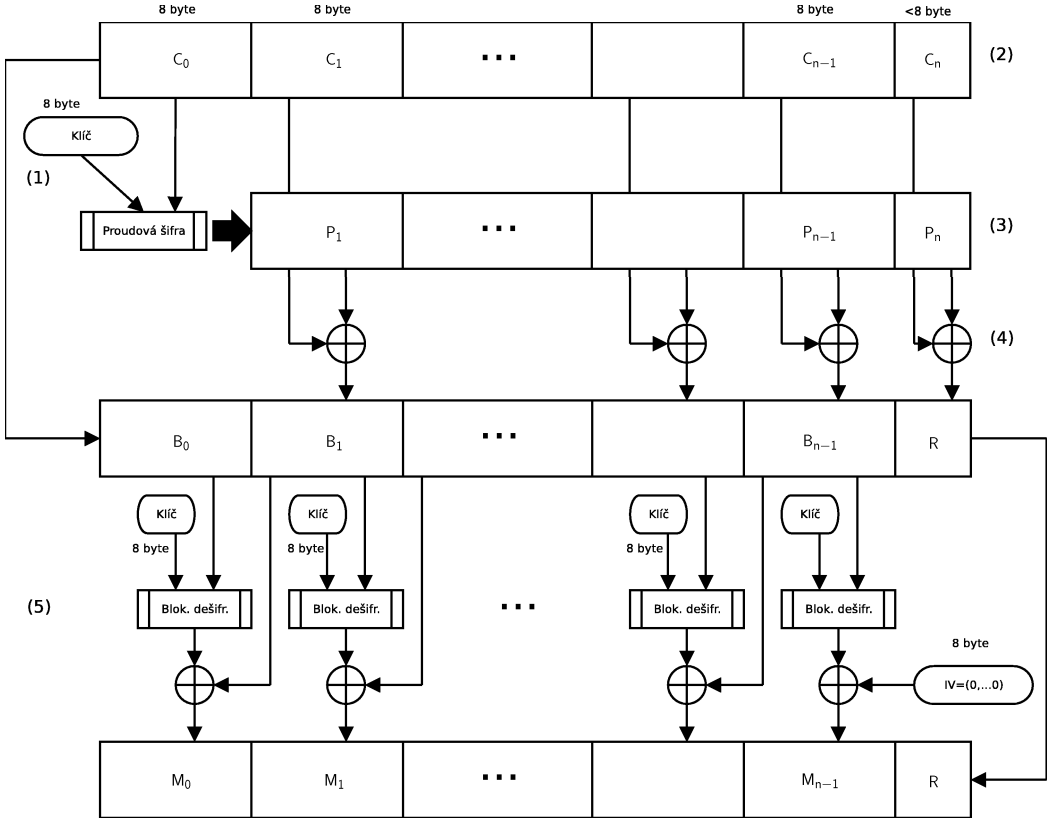
3.2 Descrambling

Vstupem pro descrambling (obrázek 3.7) je klíč K (1) a n bytů scramblovaných dat (2).

První blok C_0 (8 bytů) descramblovaných dat je inicializační hodnota pro proudovou šifru a na vstup blokového dešifrování jde nezměněna. Proudovou šifrou se vygeneruje heslo $P_1, \ldots, P_n$ (3) a tímto se xorují (4) zbylá scramblovaná data $C_1, \ldots, C_n$. Zašifrování proběhlo cestou $C_i = P_i \oplus B_i$ a proto platí $B_i = P_i \oplus C_i$. To znamená, že v případě proudové šifry s následným xorováním jsou funkce šifrování a dešifrování identické.

