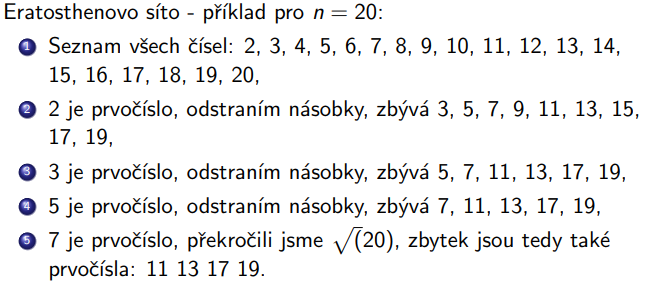
1. **Prvočísla – popište způsob generování a uveďte příklad pravděpodobnostního testu a skutečného testu.**

Jednoduchý ale nepraktický způsob je tzv **Eratosthenovo síto:**

1. Sestaví se seznam všech čísel od 2 do n,
2. První číslo na seznamu se vyřadí a určí za prvočíslo,
3. Všechny násobky nalezeného prvočísla se vyřadí ze seznamu,
4. Pokračuje se, než se všechna čísla vyřadí či dokud nedosáhneme √ (n), pak jsou všechna zbylá čísla prvočísla.



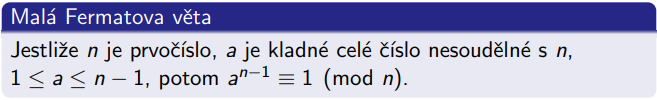
**Praktický postup:**

1. Vygeneruji náhodné číslo požadované velikosti
2. Otestuji, zda se jedná o prvočíslo
3. V Případě, že se nejedná o prvočíslo, opakuji od 1.

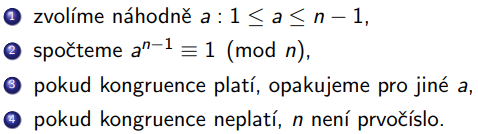
**Pravděpodobnosti Testy:**

Kvůli výkonu se nevyužívají pravé testy prvočíselnosti, ale testy pravděpodobné prvočíselnosti (Fermatův test, Miller-Rabinův test)

* Fermatův test:
  + Založen na modulárním mocnění
  + Využívá malou fermatovu větu.



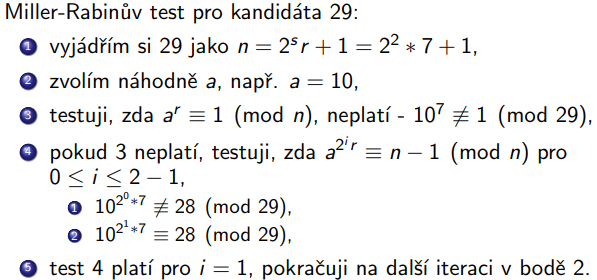
**Postup:**

****

* Nedokáže prokázat prvočíselnost
* Čím více a vyzkoušíme a kongruence platí, tím pravděpodobněji je n prvočíslo.
* Test je pouze orientační, v praxi se příliš nepoužívá.

**Miller-Rabinův test**

* Spolehlivější než Fermatův test, obsahuje méně pseudoprvočísel
* Využívá se opět rychlosti modulárního mocnění.
* Opět se volí základ a, který se dále mocní, zde ale nejen na n -1 ale i jeho dělitele.

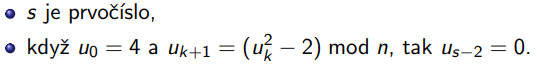


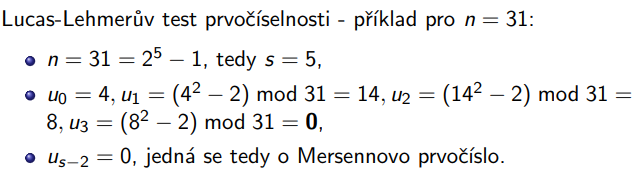
**Skutečné testy:**

Prokazuji, že vybrané číslo je skutečně prvočíslo. (Lucas-Lehmerův test).

**Lucas-Lehmerův test**

Určen pro prvočísla ve speciálním tvaru (p = 2s -1). (Mersennovo prvočíslo)





**2) Teorie čísel, algebraické struktury – popište účel a způsob výpočtu Eulerovy funkce. Popište požadavky na grupu a způsob generování grupy pro algoritmy založené na problému DL.**

**Teorie čísel**

* Zkoumání vlastností celých čísel
* Prvočíslo – číslo ≥ 2, která má pouze triviální dělitele
* Každé číslo ≥ 2 je prvočíslo nebo se dá zapsat jako součin prvočísel

Dělitelnost (zbytek a mod b), největší společný dělitel (GCD, společné prvky), nejmenší společný násobek (LCM, všechny prvky s největší mocninou)

* Pokud GCD = 1, jedná se o nesoudělná čísla. Může se jednat i o prvočísla
* GCD prvočísla a jiného čísla je buď 1 nebo prvočíslo

**Eulerova funkce φ(n)**

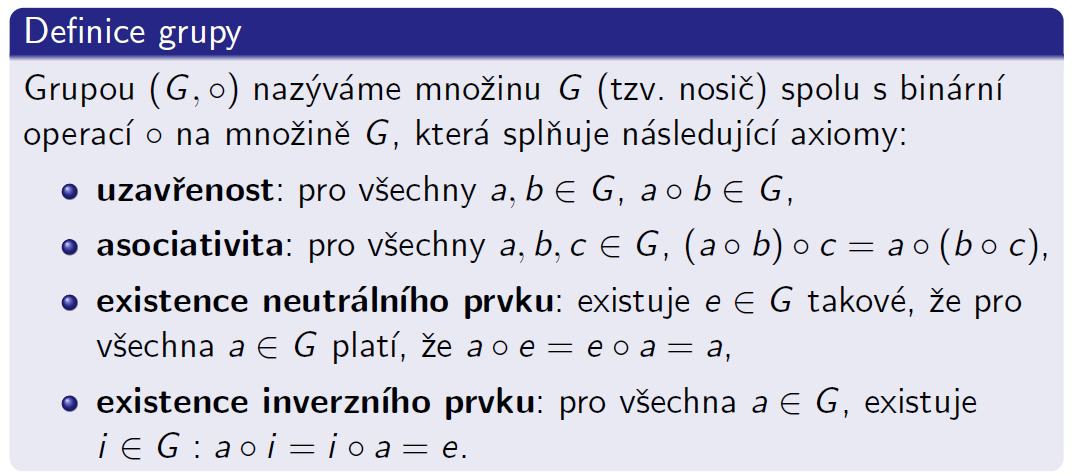
* Udává počet celých čísel, která jsou nesoudělná s n. (Větší než 1 a menší než n)
* Pro prvočíslo φ(p) = p – 1
* φ (m\*n) = φ(m) \* φ(n)
* φ(pk) = (p-1)\*pk-1

