Ochrana informace II

Petr Šťastný, 5.6.2007
petr.stastny@centrum.cz

Autor neručí za správnost informací. Použití na vlastní nebezpečí. Tonda už většinou jenom tyhle otázky nepoužívá, vezme většinou 1 z tohodle seznamu a druhou si vymyslí (nějaká přehledová).

Obsah

1. anonymní platby (tj. elektronické platby, digital cash)

- problém kreditních karet spočívá v sledovatelnosti toku peněz – hledáme protokol pro tvorbu autentizovaných ale nesledovatelných zpráv
- Zákazník A připraví 100 anonymních příkazů k platbě na stejnou částku. Každý příkaz obsahuje 100 různých párů identifikačních řetězců, každý vzniklý z řetězce obsahujícího úplnou identifikaci zákazníka vhodným algoritmem pro secrets splitting. Např. každý pár může být tvořen párem paketů podobně jako v protokolu pro bit commitment tak, aby se dalo kontrolovat rozkrytí paketu. Známe-li obě poloviny, můžeme sestavit původní řetězec.
- A "zaslepí" všechny příkazy protokolem pro podpisy naslepo a odešle je do banky B.
- B požádá A o "odslepení" náhodně zvolených 99 příkazů a rozkrytí všech identifikačních řetězců
- Je-li vše v pořádku, banka B podpisem potvrdí zbylý příkaz a vrátí jej A.
- A "odslepí" potvrzený příkaz a předá jej obchodníkovi.
- Obchodník ověří podpis banky a tím legitimnost příkazu.
- Obchodník požádá A náhodně o rozkrytí jedné poloviny každého páru identifikačních řetězců
- A splní požadavek.
- B ověří svůj podpis a zjistí, zda již nepřijala příkaz se stejným Jednozn. řetězcem. Pokud je vše v pořádku, vyplatí B peníze a příkaz archivuje.
- Pokud příkaz se stejným Jednozn. řetězcem již banka přijala provede zkoumání rozkrytých identifikačních řetězců. Jsou-li v obou případech stejný, podvádí obchodník, jinak podvádí A.
- Protokol zajišťuje, že obchodník ani A nemohou podvádět. Musí však věřit bance, že s jejich účty nakládá korektně. Na druhou stranu pokud se oba chovají korektně, banka nemá k dispozici žádnou informaci o tom, jak nakládají se svými finančními prostředky.

2. asymetrický key management

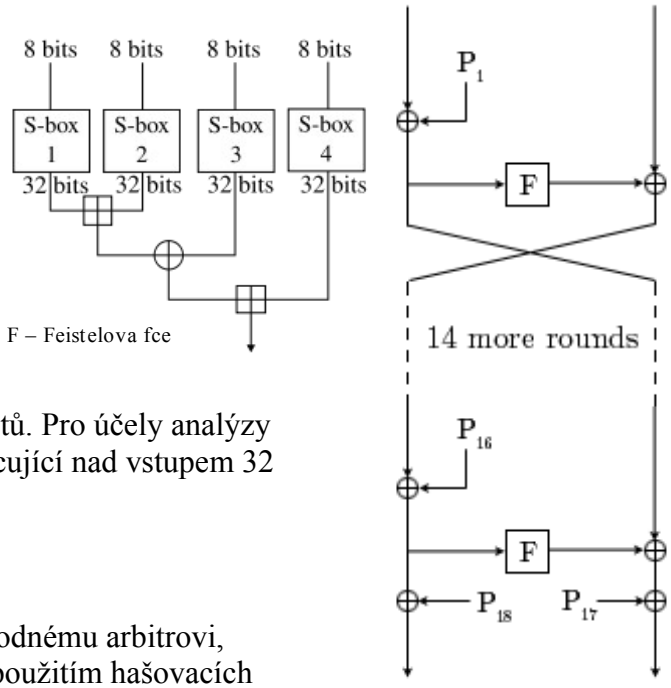
- Příklad použití:
    - Bob: Zprava + Alice_verejny_klic = Sifrovana_zprava
        - Alice: Sifrovana_zprava + Alice_privatni_klic = Zprava
    - Alice: Zprava + Alice_privatni_klic = Podepsana_zprava
        - Bob: Podepsana_zprava + Alice_verejny_klic = Zprava je opravdu od Alice
- rozlišujeme veřejné klíče ruční (přenášené v kabele) a ověřované na základě certifikátu
- **certifikát** = veřejný klíč + datum vytvoření + platnost do + vydavatel + podpis vydavatele
- certifikát se ověřuje klíčem nadřazení CA – strom důvěry
- certifikát je nutné získat osobně u certifikační autority
- norma X.500 – celý svět seřazen do stromu, na vrcholu jeden všeobecně známý certifikát
- Jak zajistit vzájemné důvěřování dvou autorit? Třeba jednoho ze systému X 500 a nějaké jiné autority?
  Nadklíč, který by ověřoval i X 500 i jinou certifikační autoritu
  Tzv. CROSS-CERTIFIKÁT, který stojí mezi těmito autoritami, a obě ho znají a důvěřují mu
  Prostředník (BRIDGE), entita, která je spojena jedním CROSS-CERTIFIKÁTEM s autoritou X-500 a jedním CROSS-CERTIFIKÁTEM s druhou autoritou.
  Osobní výměna klíčů fyzicky v kabele, nejjednodušší a nejpoužívanější.
- revokační listy (CRL) – Uživatelé nahlašují žádost o revokaci certifikátu přímo Administrátorovi CA CA vydává jednou za nějaký čas seznam zneplatněných certifikátů – nutno certifikáty ověřovat v CRL, problém se zpožděním (CA vystavuje a podepisuje list tak jednou deně). Od toho jsou OSCP, které odpovídají, co by bylo na CLV, ale není to podepsané.

3. blokové šifry

- jsou symetrické šifry které pracují se skupinami bitů stejné délky (ozn.jako bloky) s neměnnou transformací
- příklady: DES, Blowfish, IDEA, RC5, Rijndael
- Mody provozu:
- **ECB** (electronic code book) – pouze sifrování klicu, jeden blok za druhým nezávisle na sobě, ale stejná část dat je zašifrována stejně, lze poznat opakující se data, možnost změny pořadí a přidání/ubrání bloku
- **CBC** (cipher block chaining) - vhodný pro šifrování zpráv, všechny další bloky závisí na předchozích, ke vstupu se XORuje výstup z minulého bloku, ale výpadek jednoho bloku znamená ztrátu všech následujících dat
- **CFB** (cipher feed back) - pro šifrování podobné proudovým šifrám, na základě klíče se generuje stream, který se míchá se vstupním textem, porucha dat vůbec nevadí (přijdeme jen o jeden blok), žádná závislost mezi stejnými bloky
- **OFB** (output feed back) - pro aplikace, kdy je třeba eliminovat šíření přenosových chyb, vysokokapacitní spoje s velkou redundancí (řeč, video) – zpětná vazba pouze určité délky, při poruše bloku se lze po několika krocích z chyby vzpamatovat

4. Blowfish

- bloková (tzn.symetrická) šifra, zamýšlena jako náhrada za zastarávajíví DES, volně k použití, žádné patenty
- subklíče – přepočítávají se před každým šifrováním: P-pole,S-boxy
- založená na s-boxech a sofistikovaném key-schedule
- Feistelova šifra (16 iterací), délka bloku 64 bitů (dělí se na dva), proměnná délka klíče až 448 bitů
- Algoritmus provádí 16 cyklů nad vstupem délky 64 bitů. Pro účely analýzy navrženy jeho zmenšené varianty (MiniBlowfish) pracující nad vstupem 32 popřípadě 16 bitů.



