Základy moderní kryptologie - Symetrická kryptografie II.

Vlastimil Klíma

verze: 1.3, 5. 4. 2005

Abstrakt

Cílem třech přednášek (Symetrická kryptografie I, II a III) je

a) ukázat, že moderní kryptologie se zabývá mnohem širším okruhem věcí než jen utajováním zpráv a jejich luštěním,
b) seznámit s některými novými myšlenkami,
c) a věnovat se více jedné části moderní kryptologie, tzv. symetrickým schématům.

Vzhledem k rozsahu těchto přednášek, které mají úvodní přehledový charakter, nebude možné postihnout ani klíčové, ani nejkrásnější myšlenky této vědy, ale jen některé nejpoužívanější. Následující texty vychází částečně z citované a doporučené literatury, jsou však nutně zatíženy subjektivním výkladem.

Doporučená literatura:

*Základní příručka **on-line**:*
Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996, dostupné celé on-line na
http://www.cacr.math.uwaterloo.ca/hac/

Často doporučovaná alternativa:
Doug Stinson, *Cryptography: Theory and Practice*, CRC Press, 1995

Historie:
David Kahn, *The Codebreakers*, Scribner, 1996

Internetový portál:
http://theory.lcs.mit.edu/~rivest/crypto-security.html

Kontakt a osobní stránky:
v.klima@volny.cz, http://cryptography.hyperlink.cz

Obsah

7. Symetrické šifrovací systémy

Nyní se budeme věnovat **kryptografickým systémům, které zajišťují bezpečnostní službu důvěrnosti dat**, a to pomocí kryptografického nástroje šifrování dat. Tyto systémy se nazývají **šifrovací systémy** neboli **šifry**. Nejprve si uvedeme definici, která platí jak pro symetrické, tak pro asymetrické šifry. Poté se budeme věnovat už jen symetrickým šifrám.

7.1. Kryptografický systém pro šifrování zpráv (šifra) - symetrický i asymetrický

Definice: Kryptografický systém pro šifrování zpráv (šifra)
Kryptografický systém pro šifrování zpráv je pětice (M, C, K, E, D), kde M je prostor otevřených zpráv, C prostor šifrových zpráv a K prostor klíčů. E, D je dvojice **zobrazení**, které každému klíči $k \in K$ přiřazují **transformaci pro zašifrování zpráv E_k a transformaci pro dešifrování** zpráv D_k, E_k: $M \rightarrow C$: $m \rightarrow c$ a D_k: $C \rightarrow M$: $c \rightarrow m$, přičemž pro každé $k \in K$ a $m \in M$ platí $D_k(E_k(m)) = m$.

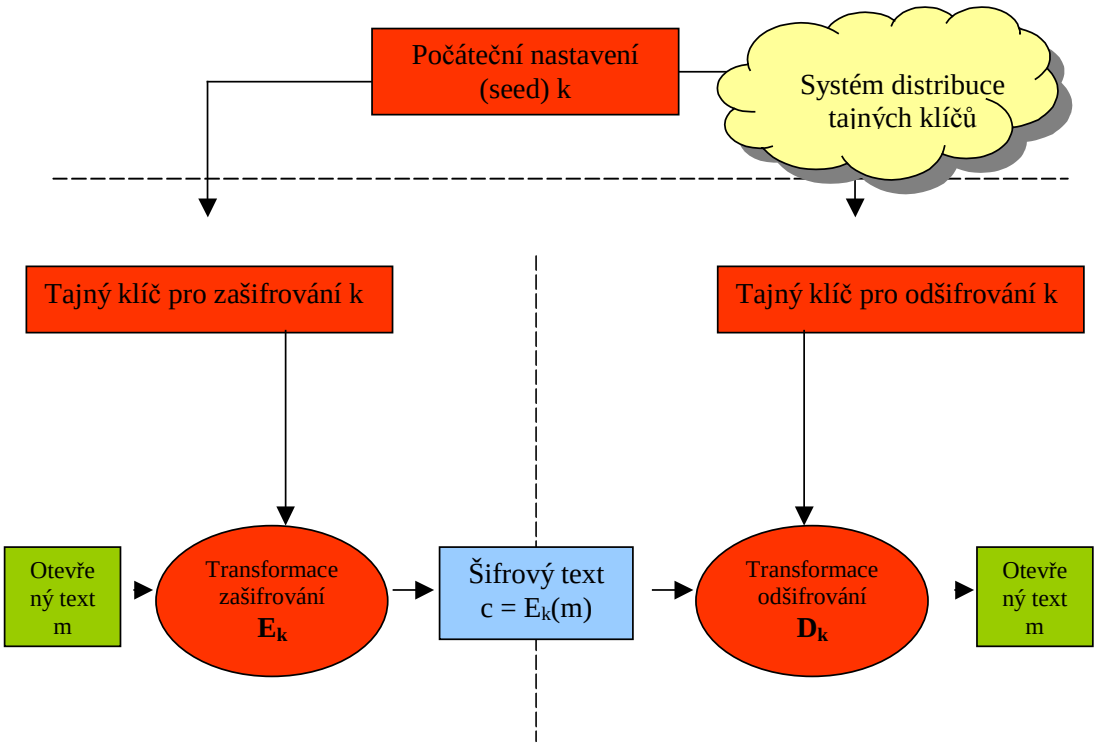
7.2. Symetrický kryptografický systém pro šifrování zpráv (symetrická šifra)

Nyní si uvedeme definici symetrické šifry.

Definice : Symetrický kryptografický systém pro šifrování zpráv (symetrická šifra)
Symetrická šifra je taková šifra, kde pro každé $k \in K$ lze z transformace zašifrování E_k určit transformaci dešifrování D_k a naopak.

Poznámka: Z důvodu této symetrie se tyto systémy nazývají **symetrické systémy** a jejich klíče **symetrické klíče**. Symetrické klíče jsou tajné, zatímco obě zobrazení E a D mohou být zcela veřejné, jako je tomu například u šifrovacích standardů DES a AES.

Poznámka: Pro úplnost si uveďme definici asymetrické šifry.
Definice: Asymetrická šifra je taková šifra, kde pro skoro všechna $k \in K$ nelze z transformace pro zašifrování E_k určit transformaci pro dešifrování D_k. V praxi je u *asymetrických* šifer klíč *k* tajným nastavením, z kterého se vhodnou transformací G vygeneruje dvojice parametrů (e,d), které se nazývají po řadě veřejný (e) a privátní (d) klíč. Ty potom parametrizují transformace zašifrování a dešifrování, takže pro jednoduchost nepíšeme E_k a D_k, ale přímo E_e a D_d. Na obrázku vidíme Shannonův model komunikačního kanálu pro potřeby symetrických šifer.



Obr.: Symetrický šifrovací systém

Jako příklad symetrické šifry si vezměme **Caesarovu šifru**, kdy se šifruje pomocí posouvání písmen o tři pozice doprava v abecedě. Slovo CAESAR bude zašifrováno na FDGWFV. Při obecnějším posouvání o k písmen je příjemci i odesílateli nutné sdělit pouze tajný šifrovací klíč *k*, zde $k = 3$. Odesílatel pak šifruje pomocí transformace E_k, tj. posouvá písmena vpravo o *k* pozic, příjemce zprávu odšifruje transformací D_k, tj. posunem písmen o *k* pozic zpět. Povšimněme si, že transformace E_k a D_k nejsou totožné (posun doprava, posun doleva), ale z jedné lze odvodit druhou.

Poznámka: V praxi se příliš nerozlišují pojmy schéma, algoritmus a transformace. V dalším textu je proto pro jednoduchost nebudeme rozlišovat, pokud to nebude nutné. Připomeňme, že šifrovací systém může kromě šifrovacího algoritmu obsahovat i další zobrazení, algoritmy a pravidla, upřesňující a doplňující šifrovací a dešifrovací transformace, například generátor hesla, algoritmus pro definici transformací E_k a D_k z klíče $k \in K$ nebo pravidla použití klíčů. Například u Vernamovy šifry je to pravidlo, že pro šifrování každé zprávy se generuje nový náhodný šifrovací klíč. Takovou podmínku může definice transformace otevřeného textu na šifrový a obráceně jen stěží zahrnovat.

7.3. Shannonova teorie

Tento odstavec je zpracován zejména s použitím [12]. Shannon v roce 1948 a 1949 položil svými dvěma pracemi [1] a [2] základy teorie informace a teorie šifrovacích systémů. Stanovil teoretickou míru bezpečnosti šifry pomocí neurčitosti otevřeného textu, když je dán šifrový text: Jestliže se nic nového o otevřeném textu nedozvíme (například zúžení prostoru možných zpráv), i když přijmeme jakékoliv množství šifrového textu, šifra dosahuje **absolutní bezpečnosti** (**perfect secrecy**). Jinými slovy v tomto případě šifrový text nenese žádnou informaci o oteveném textu.

7.3.1. Vzdálenost jednoznačnosti

Zvyšování počtu znaků šifrového textu u většiny praktických a zejména u historických šifer

poskytuje stále více **informace o otevřeném textu**. Tato informace nemusí být přímo viditelná, ale nějakým i velmi komplikovaným, nepřímým a neznatelným způsobem je v šifrovém textu přítomna. V určitém bodě je této informace v šifrovém textu obsaženo takové množství, že je možný jen jediný otevřený text. Tento počet znaků šifrového textu se nazývá **vzdálenost jednoznačnosti**.
Uvedeme si příklad. Mějme například jednoduchouz záměnu. Pokud obdržíme jeden znak šifrového textu "Z", nemáme ještě žádnou informaci o textu otevřeném. Pravděpodobnost, že se pod šifrovým textem skrývá písmeno A (resp. B, C, ..., Z) je stejná jako pravděpodobnost výskytu písmene A (B, ..., Z) v otevřeném textu p_A (resp. p_B, ... , p_Z), tj. nedověděli jsme se o otevřeném textu žádnou informaci. V případě, že zvýšíme počet písmen šifrového textu na 2 - mějme například šifrový text "ZP" - už máme určitou informaci o otevřeném textu. Otevřeným textem nemohou být všechny možné bigramy, protože vylučujeme bigramy se stejnými písmeny. Ty by totiž dávaly šifrový text typu "ZZ". Při padesáti písmenech už v šifrovém textu může i laik rozeznávat samohláskové a souhláskové pozice a může je určovat. Při tisíci znacích šifrového textu je snad již jasné, že otevřený text je plně nebo až na detaily plně určen. Někde mezi číslem 1 a 1000 je tedy bod, kdy je otevřený text už jen jeden. Z praxe víme, že při určování vzdálenosti jednoznačnosti bude také záležet na jazyku otevřeného textu. S tím souvisí pojem entropie zdroje zpráv.

7.3.2. Entropie

Teorie informace měří množství informace, obsažené ve zprávě průměrným počtem bitů, nezbytných k jejímu zakódování při optimálním kódování (optimální kódování používá co nejméně bitů k zakódování zpráv). Například při vyplňování pohlaví v dotazníku píšeme „mužské“ nebo „ženské“. Informace obsažená v tomto sdělení není 48 bitů, což zabírá zakódování uvedených šesti znaků abecedy při použití kódové stránky CP 1250 pro středoevropské jazyky v OS Windows, ale pouze jeden bit, neboť se vybírá ze dvou možností. Množství informace ve zprávě je formálně měřena entropií. Zdroj zpráv produkuje různé zprávy s obecně různou pravděpodobností, například v diplomatické zprávě můžeme slovu „diplomat“ nebo „prezident“ přiřadit větší pravděpodobnost, než slovu "homomorfismus" nebo "algebra", zatímco v algebraické práci by tomu bylo naopak. Podobně bychom mohli přiřadit pravděpodobnosti všem možným celým větám nebo všem N-znakovým zprávám. Označíme-li všechny možné zprávy daného zdroje zpráv X jako X(1), X(2),..., X(n) a jejich pravděpodobnosti jako p(1), p(2), ..., p(n), pak entropie zprávy z tohoto zdroje je daná váženým průměrem:

$$H(X) = - p(1)*\log_2 p(1) - p(2)*\log_2 p(2) - \ ... \ - p(n)*\log_2 p(n)$$

Výraz $-\log_2 p(i) = \log_2(1/p(i))$ je počet bitů, nutných k optimálnímu zakódování zprávy X(i), vážený průměr přes všechny zprávy pak dává průměrný obsah bitů, obsažených ve zprávě.

7.3.3. Příklady

Příklad: Mějme dvě možné zprávy: „mužské“ nebo „ženské“ a nechť mají stejnou pravděpodobnost. Potom entropie zprávy z tohoto zdroje je podle výše uvedeného vzorce rovna H(X) = -0.5*(-1) - 0.5*(-1) = 1, tedy jeden bit.
Příklad: Mějme zdroj, který vydává zprávy : „mužské“ s pravděpodobností 1/4 a „ženské“ s pravděpodobností 3/4. Potom entropie zprávy z tohoto zdroje je $H(X) = -0.25*(-\log_2 4) - 0.75*(-\log_2(3/4)) = 0.25 * 2 + 0.75 * (\log_2(4/3)) = 0.25 * 2 + 0.75 * 0.41 = 0.81$, tedy necelý jeden bit, takže zprávy z tohoto zdroje obsahují v průměru necelý bit informace.
Příklad: Mějme zdroj, který vydává pouze jednu zprávu s pravděpodobností 1. Potom entropie takového zdroje je H(X) = - (1 * 0) = 0, tedy 0 bitů. Daný zdroj nemá žádnou neurčitost a zprávy z něj nenesou žádnou informaci.

Příklad: Mějme *n* možných zpráv, všechny se stejnou pravděpodobností, tj. náhodné řetězce délky $\log_2 n$ bitů. Potom entropie takového zdroje je $H(X) = - (1/n * \log_2(1/n)) * n = \log_2 n$, tedy $\log_2 n$ bitů na zprávu. Pokud jsou zprávy náhodné řetězce stejné délky, obsahuje každá zpráva přirozeně přibližně tolik informace, kolik bitů má.

Poznámka: maximální entropie

Lze dokázat, že maximální entropie nabývá zdroj, který produkuje všechny zprávy se stejnou pravděpodobností. Má-li zdroj n zpráv se stejnou pravděpodobností 1/n, pak dosahuje maximální entropie $\log_2 n$.

Poznámka: komprimační programy

Máme-li k dispozici velký soubor dat, lze získat orientační představu a horní odhad pro entropii tohoto souboru použitím kvalitního komprimačního programu. Entropie souboru je obvykle o něco nižší než počet bitů komprimovaného souboru.