Blokové dešifrování (5) může nyní probíhat na rozdíl od šifrování v libovolném pořadí, neboť v CBC módu nepotřebujeme mezivýsledky pro řetězení blokové šifry. Reziduum v tomto případě zůstává nezměněno. Každý blok B_i se nejprve dešifruje a poté se xoruje s předchozím blokem B_{i-1}. Totiž při šifrování platilo

$$B_i = E_B(K, M_i \oplus B_{i-1}),$$



Obrázek 3.7: Descrambling

nyní tedy

$$M_i \oplus B_{i-1} = D_B(K, B_i)$$

podle definice šifrovací funkce a

$$M_i = D_B(K, B_i) \oplus B_{i-1}$$

podle pravidel počítání s *xor*. Dešifrovaný B_{n-1} se xoruje s inicializační hodnotou $(0, \ldots, 0)$, tedy již se nemění $(B_{n-1} \oplus (0, \ldots, 0) = B_{n-1})$.

Nakonec zbývá popsat blokové dešifrování, tj. úkolem je odvodit funkci

$$D_B : (B_0, \ldots, B_{n-1}, R) \mapsto (M_0, \ldots, M_{n-1}, R)$$

inverzní k funkci E_B shrnuté přehledně na straně 30, zejména pak nalezení inverze φ^{-1} k jednomu cyklu φ šifry.

$$\begin{aligned} R &= R \\ B_n &= (0, \ldots, 0) \\ M_i &= \varphi_0^{-1}(\varphi_1^{-1}(\ldots \varphi_{55}^{-1}(B_i)\ldots)) \oplus B_{i+1} \quad 0 \leq i \leq n-1. \end{aligned}$$

Tedy 56krát se provede inverzní transformace φ^{-1}.

Inverzní funkce $\varphi^{-1}(S, K_j)$ k funkci $\varphi(S, K_j)$ (viz strana 29) pro 64bitový mezivýsledek $S = (S_0, \ldots, S_7)$ pro krok j blokového dešifrování bude tvaru

$$\begin{aligned} \varphi^{-1}(S, K_j) = (&S_7 \oplus \tau(S_6 \oplus K_j), S_0, \\ &S_7 \oplus S_1 \oplus \tau(S_6 \oplus K_j), S_7 \oplus S_2 \oplus \tau(S_6 \oplus K_j), \\ &S_7 \oplus S_3 \oplus \tau(S_6 \oplus K_j), S_4, S_5 \oplus \tau'(S_6 \oplus K_j), S_6), \end{aligned}$$

kde τ a τ' již známe.

Tímto je descrabmling úplně popsán.

3.3 Důkaz korektnosti CSA

Je třeba dokázat, že

$$D(K, (E(K, M)) = M,$$

kde E označuje scrambling, D descrambling, K control word a $M = (M_0, \ldots, M_{n-1}, R)$ je původní zpráva. Označíme-li dále blokové šifrování E^B, dešifrování D^B a proudovou šifru E^P resp. D^P, dokazujeme

$$D^B(K, D^P(K, E^P(K, E^B(K, M)))) = M.$$

Z vlastností operace $\oplus$ plyne

$$D^P(K, E^P(K, E^B(K, M))) = E^B(K, M)$$

(korektnost proudového zašifrování) a vztah se zjednoduší na

$$D^B(K, E^B(K, M))) = M.$$

Zbývá dokázat korektnost blokové části šifry.

Vztah pro zašifrování bloků je

$$B_i = E_B(K, M_i \oplus B_{i+1}) \quad \text{pro} \quad i = 1, \ldots, n-1,$$

$B_n = (0, \ldots, 0)$, pro dešifrování

$$M_i = D_B(K, B_i) \oplus B_{i+1} \quad \text{pro} \quad i = 1, \ldots, n-1$$

a tudíž dokazujeme pro každý blok

$$M_i \oplus B_{i+1} = D_B(K, E_B(K, M_i \oplus B_{i+1})) \quad \text{pro} \quad i = 1, \ldots, n-1,$$

označíme-li $P_i = M_i \oplus B_{i+1}$, pak

$$P_i = D_B(K, E_B(K, P_i)) \quad \text{pro} \quad i = 1, \ldots, n-1.$$

Platí (viz strany 30 a 39)