Modulární aritmetika

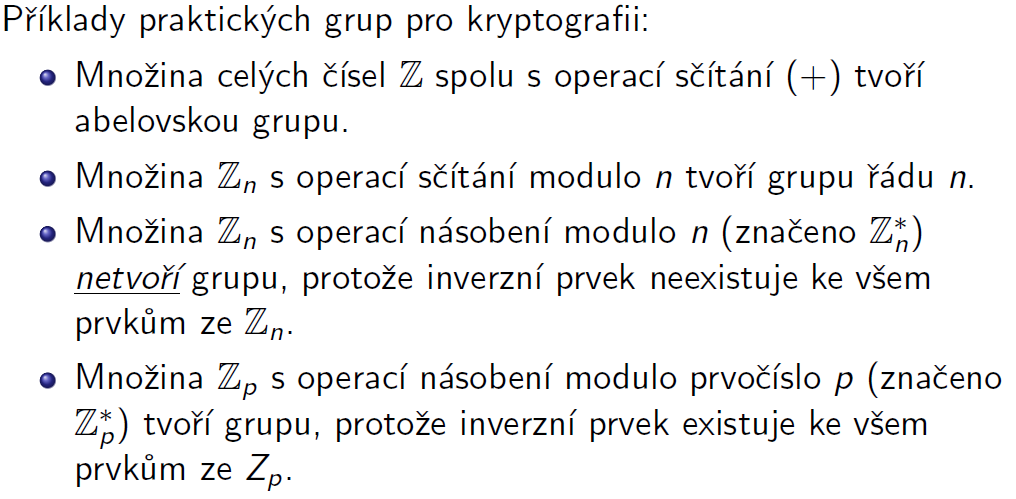
* modulo, zbytek po dělení n
* multiplikativní inverzní prvek a-1 : a \* a-1 mod n = 1
* aditivní opačný prvek (-a) : a + (-a) mod n = 0
* square and multiply nebo postupné mocnění

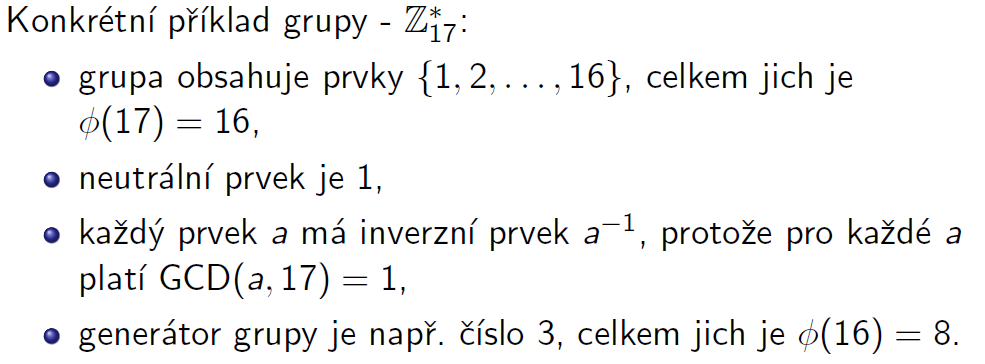
Algebraické struktury – grupy, tělesa, konečná tělesa, okruhy. Množina prvků s operacemi a vlastnostmi

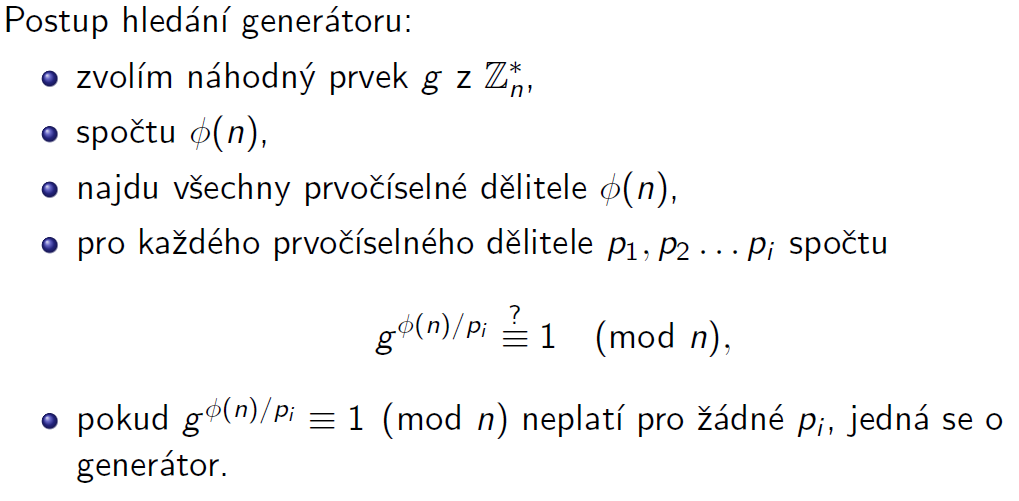
**Grupy**



Cyklická grupa – obsahuje takový prvek, který při postupném mocnění generuje všechny prvky grupy. Tento prvek je generátor značený g.

* Řád prvku – pro prvek nejmenší kladné celé číslo t, pro které platí at = 1
* Řád generátoru grupy se rovná řádu grupy.
* Počet generátorů lze zjistit ze řádu k grupy, jako φ(k)
* Pro prvočíselnou grupu je počet generátorů φ(φ(p))





1. **Modulární aritmetika – popište algoritmus Square and Multiply a Čínskou větu o zbytcích.**

**Square and Multiply**Square – mocnění x2; Multiply – násobení x \* y  
Algoritmus postupně pracuje s jednotlivými bity.   
Pokud je hodnota bitu 1, je provedeno Square and Multiply.   
Pokud je hodnota bitu 0, provede se jen operace Square.

Příklad: 413  
exponent se převede do binární soustavy: 13 = 1101  
Bity se čtou zleva doprava. První bit =1 => square and multiply.   
Začíná se mocněním, proto je základ = 1 => (12 \* 4). Násobí se mocněným číslem = 4.  
Druhá hodnota bitu = 1.   
Předchozí krok se opíše => (12 \* 4) přidá se square and multiply => ((12 \* 4)2 \* 4).  
Třetí bit = 0 => (((12 \* 4)2 \* 4)2).  
Poslední bit = 1 => ((((12 \* 4)2 \* 4)2)2 \* 4).

Teď už jen postupně umocňovat, násobit a modulovat :)

**Chinese remainder theorem**

Čínská věta o zbytcích umožňuje urychlit výpočet modulárního mocnění (třeba v RSA) pomocí rozkladu kongruence na systém více kongruencí s menším modulem.  
x ≡ . Kde M = m1m2…mk; Ni = M/mi; Li = Ni-1 mod m.

Příklad: x ≡ 3 (mod 6) => m1 = 6  
 x ≡ 2 (mod 7) => m2 = 7  
M = 6 \* 7 = 42,  
N1 = M/m1 = 7; N2 = M/m2 = 6,  
L1 = 7-1 mod 6 = 1; L2 = 6-1 mod 7 = 6,

X =

Místo #kongruence vlož (≡)

**4) Symetrická kryptografie – proudové šifry, synchronní a asynchronní proudové šifry.**

(+) Velká rychlost šifrování, pro velké objemy dat, stejný klíč

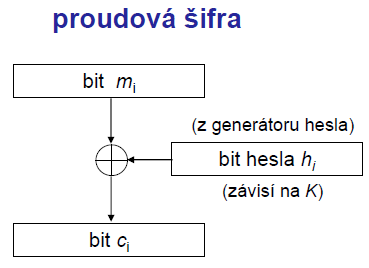
(-) nutné udržet klíč v tajnosti, často uložen např v tokenech

1. **Proudové**

Šifrován bit po bitu (Vernamova šifra), Hašovací(f)

Využití při použití s zařízením s omezenou pamětí na průchozí data.