7.3.4. Entropie, obsažnost a nadbytečnost jazyka

Pro daný jazyk uvažujme množinu X všech N-znakových zpráv. **Obsažnost jazyka** pro zprávy délky N znaků definujeme jako výraz $R_N = H(X)/N$, tj. průměrnou entropii na jeden znak (průměrný počet bitů informace v jednom znaku). Pokud by zprávy v daném jazyce tvořeném L stejně pravděpodobnými znaky, byly stejně pravděpodobné, dostali bychom $R = (\log_2(L^N)) / N = \log_2 L$. Tento výraz nazýváme **absolutní obsažnost jazyka (R)**. Absolutní obsažnost dosahuje takový jazyk, který poskytuje generátor náhodných znaků. Je to maximální neurčitost, kterou přirozené jazyky nemohou dosáhnout, neboť jednotlivé znaky tvoří slova a věty, které mají odlišné pravděpodobnosti. Dále si povšimněme, že u přirozených jazyků výraz R_N pro zvyšující se N klesá, neboť máme-li dlouhou zprávu, její další písmeno bývá v řadě případů určeno už jednoznačně nebo je možný jen malý počet variant. Například máme-li 13 znakovou zprávu „Zítra odpoled„ , je pravděpodobné, že pokračuje písmenem n. U 14. znaku nepřibude žádná entropie. U ostatních možných 13-znakových řetězců bude situace podobná - umožní jen jednu nebo několik málo variant. Entropie proto příliš nevzroste, takže výraz $R_N = H(X)/N$ o něco klesne oproti předchozí hodnotě R_N. Pokud se $R_N = H(X)/N$ pro N velké přibližuje konstantě (r), je možné tuto konstantu chápat jako **obsažnost jazyka vzhledem k jednomu písmenu (r)**. Obsažnost jazyka nám tedy říká, kolik bitů informace ve skutečnosti obsahuje průměrně jedno písmeno jazyka. Proto výraz D = R - r znamená, kolik bitů je v jednom znaku daného jazyka nadbytečných. Proto se nazývá **nadbytečnost jazyka vzhledem k jednomu písmenu (D)**. Číslo D/R pak udává, kolik bitů jazyka je nadbytečných procentuálně. Pro angličtinu je to například 3.2/4.7 = 68 % bitů.

Příklad:

Pro angličtinu máme
L = 26
$R = \log_2 26 = 4.7$ bitů na písmeno
r = 1.5 bitů na písmeno
D = 3.2 bitů na písmeno

7.3.5. Výpočet vzdálenosti jednoznačnosti

Následující úvaha je heuristická, neplatí pro všechny šifry, ale je dostatečně ilustrativní. Mějme množinu zpráv M, množinu šifrových textů C a množinu klíčů K. Uvažujeme-li klíče stejně pravděpodobné, máme $2^{H(K)}$ klíčů. Předpokládejme, že máme šifrový text c délky N znaků a že pro klíče $k \in K$ jsou odpovídající otevřené texty $D_k(c)$ vybírány z množiny všech zpráv M nezávisle a náhodně. V množině M je celkem 2^{RN} zpráv a z toho je 2^{rN} smysluplných

zpráv a $U = 2^{RN} - 2^{rN}$ zpráv nesmysluplných. Pokud provedeme dešifrování šifrového textu c všemi možnými $2^{H(K)}$ klíči, dostáváme $2^{H(K)}$ zpráv. Z nich je smysluplných průměrně pouze $S = 2^{H(K)} * (2^{rN} / 2^{RN}) = 2^{H(K)} / 2^{DN} = 2^{H(K)-DN}$. Abychom dostali pouze jednu zprávu - tu, která byla skutečně zašifrována - musí být S = 1. Odtud dostáváme H(K) = DN a N = H(K)/D. **Vzdálenost jednoznačnosti je definována jako N = H(K)/D,** kde H(K) je neurčitost klíče a D je nadbytečnost jazyka otevřené zprávy.

7.3.6. Příklady výpočtu vzdálenosti jednoznačnosti

Příklad: vzdálenost jednoznačnosti jednoduché substituce

Mějme obecnou jednoduchou substituci nad anglickou abecedou. Její vzdálenost jednoznačnosti je **$N = H(K)/D = \log_2(26!)/3.2 = 88.3/3.2 = 27.6$.** V šifrovém textu o 28 znacích je tedy dostatečné množství informace na to, aby zbýval v průměru jediný možný otevřený text. K rozluštění jednoduché substituce v angličtině postačí tedy v průměru 28 písmen šifrového textu.

Příklad: vzdálenost jednoznačnosti u Vigenerovy šifry

Mějme klíč Vigenerovy šifry o délce V *náhodných* znaků a uvažujme otevřený text z anglické abecedy. Vzdálenost jednoznačnosti je **$N = H(K)/D = \log_2(26^V)/3.2 = V * \log_2(26)/3.2 = V*4.7/3.2 = V*1.5$.** To je na první pohled optimistický výsledek, který je nutno vysvětlit. Dejme tomu, že máme šifrový text v délce jeden a půl násobku hesla, tj. oněch V*1.5 znaků. První a třetí polovina textu používá stejné heslo, které lze eliminovat odečtením a poté s určitou pravděpodobností vyluštit první a třetí polovinu otevřeného textu. Zbývá neznámá druhá polovina otevřeného textu, kde je heslo zcela náhodné a neznámé, nemáme tedy z něj žádnou informaci. Zbývá redundance otevřeného textu, z něhož známe první a třetí polovinu. Vzorec poskytuje střední hodnotu vzdálenosti jednoznačnosti, nebere v potaz, že máme k dispozici právě takové rozložení informace z otevřeného textu. V zásadě však z hlediska množství informace pokud máme dvě třetiny otevřeného textu, zbytek bychom měli být schopni u anglického jazyka doplnit. Problém je v tom, že na krátkých úsecích nedosahuje nadbytečnost jazyka hodnoty 3.2 bitu, ale méně, a druhý problém je v tom, že konkrétní rozložení informace o otevřeném textu je v daném případě atypické (je známa první a poslední třetina textu), tedy není možné na něj vztahovat výsledky, týkající se *průměru* a průměrných - obvyklých textů. Na druhou stranu, pokud bychom měli k dispozici 2*V znaků, budeme už schopni heslo eliminovat a luštit metodou knižní šifry. Kdybychom měli o pár znaků méně, byli bychom ještě schopni tyto znaky doplnit právě z důvodu nadbytečnosti zprávy. Skutečnou vzdálenost jednoznačnosti lze proto v závislosti na konkrétním problému a konkrétní hodnotě V očekávat v rozmezí 1.5 až 2.0 násobku délky hesla.

Příklad: vzdálenost jednoznačnosti u transpozice

Mějme obecnou transpozici v délce bloku V = 10 nad anglickou abecedou. Její vzdálenost jednoznačnosti je $N = H(K)/D = \log_2(V!)/3.2 = \log_2(10!)/3.2 = 21.8$. V šifrovém textu o 22 znacích (ve skutečnosti 2 nebo 3 úplné bloky o 10 písmenech) je tedy dostatečné množství informace na to, aby zbýval v průměru jediný otevřený text. Uvedený výsledek koresponduje s možností luštění do hloubky, kdy pod sebe napíšeme 2 - 3 bloky šifrového textu o 10 písmenech. Při luštění již můžeme využít 2 - 3 bigramové vazby na každou dvojici sloupců.

Poznámka: Při výpočtu vzdálenosti jednoznačnosti u transpozice bychom měli udělat korekci ve vzorci pro N. Vrátíme se proto k úvaze, kolik vznikne smysluplných textů po odšifrování daného šifrového textu všemi klíči. Protože šifrový text odpovídající transpozici zachovává četnosti znaků stejné jako v otevřeném textu, musíme vždy jeho odšifrováním obdržet text, zachovávající četnosti písmen otevřeného jazyka. Smysluplných textů je po odšifrování daného šifrového textu všemi klíči tentokrát nikoli $S = 2^{H(K)} * (2^{rN} / 2^{RN}) = 2^{H(K)} /$

2^{DN} , ale $S = 2^{H(K)} * (2^{rN} / 2^{FN})$, kde 2^{FN} je počet všech možných (smysluplných i nesmysluplných) textů, jejichž četnosti znaků odpovídají četnostem znaků v daném jazyce. Máme $F = -p(A)*\log_2 p(A) - p(B)*\log_2 p(B) - ... p(Z)*\log_2 p(Z)$. Odtud $N = H(K) / (F - r)$.

Příklad: vzdálenost jednoznačnosti u blokové šifry AES
Uvažujme otevřený text nad anglickou abecedou a blokovou šifru AES (má délku bloku 128 bitů) se 128bitovým klíčem. Důležité je, v jaké formě je otevřený text šifře předkládán, tj. jak je kódován do 128bitového vstupu. Uvažujme běžnou počítačovou praxi, kdy jeden znak otevřeného textu je reprezentován jedním bajtem. V takovém případě máme z 8 bitů vstupu pouze 1.5 bitu informace, čili nadbytečnost **D je v tomto případě 6.5 bitu na bajt**. Vzdálenost jednoznačnosti je tedy **$N = H(K)/D = \log_2(2^{128})/6.5 = 128/6.5 = 19.7$ bajtů otevřeného textu**. Potřebovali bychom tedy 19.7 bajtů šifrového textu, což je jeden celý blok (16 bajtů) a část druhého bloku. Dva bloky šifrového textu by proto měly být dostačující.

7.3.7. Vzdálenost jednoznačnosti a složitost

Vzdálenost jednoznačnosti nám dává odhad množství informace, nutného k vyluštění dané úlohy. Neříká však nic o složitosti takové úlohy. To je dobře patrné na vzdálenosti jednoznačnosti u šifry AES, kde víme, že informace, obsažená ve dvou blocích šifrového textu je dostačující k určení otevřeného textu, ale nevíme, jak tento otevřený text určit, kromě útoku hrubou silou na klíč.

8. Proudové šifry

8.1. Rozdíl mezi blokovými a proudovými šiframi

Z hlediska použití klíče ke zpracování otevřeného textu rozeznáváme dva základní druhy symetrických šifer - proudové a blokové. Nechť otevřený text používá vstupní abecedu **A** o q symbolech. **Proudová šifra šifruje zvlášť jednotlivé znaky abecedy, zatímco bloková šifra zpracovává najednou bloky (řetězce) délky *t* znaků**. Podstatné na blokových šifrách však je, že všechny bloky jsou šifrovány (dešifrovány) stejnou transformací E_k (D_k), kde *k* je šifrovací klíč. Naproti tomu proudové šifry nejprve z klíče *k* vygenerují posloupnost h(1), h(2),... a každý znak otevřeného textu šifrují jinou transformací - $E_{h(i)}$.

8.2. Definice obecné proudové šifry

Klasická definice proudových šifer zní, že zpracovávají otevřený text po znacích, zatímco blokové šifry po blocích t znaků. Proudové šifry by tedy mohly být chápány i jako blokové šifry s blokem délky t =1, avšak připomeňme, že tou podstatnou odlišností je, že u proudových šifer je každý tento "blok" zpracováván jiným způsobem, jinou substitucí.

Definice: symetrická *proudová* šifra

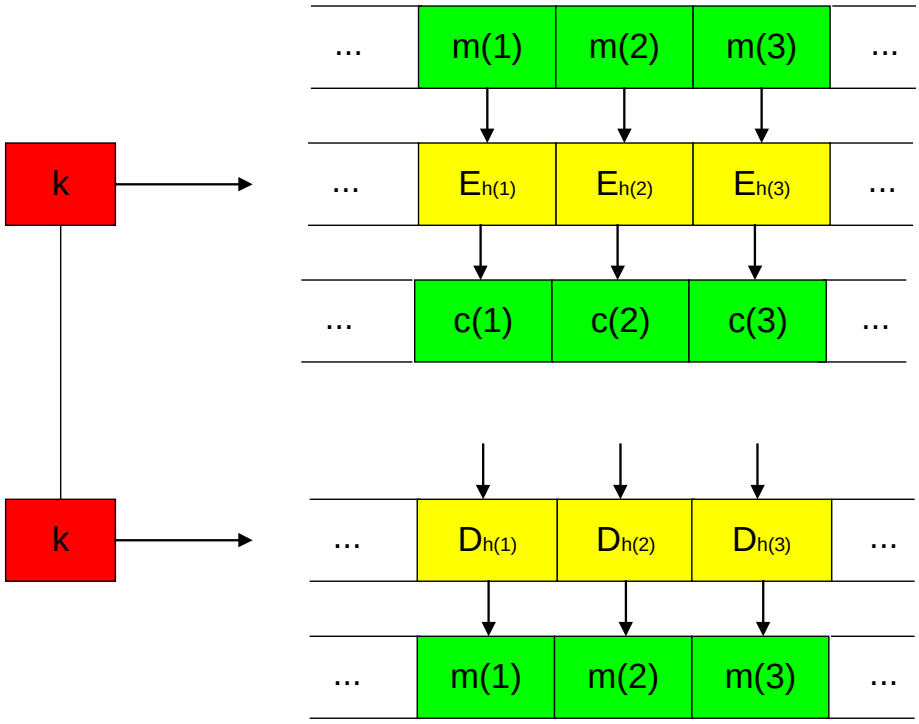
Nechť A je abeceda q symbolů, nechť M = C je množina všech konečných řetězců nad A a nechť K je množina klíčů. Proudová šifra se skládá z transformace (generátoru) G, zobrazení E a zobrazení D. Pro každý klíč k∈ K generátor G vytváří posloupnost hesla h(1), h(2),... , přičemž prvky h(i) reprezentují libovolné substituce $E_{h(1)}$, $E_{h(2)}$, ... nad abecedou A. Zobrazení E a D každému klíči k ∈ K přiřazují transformace zašifrování E_k a odšifrování D_k. Zašifrování otevřeného textu m = m(1), m(2), ... probíhá podle vztahu

$c(1) = E_{h(1)}(m(1))$, $c(2) = E_{h(2)}(m(2))$, ...

a dešifrování šifrového textu c = c(1), c(2), ... probíhá podle vztahu

$m(1) = D_{h(1)}(c(1))$, $m(2) = D_{h(2)}(c(2))$, ...

kde $D_{h(i)} = E_{h(1)}^{-1}$.



Obr.: Princip proudových šifer

Poznámka:

- Z historických důvodů nazýváme G generátor hesla, neboť h(1), h(2) ,... bývá proud znaků abecedy A a substituce $E_{h(i)}$ posunem v abecedě A o h(i) pozic, tj. c(i) = (m(i) + h(i)) mod q. Proudové šifry jsou příkladem historických tzv. **heslových systémů**. V anglické literatuře se heslo h(1), h(2) ,... nazývá **running-key** nebo **key-stream** (keystream), tj. proud klíče, i když se jedná o derivát originálního klíče k.
- Pokud se proud hesla začne od určité pozice opakovat, říkáme, že jde o periodické heslo a periodickou šifru.
- Vigenerova šifra je periodickou šifrou.