5. časové známky

- nejjednodušší metodou je zasílat kopie zpráv důvěryhodnému arbitrovi, problémy s množstvím uchovávaných dat lze vyřešit použitím hašovacích funkcí
- **Spojované (linked) známky** (aby odesílatel (adresát) společně s arbitrem nemohli podvádět)
  - odesílatel S zašle arbitrovi A hashkód zprávy, A vrátí odesilateli podepsaná data, která obsahují: pořadí zprávy, čas podpisu, informace o předešlé zprávě
  - po vyřízení následující zprávy arbitr zašle odesílateli identifikaci následujícího odesílatele
  - chce-li někdo ověřit časovou známku zprávy, kontaktuje odesílatele předchozí a následující zprávy a pomocí nich ověří platnost známky
  - Pro zvýšení bezpečnosti je možné připojit informace o více předchozích zprávách a držet seznam stejného počtu následujících odesílatelů
- alternativa: pomocí generátoru náhodných čísel se určí skupina uživatelů, kterým se zašle hash zprávy a ti k ní připojí čas a podepíší, výsledky arbitr uschová; nelze podvádět, protože není možné předem vědět, kdo bude zprávu podepisovat, a okruh verifikátorů se pokaždé náhodně mění

6. certifikáty

- certifikát = veřejný klíč + datum vytvoření + platnost do + vydavatel + podpis vydavatele

- certifikát se ověřuje klíčem nadřazení CA – strom důvěry
- certifikát je nutné získat osobně u certifikační autority
- norma X.500 – celý svět seřazen do stromu, na vrcholu jeden všeobecně známý certifikát
- revokační listy (CRL) – CA vydává seznam zneplatněných certifikátů – problém se zpožděním, nutno certifikáty ověřovat v CRL

7. co je to klíčový materiál

- klíče, certifikáty, inicializační vektory ve fyzické nebo elektronické formě
- data používaná pro vytvoření a udržování "kryptografického vztahu" entit (pomocí klíčů, …)

8. co je to s-box (substitution box)

- Jeden ze základních stavebních prvků moderní blokové šifry. V zásadě je to booleovská funkce převádějící binární vektor na jiný binární vektor. Pro kryptografické účely jsou tyto funkce studovány a jsou formulovány takové jejich vlastnosti, které poskytují záruky pro bezpečnost výsledné šifry. Tyto vlastnosti jsou obvykle statistického charakteru (např. tzv. kriterium laviny, anglicky Strict Avalanche Criterion, spočívá v požadavku, že při změně jednoho bitu vstupu do S-boxu se s pravděpodobností jedna polovina změní jednotlivé bity výstupu S-boxu).
- S box se dá implementovat jako pevná tabulka, kde se pro vstup najde výstup (DES), nebo se dá dynamicky počítat z klíče (Blowfish)

9. co jsou to inicializační vektory

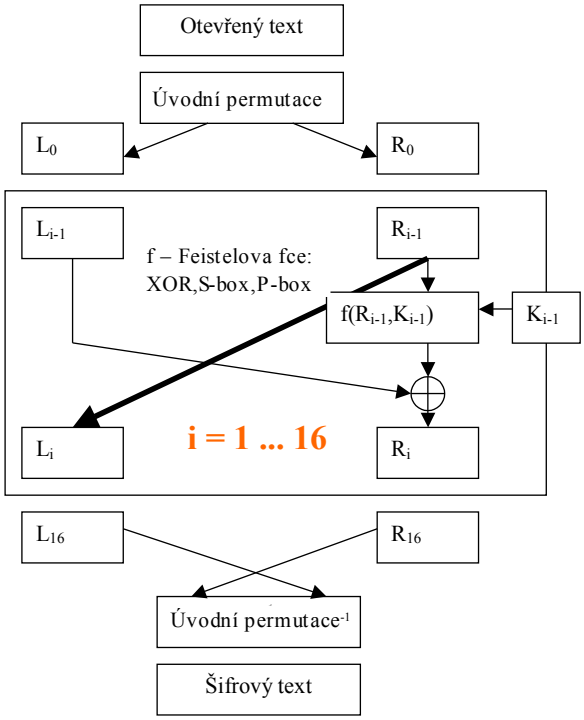
- řeší problém situace, kdy při šifrování stejného vstupního textu stejným klíčem a při použití stejného algoritmu vznikne stejná zašifrovaná zpráva (což může být někdy problém, např. útočník pozná, že byla poslána stejná věc)
- inicializační vektor je blok bitů, většinou stejně dlouhý jako zpráva nebo jako klíč (pak se před začátkem šifrování např. XORuje na zprávu nebo na klíč)
- druhá strana musí samozřejmě tento IV znát
- možnosti vzniku IV – náhodné vygenerování a jeho výměna během navazování spojení, výpočtem, měřením nějaké veličiny, ...

10. co jsou to slabé (weak) klíče

- takový klíč, který způsobí nechtěné chování šifrovacího algoritmu a odhalí něco co neměl
- např. takový klíč, při kterém se plaintext rovná zašifrované zprávě - jinak pojmenováno jako „pevný bod“ - nalezení pevných bodů šifry může být základem útoku
- účelem při tvorbě šifer je slabým klíčům se úplně vyhnout
- někdy může být malý počet slabých klíčů akceptovatelný, pokud jsou identifikovatelné a identifikované (jako např. u DES)
- také můžeme znát počet slabých klíčů (i když ne je samotné) a tím omezit odshora pravděpodobnost slabého klíče při generování
- Př - DES. Zde se 56bitový klíč rozdělí na 16 podklíčů - slabý klíč je ten, který generuje 16 identických klíčů - to nastává při 00000000000000, FFFFFFFFFFFFFF a 0000000FFFFFFF a FFFFFFF0000000