$$\begin{aligned} P_i &= \varphi_1^{-1}(\varphi_2^{-1}(\ldots \varphi_{56}^{-1}(B_i)\ldots)) \qquad \text{pro} \quad i = 1, \ldots, n-1 \\ B_i &= \varphi_{56}(\varphi_{55}(\ldots \varphi_1(P_i)\ldots)) \qquad \text{pro} \quad i = 1, \ldots, n-1, \end{aligned}$$

tj. je potřeba ukázat

$$\begin{aligned} \varphi_1^{-1}(\varphi_2^{-1}(\ldots \varphi_{56}^{-1}(\varphi_{56}(\varphi_{55}(\ldots \varphi_1(P_i)\ldots)))\ldots)) = P_i \quad \text{pro} \quad i = 1, \ldots, n-1 \\ \varphi^{-1}(\varphi(K_j^E, P), K_j^E) = P, \quad 1 \le j \le 56 \\ \varphi_j^{-1}(\varphi_j(P)) = P, \quad 1 \le j \le 56. \end{aligned}$$

Nakonec prostým dosazení ukážeme, že pro libovolný rozšířený klíč K_j je φ^{-1}

inverzní funkcí k φ

$$\begin{aligned}
\varphi(S, K_j) =& (S_1, S_2 \oplus S_0, S_3 \oplus S_0, S_4 \oplus S_0, S5, \\
& S_6 \oplus \tau'(K_j \oplus S_7), S_7, S_0 \oplus \tau(K_j \oplus S_7)) \\
\varphi^{-1}(S, K_j) =& (S_7 \oplus \tau(S_6 \oplus K_j), S_0, \\
& S_7 \oplus S_1 \oplus \tau(S_6 \oplus K_j), S_7 \oplus S_2 \oplus \tau(S_6 \oplus K_j), \\
& S_7 \oplus S_3 \oplus \tau(S_6 \oplus K_j), S_4, S_5 \oplus \tau'(S_6 \oplus K_j), S_6), \\
\varphi_j^{-1}(\varphi_j(S)) =& \varphi^{-1}(\varphi(S, Kj), Kj) = \\
& (S_0 \oplus \tau(K_j \oplus S_7) \oplus \tau(S_7 \oplus K_j), S_1, \\
& S_0 \oplus \tau(K_j \oplus S_7) \oplus S_2 \oplus S_0 \oplus \tau(S_7 \oplus K_j), \\
& S_0 \oplus \tau(K_j \oplus S_7) \oplus S_3 \oplus S_0 \oplus \tau(S_7 \oplus K_j), \\
& S_0 \oplus \tau(K_j \oplus S_7) \oplus (S_4 \oplus S_0) \oplus \tau(S_7 \oplus K_j), \\
& S_5, S_6 \oplus \tau'(K_j \oplus S_7) \oplus \tau'(S_7 \oplus K_j), S_7) \\
& = (S_0, \ldots, S_7) = S.
\end{aligned}$$

Tím je důkaz hotov.

3.4 Kryptoanalýza a útoky na DVB CSA

Ověření všech možných klíčů *(útok hrubou silou)* by trval stovky let a proto není myslitelný. K prolomení je třeba hledat alternativní cesty. Spekuluje se o proudové i blokové části šifry – v prvním případě můžeme využít *reziduum*, které není zašifrované blokovou šifrou a v druhém blok B_0, který naopak není zašifrován proudovou šifrou.

3.4.1 Proudová šifra

Výstup proudové šifry je funkcí stavu registrů a ten je dán celkem 103 bity (A – 40, B – 40, E – 4, F – 4, c – 1, Y – 4, X – 4 Z – 4, p – 1, q – 1), počítáme jen ty registry, jejichž obsah ovlivňují další průběh výpočtu. To znamená, že počet všech stavů šifry, a tím i její perioda je shora omezen číslem 2^{103}. Hledáním algebraických vztahů mezi registry se počet stavů dramaticky sníží. Označme periodu proudové šifry l.