8.3. Moderní proudové šifry

Moderní proudové šifry pracují nad abecedou A={0,1}, tj. q = 2. Sčítání modulo 2 se nazývá binární sčítání, a protože a + b je rovno a - b, je to i binární odečítání, tedy vlastně diference bitů. V počítačové praxi se pro binární sčítání používá označení xor a značka ⊕ . Jedinou smysluplnou substitucí $E_{h(i)}$ nad otevřeným bitem abecedy m(i) je vlastně transformace $E_{h(i)}(m(i)) = m(i) + h(i)$ nebo $E_{h(i)}(m(i)) = m(i) + h(i) + 1$, tedy vlastně pouze $E_{h(i)}(m(i)) = m(i) + h(i)$, jak ukazuje tabulka.

otevřený text	**1.transformace**	**2.transformace**	**3.transformace**	**4.transformace**
0	0	0	1	1
1	0	1	0	1
transformace $E_{h(i)}$	$E_{h(i)}(m(i)) = 0$	$E_{h(i)}(m(i)) = m(i)$	$E_{h(i)}(m(i)) = m(i) + 1$	$E_{h(i)}(m(i)) = 1$
poznámka	neexistuje $D_{h(i)}$	$E_{h(i)}(m(i)) =$ **$m(i) + h(i)$** je-li h(i) = 0 nebo $E_{h(i)}(m(i)) =$ **$m(i) + h(i) + 1$** je-li h(i) = 1	$E_{h(i)}(m(i)) =$ **$m(i) + h(i) + 1$** je-li h(i) = 0 nebo $E_{h(i)}(m(i)) =$ **$m(i) + h(i)$** je-li h(i) = 1	neexistuje $D_{h(i)}$

Obr.: Substituce nad binární abecedou

Skutečně, u moderních proudových šifer šifrový text (ŠT) vzniká tak, že jednotlivé bity H proudu hesla jsou postupně slučovány s jednotlivými bity proudu otevřeného textu OT binárním sčítáním. Schématicky bychom zašifrování zapsali jako

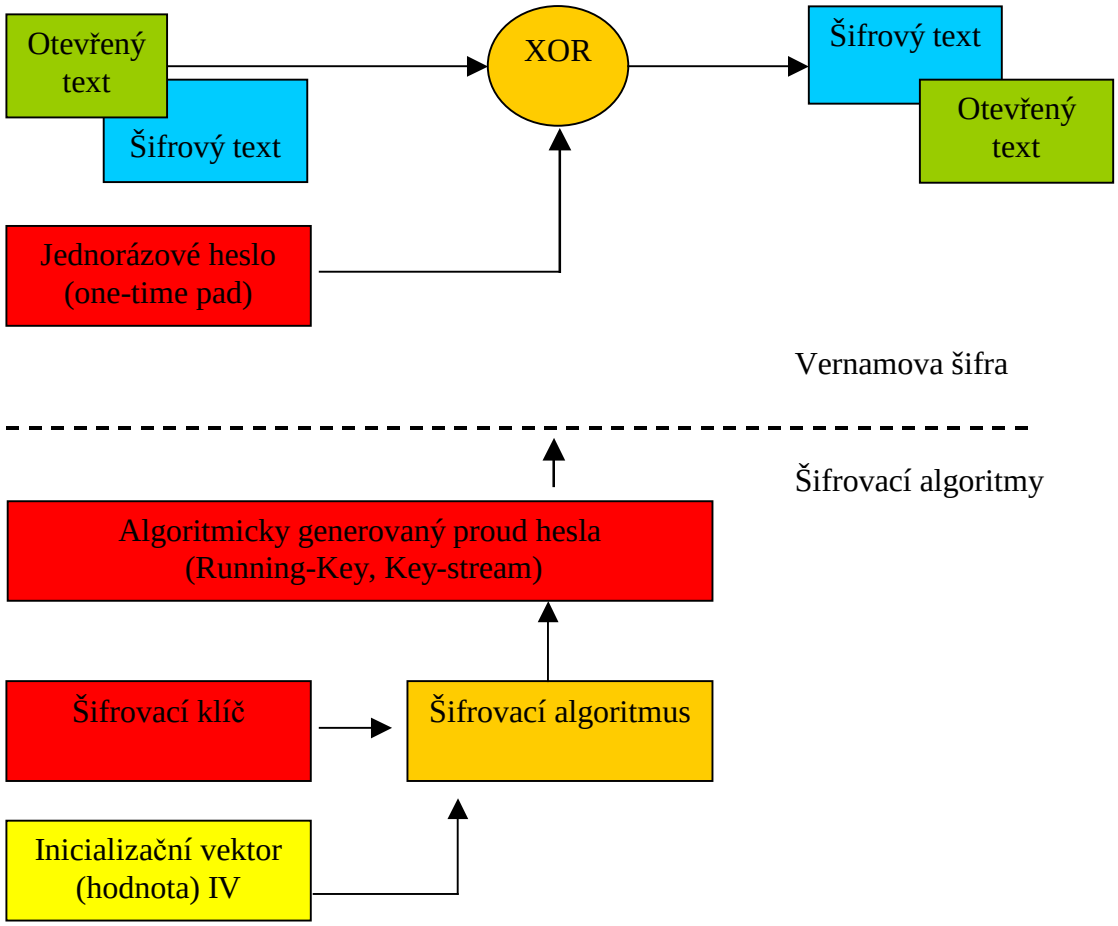
ŠT = OT + H

a odšifrování jako OT = ŠT + H. Poznamenejme, že díky rovnosti binárního sčítání a odčítání je transformace pro zašifrování a odšifrování také stejná. Jako u všech symetrických šifer, odesílatel i příjemce musí mít k dispozici tentýž klíč, tj. totéž heslo.
Heslo může být vytvořeno zcela náhodně jako u Vernamovy šifry nebo může být vygenerováno deterministicky nějakým šifrovacím algoritmem na základě šifrovacího klíče.

8.4. Vernamova šifra

Vernamova šifra používá náhodné heslo stejně dlouhé jako otevřený text a po použití se ničí, takže nikdy není použito k šifrování dvou různých otevřených textů. Tento způsob šifrování navrhl major americké armády Joseph Mauborgn krátce po první světové válce, ale nazývá se Vernamova šifra po Gilbertu Vernamovi, který si ji nechal patentovat. Gilbert Vernam si jako

zaměstnanec americké telefonní a telegrafní společnosti ATT nechal v roce 1917 patentovat šifrovací zařízení pro ochranu telegrafických zpráv, v němž na otevřený text, reprezentovaný pěticemi bitů (v 32 znakovém Baudotově kódu) se bit po bitu binárně načítá náhodná posloupnost bitů klíče, vyděrovaného na papírové pásce. Klíčová posloupnost se generovala náhodně a použité heslo se ničilo.



Obr.: Vernamova šifra a princip proudových šifer

8.5. Absolutně bezpečná šifra

Ukážeme, že Vernamova šifra má **vlastnost absolutní bezpečnosti** (anglicky perfect secrecy, dokonalé utajení, v češtině se ale vžil pojem absolutně bezpečná šifra).
Nechť h(i), o(i) a c(i) jsou po řadě bit hesla, otevřeného a šifrového textu. Máme P{o(i) = 0} = P{c(i) - h(i) = 0} = P{h(i) = c(i)}.
Tento výraz je roven
P{h(i) = 0}, v případě, že c(i) = 0
P{h(i) = 1}, v případě, že c(i) = 1.
Protože P{h(i) = 0} = P{h(i) = 1} = 1/2, je v obou případech výraz roven 1/2, tedy celkově P{o(i) = 0} = 1/2. Obdobně ukážeme, že P{o(i) = 1} = 1/2.
Odtud vyplývá, že P{o(i) = 0} = P{o(i) = 1} = 1/2 nezávisle na hodnotě šifrového textu. Jinými slovy šifrový text nenese žádnou informaci o otevřeném textu, což je definice absolutně bezpečné šifry.

8.6. Použití Vernamovy šifry

Vernamova šifra se používala a mnohde ještě používá **pro ochranu nejdůležitějších** (zejména diplomatických) **spojů, kde je nutné mít zajištěnu absolutní bezpečnost**.

Nevýhodou je nutnost distribuovat heslo na obě strany komunikačního kanálu (například na ministerstvo zahraničních věcí a na zastupitelský úřad). Dříve se heslo děrovalo do **děrných pásek** a bylo na zastupitelské úřady dopravováno **v diplomatických zavazadlech**, dnes mohou být nosiče těchto klíčových materiálů jiné, ale podstata zůstává stejná. Heslo ovšem může být použito k zašifrování jen jednou a poté se musí zničit (totéž při odšifrování). V praxi se ta část děrné pásky s heslem, která se použila, okamžitě skartovala. Víme, že pokud by se totéž heslo použilo pro šifrování ještě jiného textu, mohlo by to být detekováno a poté oba dva otevřené texty rozluštěny (viz **knižní šifra a metoda předpokládaného slova**). V praxi tyto případy skutečně nastávaly a odtajněné materiály dosvědčují, že příslušné otevřené texty, šifrované tímtéž heslem, byly také úspěšně luštěny.

8.7. Algoritmické proudové šifry

Místo transportu **děrných pásek** se časem pro generování hesla začaly používat kryptografické algoritmy. Nejprve to byly **mechanické šifrátory, poté elektrické a nakonec elektronické šifrovací stroje**. Heslo se těmito šifrátory "vypočítávalo" (generovalo) a distribuovaly se pouze šifrovací klíče pro nastavení těchto šifrátorů. Aby klíč nemusel být měněn příliš často, zavedl se **princip náhodně se měnícího inicializačního vektoru (IV)**. IV byl pro každou zprávu vybírán náhodně a byl přenášen před šifrovým textem v otevřené podobě. Inicializační vektor (za účasti tajného klíče nebo bez něj) nastavuje příslušný algoritmus (konečný automat, šifrátor) vždy do jiného (náhodného) počátečního stavu, čímž by měla být i při stejném tajném klíči generována pokaždé jiná heslová posloupnost. **Za různost hesla zodpovídá IV, za utajenost zodpovídá tajný šifrovací klíč**. Tento princip se s malou obměnou využívá i u blokových šifer.

8.7.1. Příklad proudové šifry: RC4

K nejpoužívanějším šifrám na internetu patří proudová šifra RC4, navržená prof. Rivestem v roce **1987**. Nevyužívá inicializační vektor, proto musí na každé spojení tajný klíč generovat nově, náhodně. Počítá se s tím, že příjemci se dopraví pomocí nějaké asymetrické metody. Přestože je jednou z nejpoužívanějších šifer na internetu a součástí internetových a jiných standardů, dosud její popis nebyl oficiálně publikován, i když je znám. Byl zveřejněn neznámým hackerem v roce 1994, který ho získal disassemblováním z programu BSAFE společnosti RSA. Pokud se někde setkáte se šifrou "Arcfour", je to právě RC4 a tento trik je dělán z důvodu ochrany autorských práv a obchodního tajemství společnosti RSA. RC4 používá mnoho protokolů a standardů, například S/MIME a SSL. RC4 umožňuje volit délku klíče, nejvíce používané jsou délky 40 a 128 bitů.

Z klíče substituce

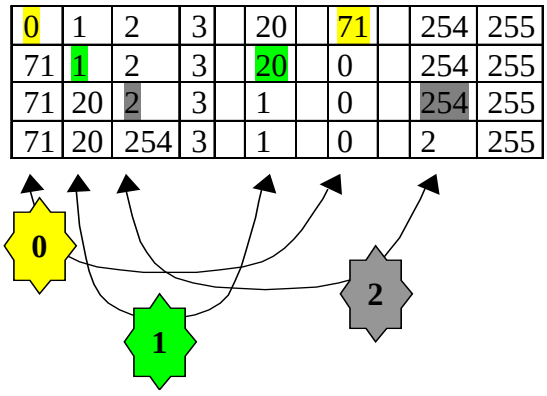
Šifrovací klíč pro RC4 se používá pouze k tomu, aby vygeneroval tajnou substituci {0, ..., 255} $\rightarrow$ {0, ..., 255}, tedy substituci bajtu za bajt. Popis tvorby tabulky pro stručnost vynecháváme, lze ho najít například v [24]. Pomocí tabulky S se pak konečným automatem generují jednotlivé bajty hesla h(0), h(1), ..., které se xorují na otevřený nebo šifrový text.

Z náhodné posloupnosti permutace

Nejprve připomeneme starou myšlenku, jak z posloupnosti 256 náhodných čísel r(0) ... r(255) v rozsahu 0 ... 255 získáme permutaci čísel 0 ... 255. V původní náhodné posloupnosti se samozřejmě mohou některá čísla z množiny {0, ..., 255} vyskytovat několikrát a některá vůbec ne. Jde o to je převést na posloupnost P, kde se každé z čísel 0 ... 255 vyskytne právě jednou.

Starý recept zní takto. P na počátku naplníme identickou permutací, tj. P(i) = i pro i = 0, ..., 255. Nyní pomocí naší náhodné posloupnosti *r* permutaci P mícháme. Míchání provádíme postupně pro i = 0 ... 255. V každém kroku *i* v permutaci P vyměníme hodnoty na pozicích *i* a

r(i), tedy hodnoty P(i) a P(r(i)) vzájemně zaměníme. Pointa je v tom, že P zůstává stále permutací. Přitom výměna postihne každou její pozici, protože index *i* projde všechny pozice. Druhý index r(i) zajišťuje náhodné míchání. Na konci máme novou permutaci *P* plně závislou na původní *náhodné posloupnosti r*. Tuto myšlenku, jen poněkud zesložitěnou, využívá algoritmus RC4 ke generování hesla.