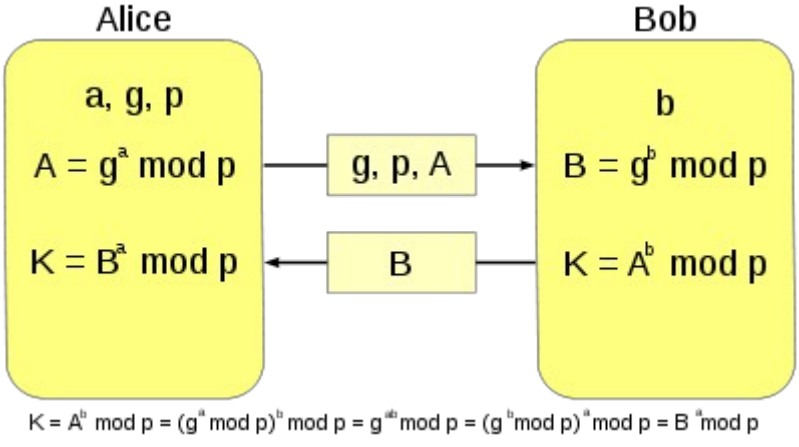
11. DES

- bloková(tzn.symetrická) šifra
- vyvinula IBM na zakázku NSA počátkem 70. let, původní název DEA, v USA DEA1. dodnes používán v komerční sféře, pro vojenské účely není certifikován ani pro ochranu neklasifikovaných informací, současnosti je tato šifra považována za nespolehlivou, patrně nejrozsáhleji používaný šifrovací algoritmus všech dob, norma ANSI X3.92
- používá Feistelovu síť (16 cyklů)
- používá se vstupní a výstupní permutace - úvodní permutace nemá prakticky žádný vliv
- šifruje 64-bitové bloky otevřeného textu na 64-bitové výstupní bloky, délka klíče 64 bitů
- na začátku se blok rozdělí na dva 32bitové
- kódování a dekódování je prakticky identické (pouze se prohodí podklíče (32 bitové půlky 64 bitového klíče)
- příliš krátký klíč, navíc efektivně pouze 56-bitový, existence slabých a poloslabých klíčů
- nevhodný návrh S-boxů
- těžko proveditelná implementace v SW (nevhodný výběr operací)
- je třeba zvýšit odolnost vůči masivně paralelnímu útoku (drahá key schedule)



12. DH-kryptosystém

- Diffie-Hellman výměna klíčů je kryptografický protokol, který umožňuje navázat bezpečné spojení. Pro bezpečné spojení je potřeba si vyměnit klíč k symetrické šifře přes ještě nezabezpečený kanál. Právě tento protokol to umožňuje aniž by byl klíč jednoduše poslán v otevřené formě.
- Alice si vymyslí a,g,p (a stejně bitů jako p) vypočte A pošle Bobovi [g,p a A], Bob vypočte B a pošle ho Alici oba si vypočítají K => muzou zacit symetricky sifrovanou komunikaci



- Puvodne nezabezpecoval autentifikaci ucastniku = nachylny k utoku man-in-the-middle. Man-in-the-middle muze vytvorit komunikaci s dvema ruznymi Diffie-Hellman klici, jeden s Alici a druhej s Bobem, a pak se tvarit jako Alice k Bobovi a obracene, treba pomoci dekodovani a rekodovani zprav mezi nimi. Nejaka metoda autentifikace mezi temito osobami je nutna.
- Problému nalezení čísla a ze znalosti $g^a \bmod p$ se říká problém diskrétního logaritmu. Tento problém je stále považován za velmi obtížný.

13. digitální podpis

- musí být nefalšovatelné, autentické, neměnitelné, "nerecyklovatelné"
- symetrické systémy – výměna zpráv přes arbitra
  - v případě, že nepožadujeme utajení přenášených dat, vystačíme s použitím MAC kodu namísto šifrování
  - na obranu proti opakovanému použití došlé zprávy lze použít vhodnou časovou známku, proti sestavování nových zpráv z částí dříve došlých poslouží časová známka v každém šifrovaném bloku.
- asymetrické systémy – podpis privátním klíčem (většinou se podepisuje hash zprávy), ověřené veřejným klíčem
  - v praxi je ovšem třeba zajistit fixaci aktu podpisu v čase, aby bylo možno zpětně ověřit platnost podpisu vzhledem k platnosti certifikátu příslušného veřejného klíče - To se řeší časovými razítky: uvnitř podepisovaných dat (nejdříve), vně podpisu (nejpozději)

14. digitální volby

- požadavky:
  - volit smí pouze oprávnění voliči
  - každý smí hlasovat nejvýše jednou
  - nikdo nesmí vědět, kdo jak volil
  - nikdo nesmí měnit volbu jiných
  - každý hlas musí být započítán
  - je zjistitelné, kdo volil
- Protokol se dvěma centrálními autoritami
  - používá registrační autoritu RA provádějící registraci voličů a sčítací autority SA, která sčítá hlasovací lístky a zveřejňuje výsledky voleb
  - všichni voliči zašlou RA žádost o validační číslo
  - RA zašle každému voliči náhodně zvolené validační číslo L, zároveň si ponechá seznam, kdo jaké číslo dostal
  - RA zašle seznam validačních čísel SA
  - každý z voličů si náhodně vybere svoje identifikační číslo Id a SA zašle zprávu (L, Id, v), kde v je volba
  - SA porovná L se seznamem validačních čísel z kroku 3. odpovídající číslo škrtne a voličovo Id přidá do seznamu asociovaného s voleným kandidátem
  - po skončení voleb SA zveřejní výsledky voleb a seznamy identifikačních čísel spojené se jmény kandidátů
  - Každý z voličů si může ověřit zda byl jeho hlas správně započítán, RA zjistí případné falešné hlasy.
  - SA však může registrovat neoprávněné voliče, případně některé voliče vícekrát. V případě že budou RA a SA spolupracovat, hrozí nebezpečí porušení anonymity voličů.
- Protokol bez centrální autority
  - všichni voliči podepisují vzájemně své hlasy veřejnými a privátními klíči, přidávají se náhodné řetězce, všichni voliči si ověří svou volbu
  - Každý může sám spočítat výsledky voleb
  - Protokol zabraňuje tomu, aby libovolná ze zúčastněných stran neoprávněně manipulovala s rozhodnutími ostatních voličů, rovněž anonymita všech voličů je zajištěna.
  - Slabinou protokolu je značná komplikovanost, všichni voliči musí spolupracovat. Navíc poslední volič má výsledky voleb k dispozici dřív než ostatní.

15. důkazy s nulovými znalostmi

- je to způsob, jak jeden uživatel druhému dokáže, že zná privátní klíč k danému veřejnému, aniž by druhý uživatel dostal po ověření sebemenší informaci, která by mu mohla pomoci k vypočítání tohoto privátního klíče
- dokazovatel (prover) nesmí podvádět, pokud důkaz nezná, jeho šance přesvědčit ověřovatele je mizivá
- ověřovatel rovněž nesmí podvádět, o samotném důkazu smí zjistit pouze to, že jej dokazovatel zná, zvláště nesmí být schopen celý důkaz rekonstruovat a sám provést
- ověřovatel se nesmí dozvědět nic, co by nebyl schopen zjistit bez pomoci dokazovatele
- není-li splněna poslední podmínka, mluvíme o důkazech s minimálním vyzrazením (minimum-disclosure proofs)
- jeden z možných důkazů založen na problematice Hamiltonovských kružnic v grafu – při izomorfismu se tyto kružnice zachovávají:
    - 1. Nechť A zná Hamiltonovskou kružnici v grafu G
    - 2. A provede náhodnou permutaci G. Původní graf a vzniklý H jsou izomorfní.
    - 3. Kopie grafu H je zaslán entitě B
    - 4. Oveřovatel B položí dokazovateli A jednu z následujících otázek:
        - a) dokázat, že G a H jsou izomorfní
        - b) ukázat Hamiltonovskou kružnici v grafu H
    - 5. opakováním kroků 1. až 4. lze docílit potřebné jistoty

16. efektivita šifrovacího algoritmu

- rychlost symetrických vs. asymetrických algoritmů – sym. jsou obecne o 3-4 řády rychlejší než asymetrické
- měřeno počtem instrukcí na zašifrovaný byte
- Symmetric-key algorithms are generally much less computationally intensive than asymmetric key algorithms. In practice, this means that a quality asymmetric key algorithm is hundreds or thousands of times slower than a quality symmetric key algorithm.