Pozorováním ihned zjistíme, že pro různé páry (klíč,nonce) je l obecně různé. Danému l však odpovídá více dvojic (klíč,nonce). Pro hledání period pro konkrétní dvojice (klíč,nonce) lze použít Floydův algoritmus na hledání délky cyklu. [Wiki-FA]

Dále si můžeme všimnout periody registru A – označme jej l_A. Zjistíme, že pokud mají různé dvojice (klíč,nonce) stejná l, potom mají stejná l_A a

dokonce celý registr $FSRA$ se chová identicky, tj. prochází stejnými stavy. To znamená, že pro zjištění period stačí zabývat se pouze množinou stavů, kterou může procházet registr $FSRA$.

Provedeme-li dostatečný počet pozorování s různými náhodnými dvojicemi (klíč,nonce) zjistíme, že drtivá většina (v pokuse [WW-04] to bylo 98,4 %) dvojic má jako periodu jedno z čísel z malé skupiny.

Předpokládejme nyní, že máme k dispozici kryptografický modul provádějící proudovou část šifry CSA a chceme najít klíč, kterým zašifrovává[1]. Ten se nachází v registrech A a B. Zašifrováním zprávy složené ze samých nul získáme pseudonáhodnou posloupnost. Zjistíme periodu generované pseudonáhodné posloupnosti a k ní najdeme příslušnou periodu registru $FSRA$.

Pokračujeme zkoumáním možných stavů $FSRA$ s danou délkou. Těch je již podstatně méně než všech možných klíčů. Navíc můžeme pro každý takový stav zjistit, zda je přípustný a tím množinu ještě více omezit.

Stav $FSRB$ je plně závislý na stavu $FSRA$ a jeho zpětnovazební funkce je lineární. Z toho vyplývá, že můžeme pro každý uvažovaný stav $FSRA$ vyjádřit stav $FSRB$ soustavou lineárních rovnic a řešit Gaussovou eliminací. Tedy pro každý stav A máme i stav B.

Můžeme proto nyní pro každý přípustný stav $FSRA$ generovat pseudonáhodnou posloupnost. Přitom použijeme nonce, které bylo předtím vloženo do kryptografického modulu. Jestliže najdeme shodu s výstupem kryptografického modulu, z A a B odečteme klíč.

Popsaný útok je již v silách běžné výpočetní techniky.

3.4.2 Bloková šifra

Expanze klíče je nelineární, což vyřazuje lineární kryptoanalýzu. Taktéž ostatní klasické metody se ukazují jako neúčinné.

K. Wirt však popsal v [Wirt-04] chosen ciphertext útok na blokovou šifru v situaci, kdyby se provedlo postranním kanálem (například laserovým působením) několik konkrétních změn v kryptografickém modulu pro blokové dešifrování.

Za předpokladu, že by se podařilo do dešifrovacího modulu vnést náhodnou chybu do několika posledních kroků $\varphi_i^{-1} \quad i = 7, \ldots, 0$, lze odvodit posledních osm byte $k_{384}, \ldots, k_{447}$, tedy $K_{48}, \ldots, K_{55}$ expandovaného klíče a z něj rekonstruovat samotný control word K.

Označme nyní P výsledek blokového dešifrování. Nechť

$$S_i = (S_i^0, \ldots, S_i^7), \quad i = 55, \ldots, 0$$

[1] Tj. provádíme *chosen ciphertext attack*, avšak u proudové šifry s xorováním je to totéž jako *chosen plaintext attack*.

je výsledek i-tého kola tohoto blokového dešifrování, potom $P = (P^0, \ldots, P^7) = S_0 = \varphi_0^{-1}(S_1)$ je poslední výsledek.