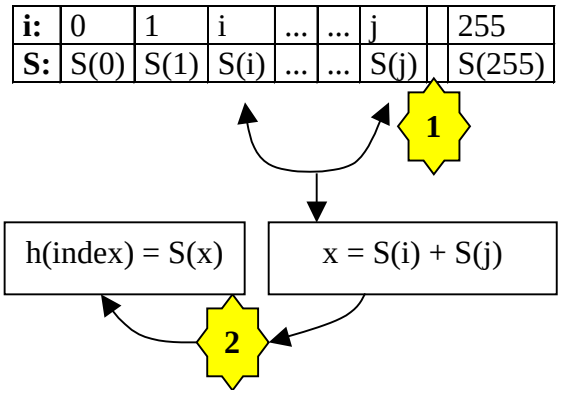


Obr.: Tvorba permutace z náhodné posloupnosti

Tvorba hesla

Poznamenejme, že v následujícím se pracuje s bajty a sčítání je proto vždy v modulu 256. Dále index *i* zde plní úlohu systematicky se zvyšujícího indexu (od 0 do 255 a cyklicky dále), zatímco úlohu náhodného indexu zajišťuje klíčově závislá proměnná *j*. Heslová posloupnost *h* je generována tímto algoritmem:

```
i = j = 0
for index = 0 to n
{
        i = i + 1
        j = j + S(i)
        vyměň mezi sebou hodnoty S(i) a S(j)
        h(index) = S( S(i) + S(j) )
}
```



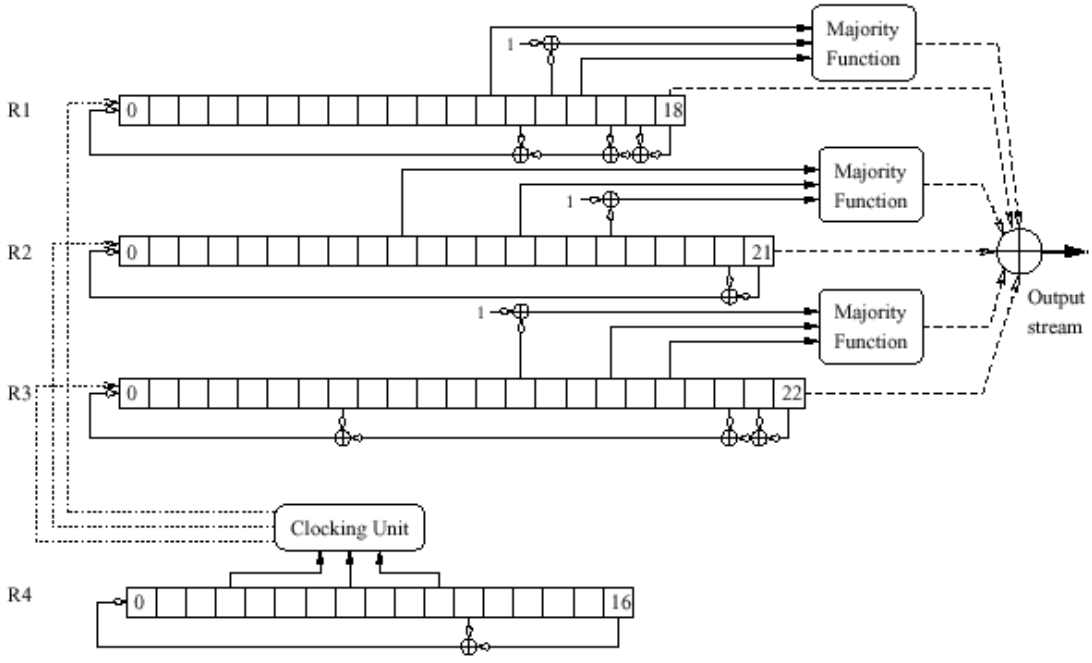
Obr.: Tvorba hesla RC4 pomocí tabulky S

8.7.2. Příklad proudové šifry: A5

Proudová šifra A5 má několik variant a používá se k šifrování hovorů mezi mobilním telefonem a základnovou stanici sítě GSM. Její popis také nikdy nebyl oficiálně publikován, zde uvedeme společné prvky variant A5/1 i A5/2. Jedná se o proudovou šifru, která také na bity otevřeného textu binárně načítá heslo. Produkuje vždy 228 bitů hesla, z toho 114 bitů pro

šifrování hovoru odesílaného z mobilního telefonu a 114 bitů pro šifrování hovoru přijímaného. Úsek 114 bitů dat, který přichází i odchází, se nazývá burst a každý burst je očíslován tzv. číslem rámce TDMA, které je 22bitové. Základem šifrování je tajný klíč Ki, který je uložen v SIM kartě telefonu. Při každém novém spojení se sítí GSM je z Ki a 128bitové náhodné výzvy RAND při autentizaci vygenerován také (na RAND a Ki závislý) klíč Kc pro šifru A5. Ten je pak prostřednictvím A5 smíchán s TDMA a vytváří počáteční naplnění registrů A5. Poté proběhne 99 (A5/2) nebo 100 (A5/1) kroků těchto registrů naprázdno a teprve pak je vygenerováno 224 bitů hesla. Tímto způsobem je zajištěno, že pro jiné číslo rámce je použito jiné heslo. Na obrázku je základní schéma A5/2. Varianta A5/1 se liší ve více detailech, nejmarkantnější je, že nepoužívá registr R4.

Poznámka k lineárním registrům a nelineárním filtrům
Použitý termín registr znamená posuvný registr s lineární zpětnou vazbou (LSFR, linear shift feedback register). V kryptografii se používaly velmi často, protože měly velmi dobré a prozkoumané teoretické a statistické vlastnosti a přitom v počátcích elektroniky byly velmi dobře realizovatelné. Protože stavy registrů jsou lineární funkcí počátečního nastavení, musí se zpracovat tzv. nelineárním filtrem, který tyto proměnné zesložití. U šifry A5 obstarává zesložitění jednak tzv. nelineární řízení, neboť registry nekrokují pravidelně, a jednak tzv. majoritní funkce (majority function) zpracovávající proměnné z registrů. Majoritní funkce Maj(a, b, c) vybírá z bitů a, b, c tu hodnotu, která mezi nimi převažuje. Platí Maj(a, b, c) = a*b + a*c + b*c, kde součet i součin je binární, přičemž součin sem přivádí onu nelinearitu.



Obr.: Základní schéma šifry A5/2 [23]

Posuvy registrů jsou nelineárně řízeny prostřednictvím registru R4: R1 se posune, pokud R4[10] = Maj(R4[3], R4[7], R4[10]), R2 se posune, pokud R4[3] = Maj(R4[3], R4[7], R4[10]), R3 se posune, pokud R4[7] = Maj(R4[3], R4[7], R4[10]). To také zajišťuje, že se vždy posunou 2 nebo 3 z registrů R1, R2 a R3.

8.8. Dvojí a vícenásobná použití hesel

Toto je obecná poznámka k vícenásobnému použití hesla u proudových šifer. Pokud máme k dispozici dostatečné množství šifrových textů, šifrovaných stejným heslem, můžeme tyto šifrové texty zapsat pod sebe a provádět tzv. luštění do hloubky. Podle definice proudové šifry pak na každé pozici (i) dostáváme jednoduchou záměnu $E_{h(i)}$ nad abecedou A.

$E_{h(1)}(m_{1,1})$	$E_{h(2)}(m_{1,2})$	$E_{h(3)}(m_{1,3})$	$E_{h(4)}(m_{1,4})$	.....
$E_{h(1)}(m_{2,1})$	$E_{h(2)}(m_{2,2})$	$E_{h(3)}(m_{2,3})$	$E_{h(4)}(m_{2,4})$	.....
$E_{h(1)}(m_{3,1})$	$E_{h(2)}(m_{3,2})$	$E_{h(3)}(m_{3,3})$	$E_{h(4)}(m_{3,4})$	.....
$E_{h(1)}(m_{4,1})$	$E_{h(2)}(m_{4,2})$	$E_{h(3)}(m_{4,3})$	$E_{h(4)}(m_{4,4})$	.....
$E_{h(1)}(m_{5,1})$	$E_{h(2)}(m_{5,2})$	$E_{h(3)}(m_{5,3})$	$E_{h(4)}(m_{5,4})$	.....
$E_{h(1)}(m_{6,1})$	$E_{h(2)}(m_{6,2})$	$E_{h(3)}(m_{6,3})$	$E_{h(4)}(m_{6,4})$	.....
.....				
.....				
$E_{h(1)}(m_{n,1})$	$E_{h(2)}(m_{n,2})$	$E_{h(3)}(m_{n,3})$	$E_{h(4)}(m_{n,4})$	.....
každý sloupec tvoří jednoduchou substituci nad abecedou A, většinou se jedná o posuv písmen o h(i)				

Obr.: Luštění několikanásobného použití hesla

U proudových šifer je jednoduchá substituce $E_{h(i)}$ většinou pouhý posun o písmeno, takže se může aplikovat postup luštění Vigenerovy šifry.

8.8.1. "Upravená" Vernamova šifra

Šifréři, kteří pracovali s děrnými páskami, přišli časem s myšlenkou jak ušetřit heslovou pásku u Vernamovy šifry. Vzali nějaký kus heslové pásky a jeho konce slepili. Tím vznikla páska nekonečná a problém s distribucí hesla byl "vyřešen". Další variantou bylo, že se do snímače vložily takto slepené pásky současně dvě, a to o různých délkách. Heslo pak vzniká součtem bitů z obou pásek. V těchto variantách vzniklo vždy periodické heslo.

8.8.2. Dvojí použití hesla u aditivních šifer

U proudových šifer, kde každá parciální substituce je posunem otevřeného textu v abecedě A o h(i), **postačí k luštění dva šifrové texty** (c, c´), **šifrované tímtéž heslem**. Máme totiž
c(i) = (m(i) + h(i)) mod q
a
c´(i) = (m´(i) + h(i)) mod q.
Odečtením šifrových textů od sebe modulo q eliminujeme heslo a dostáváme tak známý rozdíl dvou otevřených textů m a m´:
(c(i) - c´(i)) mod q = **((m(i) - m´(i)) mod q**.
Z obdržené posloupnosti ((m(i) - m´(i)) mod q pro i = 0, 1, ... pak **oba původní texty m a m´ luštíme jako při použití knižní šifry**, kde v roli původního otevřeného textu vystupuje m a v roli knižního hesla m´. Často se přitom používá **metoda předpokládaného slova**, kdy za m´ zkoušíme nějaké slovo postupně od první do poslední pozice, dopočteme m´ a sledujeme, zda dává smysl.

8.8.3. Příklad - šifrování on-the-fly

Řada prostředků pro šifrování dat na osobních počítačích umožňuje vytvoření adresáře nebo virtuálního logického disku nebo skutečného oddílu pevného disku tak, že všechna data, která se sem ukládají, jsou šifrována, a to na pozadí činnosti uživatele (on-the-fly), za chodu

operačního systému. Vše, co se z tohoto oddílu čte nebo se sem zapisuje, prochází speciálním (softwarovým nebo hardwarovým) řadičem, který vstupní data při zápisu zašifrovává a při čtení odšifrovává proudovou šifrou. Takový prostředek může být napadnutelný následujícím způsobem. Předpokládejme, že útočník má možnost v době nečinnosti počítače okopírovat zašifrovaná data (například vyjmutím a okopírováním pevného disku nebo přihlášením se k počítači jako jiný uživatel apod.). Dále předpokládejme že útočník může danému uživateli vnutit nějaká data k zašifrování. Například to může být zaslání e-mailu, který se jako součást archivu poštovních zpráv také ukládá do šifrované oblasti. Pro jednoduchost předpokládejme, že je to jeden pouhý znak o(0). Dojde tedy k jeho vložení před původní neznámý otevřený text o(1), o(2),..... Předpokládejme, že útočník je opět schopen nyní nějakým způsobem získat zašifrovaná data. Jestliže ano, může si zapsat předchozí a současný šifrový pod sebou (součet je mod q):

původní šifrový text: o(1) + h(1), o(2) + h(2), o(3) + h(3), ...
nový šifrový text: o(0) + h(1), o(1) + h(2), o(2) + h(3), ...

Nyní ze znalosti o(0) a prvního šifrového znaku z druhého řádku získá h(1) a poté střídavě z prvního a druhého řádku získává o(1), h(2), o(2), h(3), o(3), atd. Tímto způsobem odšifruje obsah celý otevřený text. Záleží na operačním systému a konkrétní realizaci, jak dlouhou posloupnost je možné takto odšifrovat. Obvykle to bude v délce alespoň jednoho sektoru, tedy 4 nebo 8 kByte.

8.9. Vlastnosti proudových šifer, synchronní a asynchronní šifry

8.9.1. Použití

Proudové šifry se používají zejména u tzv. **linkových šifrátorů**, kdy do komunikačního kanálu přichází jednotlivé znaky v pravidelných nebo nepravidelných časových intervalech, přičemž v daném okamžiku je nutné tento znak okamžitě přenést, takže není vhodné nebo možné čekat na zbývající znaky bloku. To je příklad tzv. terminálového spojení, kdy jsou spojeny dva počítače, přičemž to, co uživatel píše na klávesnici na jedné straně, se objevuje na monitoru počítače druhého uživatele. Proudové šifry se také používají v případech, kde šifrovací zařízení má omezenou paměť na průchozí data.

8.9.2. Propagace chyby

Další výhodou proudových šifer oproti blokovým je malá "**propagace chyby**". Pokud vznikne chyba na komunikačním kanálu v jednom znaku šifrového textu, projeví se tato chyba u proudových šifer pouze v jednom odpovídajícím znaku otevřeného textu, u blokové šifry má vliv na celý blok znaků.

8.9.3. Synchronní proudové šifry

V případě, že proud hesla nezávisí na otevřeném ani šifrovém textu, hovoříme o **synchronních** proudových šifrách. V tomto případě musí být příjemce a odesílatel přesně synchronizováni, protože ***výpadek* jednoho znaku šifrového textu naruší veškerý následující otevřený text**. Šifry tohoto typu jsou plně definovány generátorem hesla G, který pro daný tajný klíč k vytvoří posloupnost h(1), h(2),.... a zašifrování probíhá podle vztahu

$c(i) = E_{h(i)}(m(i),$

a odšifrování jako

$m(i) = D_{h(i)}(c(i)),$

přičemž v drtivé většině případů je $E_{h(i)}$ a $D_{h(i)}$ prostá operace binárního sčítání, tj.

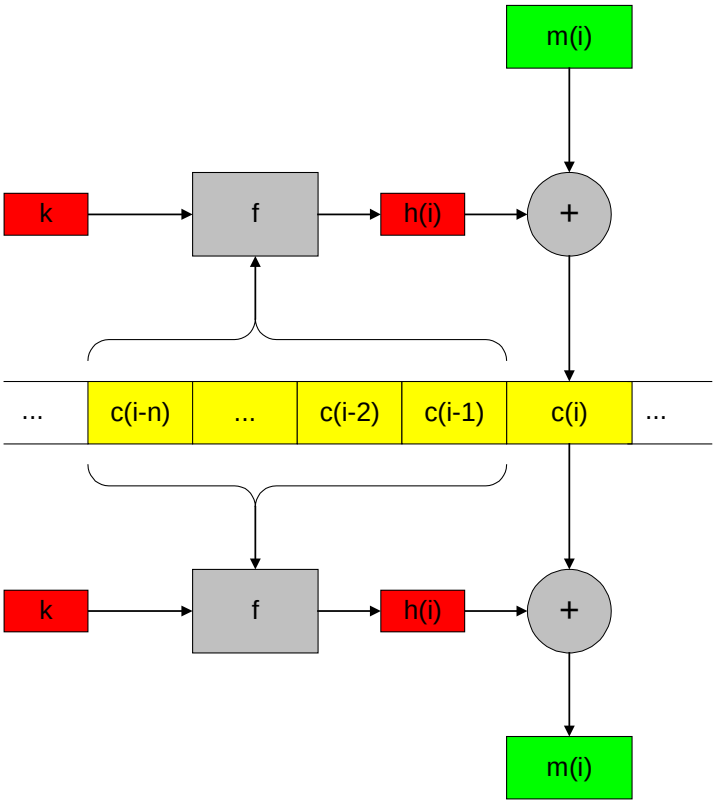
$c(i) = m(i) + h(i)$
$m(i) = c(i) + h(i)$

Pokud dojde k chybě na komunikačním kanálu, kdy vypadne (nebo přibude) jeden nebo více

znaků šifrového textu, dojde k **rozsynchronizování proudu hesla** a proudu šifrového textu, v důsledku čehož bude chybně dešifrován celý zbytek otevřeného textu.