17. el. podepisovaná pošta

- **Podepisování kontraktů (Contract signing)**
    - V každém okamžiku musí být obě smluvní strany vázány stejně moc
    - nejjednodušším řešením je arbitrovaný protokol, kde obě strany předají centrální autoritě své podepsané kopie a tato třetí strana zajistí výměnu po obdržení obou kopií
    - daleko lepší je následující distribuovaný protokol:
        - A i B náhodně vygenerují 200 konvenčních klíčů, které rozdělí do dvouprvkových množin
        - A i B vytvoří 100 párů zpráv Ln a Rn, zhruba ve tvaru “Toto je levá část n-tého podpisu smlouvy ”…, každá ze zpráv navíc obsahuje polovinu elektronického popisu smlouvy, timestamp atd. Kontrakt považujeme za podepsaný druhou stranou, pokud se můžeme prokázat oběma polovinami některého z podpisů.
        - A i B zašifrují i-tý pár zpráv i-tým párem klíčů (“levou” zprávu jedním, “pravou” druhým z klíčů)
        - obě strany si navzájem zašlou páry zašifrovaných zpráv (200 zpráv každý)
        - použitím protokolu pro Oblivious transfer si A a B navzájem zašlou všechny šifrovací klíče - druhá strana má tedy z každého páru jeden (který ?) klíč
        - A i B provedou dešifrování těch zpráv, ke kterým mají k dispozici odpovídající klíč
        - A zašle B první bit všech 200 konvenčních klíčů, obdobně B

        - předchozí krok opakujeme dokud nejsou přeneseny všechny bity klíčů
        - obě smluvní strany mohou dešifrovat všechny zprávy -> kontrakt je podepsán
    - Pokud by se některá ze stran pokusila o podvod v kroku 4. nebo 5., bude podvod odhalen v kroku 6. Podvod v kroku 7. bude s velkou pravděpodobností objeven okamžitě. Ukončí-li jeden z účastníků protokol před zasláním všech bitů klíčů, mají oba stejnou šanci dopočítat některý z klíčů a získat elektronický podpis druhého.
    - Problémem je, má-li někdo z podepisujících výrazně větší výpočetní kapacitu, v protokolu není zlom, ve kterém by se výrazně změnila míra vázanosti účastníků.
- **Elektronická potvrzovaná pošta (digital certified mail)**
    - chceme, aby adresát mohl přečíst naši zprávu až poté, co získáme potvrzení o tom, že ji obdržel (elektronický doporučený dopis)
    - protokol:
        - A zašifruje posílanou zprávu náhodně zvoleným konvenčním klíčem K
        - A vytvoří 100 párů konvenčních klíčů - první klíč každého páru je generován náhodně, druhý je XOR prvního klíče a klíče K
        - A zašifruje pomocnou zprávu každým ze 100 párů těchto klíčů (200 šifer)
        - všechny výsledné páry šifer zašle B
        - B vygeneruje 100 párů náhodných konvenčních klíčů
        - B vytvoří 100 párů zpráv tvaru "Toto je levá část mého potvrzení číslo n". Opět potvrzení o přjetí je platné, pokud se protější strana může prokázat oběma polovinami jednoho z exemplářů potvrzení.
        - B zašifruje i-tý pár zpráv i-tým párem klíčů ("levou" zprávu jedním, "pravou" druhým z klíčů)
        - výsledné páry šifer zašle B protější straně
        - s použitím protokolu pro Oblivious transfer si A i B navzájem pošlou všech 100 párů svých klíčů - žádná ze stran neví, který klíč z kterého páru protějšek má k dispozici
        - A i B dešifrují všechny zprávy, ke kterým mají klíče a ověří jejich smysluplnost
        - obě strany si zašlou první bit všech 200 svých šifrovacích klíčů
        - předchozí krok je opakován dokud nejsou přeneseny všechny bity
        - A i B dešifrují zbývající části párů zpráv, které v předchozích krocích obdrželi - A má k dispozici potvrzení o přijetí zprávy od B a B může provést XOR libovolného páru klíčů a dešifrovat zprávu.
    - Použití pomocné zprávy dává straně B možnost odhalení podvodu v kroku 10.
    - V ostatních ohledech má protokol obdobné vlastnosti jako protokol pro podepisování kontraktů.

18. el. podpis symetrický

- výměna zpráv přes arbitra, který ověřuje totožnost odesílatele (na základě sdíleného tajemství) a ručí příjemci
- v případě, že nepožadujeme utajení přenášených dat, vystačíme s použitím MAC namísto šifrování
- na obranu proti opakovanému použití došlé zprávy lze použít vhodnou časovou známku, proti sestavování nových zpráv z částí dříve došlých poslouží časová známka v každém šifrovaném bloku

19. ElGamal kryptosystém

- asymetrický, založen na obtížnosti výpočtu diskrétního logaritmu nad okruhem
- kryptosystém je považován za bezpečný a jednoduchy
- nevýhodou je nutnost generování náhodných čísel k a zdvojnásobení objemu dat při šifrování, je relativně pomalý
- sklada se z 3 casti: generator klicu, sifrovaci a desifrovaci algoritmu

- ElGamal is malleable in an extreme way: for example, given an encryption (c1,c2) of some (possibly unknown) message m, one can easily construct an encryption (c_1, 2 * c_2) of the message 2m. Therefore ElGamal is not secure under chosen ciphertext attack.

20. FISH

- proudová (tzn.symetrická) šifra založená na Fibonacciho generátoru pseudonáhodných čísel
- koncept vypouštěcích generátorů
- Šifrování se provádí např. xorováním výsledné posloupnosti s otevřeným textem
- Algoritmus byl publikován koncem roku 1993, nebyl nikdy šířeji používán, není známo, že by existoval efektivní útok.
- FISH je velice rychlý, ale obrovská délka klíče

21. hašování

- použití:
  - symetrické systémy - slouží k rozpoznání pravosti dešifrované zprávy
  - asymetrické systémy - rychlejší autentizace: spočítá se hašovací funkce nad danou zprávou a elektronicky se podepíše až výsledný hashkód
  - ochrana hesel a passphrases
- **MAC** kódy – hašování závislé na klíči
- **MDC** kódy – není závislé na klíči
  - **OWHF (one-way hash function)** – odolné proti nalezení vzoru či vzoru se stejným výsledkem na základě znalosti hašovací funkce a dvojic vzor-výsledek
  - **CRHF (collision resistant hash function)** – navíc odolné proti nalezení dvou vzorů se stejným výsledkem
- typy:
  - Hašovací funkce založené na blokových šifrách (MDC-2, MDC-4)
  - Hašovací funkce založené na modulární aritmetice
  - Hašovací funkce založené na problému batohu
  - Speciální MDC funkce – algoritmy speciálně navrhované od počátku přímo pro hashování (MD4, MD5)