Předpokládejme, že útočník vnese před prováděním posledního kola φ_0 náhodnou chybu do sedmého byte registru, tedy náhodně změní S_1^6 na $\overline{S}_1^6$ a ta se dostane na vstup φ_0. K dispozici pak má původní správný výstup $P = (P^0, \ldots, P^7)$ a modifikovaný $\overline{P}^0 = (\overline{P}^0, \ldots, \overline{P}^7)$ (oba 8 byte).

Podle definice funkce φ^{-1} (na straně 39) platí:

$$S_0^7 = S_1^6 = P^7$$
$$\overline{S}_0^7 = \overline{S}_1^6 = \overline{P}^7.$$

Zavedeme funkci g jednobytového bloku expandovaného klíče takto:

$$\begin{aligned} g(k_{440}^E, \ldots, k_{447}^E) = g(K_{55}) &= \tau(S_1^6 \oplus K_{55}) \oplus \tau(\overline{S}_1^6 \oplus K_{55}) \\ &= S_0^0 \oplus \overline{S}_0^0 = P^0 \oplus \overline{P}^0. \end{aligned}$$

Poté mezi všemi možnými bloky klíči $k' \in (0, \ldots, 255)$ najdeme ty, pro které $g(k')$ a $g(K_{55})$. Opakováním pokusu zjistíme nejpravděpodobnější blok expandovaného klíče K_{55}.

Podobně se provede zanesení náhodných hodnot do dalších kroků a výpočet funkce g se mírně modifikuje na základě vztahů pro výpočet expandovaného klíče. K odvození celého klíče K je pak potřeba osm takových pozorování – viz [WW-04, strana 5].

3.4.3 Shrnutí

Dva možné útoky, které jsme zde popsali, samozřejmě CSA přímo neohrožují. Ve skutečnosti je nelineární expanze klíče v blokové části šifry, řetězení CBC módu, použití nonce až z dešifrovaného posledního bloku a nakonec samotná kombinace blokové a proudové části šifry velmi silná. Naznačují však cestu jakým se útočník může vydat při prolamování CSA. Autoři věří, že CSA je napadnutelný.

Kapitola 4

RSA

RSA je zkratka autorů algoritmu (Rivest, Shamir, Adleman). Algoritmus vznikl v roce 1977 a stal se zřejmě nejpoužívanější asymetrickou šifrou vůbec. Jeho idea spočívá ve faktu, že je snadné vynásobit dvě velká prvočísla ale je velký problém najít pro tento součin zpět prvočinitele. V této kapitole RSA popíšeme a nastíníme možnosti útoků na něj.

Narozdíl od jiných systémů nebyl Cryptoworks dosud vážněji ohrožen a za to vděčí mimo jiné také RSA. Klíče pro scrambling a descrambling se v pravidelných intervalech mění a informace o výměně klíčů centrála kartě posílá zašifrované proprietární modifikací RSA. Zprávy s novými klíči a další informce pro Conditional Access, tzv. *Control Messages*, jsou přidány do transportního proudu ke scramblovaným audiovizuálním datům. Conditional Access Module přijme Control Message a na jeho základě předá kartě *instrukci*. Instrukce je uvozena sekvencí udávající její typ (nový klíč, výměna klíče, změna oprávnění) a ta je následována blokem dat s vlastním obsahem instrukce, stále ještě zašifrovaným RSA.

Zatímco předání Control Message přes DVB Common Interface je přesně specifikováno, protokol komunikace Cryptoworks CAM s kartou veřejně příliš znám není. Některé informace se objevují v dokumentu „cwfaq“ [CWFaq] anonymních autorů, který koluje na WWW.

Není ani veřejně známo, jakou podobu RSA systém používá. Navíc samotný algoritmus RSA neříká nic o implementaci. Firma RSA Security vydala několik standardů, které říkají, jakým způsobem má zpráva být rozdělena, šifrována a podepisována [Web-PKCS]. Dále existuje také modifikace algoritmu RSA vycházející z tzv. *čínské věty o zbytku* [Ber-04, strana 12], [CRT], Tato modifikace je ovšem například v USA patentována, zatímco původní algoritmus je volně použitelný.