8.9.4. Asynchronní proudové šifry

Šifry, které umí eliminovat takové chyby, se nazývají **asynchronní** nebo **samosynchronizující se šifry**. U nich dojde v krátké době k synchronizaci a správné dešifraci zbývajícího otevřeného textu. Této vlastnosti se může docílit například tím, že **proud hesla je generován pomocí klíče a *n* předchozích znaků *šifrového* textu**:
h(i) = f(k, c(i-n), ..., c(i-1)).
V tomto případě se výpadek některého znaku šifrového textu projeví celkem na n sousledných znacích otevřeného textu, ale další otevřené znaky budou již správně dešifrovány. Z definice tvorby hesla je vidět, že k synchronizaci dojde, jakmile se přijme souvislá posloupnost *n+1* správných znaků šifrového textu.



Obr.: Asynchronní (samosynchronizující se) šifry

8.9.5. Příklad: historická asynchronní šifra (Vigenerův autokláv)

Zvláštním případem asynchronní proudové šifry je tzv. Vigenerův autokláv. Vigenere navrhl použít jako heslo pouze jedno písmeno klíče (v příkladu je to první B), přičemž další znaky hesla byly tvořeny už přímo předchozím znakem šifrového textu (sčítání v modulu 26). Taková šifra je asynchronní.
c(1) = p(1) + h(1), kde h(1) = k
a pro i = 2, 3, ... : c(i) = p(i) + h(i), kde h(i) = c(i-1).

OT: A H O J
H: **B** B I W
ŠT: B I W F
(a = 0, b = 1, c = 2, d = 3, e = 4, f = 5, g = 6, h = 7, i = 8, j = 9, k = 10, l = 11, m = 12, n = 13, o = 14, p = 15, q = 16, r = 17, s = 18, t = 19, u = 20, v = 21, w = 22, x = 23, y = 24, z = 25)

9. Blokové šifry

9.1. Definice

Definice: Bloková šifra

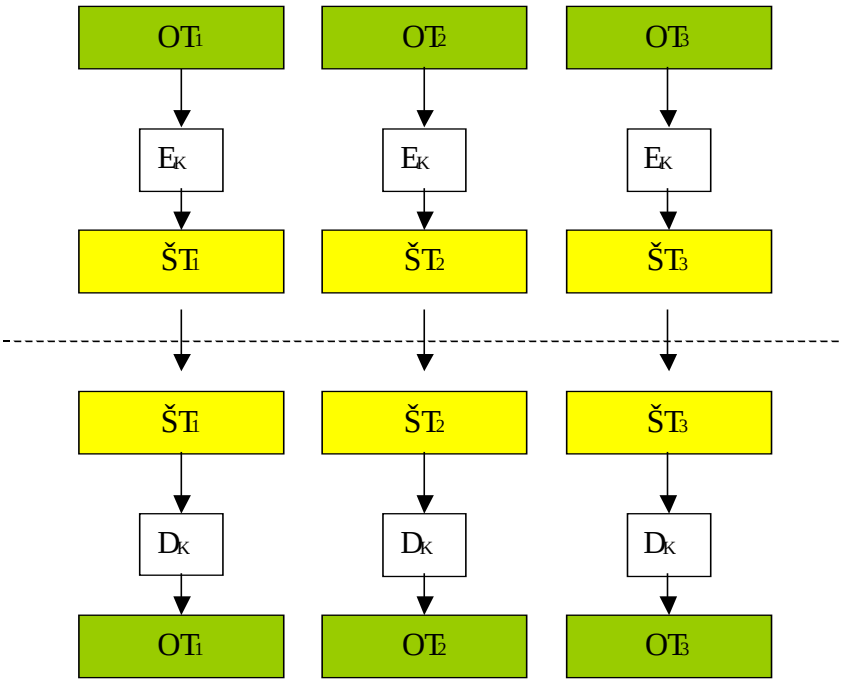
Nechť A je abeceda q symbolů, $t \in N$ a $M = C$ je množina všech řetězců délky t nad A. Nechť K je množina klíčů. Bloková šifra je šifrovací systém (M, C, K, E, D), kde E a D jsou zobrazení, definující pro každé $k \in K$ transformaci zašifrování E_k a dešifrování D_k tak, že zašifrování bloků otevřeného textu m(1), m(2), m(3),..., (kde $m(i) \in M$ pro každé $i \in N$) probíhá podle vztahu

$c(i) = E_k(m(i))$ pro každé $i \in N$

a dešifrování podle vztahu

$m(i) = D_k(c(i))$ pro každé $i \in N$.

Pro definici blokové šifry je podstatné, že všechny bloky otevřeného textu jsou šifrovány toutéž transformací a všechny bloky šifrového textu jsou dešifrovány toutéž transformací.



Obr.: Bloková šifra

Příkladem blokových šifer je substituční a transpoziční šifra.

9.2. Substituční šifra

Definice: Substituční šifra (bloková šifra s délkou bloku 1)

Nechť A je abeceda q symbolů a $M = C$ je množina všech konečných řetězců nad A. Nechť K je množina všech permutací na množině A. Substituční šifra je bloková šifra s délkou bloku 1 znak. Je tvořená pěticí (M, C, K, E, D), kde E a D jsou zobrazení, definující pro každé $k \in K$ transformaci zašifrování E_k a dešifrování D_k tak, že $E_k = k$ a $D_k = k^{-1}$, tedy pro každé $i \in N$ zašifrování znaku $m(i) \in A$ otevřeného textu na šifrový text c(i) probíhá podle vztahu

$c(i) = E_k(m(i))$

a odšifrování podle vztahu

$m(i) = D_k(c(i))$.

9.3. Transpoziční šifra

Definice: Transpoziční šifra

Nechť A je abeceda q symbolů, $t \in N$ a M = C je množina všech řetězců délky t nad A. Nechť K je množina všech permutací na množině {1, 2, ..., t}. Transpoziční šifra je bloková šifra, tvořená pěticí (M, C, K, E, D), kde E a D jsou zobrazení, definující pro každé $k \in K$ transformaci zašifrování E_k a dešifrování D_k tak, že

$E_k: M \rightarrow C: m \rightarrow c = (c_1,..., c_t) = (m_{k(1)}, m_{k(2)},..., m_{k(t)})$

a

$D_k : C \rightarrow M: c \rightarrow m = (m_1, m_2,...,m_t) = (c_{l(1)}, c_{l(2)},..., c_{l(t)})$, kde $l = k^{-1}$.

Poznámka: permutace bývají definovány dvěma způsoby: k(1) může znamenat původní pozici, z které se prvek přemísťuje na první pozici nebo to může znamenat novou pozici, kam se přemístí původní první prvek. V uvedené definici se jedná o první interpretaci.

9.4. Příklady

9.4.1. Polygramová šifra

Abeceda A může být libovolná množina, v moderních šifrách je to obvykle binární abeceda, A = {0,1}. Pokud je A abecedou nějakého přirozeného jazyka, například češtiny nebo angličtiny, bloková šifra je vlastně polygramovou šifrou, která t-tici znaků přiřadí obvykle t-tici jiných znaků. Pro konkrétní t se nazývá t-gramová a pro t = 2 hovoříme o bigramové šifře. Příkladem bigramové šifry je šifra Playfair.

9.4.2. Šifra Playfair

Šifra Playfair je historická bigramová substituce, jejíž klíč je tvořen tabulkou o rozměru 5x5, kde je vepsána obecná permutace 25 písmen abecedy (J se nevyskytuje a je nahrazeno v otevřeném textu hláskou I).

L	Z	Q	C	P
A	G	N	O	U
R	D	M	I	F
K	Y	H	V	S
X	B	T	E	W

Obr.: Šifra Playfair

Potom se daná dvojici písmen otevřeného textu nahradí dvěma písmeny, která s danými tvoří obdélník. Pokud jsou vzory v jednom řádku nebo sloupci, berou se písmena následující v řádku nebo písmena pod v daném sloupci. Například máme substituce: AC → LO, RI → DF, RF → DR, PS → UW.

9.5. Difúze

Důvod, proč historické šifry jsou luštitelné přímo ze šifrového textu je ten, že šifrový text u nich příliš dobře a přímočaře odráží statistické charakteristiky otevřeného textu. **Shannon navrhl dvě metody**, jak tomu zabránit - **difúzi a konfúzi**. **Difúze rozprostírá statistické charakteristiky otevřeného textu do delších úseků šifrového textu.** Dobrou difúzi například docílíme zobrazením s znaků otevřeného textu do jednoho znaku šifrového textu předpisem

$c(n) = (o(n – s + 1) + o(n – s + 2) + ... + o(n)) \bmod 26.$

Jednoduchá záměna ponechává zcela nepozměněny vazby mezi hláskami, neboť četnosti jednotlivých hlásek, bigramů, trigramů atd. zůstávají stejné v šifrovém textu jako v otevřeném

textu.
Transpozice rozbíjí tyto vazby, ale promítá beze změny rozdělení pravděpodobností jednotlivých hlásek otevřeného textu přímo do šifrového textu.
Vigenerova šifra částečně vyhlazuje rozdělení pravděpodobností hlásek otevřeného textu, ale ponechává nezměněnu určitou část N-gramových závislostí. Jakmile se zjistí délka hesla, je možné využít statistických vlastností otevřeného textu, neboť na hláskách vzdálených od sebe o délku hesla se přímo projeví rozložení četností jednotlivých hlásek otevřeného textu.

9.5.1. N-gramová substituce tvořená klíčovou maticí

Velmi dobrá difúze je dosažena, když jakákoliv redundance v otevřeném textu je rozprostřena do celého šifrového textu. Například uvažujme anglickou abecedu A až Z, převedenou klasicky na čísla 0 až 25, a klíč jako invertibilní matici $K = (k_{i,j})_{i=1..10,\ j=1..10}$ čísel v rozmezí 0 - 25. Uvažujme blokovou šifru, převádějící otevřený blok 10 znaků otevřeného textu OT na šifrový blok ŠT 10 znaků předpisem ŠT = K* OT, tj. $ŠT(i) = (k_{i,1}*o(1) + k_{i,2}*o(2) + ... + k_{i,10}*o(10)) \bmod 26$ pro i = 1 ... 10. Tato bloková šifra bude mít velmi dobrou difúzi, protože každý znak, dvojice nebo n-tice znaků otevřeného textu se promítá do každého znaku šifrového textu, takže tyto a podobné charakteristiky otevřeného textu se promítají do všech znaků (nikoli jen části) šifrového textu. Při luštění této šifry bychom se však nezaměřili na luštění otevřeného textu, ale přímo klíče. Při znalosti deseti bloků otevřeného a šifrového textu (OT^x, $ŠT^x$) lze snadno vypočítat celou klíčovou matici prostým řešením soustavy lineárních rovnic (modulo 26). Pro každé *i* = 1, ..., 10 dostáváme 10 lineárních rovnic pro klíčové neznámé $k_{i,1}$ až $k_{i,10}$:

$$ŠT_i^x = k_{i,1}*OT_1^x + k_{i,2}*OT_2^x + ... + k_{i,10}*OT_{10}^x \text{ pro } x= 1, ..., 10,$$

z nichž je lze snadno vypočítat.

9.6. Konfúze

Dále Shannon definoval druhou metodu, jak znesnadnit statistickou kryptoanalýzu, tzv. konfúzi. **Konfúze je metoda, jejímž cílem je učinit vztah mezi statistickými vlastnostmi šifrového textu a klíčem co nejsložitější a nejobsažnější** (tj. zahrnující co největší části šifrového textu a klíče), viz originální definice „The method of *confusion* is to make the relation between the simple statistics of E and the simple description of K a very complex and involved one“. Konfúze u n-gramové šifry, spočívající v násobení klíčovou maticí z výše uvedeného příkladu není dobrá, neboť vztahy mezi šifrovým textem a klíčem jsou sice "obsažné", ale přesto poměrně jednoduché (lineární).

9.7. Součinové šifry

Pro docílení dobré vlastnosti promíchávání otevřeného textu s klíčem Shannon navrhl metodu vytváření šifer opakovaným skládáním - součinem - několika šifer různých typů. Označíme-li substituci S, transpozici T a lineární šifru L (například aditivní šifra Vigenerova typu), pak LSLSLT je součinová šifra, která na otevřený text aplikuje nejprve transpozici T, na výsledek lineární šifru L, substituci S atd. Každá z dílčích šifer má svůj význam pro tvorbu celkové difúze a konfúze součinové šifry.

Definice: součinová šifra
Mějme n šifrovacích systémů $S^i = (M^i, C^i, K^i, E^i, D^i)$, kde pro jednoduchost nechť $M = C = K = M^i = C^i = K^i$ pro všechna i = 1,..., n. Nechť G je generátor, který každému klíči $k \in K$

přiřadí n-tici klíčů {k(1), ..., k(n)} ∈ K^n. Potom definujeme součinovou šifru $S = S^n S^{n-1} ....S^1$ jako šifrovací systém s množinou klíčů K, množinou otevřených textů M, množinou šifrových textů C a dvojicí zobrazení {E, D} tak, že pro každé k ∈ K definujeme {k(1),..., k(n)} = G(k), kde G je vhodný generátor posloupnosti {k(1),..., k(n)}, kterou nazýváme expandovaný klíč k, a
pro každé m ∈ M definujeme $c = E_k(m) = E^n_{k(n)}(E^{n-1}_{k(n-1)}(... E^1_{k(1)}(m)))$,
pro každé c ∈ C definujeme $m = D_k(c) = D^1_{k(1)}(D^2_{k(2)}(... D^n_{k(n)}(c)))$.

9.8. Úplnost

Úplnost patří k dobrým vlastnostem blokových šifer.

Definice: Úplnost je vlastnost, kdy každý bit šifrového textu funkčně závisí na každém bitu otevřeného textu (**úplnost vzhledem k otevřenému textu**) a na každém bitu klíče (**úplnost vzhledem ke klíči**).

9.9. Poznámka: Obecnější pojetí difúze a konfúze

Výše jsme použili původní Shannonovu definici difúze a konfúze. U moderních šifer je však vhodné vztahovat vlastnosti difúze a konfúze současně jak na klíč, tak na otevřený text. Proto bychom mohli definovat difúzi obecněji, například takto:

- **Difúze** je vlastnost šifry, odrážející vliv otevřeného textu a klíče na šifrový text. Můžeme ji dělit zvlášť na difúzi vzhledem k otevřenému textu a na difúzi vzhledem ke klíči:
- **Difúze vzhledem k otevřenému textu** je vlastnost šifry, odrážející vliv otevřeného textu na šifrový text.
- **Difúze vzhledem ke klíči** je vlastnost šifry, odrážející vliv klíče na šifrový text.