22. hoďte korunou (oblivious transfer)

- protokol umožňuje, aby si adresát vybral z několika nabízených možností aniž by odesílatel předem znal jeho volbu, možné doplnění o následnou vzájemnou kontrolu
- A vygeneruje dva páry public-key klíčů, oba veřejné klíče zašle B
- B vytvoří klíč K pro symetrický algoritmus, tento klíč zašifruje jedním z přijatých klíčů a výsledek vrátí A
- A dešifruje přijatou zprávu oběma tajnými klíči, čímž získá K1 a K2
- Klíčem K1 zašifruje A jednu z posílaných zpráv, klíčem K2 druhou a oba výsledky zašle B.
- B se pokusí dešifrovat přijaté zprávy, přičemž v jednom případě získá smysluplný výsledek
- Případné ověření se provede tak, že A zveřejní své tajné klíče.
- Protokol sám o sobě lze používat k distribuci šifrovacích klíčů, případně jako obdobu házení korunou. Větší význam má jako součást následujícího protokolu.

23. HW generátory (jake znate, jak se pouzivaji,k cemu se pouzivaji)

- odvíjejí svoji činnost od sledování nějakého fyzikálního jevu, který je ve své podstatě nepredikovatelný, zhusta dosti sofistikovaně „nevhodným“ způsobem
  - měření záření kousku radioaktivního materiálu
  - termální šum polovodiče
  - měření stavu volně běžícího oscilátoru
  - vnitřní šum zesilovače
  - samovolné vybíjení kondenzátoru

24. SW generátory

- jsou založeny na pozorování jevů v počítači, které jsou z hlediska programu náhodné:
  - doba odezvy diskového systému
  - čas mezi stisky kláves uživatelem
  - pohyb myši
  - systémový časovač
  - stav vnitřních tabulek OS
  - statistiky síťové komunikace

25. hybridní šifrování

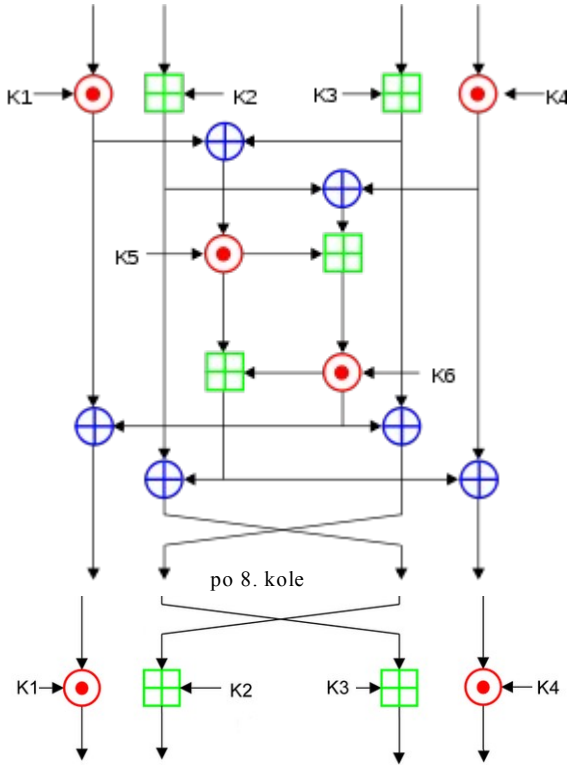
- Asymetrické šifrování má jednu velkou nevýhodu. Je velmi náročné na matematické operace, tedy i na výkon počítače. V praxi se proto používá kombinace symetrického a asymetrického šifrování. Tomuto způsobu se říká hybridní šifrování. Využijeme výhod obou: rychlost symetrického šifrování a „použitelnost" asymetrického šifrování.
- Pokud chtějí dva počítače komunikovat přes otevřenou síť, kde je každý může „odposlechnout“, vytvoří relaci. Na začátku relace vygeneruje jeden z nich klíč, zašifruje ho veřejným klíčem druhého počítače a pošle mu ho. Druhý počítač si klíč dešifruje, takže teď už mají oba dva stejný klíč, který kromě nich nikdo jiný nezná. Můžou už tedy používat symetrické šifrování, protože to je daleko rychlejší. Každá jejich další komunikace, která probíhá během relace, je symetricky zašifrovaná.

26. chosen plaintext/ciphertext attack

- Zvolený otevřený text (choosen plaintext att.) - útočník může získat k libovolnému otevřenému textu odpovídající šifru, velmi používaný útok, vhodný i proti statistickým databázím
- Zvolená šifra (choosen ciphertext att.) - používá se v případě, že útočník může šifrovacím algoritmem zašifrovat velké množství zpráv, aby našel plaintext odpovídající zvolené šifře
- Znalost zašifrovaného textu (ciphertext only att.)
  - útočník má k dispozici pouze zachycený zašifrovaný text, dále může využívat apriorních informací - pracuje tedy jen se statistickými rozbory, distribucí, pravděpodobností
  - Systém, který není odolný vůči tomuto útoku nelze označit za bezpečný
- Znalost otevřeného textu (known plaintext att.) - předpokládá se, že útočník má k dispozici pár otevřený text + odpovídající šifra
- Pravděpodobný text (probably plaintext att.) - útočník na základě okolností odeslání zprávy může učinit částečný odhad obsahu zprávy
- Digitální podpisy
  - key-only attack – zná pouze veřejný klíč oběti
  - known signature attack – dtto + má k dispozici pár (zpráva-podpis)

- chosen signature attack – útočník si může vybrat zprávy, které si nechá podepsat

27. IDEA



- symetrická bloková šifra, publikován v roce 1991 pod názvem IPES
- současný název akronymem za International Data Encryption Algorithm
- bloková šifra s délkou bloku 64 bitů, pracující s klíčem o délce 128 bitů
- algoritmus je patentován, nelze volně používat
- Blok otevřeného textu je rozdělen na čtyři části, každá o délce šestnáct bitů. Poté je provedeno osm kol šifrovacího procesu.
- celkem 52 subklíčů – generovány rozdělením klíče a prováděním rotací

  Tvorba subklíčů:

  1. klíč rozdělen na osm částí - vznikne prvních osm subklíčů
  2. je provedena rotace klíče vlevo o 25 bitů
  3. vzniklý řetězec je opět rozdělen na osm částí - subklíčů
  4. Opakováním 2 a 3 získáme potřebné množství subklíčů.
- dešifrování s použitím podobných subklíčů (jsou v trochu jinm tvaru)
- IDEA může být používána v libovolném pracovním módu pro blokové šifry, zejména v módech ECB, CBC, OFB, CFB
- lze použít trojnásobné šifrování EDE Triple-IDEA se dvěma 128-bitovými klíči, použít 52 nezávislých řetězců
- Je zajímavé, že pokud bychom algoritmus upravili tím způsobem, že zvětšíme délku všech řetězců, se kterými pracuje na dvojnásobek, dojde ke ztrátě bezpečnosti.
- Algoritmus je považován za bezpečný.