Ci = f(mi, K)



**Základní vlastnosti:**

* **Konfuse =** přerušení vztahu mezi M a C (substituce, S-box, nelineární transformace)
* **Difuse =** Změna M se promítá do mnoha míst C (permutace P-box, permutace nezvyšuje bezpečnost)
* **Výhody proud šifry =** velká rychlost šifrování, každý symbol šifrován samostatně.

Malé šíření chyb – případná chyba ovlivní pouze jeden znak

* **Nevýhody =** Nízká úroveň difůze – všechny informace o znaku transformovány opět do jednoho znaku. Nebezpečí při použití stejného hesla 2x (Stejný výstup).
* **Synchronní šifry**

Generuje se zde proud pseudonáhodných čísel nezávisle na vstupním textu, nebo zašifrované zprávě. Následně dojde ke kombinaci generovaných čísel se vstupujícím textem (kódování) , nebo se šifrovaným textem (dekódování). Nejčastěji je využita funkce XOR.

Pokud jsou čísla přidána nebo odebrána ze zprávy v průběhu přenosu, synchronizování je ztraceno. Jestli je chyba zanesena do jednoho znaku, chyba se nesíří do dalších části zprávy.

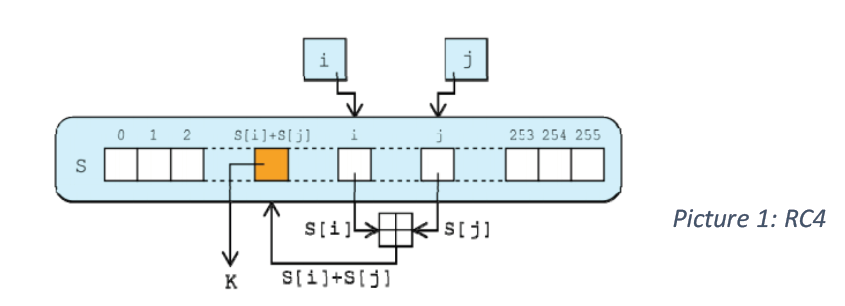
* **Asynchronní**

Používá několik předchozích N čísel šifrovaného textu k vypočítání šifrovacího klíče.

Příjemce bude automaticky synchronizován s generátorem šifrovacích klíčů po obdržení N čísel zašifrovaného textu. (CFB mode).

**Např : RC4**

* Šifra použita např pro šifrovaný přenos webových stránek (HTTPS).
* Jednoduchá a rychlá
* Generuje náhodný proud bitů (keystream) – přitom se vytvoří pole bajtů o délce 256 a dvě celočíselné proměnné i,j.
* Pomocí XORU se smýchá keystream s plaintextem a vznikne šifrovaný text
* Je možný útok hrubou silou (krátká délka klíče).



**5)Blokové šifry - Product Ciphers, konstrukce, Feistelova síť, DES, AES, základní módy blokových šifer.**

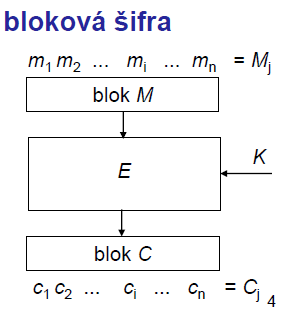
* **Blokové šifry**

Otevřený text se zpracovává po n-bitových blocích dat (Šifruje se skupina symbolů otevřeného textu jako jeden blok).

Blok šifrovaného textu má stejnou délku jako blok otevřeného textu.

Velikost bloku má zásadní význam pro bezpečnost algoritmu. Převážně bloky o 64 – 128 bitech.

**Bloková šifra =** Ci = f(m1…mn, K)



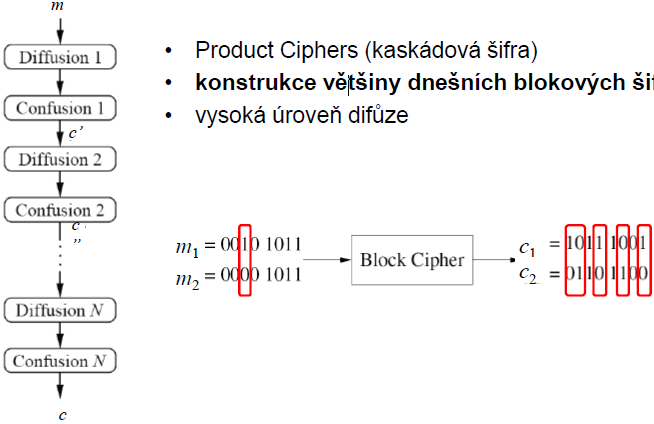
(+) **Vysoká úroveň difůze** = informace otevřeného textu se difunduje (promítá) do několika symbolů šifrového textu.

**Imunita vůči narušení** = nelze změnit symbol v bloku, aniž by to bylo při dešifrování odhaleno.

(-) **Zpoždění** = pro započetí šifrování je nutno přijmout celý blok.

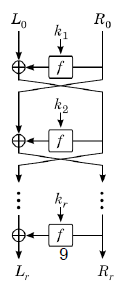
**Šíření chyb** = chyba ovlivní transformaci všech ostatních znaků téhož bloku.

**Kaskádová šifra**



**Iterativní provádění =** jednoduchá funkce provádějící šifrování jednoho bloku, která je několikrát po sobě zopakována. (**Runda**).

**Feistelová síť**



vstupní blok o **n** bitech rozdělíme na 2 části (**n**/2) (dostaneme L0 a R0 viz obr).

Vstup do i-té rundy je výstupem vždy rundy předcházející. Kde *f*(*x*) je libovolná funkce.

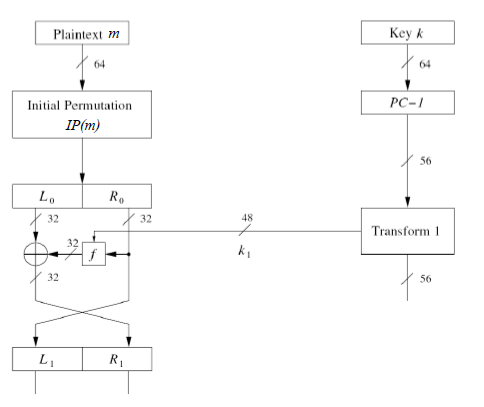
**DES ( Data Encryption Standard)**

**Bloky o velikosti 64bitů, klíč dlouhý 56 bitů, 16 rund a také 16 rundovních klíčů odvozených z hlavního DES klíče (hlavní = 64b, 56b klíče +8 paritních).**

Paritní bit = sudá, lichá parita, vztaženo k počtu jedniček ve slově.

**Zvláštní klíč pro každou rundu.**

V dnešní době rozluštitelný hrubou silou, zesílení triple DES.

Zprávy 64b po počáteční permutaci rozděleny do dvou registrů 2x32b L**i** a R**i**.

Výstup 32b Ri je přiveden do funkce f, její výstup 32b je XOR s 32B Li.

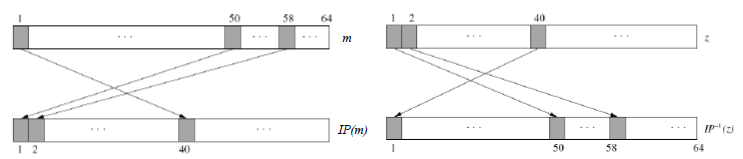
Následuje prohození hodnot pravého registru s Ri-1 s Levým Li.