Podobně bychom mohli definovat obecněji i konfúzi:

- **Konfúze** je vlastnost šifry, odrážející složitost vztahu mezi otevřeným textem, klíčem a šifrovým textem. Můžeme ji dělit zvlášť na konfúzi vzhledem k otevřenému textu a konfúzi vzhledem ke klíči:
- **Konfúze vzhledem k otevřenému textu** je vlastnost šifry, odrážející složitost vztahu mezi otevřeným a šifrovým textem.
- **Konfúze vzhledem ke klíči** je vlastnost šifry, odrážející složitost vztahu mezi šifrovým textem a klíčem.

9.10. Iterovaná šifra

Definice: iterovaná šifra
Jestliže v součinové šifře $S = S^n S^{n-1} .... S^1$ jsou dílčí šifry stejné (až na možnou výjimku první a poslední šifry), nazýváme S iterovanou šifrou. Jednotlivé iterace S^i nazýváme **rundy**, jim odpovídající transformace zašifrování (E^i) a odšifrování (D^i) a jejich dílčí klíče (k(i)) nazýváme po řadě **rundovní funkce** zašifrování a odšifrování a **rundovní klíče**.

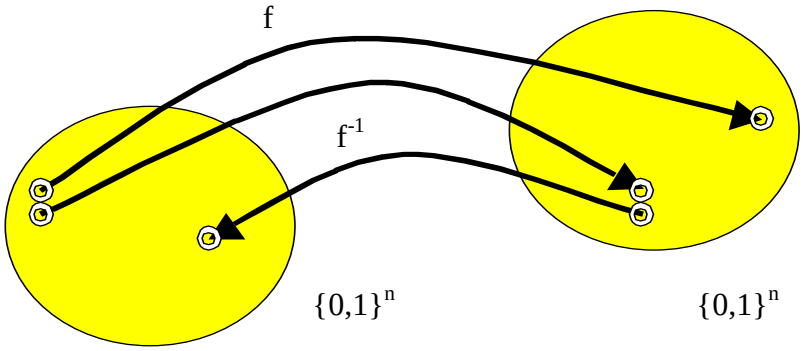
9.11. Obecné metody dosažení difúze a konfúze

U moderních šifer, které vznikaly zejména pro počítačové použití, je možné sledovat řadu originálních myšlenek, které byly použity při jejich kryptografickém návrhu. **Většina moderních šifer jsou iterované šifry typu $E_{k(n)} \bullet E_{k(n-1)} \bullet$ ... $\bullet E_{k(1)}$.** Dílčí iterace jsou stejné transformace, které jsou řízeny jen jinými rundovními klíči k(1), ..., k(n), vzniklými expanzí z původního šifrovacího klíče k . Expanze většinou probíhá v tzv. přípravné fázi šifrování, kdy se využívá faktu, že je relativně dost času na složité výpočty. Výjimku tvoří případy, kdy dané

šifrovací zařízení má dost vysoký výkon, ale málo paměti. V takovém případě je možné rundovní klíče vytvářet za chodu schématu postupně a neukládat je. Při tvorbě rundovní funkce je třeba dbát na to, že za její poslední operací následuje opět první operace následující rundovní funkce, tj. první a poslední operace rundovní funkce by měly být jiného typu, aby jejich složením v součinu nedocházelo ke zjednodušení (aby to nebyly komutující šifry, například dvě substituce nebo dvě transpozice apod.).
Další často užívanou technikou je tzv. **zašumění (whitenning)**, a to dvakrát – jednou před zpracováním otevřeného textu (počáteční zašumění, pre - whitenning) a podruhé těsně před výstupem z blokové šifry (závěrečné zašumění, post - whitenning). Zašumění se prování prostým načtením pro tento případ speciálně vytvořených rundovních klíčů na vstup. Tyto rundovní klíče jsou použity jen pro zašumění, v řádných rundách se nepoužijí. Uvedené principy použila řada moderních počítačových šifer, včetně standardů DES a AES.

9.12. Náhodné permutace na množině $\{0,1\}^n$

Moderní blokové šifry by měly mít takovou vlastnost difúze a konfúze, že bez znalosti šifrovacího klíče by se měly jevit a být nerozlišitelné od náhodných permutací na množině $\{0,1\}^n$, kde n je délka bloku. Potom znalost jakékoli dvojice (OT, ŠT) nepomáhá k odvození možných šifrových textů pro blízké otevřené texty nebo možných šifrových textů pro blízké otevřené texty apod.



Obr. : Náhodná permutace f na množině $\{0,1\}^n$

Navíc pochopitelně vlastnosti konfúze a difúze by měly zabránit tomu, aby ze znalosti mnoha dvojit (OT, ŠT) šel odvodit použitý šifrovací klíč nebo se o něm získávala nějaká *užitečná* informace. Zajistit takové požadavky je velmi složitý úkol při kryptografickém návrhu šifry, protože ze Shannonovy teorie víme, že informace o klíči je dostatek už v několika párech otevřeného a šifrového textu.

9.13. Příklad: iterovaná šifra MBTM

Uvažujme, že šifra MBTM (zmíněná v předchozích přednáškách) je sama jednou rundou S součinové šifry S • S • S • ... • S a sledujme zašifrování otevřeného textu při zvyšování počtu rund. Zvolme proto následující jednoduchý konkrétní příklad, v němž transpozice (16 na 16) je dána klíčovým slovem VŘES a substituce klíčovým slovem ÚTERÝ, viz obrázek.

	A	B	C	D	E
A	U	T	E	R	Y
B	A	B	C	D	F
C	G	H	I	J	K
D	L	M	N	O	P
E	Q	S	V	W	Z

V	R	E	S
4	2	1	3
1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
T	3	7	11	15	2	6	10	14	4	8	12	16	1	5	9	13

V	E	S	E	L	E	V	**A**								
E	C	A	C	E	B	A	C	D	A	A	C	E	C	**B**	**A**
A	A	A	**B**	C	B	A	C	C	C	C	**A**	E	E	D	E
U	**T**	H	E	I	**G**	Z	P									1.
A	A	A	**B**	C	B	A	C	C	C	C	**A**	E	E	D	E
A	A	C	D	A	B	C	E	**B**	C	**A**	E	A	C	C	E
U	J	T	K	**C**	**Y**	E	K									2.
A	A	C	D	A	B	C	E	**B**	C	**A**	E	A	C	C	E
C	C	**A**	C	A	B	C	C	D	E	E	E	A	A	**B**	A
I	**E**	T	I	P	Z	U	**A**									3.
C	C	A	C	A	B	C	C	D	E	E	E	A	A	B	A
A	C	E	B	C	B	E	A	C	C	E	A	C	A	D	A
E	**S**	H	Q	I	Q	G	L									4.
A	C	E	B	C	B	E	A	C	C	E	A	C	A	D	A
E	E	E	D	C	B	C	A	B	A	A	A	A	C	C	C
Z	W	H	G	**A**	U	**E**	I									5.
E	E	E	D	C	B	C	A	B	A	A	A	A	C	C	C
E	C	A	C	E	B	A	C	D	A	A	C	E	C	B	A
V	E	S	E	L	E	V	**A**									6.

Obr.: Příklad iterované šifry MBTM

Zjistíme, že $S^6 = I$ a že skládání nezvyšuje difúzi šifry. Příčina obou jevů je následující. S jako šifru MBTM můžeme zapsat jako součin $S = S_{2\_to\_1} \bullet T_{1\_to\_1} \bullet S_{1\_to\_2}$, kde $S_{1\_to\_2}$ je prvotní substituce monogramu na bigram, bigramy se pak chápou jako monogramy, na něž se aplikuje následná transpozice $T = T_{1\_to\_1}$ a poté se monogramy spojí v bigramy, na něž se aplikuje

zpětná substituce $S_{2\_to\_1}$. Z definice substituce bigram-monogram víme, že platí $S_{2\_to\_1} \bullet S_{1\_to\_2} = I$.
Dále máme
$S^2 = S \bullet S = S_{2\_to\_1} \bullet T \bullet S_{1\_to\_2} \bullet S_{2\_to\_1} \bullet T \bullet S_{1\_to\_2} = S_{2\_to\_1} \bullet T^2 \bullet S_{1\_to\_2}$
$S^3 = S \bullet S^2 = S_{2\_to\_1} \bullet T \bullet S_{1\_to\_2} \bullet (S_{2\_to\_1} \bullet T^2 \bullet S_{1\_to\_2}) = S_{2\_to\_1} \bullet T^3 \bullet S_{1\_to\_2}$
$S^4 = S_{2\_to\_1} \bullet T^4 \bullet S_{1\_to\_2}$
$S^5 = S_{2\_to\_1} \bullet T^5 \bullet S_{1\_to\_2}$
$S^6 = S_{2\_to\_1} \bullet T^6 \bullet S_{1\_to\_2}$
Použitá transformace T je permutací na množině {1, ..., 16} šestého řádu, jak ukazuje obrázek, takže $T^6 = I$. Odtud
$S^6 = S_{2\_to\_1} \bullet T^6 \bullet S_{1\_to\_2} = S_{2\_to\_1} \bullet I \bullet S_{1\_to\_2} = I$.

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
3	7	11	15	2	6	10	14	4	8	12	16	1	5	9	13
11	10	12	9	7	6	8	5	15	14	16	13	3	2	4	1
12	8	16	4	10	6	14	2	9	5	13	1	11	7	15	3
16	14	13	15	8	6	5	7	4	2	1	3	12	10	9	11
13	5	1	9	14	6	2	10	15	7	3	11	16	8	4	12
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Obr.: Transpozice T

V tomto příkladě jsme si ukázali, že je nutné dbát na vhodné řazení dílčích šifer ve výsledné součinové šifře.

9.14. Příklady součinových šifer: Hlavní šifry československé vojenské zpravodajské služby za 2.sv.války

Tento odstavec vychází zejména z ojedinělé knihy [16]. O tom, že naše šifry luštili němečtí i švédští kryptoanalytici přinesl informaci i David Kahn v [8]. Od září 1939 do roku 1940 se používala zejména **šifra TTS** (až na krátké období, kdy se používal tzv. čtvercový klíč, tj. „kódová tabulka“ 10x10) a poté v závěru roku 1940 šifra **STT**. Jednalo se o součinové šifry složené ze dvou transpozic a jedné substituce. Kromě nich byly ještě používány šifry čtvercový klíč, ST, SP, STP a tzv. zubatka. Němci je uměli luštit, od poloviny roku 1940 do konce války luštili až na výjimky téměř všechny zprávy našich zpravodajců.

9.14.1. Substituční tabulky

Substituční tabulky byly podobné.
Podle čísla dne v měsíci se začínala číslovat tato abeceda 45 otevřených znaků:
A, B, C, ... , Z, Č, Ě, Ř, Š, Ž, ., ?, -, /,1, 2, ..., 9, 0
Například 12.den v měsíci se použilo A = 12, ..., - = 45, / = 01, 1 = 02, ..., 0 = 11. Používala se také varianta, že 12.den v měsíci platilo A = 02,..., tj. začínalo se číslem dne modulo 10.

9.14.2. Klíče pro transpozice

K definování transpozic se na počátku války určovala hesla ze seznamu 11 hesel (tzv. PB-1), později se do seznamu přidávala další slovní hesla.
U šifer TTS a STT byla vždy dvě hesla pro transpozici na každý den vybírána z knihy.
Základní varianta byla následující: V určené knize byla na daný měsíc určena jedna stránka. V daný den se na řádku téhož čísla jako den vybralo první heslo (tzv. zašifrovací) ze začátku řádku (všechna celá slova až do slova, obsahujícího 15. písmeno od začátku řádku) a druhé

heslo (tzv. přešifrovací) od konce řádku (všechna celá slova až do slova, obsahujícího 15. písmeno od konce řádku). Očíslováním jednotlivých písmen ve slovním hesle se získala řada čísel o proměnné délce, minimálně však 15 čísel. Například pro 14.den se použilo zašifrovací heslo ze začátku 14. řádku **C**OTEDY**C**H**C**EMEMLUVÍME = **1** 15 16 4 19 **2** 9 **3** 6 12 7 13 11 17 18 10 14 8, jako přešifrovací heslo POTŘEBĚNÁBOŽENSTVÍ = 12 10 15 13 4 2 6 8 1 3 11 18 5 9 14 16 17 7 z konce téhož řádku.
Zdrojem byla zejména kniha: T.G. Masaryk: Světová revoluce 1914-1918, vydání z r. 1925, nakladatelství Orbis a ČIN, Praha, série 41. až 45. tisíc a dále text básně *Nadšení*, mající 32 řádků.

9.14.3. Šifra TTS

Šifra TTS použila nejprve první transpozici (tzv. zašifrovací), poté druhou transpozici (tzv. přešifrovací) a pak substituci, kdy se každé písmeno nahradilo dvojciferným číslem podle ten den platné substituční tabulky.
U šifrogramů systému TTS bylo možné po rozdělení šifrového textu na dvojice číslic rozpoznat substituci velmi jednoduše podle nerovnoměrného výskytu znaků „01“ až „45“, přičemž jiné znaky se nevyskytovaly. Určení substituce ještě zjednodušovalo to, že se nejednalo o obecnou substituci, ale o posun o konstantu. Po odstranění této substituce se luštěním do hloubky zjistila výsledná transpozice, z níž se poté oddělily obě dílčí transpozice. Luštění do hloubky umožňovalo vysílání zpráv stejné délky, což se dělo dosti často u delších zpráv, které se rozdělovaly na několik dílčích.

9.14.4. Šifra STT

Šifra STT použila nejprve substituci (definovanou výše pro daný den), poté první transpozici (zašifrovací) a nato druhou transpozici (přešifrovací). Výhodou bylo, že transpozice rozdělila původní desítkové a jednotkové cifry substituční tabulky. Luštění popsali němečtí zajatci, kteří byli ve skupině, luštící československé šifry. Hlavní idea spočívala v určení, zda zašifrovací transformace měla sudý nebo lichý počet cifer. V obou případech docházelo k různým charakteristikám výskytu jednotkových a desítkových číslic ve výsledném textu. Těchto pravidelností si povšimli jak němečtí, tak švédští luštitelé, což vedlo k rutinnímu luštění šifrogramů.

9.14.5. Čtvercový klíč

Čtvercový klíč byl pouze čtverec 10x10 polí, kde písmena abecedy, časté bigramy a trigramy, interpunkce a číslice se zašifrovávaly jako číselné dvojice podle souřadnic daného výrazu.