28. integrita zpráv

•Checksum nestaci, protoze neoveris jestli nebyla modifikovana. Resp. tu checksum utocnik lehko spocita sam. Hash s klicem ne. Potrebujes tim padem nejakou hashovaci funkci a potom zalezi na tom, jestli chces, aby nepritel mohl cist tvou zpravu nebo ne.

•**jak to udelat jednoduse:** No, vymyslel jsem tam DES v CBC modu s klicem jako uvodni hodnotou. To pak posles spolu se zpravou.

•**jak to udelat robustne:** V tom slozitym jsem to jeste nejak zasifroval AES(Rijandel)em a pridal tam timestampy nebo pocitani zprav.

29. autenticita zpráv (TODO: Napsat cely lip)

- Autenticita: • identifikace entit • autenticita obsahu zprávy • autenticita původu zprávy
- A cryptographic message authentication code (MAC) is a short piece of information used to authenticate a message. A MAC algorithm accepts as input a secret key and an arbitrary-length message to be authenticated, and outputs a MAC (sometimes known as a tag). The MAC value protects both a message's integrity as well as its authenticity, by allowing verifiers (who also possess the secret key) to detect any changes to the message content.
- MACs differ from digital signatures, as MAC values are both generated and verified using the same secret key. This implies that the sender and receiver of a message must agree on keys before initiating communications, as is the case with symmetric encryption. For the same reason, MACs do not provide the

property of non-repudiation offered by signatures: any user who can verify a MAC is also capable of generating MACs for other messages. In contrast, a digital signature is generated using the private key of a key pair, which is asymmetric encryption. Since this private key is only accessible to its holder, a digital signature proves that a document was signed by none other than that holder.

30. jak poznat dobrou šifru

- Množství práce vynaložené na šifrování a dešifrování by mělo být úměrné požadovanému stupni utajení.
- Šifrovací algoritmus by neměl obsahovat zbytečná omezení.
- Implementace algoritmu by měla být co nejjednodušší.
- Chyby při šifrování by se neměly příliš šířit a ovlivňovat následující komunikaci.
- Zprávy by se zašifrováním neměly zvětšovat.
- Security through obscurity NEFUNGUJE!
- Zmatení (confusion) - nelze predikovat, jakou změnu zašifrovaného textu vyvolá byť jen malá změna otevřeného textu, složitá funkční závislost mezi zašifrovaným textem a párem klíč - otevřený text.
- Difuze (diffusion) - změna otevřeného textu se promítá do mnoha míst zašifrovaného textu
- Bezpečný systém - nelze získat otevřený text na základě znalosti odpovídající šifry

31. jak poznat dobrý generátor

- **Monobit test** - zda počet „1“ a „0“ je přibližně ‚správný’
- **Serial test** - zda počet výskytů „00“, „01“, „10“ a „11“ je přibližně stejný, jednotlivé podřetězce se mohou překrývat.
- **Poker test** - zkoumá, zda posloupnost obsahuje zhruba stejný počet všech možných podposloupností délky m
- **Runs test** - zda vstupní posloupnost obsahuje správný počet běhů, tj. posloupností samých nul (díry) a jedniček (bloky)
- **FIPS140-1 test náhodnosti** - vstupem testu je výstupní posloupnost generátoru o délce 20 000 bitů, která musí splnit všechna čtyři výše uvedená kriteria
- **Autocorellation test** - cílem je srovnání korelace mezi vstupní posloupností s a její posunutou verzí

32. jak volit asymetrické klíče

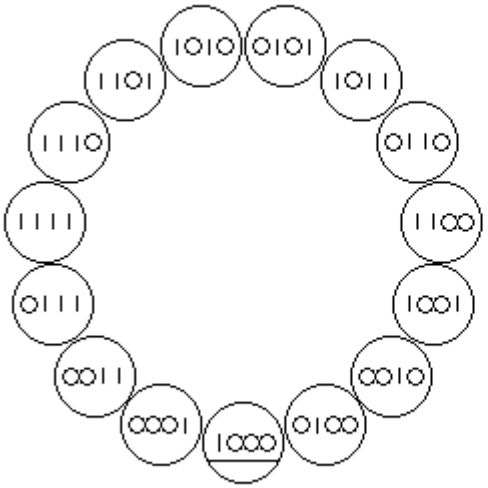
- A key length of 80 bits is generally considered the minimum for strong security with symmetric encryption algorithms. 128-bit keys are commonly used and considered very strong. See the key size article for a fuller discussion.
- The keys used in public key cryptography have some mathematical structure. For example, public keys used in the RSA system are the product(součin) of two prime numbers. Thus public key systems require longer key lengths than symmetric systems for an equivalent level of security. 3072 bits is the suggested key length for systems based on factoring and integer discrete logarithms which aim to have security equivalent to a 128 bit symmetric cipher
- In a secure asymmetric key encryption scheme, the decryption key should not be deducible from the encryption key

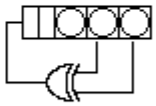
33. k čemu je dobré LFSR (Linear Feedback Shift Register)

- čítač založený na principu posuvného registru s lineární zpětnou vazbou
- používá se mimo jiné jako generátor pseudonáhodných posloupností bitů
- sestávají z posuvného registru a vypouštěcí sekvence (tap sequence), což je polynom stupně max. délky posuvného registru
- v každém kroku je obsah registru posunut o bit doprava, výstupem je nejpravější bit, registr se zleva doplní o

XOR těch bitů v registru, které odpovídají koeficientům vypouštěcího polynomu

- např.:
  - **A5/1, A5/2** (GSM)
  - **Střídající stop-and-go generátor** - používá tři posuvné registry s lineární zpětnou vazbou, impuls hodin je v závislosti na hodnotě LFSR-1 přiveden na LFSR-2 nebo LFSR-3, výstup těchto registrů je XORován, čímž vzniká výsledná hodnota. Generátor má velmi dlouhou periodu.
  - 4-bit Fibonacci LFSR





34. klíče pro konference

- schopnost vytvořit společný sdílený klíč pro libovolnou skupinu komunikujících – konferenci
- umožňuje ustanovit sdílený tajný klíč bez předchozí komunikace jednotlivých členů zamýšlené konference, autentizaci informací, předávaných v rámci procesu ustanovování klíče. Opět je nezbytná existence důvěryhodného centra.
- založeno na generování prvočísel na straně důvěryhodného centra, způsob velmi podobný jako u RSA
- Útok na jednotlivé části protokolu by měl být ekvivalentní s vyřešením alespoň jednoho ze dvou známých obtížných problémů - výpočet prvočíselného rozkladu, nebo výpočet logaritmu nad Galoisovým polem.