Postup opakován až do 16 rundy, po které nastává permutace inverzní k počáteční permutaci.

Šifrování a dešifrování je rozdílné v pořadí použití rundovních klíčů.

**Kroky :**

* **Počáteční a konečná permutace**

****

Bitová permutace, inverzní operace

* **Funkce *F(x)***

Vstupy jsou Ri a rundovní klíč ki

Expanze E (rozšíření n/2 bloků z 32 na 48b) , následně se XORuje rundovní klíč ki. Probíhá substituce S-box a následně permutace P.

* Expanzní funkce E
* Přičtení rundovního klíče
* Substituce S box = **klíčový prvek pro bezpečnost DES (**Substituce = dochází k záměně množiny symbolů za jinou množinu smybolů)
* Permutace P = zavádí difuzi
* **Plánování klíčů**

Klíč 56b dlouhý rozdělen na 2\*28b, registry C0, D0

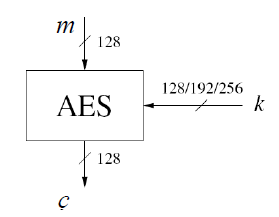
V rundách **i = 1,2,9,16,** je každý registr rotován o 1 bit. V **ostatních rundách** je rotace o **2bity**.

**Při dešifrování je třeba rundovní klíče generovat v opačném pořadí.**

1 runda bez rotace, v dalších rundách **i = 2, 9, 16**, rotace o **1bit**. V ostatních rundách rotace o **2 bity**.

AES

**Počet rund závisí na délce klíče**. ( Symetrická šifra, jeden klíč pro šifrování i dešifrování)

**Klíč [b] Počet Rund**

128 10

192 12

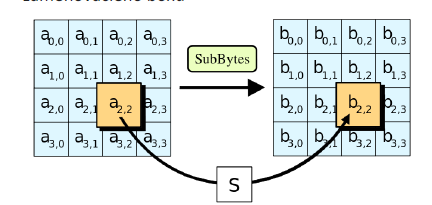
256 14

* Data jsou rozděleny do bloků o pevné délce délky 16B
* Blok je matice s rozměry 4x4B.

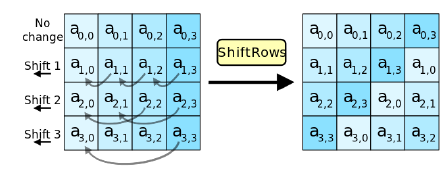
Každá runda se skládá z vrstev „layers“.

**Postup Algoritmu:**

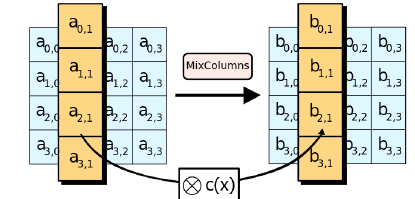
* Expanze klíče (Key Expansion) = vytvoření podklíčů ( odvozených z klíčů šifry).
* Inicializační část = Přidání podklíče pomocí XORu.
* Iterace = Záměna bytů = nelineární nahrazovací krok. V Matici nahrazujeme každý bit jiným bitem pomocí 8bitové vyhledávací tabulky.

****

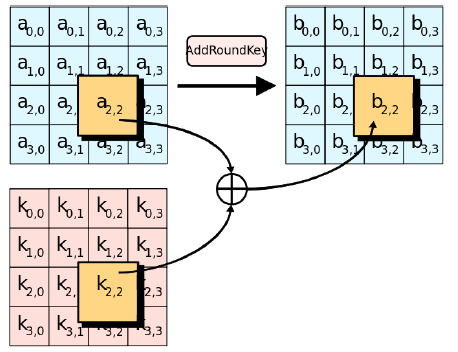
* Prohození řádků = Jednotlivě se přehazují byty v každém řádku o určitý posun.



* Kombinování sloupců = V každém sloupci se kombinují 4 bity. Funkce vezme 4 vstupní bity a vrátí 4 bitový výstup.

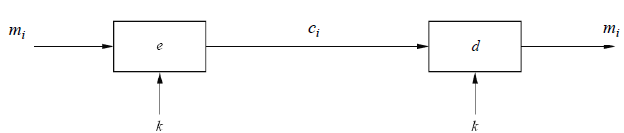


* Přidání podklíče = Kazdý bite je kombinovaný s podklíčem. Pro každou iteraci je klíč odvozený z hlavního klíče. Podklíče mají stejnou velikost.



Módy blokových šifer

* ECB – Electronic Code Block
* CBC – Cipher Block Chaining
* OFB – Output FeedBack
* CFB – Ciphertext FeedBack
* CTR – Counter mode
* MAC – Message Authentication Code
* GCM – Galois Counter Mode

**ECB**

Každý blok šifrován samostatně.

Opakovaný blok je shodné šifrován.

Pro krátké zprávy, rozesílání klíčů.

Nezajišťuje integritu otevřeného textu.

Šifrování ci =e(k, m) Dešifrování mi =d(k, ci)

**Produkuje identické bloky i po šifrování**

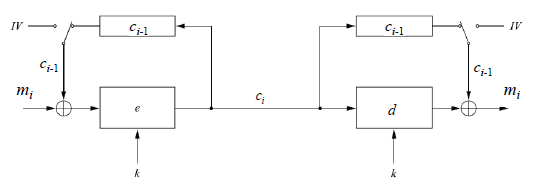
**CBC (Cipher Block Chaining)**

Řetězení šifrového textu

Každý blok před šifrováním XOR s předchozím zašifrovaným blokem.

První blok XOR s IV (inicializačním blokem) tzv „Nonce“, který se příjmací straně vyšle otevřeně.

Poslední blok doplněn na potřebou délku, kontrola při dešifrování.

****

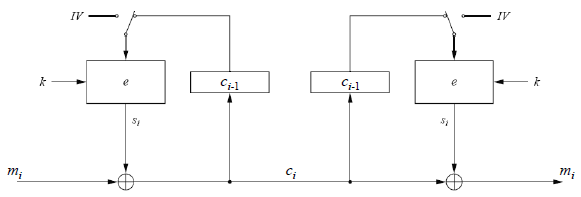
**CFB (Ciphertext FeedBack)**

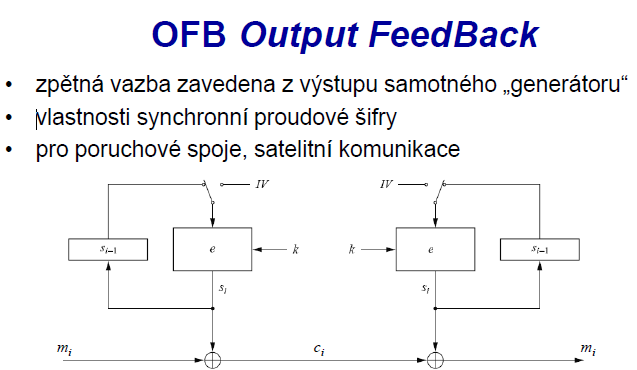
Využití blokové šifry jako zdroje „hesla“ pro proudovou šifru.

Zdroj hesla ovlivněn zpětnou vazbou branou ze zašifrovaného textu.

Při dešifrování náchylné na chybovost spoje.

Schopnost „samosynchronizace“.