10. Nejznámější blokové šifry

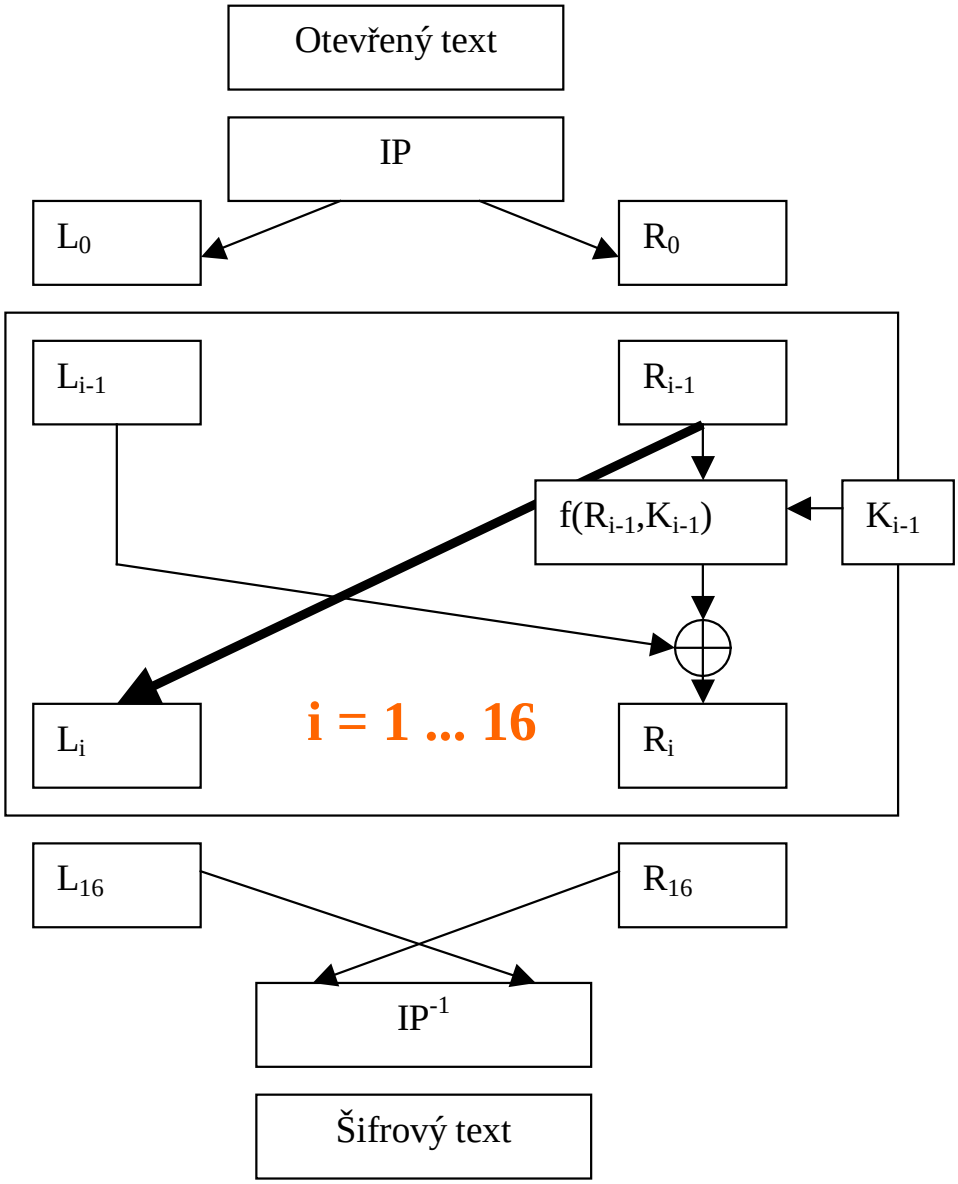
Nejznámější blokové šifry používaly a používají blok o délce 64 bitů (DES, TripleDES, IDEA, CAST aj.), v současné době se přechází na blok 128 bitů, který používá standard AES.

10.1. DES

DES (Data Encryption Standard) je nejpoužívanější šifra na světě [18]. Byla jako výsledek veřejné soutěže v roce 1977 schválena jako šifrovací standard (Federal Information Processing Standard FIPS 46-3) v USA pro ochranu citlivých, ale neutajovaných dat ve státní správě. Je součástí mnoha průmyslových, internetových a bankovních standardů (např. ANSI standard X9.32). Už v roce 1977 mnozí upozorňovali na její příliš krátký klíč 56 bitů, který byl do původního návrhu IBM zanesen vlivem americké tajné služby NSA. DES se stala předmětem intenzivního výzkumu a mnoha útoků a díky tomu byly objeveny některé teoretické negativní vlastnosti. Jedná se zejména o objev tzv. slabých a poloslabých klíčů, vlastnost komplementárnosti a později i teoreticky úspěšnou lineární a diferenciální kryptoanalýzu. V praxi však jedinou zásadní nevýhodou zůstával pouze krátký klíč. V roce 1998 byl zkonstruován stroj (viz DES-Cracker), luštící DES hrubou silou, tj. vyzkoušením všech možných klíčů. V současné době DES jako americký standard již skončil (může být používán jen v "dobíhajících" systémech a kvůli kompatibilitě) a místo něj byl přijat Triple-DES, definovaný normou FIPS 46-3, od 26.května 2002 je v platnosti šifrovací standard nové generace AES.

10.1.1. Stavební prvky DES

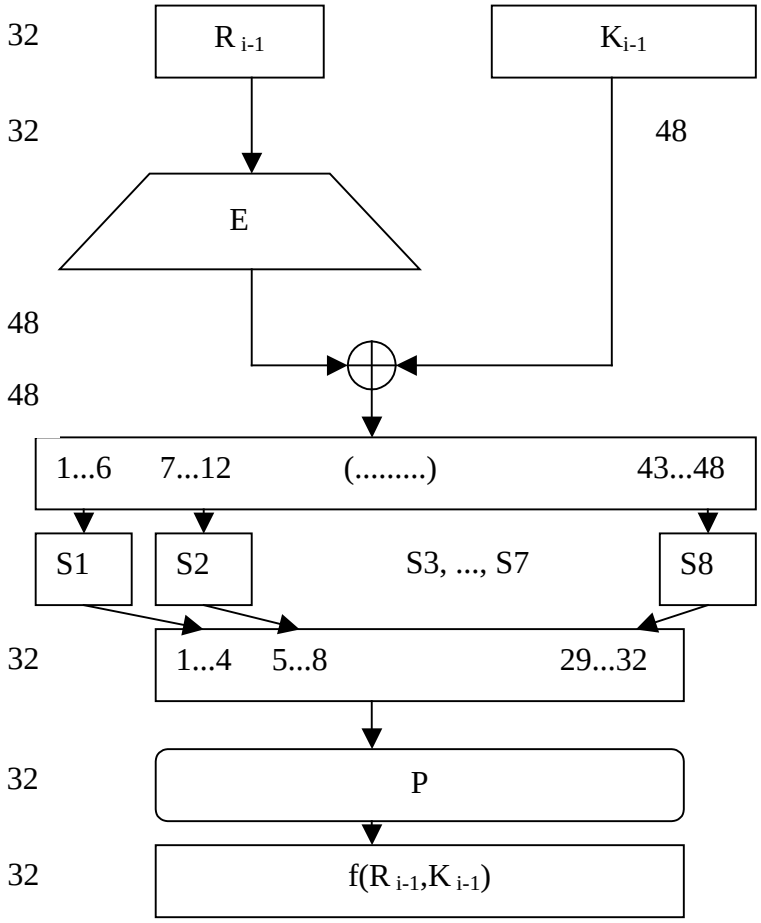
DES je iterovaná šifra typu $E_{k(16)} \bullet E_{k(15)} \bullet \ ... \bullet E_{k(1)}$, používající 16 rund a blok délky 64 bitů. Šifrovací klíč k má délku 56 bitů a je v inicializační fázi nebo za chodu algoritmu expandován na 16 rundovních klíčů k(1) až k(16), které jsou řetězci 48 bitů, každý z těchto bitů je některým bitem původního klíče k. Místo počátečního zašumění otevřeného textu se používá bezklíčová pevná permutace IP a místo závěrečného zašumění permutace k ní inverzní IP^{-1}. Po počáteční permutaci je blok rozdělen na dvě poloviny (L, R) o 32 bitech. Každá ze 16 rund (i) transformuje (L, R) na novou hodnotu (L, R) = (R, f(R, k(i)) xor L), liší se jen použitím jiného rundovního klíče k(i). Po 16. rundě dochází ještě k výměně pravé a levé strany: (L, R) = (R, L) a závěrečné permutaci IP-1. Dešifrování probíhá stejným způsobem jako zašifrování, pouze se obrátí pořadí výběru rundovních klíčů.



Obr.: Základní schéma DES

10.1.2. Rundovní funkce

Rundovní funkce f se skládá z binárního načtení klíče na vstup, následné pevné (nelineární) substituce na úrovni 6bitových znaků a poté transpozice na úrovni bitů. Tímto způsobem se dosahuje dobré difúze i konfúze.



Obr.: Rundovní funkce DES

10.1.3. S-boxy

Použité substituce se nazývají substituční boxy (S-boxy), jsou jediným nelineárním prvkem schématu. Pokud bychom substituce vynechali, mohli bychom vztahy mezi šifrovým textem, otevřeným textem a klíčem popsat pomocí operace binárního sčítání (xor), tedy lineárními vztahy. Při znalosti jednoho bloku otevřeného textu bychom tak mohli sestavit soustavu 64 lineárních rovnic se známými bity OT a ŠT a 56 neznámými bity klíče K.
Vyjádříme-li však vztah mezi výstupním bitem S-boxu a vstupními bity, obdržíme nelineární vztah, obsahující kromě binárního součtu i součiny vstupních bitů. Tato nelinearita je překážkou jednoduchého řešení rovnic, vyjadřující vztah mezi OT, ŠT a K. Tento vztah si ukážeme na příkladu S-boxu 3 na 3.

S-box nechť je dán následující tabulkou.

x_1	x_2	x_3	y_1	y_2	y_3
0	0	0	0	1	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	1	0	0
1	0	0	0	1	1
1	0	1	1	0	1
1	1	0	1	0	1
1	1	1	1	0	0

Obr.: Příklad S-boxu

Proměnné y lze vyjádřit jako funkce x pomocí binárních operací + a * (v počítačové praxi označované jako XOR a AND). Při vyjádření jako mezikrok použijeme negaci (x´= non x = x + 1), kterou označujeme čárkou. Podle tabulky vypočteme
$y_1 = x_1´*x_2*x_3 + x_1*x_2´*x_3 + x_1*x_2*x_3´ + x_1*x_2*x_3$.
(Pozn.: každý člen ve výrazu pro y_1 odpovídá hodnotě $y_1 = 1$). Upravíme vztah tak, aby neobsahoval negace a zjistíme, zda se nedá zjednodušit.

$y_1 = x_1´*x_2*x_3 + x_1*x_2´*x_3 + x_1*x_2*x_3´ + x_1*x_2*x_3 =$
$= (x_1 + 1)*x_2*x_3 + x_1*(x_2 + 1)*x_3 + x_1*x_2*(x_3 + 1) + x_1*x_2*x_3 =$
$= (x_1*x_2*x_3 + x_2*x_3) + (x_1*x_2*x_3 + x_1*x_3) + (x_1*x_2*x_3 + x_1*x_2) + (x_1*x_2*x_3)$, tj.

$y_1 = x_1*x_2 + x_1*x_3 + x_2*x_3$
$y_2 = y_1 + 1 = x_1*x_2 + x_1*x_3 + x_2*x_3 + 1$
$y_3 = x_1 + x_1*x_2*x_3$
Vidíme, že výstupní bity jsou nelineárními funkcemi druhého a třetího řádu.

10.1.4. Lineární kryptoanalýza

Uvedené nelineární vztahy se s určitou pravděpodobností dají nahradit lineárními, na čemž je založena metoda lineární kryptoanalýzy. Ta byla teoreticky úspěšně použita proti DES v roce 1993 [21]. V roce 1994 byla tato metoda použita pro určení klíče s 2^{43} známých náhodně generovaných otevřených textů a se složitostí 2^{43} operací (na 12 PC 99MHz trvala tato úloha 50 dní).

10.1.5. Komplementárnost

Je zajímavé, že přes tyto nelinearity platí tzv. vlastnost komplementárnosti, objevená v roce 1976. Jde o to, že pro každý klíč K a každý otevřený text M platí

Je-li $C = DES_K(M)$, potom non $C = DES_{nonK}$(non M).

Jinými slovy, pokud použijeme negovaný klíč na negovaný otevřený text, měli bychom dostat náhodný šifrový text, místo toho dostáváme negaci šifrového textu, patřícího k původnímu otevřenému textu a klíči. Důkaz vyplývá z této vlastnosti rundovní funkce: f(R, Ki) = f(nonR, nonKi). Komplementárnost snižuje složitost útoku hrubou silou o jeden bit.

10.1.6. Diferenciální kryptoanalýza

Jinou metodou, která byla teoreticky úspěšně použita proti DES, je metoda diferenciální kryptoanalýzy [22]. Ta zkoumá jak se (bitové) diference v otevřených textech projevují na (bitových) diferencích odpovídajících šifrových textů. Využívá toho, že některé diference otevřených textů se projevují na určitých diferencích šifrových textů statisticky významnějším způsobem. Metoda diferenciální kryptoanalýzy DES byla popsána v roce 1990 (Biham, Shamir). V roce 1992 byl pomocí ní vyluštěn klíč DES s možností volby 2^{47} otevřených textů, analyzovalo se 2^{36} otevřených textů a provedlo se 2^{37} operací šifrování

10.1.7. Slabé a poloslabé klíče

V roce 1976 byly popsány slabé a poloslabé klíče. Pro slabé klíče K platí: $E_K(X) = X$ pro každé X
Jsou to (hexadecimálně) tyto čtyři hodnoty:

- 0101 0101 0101 0101,
- FEFE FEFE FEFE FEFE,
- 1F1F 1F1F 0E0E 0E0E,
- E0E0 E0E0 F1F1 F1F1.

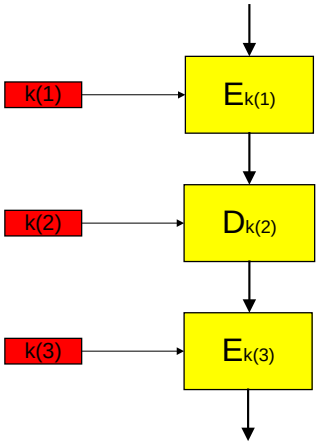
Poloslabé klíče vystupují ve dvojicích (K1, K2) a platí pro ně $E_{K2}(E_{K1}(X)) = X$ pro každé X.
Bylo nalezeno 6 dvojic poloslabých klíčů (K1,K2) :

- 01FE01FE01FE01FE, FE01FE01FE01FE01,
- 1FE01FE00EF10EF1, E01FE01FF10EF10E,
- 01E001E001F101F1, E001E001F101F101,
- 1FFE1FFE0EFE0EFE, FE1FFE1FFE0EFE0E,
- 011F011F010E010, E1F011F010E010E01,
- E0FEE0FEF1FEF1FE, FEE0FEE0FEF1FEF1.