35. kongruenční generátory

- generátor pseudonáhodných sekvencí
- generování čísel pomocí parametrizované lineární rekurzivní funkce, variantou mohou být: kvadratický, kubický (využívají více parametrů a počítají mocniny z výsledků předchozího kroku)
- klasický vzorec: $X_{n+1} = (a X_n + c) \bmod m$
- kongruenční generátory jsou velmi rychlé, avšak predikovatelné, pokud útočník získá $X_i,..,X_{i+3}$, může dopočítat a, c a m.
- Používají se např. v *rand()* funkcích kompilátorů (gcc,MS Visual C++ atd)

36. konstrukce hašovacích funkcí

- nejčastěji používanou metodou je počítání DES nebo podobného algoritmu v modu CBC, resp CFB, konkrétní schemata se liší volbou případně kombinací zmíněných módů, použitím různých zarovnání
- **Stream MAC** – potřebujeme kryptograficky bezpečný generátor pseudonáhodných sekvencí, podle výsledku generátoru je vstupní bit přesunut do prvního nebo druhého posuvného registru se zpětnou vazbou, výsledek určen konečným obsahem posuvných registrů
- **HMAC** – vlastně obecný návod jako zkonstruovat MAC na základě jakékoliv hashovací (MDC) funkce –

pomocí MDC funkce se hašuje zpráva ve vhodné kombinaci s klíčem

37. Merkle-Helmann kryptosystém

- asymetrický, založen na problému batohu, přenášená zpráva je chápána jako vektor řešení, přenášena je výsledná suma - "hmotnost batohu"
- jednosměrný, veřejný klíč pouze pro zašifrování a privátní jenom na dešifrování – nejde použít na elektronické podepisování
- prolomen polynomialnim alg., M-H systém tedy není vhodný k ochraně důležitých informací.

38. nepopiratelnost

- definice: Cílem služby nepopiratelnosti je vytvářet, shromažďovat, udržovat, zajistit dostupnost a ověřovat důkazy týkající se údajné události nebo činnosti, aby bylo možné řešit spory o tom, zda se událost nebo činnost vyskytla či nikoliv.
- pozor na změnu pohledu: Útočníkem je zde často sám majitel privátního klíče!
- asymetrické šifrování je schopné nepopíratelnst zajistit, symetrické ne

39. operace asymetrických šifer

- Klasický příklad použití:
    - Bob: Zprava + Alice_verejny_klic = Sifrovana_zprava
        - Alice: Sifrovana_zprava + Alice_privatni_klic = Zprava
    - Alice: Zprava + Alice_privatni_klic = Podepsana_zprava
        - Bob: Podepsana_zprava + Alice_verejny_klic = Zprava je opravdu od Alice
- Správně by privatni_klic neměl být odvoditelný z verejneho. Nicméně každý algoritmus s veřejným klíčem se dá časem prolomit brute-force útokem, jde o to aby ten čas byl co nejdelší.

40. operace symetrických šifer (režimy)

- režimy činnosti blokových šifer anebo jak přejít od šifrování jednotlivých bloků k celým datům
- **ECB** (electronic code book) – pouze sifrování klicu, jeden blok za druhým nezávisle na sobě, ale stejná část dat je zašifrována stejně, lze poznat opakující se data, možnost změny pořadí a přidání/ubrání bloku
- **CBC** (cipher block chaining) - vhodný pro šifrování zpráv, všechny další bloky závisí na předchozích, ke vstupu se XORuje výstup z minulého bloku, ale výpadek jednoho bloku znamená ztrátu všech následujících dat
- **CFB** (cipher feed back) - pro šifrování podobné proudovým šifrám, na základě klíče se generuje stream, který se míchá se vstupním textem, porucha dat vůbec nevadí (přijdeme jen o jeden blok), žádná závislost mezi stejnými bloky
- **OFB** (output feed back) - pro aplikace, kdy je třeba eliminovat šíření přenosových chyb, vysokokapacitní spoje s velkou redundancí (řeč, video) – zpětná vazba pouze určité délky, při poruše bloku se lze po několika krocích z chyby vzpamatovat

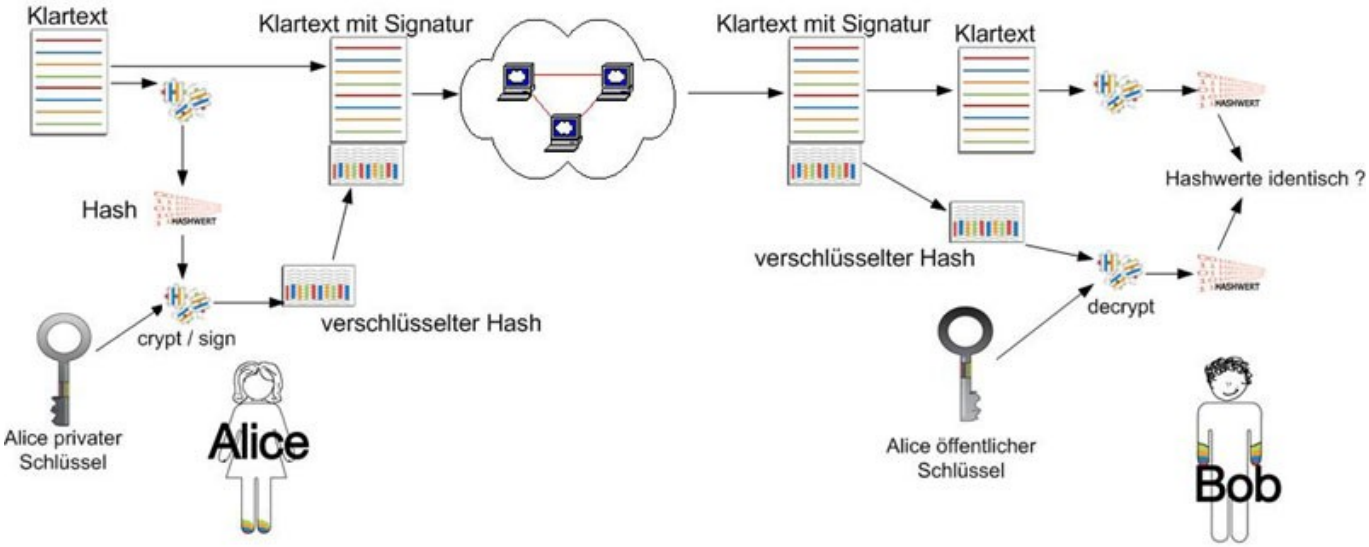
41. perfektní šifra

- složitost analýzy polyalfabetických šifer roste s počtem použitých substitucí. -> co třeba použít každou substituci jen jednou
- **one-time pad**
    - Předpokládáme že máme k dispozici klíče v celkové délce větší, než přenášená zpráva.
    - Každý tento klíč použijeme jen jednou, t.j. k zašifrování tolika znaků, jakou má délku. Šifrovat můžeme např. pomocí Vigenêrovy tabulky.

- K dešifrování je třeba mít stejnou sadu klíčů.
- Dlouhé sekvence náhodných čísel - mohou být použity namísto klíčů pro one-time pad systémy
- Sekvence textů z knih
  - mohou být použity namísto náhodných čísel.
  - otevřený text a takto vytvořený klíč mají charakteristické rozložení pravděpodobnosti výskytu znaků.
  - až 25% znaků šifrovaného textu vzniká kombinací několika nejčastějších symbolů -> lze rozkrýt část zprávy, což nám umožní efektivně odhadnout zbytek.