4.1 Popis algoritmu RSA

Najdeme dvě velká prvočísla p, q. Jejich součin označíme n:

$$n = pq.$$

Najdeme d, $1 < d < pq$ tak, aby bylo nesoudělné s $(p-1)(q-1)$:

$$(d, (p-1)(q-1)) = 1.$$

$(p-1)(q-1)$ je sudé a proto d musí být liché. Nakonec vypočítáme e tak, aby $(de-1)$ bylo beze zbytku dělitelné $(p-1)(q-1)$, tj. aby platilo

$$de = 1 \bmod (p-1)(q-1).$$

Dvojice (e, n) tvoří veřejný klíč, číslo d tvoří tajný klíč.

4.1.1 Šifrování zprávy pomocí RSA

zprávu M zašifrujeme do kryptogramu C jako

$$C = M^e \bmod n.$$

Od tohoto okamžiku zprávu C dešifrujeme jen za předpokladu znalosti tajného klíče d:

$$M = C^d \bmod n.$$

Důkaz korektnosti šifrování RSA viz např. [Paseka, strana 101].

4.1.2 Dvoustranná komunikace šifrovaná RSA

Přijímající straně stačí vygenerovat dvojici klíčů a zveřejnit (n, e). Odesílající strana vytvoří kryptogram. Při dvoustranné komunikaci každá ze stran A, B vytvoří svoji dvojici klíčů (n_A, e_A), resp. (n_B, e_B), a ponechá si v tajnosti privátní klíč d_A resp. d_B.

Strana A zašifruje	Analogicky B šifruje
$C = M^{e_B} \bmod n_B.$	$\overline{C} = \overline{M}^{e_A} \bmod n_A$
Strana B dešifruje	a A dešifruje
$M = C^{d_B} \bmod n_B.$	$\overline{M} = \overline{C}^{d_A} \bmod n_A.$

4.1.3 RSA podepsiovací schéma

Dvojici klíčů lze použít k *autentizaci* zprávy takto: odesilatel podepíše zprávu M svým tajným klíčem d a kdokoli pak může na základě znalosti veřejného klíče (e, n) ověřit autentičnost odesilatele. Jinými slovy: veřejným klíčem ověříme, že odesílací strana je držitelem příslušného tajného klíče.

Ze zprávy M vytvoří odesilatel *signaturu*

$$S = M^d \bmod n.$$

Signaturu přiloží ke zprávě a přijímající strana veřejným klíčem (e, n) ověří, že byla vytvořena tajným klíčem d:

$$M = S^e \bmod n.$$

Stejně jako u šifrování ukážeme, že $(M^d \bmod n)^e \bmod n = M$:

$$(M^d \bmod n)^e \bmod n = (M^d)^e \bmod n = M^n \bmod n = M.$$

Dvoustranná šifrovaná a podepsaná komunikace potom probíhá takto:

strana A podepíše zprávu	analogicky B podepíše zprávu
$S = M^{d_A} \bmod n_A$	$\overline{S} = \overline{M}^{d_B} \bmod n_B$
a zašifruje	a zašifruje
$C = S^{e_B} \bmod n_B.$	$\overline{C} = \overline{S}^{e_A} \bmod n_A.$
Strana B dešfiruje	Strana A dešfiruje
$S = C^{d_B} \bmod n_B$	$\overline{S} = \overline{C}^{d_A} \bmod n_A$
a ověří signaturu	a ověří signaturu
$M = S^{e_A} \bmod n_A.$	$\overline{M} = \overline{S}^{e_B} \bmod n_B.$

Odkazy: [Web-RSA],[Wiki-RSA]

4.1.4 Útoky na RSA

Jsou-li prvočísla p a q dostatečně velká, je útok hrubou silou prakticky vyloučen. Velikost čísla n udávaná v bitech se nazývá *délka klíče* a volí se 1024, 2048 a někdy i více bitů. Lze se však pokusit o faktorizaci $n = pq$. Jestliže

útočník zná p a q, dá se dopočítat d, viz [Paseka, strana 118]. Počítání je snadnější, pokud p i q jsou konkruentní s 3 modulo 4, tj. $p \equiv q \equiv 3(\mathrm{mod}4)$, proto se taková čísla již nevolí.