Jedná se o další teoretickou slabinu.

10.2. TripleDES

TripleDES [18] uměle prodlužuje klíč originální DES tím, že používá DES jako stavební prvek celkem třikrát s dvěma nebo třemi různými klíči. Nejčastěji se používá tzv. varianta EDE této šifry, která je definována ve standardu FIPS PUB 46-3 a v bankovní normě X9.52. Vstupní data OT jsou zašifrována podle vztahu $ŠT = E_{K3}(D_{K2} (E_{K1}(OT)))$, kde K1, K2 a K3 jsou buď tři různé klíče nebo K3 = K1. Varianta EDE byla zavedena z důvodu kompatibility, neboť při rovnosti všech klíčů se z TripleDES stává původní DES. Klíč TripleDES je tedy buď 112 bitů (dva klíče) nebo 168 bitů (tři klíče). I když DES byla zlomena hrubou silou, TripleDES (3DES) se považuje za spolehlivou, protože klíč je dostatečně dlouhý a teoretickým slabinám (komplementárnost, slabé klíče) se dá předcházet. Proto je TripleDES vedle AES platným oficiálním standardem, nahrazujícím DES. Pokud se setkáte se zkratkou 3DES-EDE, je to právě popsaná varianta. Navíc ji lze, jako jakoukoliv jinou blokovou šifru, použít v různých operačních modech, například CBC (viz dále). Taková šifra je pak označena často jako 3DES-EDE-CBC nebo krátce jen DES-EDE-CBC.



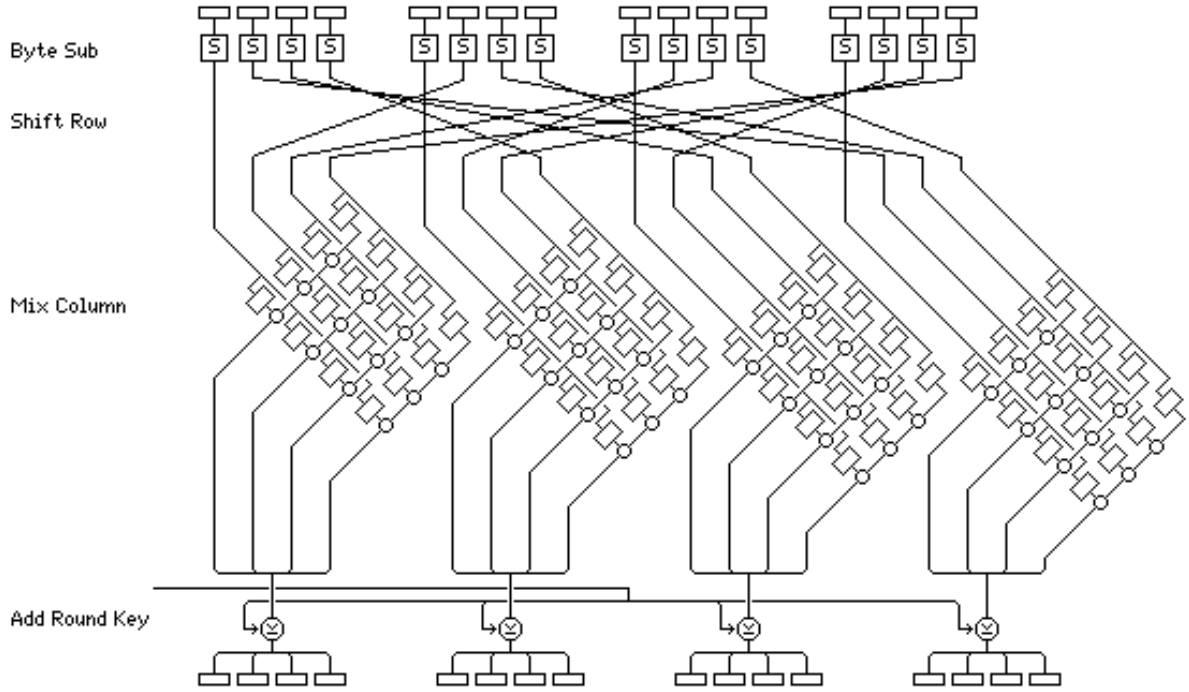
Obr.: TripleDES-EDE

10.2.1. Varianty DES

Algoritmus DES má více variant. Nejznámější je DESX, drobná modifikace DES, používaná firmou RSA Security. Používá 128 bitů klíče, přičemž jeho první polovina je naxorována na otevřený text před průchodem DES a druhá polovina je použita jako klíč k DES. Na výstup z DES je ještě naxorována další hodnota, odvozená z původních 128 bitů klíče.

10.3. AES

Jak postupovaly snahy o útok hrubou silou na DES, americký standardizační úřad začal připravovat jeho náhradu - Advanced Encryption Standard (AES). Proto bylo 2.1. 1997 vypsáno výběrové řízení na AES, do něhož se přihlásilo 15 kandidátů. NIST uspořádal celkem čtyři pracovní konference a několik výběrových kol. Nakonec z pěti finalistů byl vybrán jediný vítěz, kterým se stal algoritmus Rijndael (doporučená výslovnost je "Rájndol" s mírně polknutým o nebo "Rejndál"). Jako AES byl přijat s účinností od 26. května 2002 a byl vydán jako standard v oficiální publikaci FIPS PUB 197 [17].
AES je bloková šifra s délkou bloku 128 bitů, čímž se odlišuje od současných blokových šifer, které měly blok 64 bitový. AES podporuje tři délky klíče: 128, 192 a 256 bitů a v závislosti na tom se částečně mění algoritmus (počet rund je po řadě 10, 12 a 14). Větší délka bloku a delší klíče zabraňují mnoha útokům, které byly aplikovatelné na DES a jiné blokové šifry. AES má domovskou stránku http://csrc.nist.gov/encryption/aes/, která je věnována vzniku, konferencím, vědeckým zprávám a dalším informacím. AES nemá slabé klíče jako jeho předchůdce a měl by být odolný proti všem známým útokům, i proti nejnovějším metodám lineární a diferenciální kryptoanalýzy. Na referenčním počítači 200 MHz Pentiu Pro PC byla dosažena rychlost zašifrování cca 30 - 70 Mbitů/s podle použitého programovacího jazyka a délkách klíče. Algoritmus zašifrování i odšifrování se dá výhodně programovat na různých typech procesorů, má malé nároky na paměť i velikost kódu a je vhodný i pro paralelní zpracování. Pokud půjde vše podle předpokladů, AES bude platným šifrovacím standardem po několik desetiletí. Proto se předpokládá, že bude mít obrovský vliv na počítačovou bezpečnost.



Obr.: Rundovní funkce AES

10.3.1. Popis

Autory AES jsou belgičtí kryptologové Joan Daemen a Vincent Rijmen. AES je 128bitová bloková šifra, která podporuje tři délky klíče: 128, 192 a 256 bitů. Označíme-li délku klíče (Nk) jako počet 32-bitových slov, máme Nk = 4,6 a 8. AES je iterativní šifra, přičemž počet rund Nr se mění podle toho, jak dlouhý klíč se zpracovává: Nr = Nk + 6, tj. je to 10, 12 nebo 14 rund. Toto opatření odráží nutnost zajistit konfúzi vzhledem ke klíči. Algoritmus pracuje s prvky Galoisova tělesa GF(2^8) a s polynomy, jejichž koeficienty jsou prvky z GF(2^8). Bajt s bity (b_7,...,b_0) je proto chápán jako polynom $b_7x^7 + ... + b_1x^1 + b_0$ a operace "násobení bajtů" odpovídá násobení těchto polynomů modulo $m(x) = x^8 + x^4 + x^3 + x^1 + 1$.

10.3.2. Rundovní klíče

AES využívá 4 + Nr*4 rundovních klíčů (32-bitových slov), které se definovaným způsobem derivují ze šifrovacího klíče. Před zahájením první rundy zašifrování se provede úvodní zašumění, kdy se na otevřený text naxorují první čtyři rundovní klíče (128 bitů na 128 bitů). Potom následuje Nr shodných rund (s výjimkou poslední, kdy se neprovede operace MixColumn), při kterých výstup z každé předchozí rundy slouží jako vstup do rundy následující. Tím dochází k postupnému mnohonásobnému zesložiťování výstupu.

10.3.3. Runda

Na počátku každé rundy se vždy vstup (16 bajtů) naplní postupně zleva doprava a shora dolů po sloupcích do matice 4x4 bajtů $A = (a_{ij})_{i=0..3,j=0..3}$. Na každý bajt matice A se zvlášť aplikuje substituce, daná pevnou substituční tabulkou **ByteSub**. Potom se řádky matice A cyklicky posunou postupně o 0-3 bajty doleva (operace **ShiftRow**, první řádek o 0, druhý o 1, třetí o 2 a čtvrtý o 3), čímž dochází k transpozici na úrovni bajtů. Dále se na každý jednotlivý sloupec matice aplikuje operace **MixColumn**, která je substitucí 32 bitů na 32 bitů. Tuto substituci lze však popsat lineárními vztahy – všechny výstupní bity jsou nějakou lineární kombinací vstupních bitů. Označíme-li jednotlivé bajty v rámci daného sloupce matice A (shora dolů) jako a_0 až a_3, pak výstupem budou jejich nové hodnoty b_0 až b_3, podle vztahů

$b_0 = 0x02*a_0 \oplus 0x03*a_1 \oplus 0x01*a_2 \oplus 0x01*a_3$,
$b_1 = 0x01*a_0 \oplus 0x02*a_1 \oplus 0x03*a_2 \oplus 0x01*a_3$,
$b_2 = 0x01*a_0 \oplus 0x01*a_1 \oplus 0x02*a_2 \oplus 0x03*a_3$,
$b_3 = 0x03*a_0 \oplus 0x01*a_1 \oplus 0x01*a_2 \oplus 0x02*a_3$,

kde násobení je zmíněné násobení prvků $GF(2^8)$. Konstantní prvky tohoto pole jsou v soustavě rovnic vyjádřeny hexadecimálně (s prefixem 0x). Jako poslední operace rundy se vykoná transformace **AddRoundKey**, v rámci níž se na jednotlivé sloupce matice A zleva doprava naxorují 4 odpovídající rundovní klíče. Tím je jedna runda popsána a začíná další. Po poslední rundě se šifrový text jen vyčte z matice A.
Při odšifrování se používají operace inverzní k operacím, použitým při zašifrování, neboť všechny jsou reverzibilní.
Podrobný popis lze nalézt v [17].

10.3.4. Poznámka: Vlastnosti substitučního boxu AES

Nelinearity v AES se objevují pouze v substituci ByteSub. V roce 2002 bylo zjištěno, že vzájemné vztahy výstupních (y1,..,y8) a vstupních (x1,..,x8) bitů lze popsat implicitními rovnicemi f(x1,..,x8,y1,..,y8) = 0 pouze druhého řádu [19]. Dále bylo v témže roce zjištěno, že funkce jednotlivých výstupních bitů mají mezi sebou tento lineární vztah: $y_j = y_1(D_{1,j} * x) + c$, kde c je binární konstanta a $D_{1,j}$ jsou konstantní binární matice 8x8, j = 2...8 [20].

10.3.5. Poznámka: MixColumn jako lineární transformace

Podívejme se na první rovnici v transformaci MixColumn:
$b_0 = 0x02*a_0 \oplus 0x03*a_1 \oplus 0x01*a_2 \oplus 0x01*a_3$.
Ukážeme, že výstup je lineární funkcí vstupů, a to například na prvním sčítanci **v = {02}*a**, kde $a = \{a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0\}$. Máme

$$v = \{02\}*a = (x^1)*(a_7x^7+a_6x^6+a_5x^5+a_4x^4+a_3x^3+a_2x^2+a_1x^1+a_0x^0) =$$
$$= (a_7x^8+a_6x^7+a_5x^6+a_4x^5+a_3x^4+a_2x^3+a_1x^2+a_0x^1) =$$
$$= (a_7x^8+a_6x^7+a_5x^6+a_4x^5+a_3x^4+a_2x^3+a_1x^2+a_0x^1) + a_7*m(x) =$$
$$= (a_7x^8+a_6x^7+a_5x^6+a_4x^5+a_3x^4+a_2x^3+a_1x^2+a_0x^1) +$$
$$a_7x^8 \quad +a_7x^4+a_7x^3 \quad +a_7x^1 + a_7 =$$
$$= (a_6x^7+a_5x^6+a_4x^5+(a_3 + a_7)x^4+ (a_2 + a_7)x^3+a_1x^2+ (a_0 + a_7)x^1 +a_7 =$$
$$= \{a_6, a_5, a_4, (a_3 + a_7), (a_2 + a_7), a_1, (a_0 + a_7), a_7\}.$$

Ukázali jsme, že transformace a $\rightarrow$ **{02}*a** je lineární, a to:
$$\mathbf{\{a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0\} \rightarrow \{a_6, a_5, a_4, (a_3 + a_7), (a_2 + a_7), a_1, (a_0 + a_7), a_7\}.}$$

Podobně bychom vyjádřili i ostatní sčítance v rovnicích pro MixColumn. Celá operace MixColumn, převádějící 32bitový sloupec na 32bitový sloupec je tak lineární transformací 32 vstupních bitů na 32 výstupních bitů a lze popsat konstantní maticí.

11. Literatura

[16] Jiří Janeček, Válka šifer – výhry a prohry československé vojenské rozvědky (1939 – 1945), Votobia, 2001, ISBN 80-7198-505-8

[17] AES, FIPS PUB 197, domovská stránka http://csrc.nist.gov/encryption/aes/

[18] Data Encryption Standard (DES), FIPS 46-3, October 1999, http://csrc.nist.gov/CryptoToolkit/tkencryption.html

[19] N. Courtois, J. Pieprzyk, *Cryptanalysis of Block Ciphers with Overdefined Systems of Equations*, ASIACRYPT 2002, pp. 267 – 287, Cryptology ePrint Archive: Report 2002/044, 2002, http://eprint.iacr.org/2002/044/

[20] J. Fuller and W. Millan, *On Linear Redundancy in the AES S-Box*, FSE 2003, pp. 74 – 86, Cryptology ePrint Archive: Report 2002/111, 2002, http://eprint.iacr.org/2002/111/

[21] M. Matsui, *Linear cryptanalysis method for DES cipher*, EUROCRYPT 1993, pp. 386 – 397

[22] E. Biham, A. Shamir, *Differential cryptanalysis of DES-like cryptosystems*, CRYPTO 1990, pp. 2 – 21

[23] E. Barkan, E. Biham, N. Keller, *Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communications*, CRYPTO 2003, pp. 600 – 616, http://cryptome.org/gsm-crack-bbk.pdf

[24] S. R. Fluhrer, D. A. McGrew, *Statistical Analysis of the Alleged RC4 Stream Cipher*, FSE 2000, http://www.mindspring.com/%7Edmcgrew/rc4-03.pdf