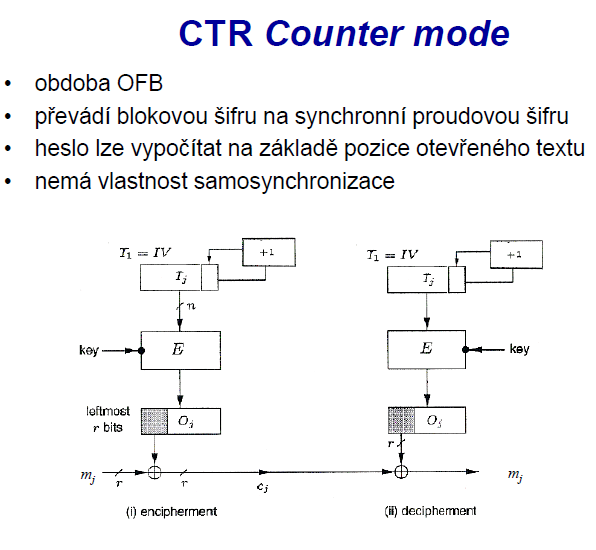
**Vlastnosti CFB + OFB**

Používá se šifrovací transformace E, stačí zde pouze šifrovací funkce.

Z Výstupního bloku (OFB) nebo šifrového textu (CFB) stačí přenést pouze část náhodných **r** bitů.

OFB je čistě synchronní proudová šifra, proud klíče není ovlivněn ani otevřeným ani šifrovým textem.

Vstupní blok mé nejvýše 2N stavů, kde N je délka vstupního bloku v bitech. Po 2N blocích se začne proud klíče opakovat.



**Metoda solení**

U všech módu kde se využívá inicializační hodnota **IV** (CBC, OFB, CFB, CTR) lze použít tuto metodu k utajení IV.

Hodnota IV se pošle druhé straně otevřeně, k šifrování se ale používá jiná inicializační hodnota **IV´.** Získá se z IV a klíče K na obou stranách komunikačního kanálu stejným postupem.

**6)Asymetrické algoritmy - RSA, Diffie-Hellman, ECDH systém a jejich využití pro digitální podpis.**

Otevřený Text: **M** – message = zpráva Z, někdy P plaint text = otevřený text (OT).

Šifrovací funkce **E** – Encryption function

Šifrový text **C** – cipher text = šifrovaný text (ŠT)

Dešifrovací funkce **D** – Decryption function

**K1** = public key, veřejný klíč, který slouží k šifrování

**K2** = private key, soukromý klíč, slouží k dešifrování.

**Šifrování C = E (K1, M)**

**Dešifrování M = D (K2, C)**

**Oblasti použití:** šifrování zpráv, výměna klíčů, autentizace, digitální podpis

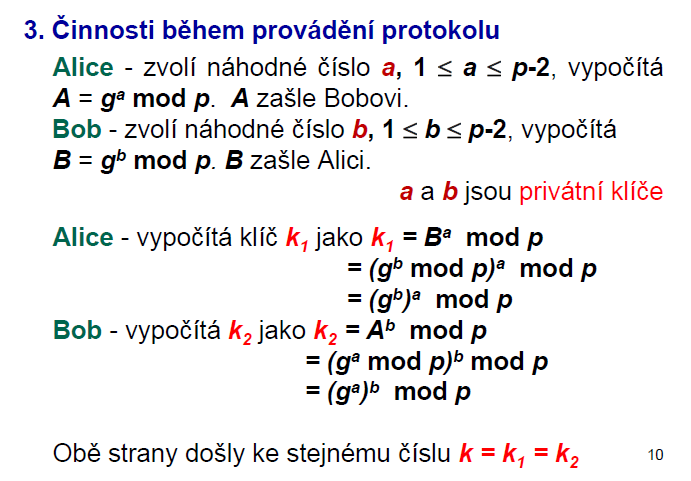
**Vlastnosti:** Pomalejší než symetrické algoritmy (100-1000x), velké délky klíčů.

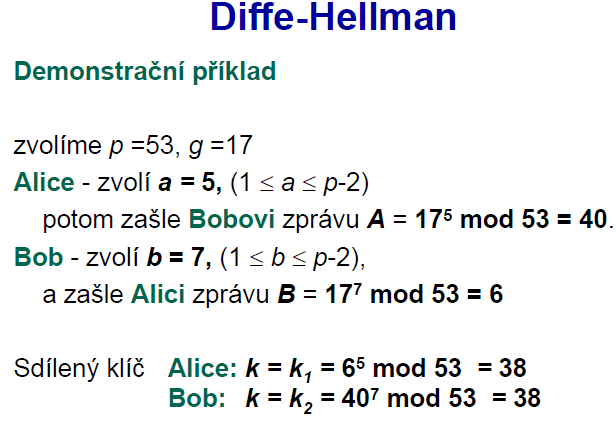
**Diffe-Hellman**

Bezpečnost založena na obtížnosti výpočtu diskrétních logaritmů (y=gx mod p)

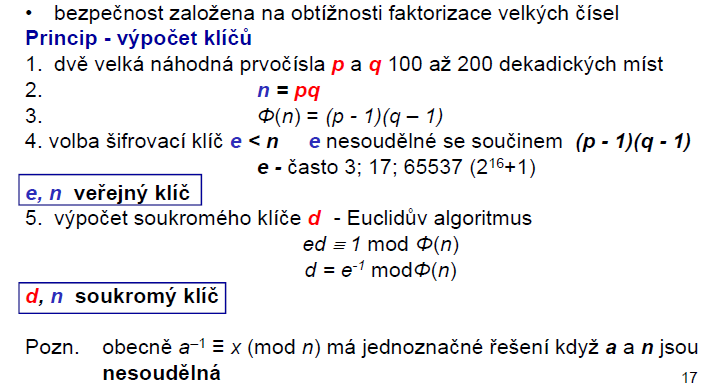
Slouží pro výměnu klíčů symetrických systému.

1. Zvoleno vhodné prvočíslo **p** a generátor **g** je elementem Zp\* ( každé číslo od 1 do **p** může být generováno jako nějaká mocnina g).
2. Alice> Bob = **A** = ga mod p …… Bob > Alice **B** = **gb mod p**





**RSA**

****

****

Se vzrůstající délkou modulu roste bezpečnost pomalu, ale výpočetní nároky rychle.

RSA-1024 srovnatelné s symetrickou šifrou s klíčem 80 bitů.

RSA-2048 přibližně odpovída klíči délky 112 bitů (3-DES)

RSA-3072 odpovídá klíči délky 128 bitů (AES-128)

**HW** 1000x pomalejší než DES, SW 100x pomalejší než DES

HW čipy 512, 1024 bitů rychlost šifrování až 1 Mbit/s, využíváno v čipových kartách

Výkon RSA s velkými moduly je velmi nízký.

**Čas nutný k podepisování roste s třetí mocninou délky klíče.**

K výpoćtu podpisu pomocí RSA-2048 je potřeba 8x delší čas než s RSA-1024.