Mnohem vážněji je však RSA ohroženo postranními kanály, viz [Klíma-02]. Učebnicovým příkladem postranních útoků je karta Cryptoworks. V její verzi tzv. BIOS 3 bylo možné zjistit časovým postranním kanálem (*timingem*) signaturu zprávy, která kartu informovala o oprávnění pro příjem dalších kanálů a uživateli proto stačilo mít kartu libovolnou, aby mohl sledovat i nepředplacené kanály.

Odesláním instrukce `00 48` se na kartu dohrávají další oprávnění a tato zpráva musí být ukončena *signaturou*, což je zřejmě signatura RSA podepisovacího schématu. Díky chybě na kartě šlo zkoušet postupně pro každý byte signatury jednotlivé hodnoty. Krátká doba odezvy znamenala úspěch, delší neúspěch. Takto byla signatura rekonstruována během několika minut a kartě podstrčena identita centrály Conditional Access systému.

Literatura

[Web-irde.to] WWW stránka, na které byl poprvé uveřejněn Common Scrambling Algorithm.
http://csa.irde.to

[DRM-02] Popis základních analogových a digitálních CA systémů.
http://www1.his.no/prosjekt/sikt/SIKT-rapporter%5CR_4_1_DRM_VDSL_May_2002.pdf

[VC-97] Popis systému VideoCrypt.
http://www.cl.cam.ac.uk/~mgk25/vc-slides.pdf

[Web-YV] Přehled formátů YUV obrazových dat.
http://graphics.cs.uni-sb.de/Courses/ss03/Multimedia/uebungen/YUV.html

[Web-MPEG] Základní informace o skupině MPEG.
http://www.mpeg.org/MPEG/starting-points.html#mpeg

[FAQ-MPEG2] Základní informace o MPEG a MPEG2.
http://bmrc.berkeley.edu/frame/research/mpeg/mpeg2faq.html

[MPEG-TS] Popis transportního proudu MPEG2.
http://erg.abdn.ac.uk/research/future-net/digital-video/mpeg2-trans.html

[ETSI] Hlavní stránka ETSI.
http://www.etsi.org

[WWW-CA] Základní informace o Conditional Access.
http://www.dtg.org.uk/reference/tutorial_ca.html

[FAQ-SC] Smartcard FAQ.
http://smartcard.nist.gov/faq.html

[ALT-SC] Smartcard FAQ.
http://www.scdk.com/atsfaq.htm

[Web-CAM] WWW stránka s informacemi o Cryptoworks CAM.
http://www.software.philips.com/cryptotec/section-13549/index.html

[LET-CAM] Propagační leták Philips s vlastnostmi Cryptoworks CAM.
http://www.software.philips.com/assets/Downloadablefile/CASmodule(1)-12845.pdf

[Laws] Neal R. Wagner. The Laws of Cryptography with Java Code.
http://www.cs.utsa.edu/~wagner/lawsbookcolor/laws.pdf

[Paseka] Jan Paseka. Kryptografie. Učební text na PřF MU Brno. 1998
ftp://www.math.muni.cz/pub/math/people/Paseka/lectures/2stkryptografie.ps

[Klíma-03] Vlastimil Klíma, Tomáš Rosa. Kryptologie pro praxi (2) – symetrická a asymetrická kryptografie, Sdělovací technika, 7/2003, str. 16.
http://cryptography.hyperlink.cz/2003/st_2003_07_16_16.pdf