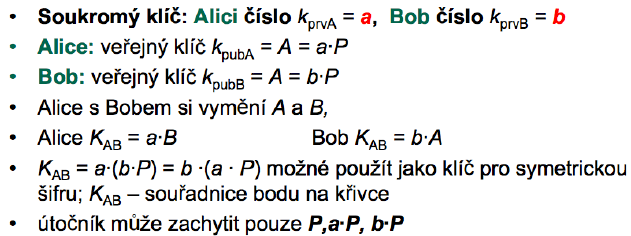
**ECDH systém a jejich využití pro digitální podpis.**

* Umožňuje dvěma stranám získat sdílenou tajnou informaci jako je například klíč pro klasickou symetrickou šifru. Na rozdíl od asymetrických systému je zde potřeba kratší délka klíče.
* Eliptická křivka je definována nad Fp a bod P na eliptické křivce, který je generátorem cyklické grupy.
* Eliptická křivka Fp je tvořena konečným počtem bodů při rovnici:

y2=x3+ax+b

* Počet bodů křivky udává tzv řád křivky.

**Problematika postavena na diskrétním logaritmu.**



ECDH

* Slouží pro podpis a ověření podpisů.
* Nemožné použít na šifrování
* Analogicky k DSA

**7)Hašovací funkce vlastnosti, princip, použití, kolize, odolnost proti kolizím, příklady.**

**Vlastnosti:**

* K „libovolně“ velkému vstupu M – pevná délka výstupu h
* **Jednocestnost**

Pro danou zprávu M lze snadno spočítat h= H(M),

Je-li dáno h je velmi těžké vypočítat M, ne však nemožné.

* **Bezkoliznost**

Je velmi těžké nalézt různá M a M´ tak aby **H(M) = H(M´)**

Heš, hash, hašový kód, otisk, miniatura, parita= výstupní kód s předem pevně definovanou délkou.

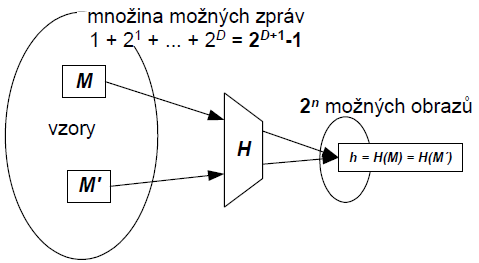
**Odolnost proti kolizím**

* **Kolize prvního řádu**

Nalezení dvou libovolných zpráv **M** a **M´** pro které platí **H(M) = H(M´)**

* **Kolize druhého řádu**

K dané zprávě Mx nalézt zprávu My tak aby platilo **H(Mx)=H(My).**

****

**Použití:**

* Pro uložení hesel – otisky hesel
* Uložení klíčů – otisky klíčů
* Jednoznačná identifikace dat, tvorba digitálních podpisů, otisky zprávy,
* Kontrola Integrity, porovnání souborů – otisk souboru,
* Prokazování autorství

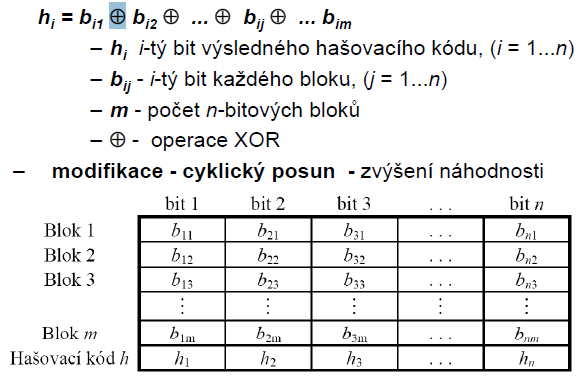
**Typy H(f)**

* Jednoduché hašovací funkce
* Zřetězené hašovací funkce (blokové šifry)
* Iterační hašovací funkce (MD, SHA1,2)
* Konstrukce typu „houba“ (SHA-3)

**Jednoduchá H(f)**

Zpráva je rozdělena na posloupnost n-bitových bloků, které jsou zpracovány lg operací XOR (bit po bitě).

h = h1 h2… hn



**Zřetězené H(f)**

Využívá symetrické blokové šifry, obdoba módu CBC bez použití tajného klíče.

Zpráva **M** rozdělena na n bloků (m1,m2,….mn).

IV = h0 = počáteční hodnota

hi = E (mi, hi-1)

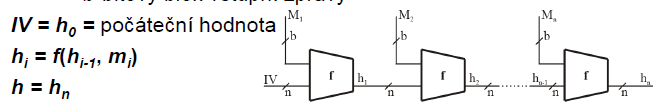
h= hn

**Iteračni H(f)**

Oproti předchozí variantě využívá speciální kompresní funkci f, která je funkcí dvou proměnných (má dva vstupy).

**n**-bitový vstup předchozího kroku

**b**-bitový blok vstupní zprávy



**Princip H(f)**

1. Zpráva se zpracuje po blocích
2. Doplnění posledního bloku na určitou délku.
3. Následuje iterační proces f – kompresní funkce
4. Máme výstupní transformaci

**8) PKI - certifikát X.509 struktura, certifikační autorita základní části, časová razítka, autorita časových razítek.**

PKI = public key infrastructure

Souhrn technických a organizačních prostředků spojených s vydáváním, správou a používáním a odvoláním platnosti kryptografických klíčů a certifikátů.

Zabraňuje používání falešné identity

**Veřejný klíč platný pouze v případě jeho potvrzení důvěryhodnou stranou. (Např. Certifikační autoritou.)**

**Certifikační autorita:**

**Nejdůležitější aktiva potřebné ochraňovat u CA:**

* Soukromý klíč CA
* Databáze uživatelů CA
* Archiv soukromých šifrovacích klíčů uživatelů, pokud CA tuto službu poskytuje

**Základní časti CA**

* **Třída 1**

CA (Certifikační autorita) ručí pouze za jednoznačnost certifikátu.

Žadatel vyplní formulář serveru a protokolem HTTPS a jej odešle (stačí autentizace serveru – klient anonymní).

* **Třída 2**

Jako třída 1 + **CA kontroluje totožnost uživatele.**

Uživatelé osobně dochází se svými žádostmi. RA (registrační autorita) ověří totožnost uživatele a odesílá žádost k vyřízení CA.

* **Třída 3**

Jako třída 2, ale **vydané certifikáty jsou určeny výhradně pro konkrétní aplikaci**.

CA třídy 3 uchovává svůj soukromý klíč v bezpečném hadwaru.

**Další časti CA**

* **Registrační autority** – žádost o certifikát. Ověří totožnost uživatele.

Verifikuje žádost o certifikát, předá žádost CA. CA ověří údaje z žádosti uživatele a údaje doplněné RA a vydá (nebo nevydá) příslušný certifikát soukromý klíč CA.

* **OnLine RA –** přijímání žádostí elektronickou cestou a lze zažádat o :

1. Obnovení certifikátu v době platnosti starého certifikátu.
2. Vydání nového certifikátu na základě jednorázového hesla pro vydání certifikátu.
3. Další certifikáty, žádosti podepisuje platným certifikátem
4. Odvolání certifikátu

* **IVR nebo telefonní záznamník** – k odvolání certifikátu jinou cestou (telefon)
* **Adresářové služby –** informace o uživatelích, kteří chtějí info o sobě publikovat.
* **DVCS server –** poskytuje informace o platnosti certifikátu, o platnosti listin a dále může poskytovat časová razítka.

**Certifikát X.509**

**Struktura:**

* **Verze**

0…X.509 verze 1

1…X.509 verze 2

2…X.509 verze 3

* **Sériové číslo**
* **Algoritmus použitý pro podpis**
* **Vydal (identifikace CA podle X.500)**
* **Platnost od-do**
* **Jméno a Adresa (identifikace vlastníka)**
* **Veřejný klíč**
* **Rozšíření certifikátu**
* **Digitální podpis certifikátu**

**Časová razítka**

* Elektronický ekvivalent časového určení a místa vlastního podpisu na listině.
* Řeší možné problémy vzniklé odvoláním certifikátu, (byl elektronický dokument podepsán před odvoláním)?
* **Časové razítko tedy především zajišťuje důkaz o existenci dokumentu v daném čase.**
* Nutné pro poskytování elektronických notářských služeb a zajištění dlouhodobé archivace elektronicky podepsaných dokumentů.
* Časové razítko je struktura obdobná certifikátu, která svazuje kontrolní součet (Hash) z dokumentu s časem.
* Je elektronicky podepsáno (vydáváno) autoritou pro vydávání časových razítek (TSA)

**Elektronicky podepsána struktura časového razítka obsahuje:**

1. Jméno vydavatele (jméno TSA (Jméno autority časových razítek))
2. Jedinečné sériové číslo razítka
3. Kontrolní součet (Hash) z dokumentu a čas

**Autorita časových razítek (TSA)**

* TSA musí podepisovat každé časové razítko privátním klíčem vyhrazeným pouze k tomuto účelu
* TSA musí být bezpečně napojena a v pravidelných intervalech synchronizována proti nejlépe **třem na sobě nezávislým zdrojům času**.

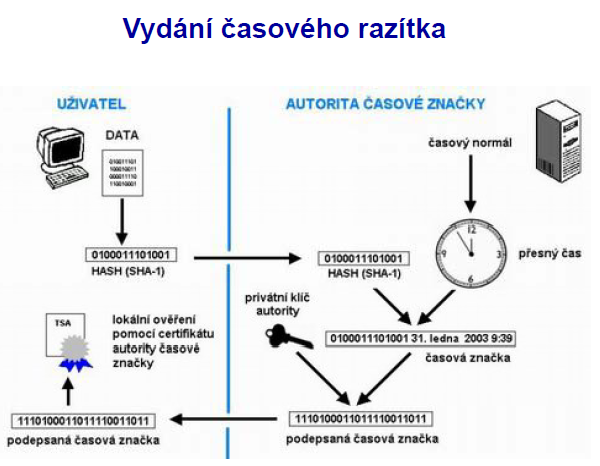
**Požadavky na zdroj času:**

1. Musí pocházet z oficiálního důvěryhodného zdroje – např od národní časové autority.
2. Čas nesmí být možné cestou změnit.
3. Vždy musí být možné zpětné dosledovat zdroj času, tedy celou hierarchii časových serverů.

Těchto požadavků dosahujeme využitím PKI, zajištujícím důvěrnost, integritu, neodmítnutelnost a autenticitu při časové synchronizaci.

**Vydání časového razítka:**

* Žádá se prostřednictvím klientské aplikace
* Klient vytvoří a odešle žádost o časové razítko ve standardizovaném formátu
* Žádost o časové razítko je datová struktura obsahující kontrolní součet z dokumentu.
* TSA v případě kladné odpovědi odesílá odpověď na žádost obsahující časové razítko.
* Časové razítko je elektronicky podepsaná datová struktura obsahující číslo časového razítka, kontrolní součet z dokumentu, čas a název vydavatele časového razítka elektronický ekvivalent časového určení a místa vlastního podpisu na listině.



# **9) Generování náhodných čísel - kryptografické generátory, požadavky, použití, princip, testování generátorů.**

**Generátory náhodných čísel:**

**Požadavky:**

* Statistická nezávislost
* Nepredikovatelnost
* Rychlé generování

**Použití:**

* Přímé šifrování náhodným heslem
* Generování klíčů pro symetrické šifry, asymetrické
* Výplň(padding) doplnění krátké zprávy na délku bloku
* Jednorázově použitá hodnota (nonce) v autentizačních protokolech, chrání před útoky typu replay attack
* Obrana proti útokům postranními kanály

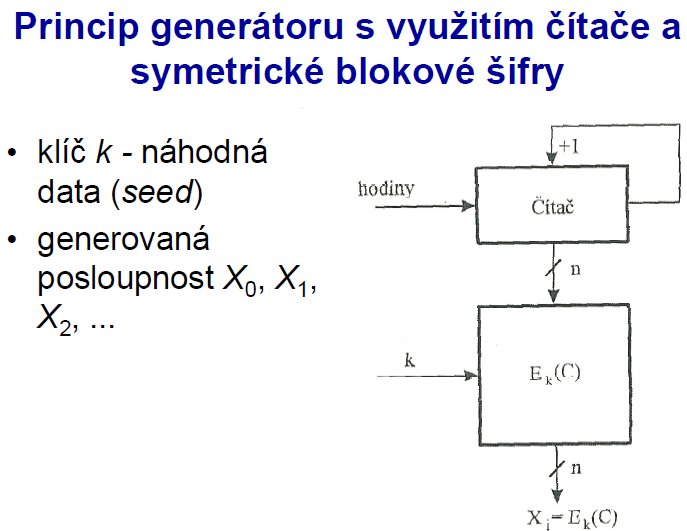
**Entropie:** Popisuje míru náhodnosti (Jak obtížné je hodnotu uhodnout).

* Entropie je mírou nejistoty hodnoty a závisí na pravděpodobnostech možných výsledků procesu, který ji generuje.
* Entropie se vztahuje k útočníkovi a jeho schopnosti předpovědět vygenerovanou hodnotu. **Pokud útočník následující generovanou hodnotu s jistotou zná, entropie je nulová –** Nulová bezpečnost aplikace, která takto generovanou náhodnou hodnotu využívá.

Fyzikální generátory ( true random posloupnost TRNG), Generátory Algoritmické (pseudo-random posloupnost PRNG), generátory smíšené.

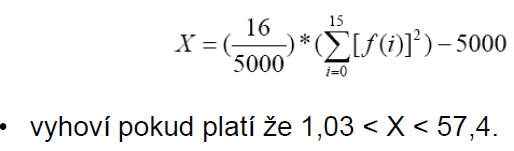
**Kryptograficky bezpečné PRNG**

* **Bloková šifra v režimu čitače:** náhodně se zvolí klíč (seed) a počáteční hodnota čítače *i*. Zvoleným klíčem se postupně šifrují hodnoty *i, i+1.* Nesmí dojít k vyzrazení klíče a počáteční hodnoty čítače.
* **Hašovací funkce aplikovaná na čítač:** náhodně se zvolí počáteční hodnota čítače *i.* Postupně se hašuje *i, i+1.* Nesmí dojít k prozrazení počáteční hodnoty čítače.
* **Proudové šifry:** Jsou v zásadě PRNG, s jehož výstupem se provádí XOR s otevřeným textem.
* **Algoritmy založené na teorii čísel,** u kterých byl proveden důkaz bezpečnosti.

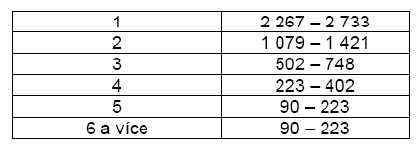
**Kryptograficky bezpečné PRNG**

* Použití v linuxových jádrech.
* Entropie získávána sběrem událostí z pohybu myši, klávesnice, HD a některých přerušení. Extrakce pomocí SHA-1.