[Wiki-FSR] Definice posouvacího registru.
http://en.wikipedia.org/wiki/Shift_register

[Wiki-LSR] Definice posouvacího registru se zpětnou vazbou.
http://en.wikipedia.org/wiki/Linear_feedback_shift_register

[Wiki-Perm] Definice permutace.
http://en.wikipedia.org/wiki/Permutation

[Philips] Popis přístroje Philips DSX 6010.
http://www.antenysatelity.cz/upcdirect/popisproduktu.htm

[Klíma-02] Vlastimil Klíma. RSA v novém světle. Ochrana před postrannímí útoky na RSA – článek v časopise Chip.
http://cryptography.hyperlink.cz/2002/chip-2002-01-126-129.pdf

[Wirt-04] Kai Wirt. Fault Attack on the DVB Common Scrambling Algorithm
http://eprint.iacr.org/2004/289.pdf

[WW-04] Ralf-Philipp Weinmann, Kai Wirt. Analysis of the DVB Common Scrambling Algorithm.
http://www.cdc.informatik.tu-darmstadt.de/~kwirt/csa.pdf

[Web-ICZ] Přednáškové slidy o postranních útocích
ftp://ftp.decros.cz/pub/Archiv/Publications/2002/bin_vktr.ppt

[Wiki-FA] Popis Floydova algoritmu na hledání cyklů
http://en.wikipedia.org/wiki/Floyd's_cycle-finding_algorithm

[DEF-LK] Definice lineární kryptoanalýzy
http://www.x5.net/faqs/crypto/q59.html

[Web-RSA] Paul Johnston. Popis algoritmu RSA
http://pajhome.org.uk/crypt/rsa/rsa.html

[Wiki-RSA] Popis algoritmu RSA – Webová encyklopedie
http://en.wikipedia.org/wiki/RSA

[KR-02] Vlastimil Klíma, Tomáš Rosa. Further Results and Considerations on Side Channel Attacks on RSA
http://eprint.iacr.org/2002/071

[Web-PKCS] Public-Key Cryptography Standards vydané firmou RSA Security
http://www.rsasecurity.com/rsalabs/node.asp?id=2124

[KR-05] Vlastimil Klíma, Tomáš Rosa: Kryptologie pro praxi (21) Achillova pata RSA, Sdělovací technika, 4/2005, str. 18-19
http://cryptography.hyperlink.cz/2005/ST_2005_04_18_19.pdf

[CRT] A. Bogomolny. Čínská věta o zbytku
http://www.cut-the-knot.org/blue/chinese.shtml

[CWFaq] CRYPTO****S FAQ. Popis známých instrukcí pro smartkartu Cryptoworks
http://www.id2.cz/sw_upc/CWfaq15_250801.zip

[CWPh] Diskusní skupina CRYPTO****S na serveru satellite.center.sk
http://satellite.center.sk/forumdisplay.php?forumid=4

[MITM] N. Asokan, Valtteri Niemi, Kaisa Nyberg. Man-in-the-Middle in Tunnelled Authentication Protocols
http://eprint.iacr.org/2002/163

[Klíma-VA] Vlastimil Klíma, Tomáš Rosa. Vybrané aspekty moderní kryptoanalýzy. Sdělovací technika, 3/2003, str. 3-7
http://cryptography.hyperlink.cz/2003/st_2003_03_03_07.pdf

[DL-tut] Howard M. Heys. A tutorial on Linear and Differential cryptanalysis.
http://www.engr.mun.ca/ howard/PAPERS/ldc_tutorial.pdf

[Wiki-CA] Definice pojmu kryptoanalýza a výčet jejích základních směrů.
http://en.wikipedia.org/wiki/Cryptanalysis

[Ber-04] Jan Berousek. Fermatova čísla. Bakalářská práce na PřF MUNI Brno.

Seznam tabulek

Seznam obrázků