**Testování:**

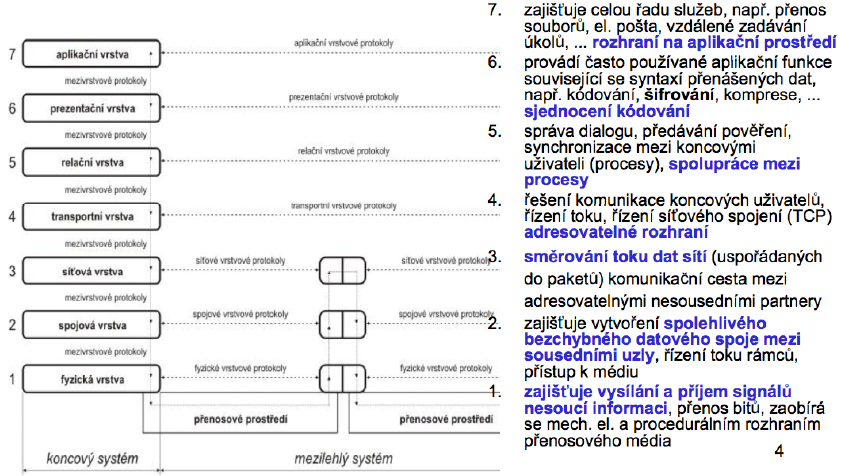
* **Teoretické testy**
  + Metodika návrhu
  + Zobrazení ve vícedimenzionálních prostorech
  + Zjištění entropie, kompresní techniky
* **Klasické testy**
  + Test Frekvence (Monobit test)
    - Test četnosti jedniček
    - Vyhovuje tomuto testu, pokud ve vygenerované posloupnosti **20 000 bitů** je počet jedniček v rozmezí od **9 654 do 10 346.**
  + Poker test
    - Posloupnost 20 000 náhodných bitů je neřetězovitě rozdělena na 5000 čtyřbitových úseků, které reprezentují hodnotu i=0…15. Počet úseků s hodnotou i si označme jako f(i). Podle následujícího vzorce pak vypočteme testovací hodnotu X.
    - 
  + Run Test
    - Termínem run se označuje úsek dané posloupnosti, který je složen se samých **nul** (pak se nazývá gap) nebo **jedniček** (v tom případě se nazývá blok)**.**

Například v posloupnosti 00101111100000001 je na začátku gap délky 2, poté je blok délky 1, následuje gap délky 1, blok délky 5, gap délky 7 atd.. Při testu spočítáme v dané posloupnosti počet gapů a počet bloků délky **1,2,3,4,5,6 a více**. Všech dvanáct vypočítaných čísel musí ležet v následujících intervalech.



# **10) Bezpečnostní architektura RM OSI - služby bezpečnosti, mechanizmy bezpečnosti, útoky na bezpečnost, příklady implementace bezpečnostních mechanizmů v jednotlivých vrstvách.**

Bezpečnostní mechanismy jsou zabudované do aplikačních programů a operačních systémů (7vrstva a 4.). Nebo do propojovacích zařízení (3.vrstva)



**Služby bezpečnosti =** Služba realizovaná protokolem příslušné vrstvy. **5 kategorii:**

* Autentizace: Proces ověřování identity uživatele.
  + Uživatelů = neeliminuje útoky replay attack.
  + Zdroje dat = Provádí autentizaci všech dat, eliminují útoky replay attack.
* Řízení přístupu: Možnost povolit/odepřít použití určitého zdroje subjektu.
  + Ochrana před neautorizovaným přístupem = implementace v OS nebo v aplikačním programu.
* Zabezpečení důvěrnosti dat = Ochrana obsahu dat, ochrana toku dat při přenosu (proti analýze).
  + Služby pro důvěrnosti přenosu zpráv
  + Služby pro důvěrnost spojení = ochrana důvěrnosti v rámci navázaného spojení.
  + Služby pro důvěrnost toku dat
  + Služby selektivní důvěrnosti = Ochrana pouze určených částí informace.
* Zabezpečení integrity dat = Zabezpečení proti neautorizované modifikaci.
  + Služby integrity přenosu zpráv (ochrana integrity všech přenášených zpráv)
  + Služba integrity spojení (ochrana přenosu v rámci určitého navázaného spojení)
  + Služby selektivní integrity spojení a selektivní integrity zpráv

**„Slabá“ integrita =** Pro objektivní útoky (modifikace zprávy šumem, náhodná změna pořadí paketů) – Aplikace kontrolních součtů, CRC, pořadová čísla paketů apod..

**„Silná“ integrita =** Subjektivní (úmyslné, aktivní útoky) – Podvržené zprávy, úmyslně pozměněné zprávy – Prostředky pro zajištění slabé integrity + kryptografické prostředky.

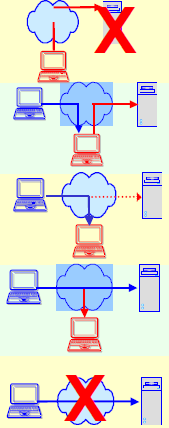
* Ochrana proti odmítnutí původu zprávy
  + Zajišťuje důkaz o původu dat
  + Prokázání původu
  + Prokázaní doručení

Autentizace = vím s kým komunikuji.

Nepopíratelnost = vím s kým komunikuji a lze mu to dokázat.

**Mechanismy bezpečnosti:**

* Šifrování
* Digitální podpis
* Řízení přístupu
* Integrita dat
* Výplň „Padding“
* Řízení směrování
* Ověření třetím subjektem

**Útoky na bezpečnost:**

* Destruction (Zničení dat)
* Corruption (Útok na integritu)

Neautorizovaná modifikace aktiv/dat

* Removal (útok na dostupnost)

Krádež, odebrání či ztráta informací nebo jiných zdrojů.

* Disclosure (útok na důvěrnost)

Neautorizovaný přístup k aktivům/datům.

* Interruption (útok na dostupnost)

Přerušení služeb – Spojení začne být nepoužitelné

* IPsec
  + V Síťové vrstvě, využívá správu klíčů
  + Zajišťuje důvěrnost (šifrování), integritu (autentičnost)
* Módy:
  + Transportní mód: zabezpečený paket má nezměněné záhlaví.
  + Tunelový mód: k původnímu paketu je připojeno nové záhlaví.
  + Transportní mód: se **používá pro přenos mezi koncovými zařízeními** s veřejnou IP adresou**.**
  + Tunelový mód: má použití pro bezpečný přenos paketů **mezi branami různých sítí.**
  + Integrita dat pomoci MAC
* SSL/TLS
  + **Mezi transportní a aplikační vrstvou**
  + Umožňuje autentizaci mezi serverem a klientem
  + Zajišťuje vytvoření šifrovaného spojení.
  + Komunikace typu end-to-end
  + Integrita toku zpráv je chráněna pořadovým číslováním

Autentizace serveru:

* + Ověření identity
  + Klient vyžaduje autentizaci serveru, zašle se certifikát klientovi, na kterém je, aby správně udělal autentizaci.
  + Ověření doby platnosti certifikátu, je vydávající CA důvěryhodnou?, Možné veřejným klíčem CA ověřit pravost podpisu vydavatele?

Odpovídá doménové jméno certifikátu skutečnému jménu serveru?

* HTTPS
  + Šifrování dat mezi klientem a webovým serverem na protokole http
  + Certifikáty typu X.509 k autentizaci serveru, případně klienta

**Příklady Implementace bezpečnostních mechanizmů v jednotlivých vrstvách.**

IPsec(L3), SSL/TLS (L4-L7), HTTPS(L7)