42. podepisovací schémata

- „obracene“ asymetricke sifrovani
- s příponou: **Alice** zprávu zahashuje a hash zasifruje svým privatnim klicem, ziska podpis, preida podpis ke sprave jako priponu (prilohu?) a odesle ; **Bob** ze zprávy oddeli podpis, zahashuje zpravu a hash porovna s rozsifrovanym (pomoci verejneho klice Alice) podpisem v pripone pokud jsou stejne zprava je autenticka



  - př: DSA, ElGamal, Merkle, Chaum-van Antverpen
- s obnovou zprávy: **Alice** doplni redundanci (aby nam nevznikl ze dvou ruznych zprav jeden stejny podpis, je to tam místo hashe, deplneni redundance musí byt invertibilní) a zašifruje zpravu svým privatnim klicem pošle zašifrovanou zpravu ; **Bob** zpravu rozsifruje verejnym klicem Alice, odstraní redundanci a tim zrekonstruuje zpravu
  - př: RSA, Rabin
- RSA popis– deterministické pod.schéma s obnovou zprávy, obtížnost faktorizace celých čísel
  - bezpečnost: pokud je znám podpis dvou zpráv, muzu bez znalosti klíče sestavit podpis 3. zprávy (je jejich součinem), pokud by funkce pro přidání redundance byla sama multiplikativní volba parametrů odpovídá volbě pro RSA šifrování

43. počítání průměru (bezpečné spolupočítání)

- Protokol umožňuje skupině uživatelů počítat funkci nad vstupními daty tak, aby všichni znali výsledek ale ne vstup ostatních.
  1. A přičte ke své hodnotě tajné náhodné číslo, výsledek zašifruje veřejným klíčem účastníka B a předá výsledek B.
  2. B dešifruje zprávu, přičte svoji hodnotu, zašifruje a pošle dalšímu účastníku.
  3. Poslední z účastníků dešifruje přijatou zprávu, přidá svoji hodnotu, výsledek zašle A.
  4. A po dešifrování odečte své náhodné číslo, spočítá výsledek a zveřejní jej.
- Protokol nezabrání účastníku A podvádět, ani nezajišťuje, aby ostatní zadali správné hodnoty.

44. porovnejte dvě čísla (bezpečné spolupočítání)

- Protokol umožňuje skupině uživatelů počítat funkci nad vstupními daty tak, aby všichni znali výsledek ale ne vstup ostatních.

- Nechť A má tajné číslo i a B má tajné číslo j. Nechť i a j jsou celá čísla mezi 1 a 100
    1. A náhodně zvolí velké číslo x zašifuje je veřejným klíčem entity B a zašle: c-i=$E(K_B,x)$-i
    2. B spočítá těchto 100 čísel: $y_u=D(K_B,c-i+u)$, $1\leq u\leq 100$, vybere náhodně prvočíslo p o málo menší než x a spočítá všechna $z_u=(y_u \bmod p)$, $1\leq u\leq 100$
    3. B ověří, že pro všechna $u \neq v$: $|z_u-z_v|\geq 2$ a pro všechna u: $0<z_u<p-1$. V případě neúspěchu opakuje předchozí bod.
    4. B zašle A následující sekvenci: $z_1, z_2,...,z_j, z_{j+1}+1, z_{j+2}+1,..., z_{100}+1$, p
    5. A zjistí, zda i-tý prvek posloupnosti je kongruentní s x mod p. Tehdy a jen tehdy je $i \leq j$. A oznámí výsledek.
- Vadou protokolu je fakt, že v posledním kroku může A zastavit výpočet, nebo podvádět. Řešením paralelní provádění protokolu oběma stranami v kombinaci s vhodným protokolem pro výměnu zpráv.

45. prahová schémata (treshold schemes)

- protokol, k jehož úspěšnému provedení musí spojit síly více (např. t) účastníků, t nazýváme prahem (threshold value)
- pokud t - 1 a méně účastníků nemůže získat ani částečný výsledek, mluvíme o perfektním schématu
- **Sun a Shieh** (t, n) prahové schéma
    - vedoucí skupiny, který zná sdílenou informaci, generuje počáteční náhodná data a řídí práci účastníků (n účastníků) při uschovávání sdílené informace – každý účastník bude mít jen kousíček dat k sestavení celého tajemství
    - sdílená informace se schová do vhodného polynomu stupně t-1
    - každý účastník může spočítat jeden bod tohoto polynomu, ke spočítání celého původního polynomu Langrangeovou interpolací je nutno znát všechny

46. pravděpodobnostní šifrování

•zajišťuje, že stejný plaintext je při opakovaném použití stejného klíče šifrován na jiný zašifrovaný text

•**Blum-Goldwasser**

•The Blum-Goldwasser (BG) cryptosystem is an asymmetric key encryption algorithm. Blum-Goldwasser is a probabilistic, semantically secure cryptosystem with a constant-size ciphertext expansion. The encryption algorithm implements an XOR-based stream cipher using the Blum Blum Shub (BBS) pseudo-random number generator to generate the keystream. Decryption is accomplished by manipulating the final state of the BBS generator using the secret key, in order to find the initial seed and reconstruct the keystream.

•The BG cryptosystem is semantically secure based on the assumed intractability of integer factorization; specifically, factoring a composite value N = pq where p,q are large primes. BG has multiple advantages over earlier probabilistic encryption schemes such as the Goldwasser-Micali cryptosystem. First, its semantic security reduces solely to integer factorization, without requiring any additional assumptions (e.g., hardness of the quadratic residuosity problem or the RSA problem). Secondly, BG is efficient in terms of storage, inducing a constant-size ciphertext expansion regardless of message length. BG is also relatively efficient in terms of computation, and fairs well even in comparison with cryptosystems such as RSA (depending on message length and exponent choices). However, BG is highly vulnerable to adaptive chosen ciphertext attacks (see below).

•Because encryption is performed using a probabilistic algorithm, a given plaintext may produce very different ciphertexts each time it is encrypted. This has significant advantages, as it prevents an adversary from recognizing intercepted messages by comparing them to a dictionary of known ciphertexts.

47. produkční šifra

- obecné pojmenování pro blokové šifry, které fungují tak, že opakovaně (v kolech) provádí stejné transformace řízené klíčem – substituce, permutace, výpočty, dokud není zmatení nepřítele dostatečné
- přestože šifra po provedení jednoho kola nebývá bezpečná, předpokládá se, že se to velmi zlepší při vyšším počtu opakování stejné operace

- většinou se celý klíč rozdělí na části podle počtu kol (a následně nějakým způsobem rozšíří), v každém kole se tedy použije jiná jeho část
- **Feistelovy šifry**
  - podtyp produkční šifry
  - šifrování a dešifrování se většinou provádí velmi podobným způsobem (např. stejným postupem, jen s jinak inicializovaným klíčem apod.) -> nemusí se obojí implementovat zvlášť
  - Feistelova síť: v každém kole se vstup rozděluje na 2 poloviny, každá se nějak zpracuje i pomocí v algoritmu definované Feistelovy funkce a poté se strany prohodí
  - např.: Blowfish, DES, RC5, 3DES

48. proudové šifry

- symetrické šifry, zpracovávají otevřený text po jednotlivých bitech
- vždy nějaký pseudonáhodný generátor streamu (stream generován podle klíče), který se např. XORruje s procházejícím plaintextem
- RC4, FISH, A5/1, A5/2 (GSM mobilní telefony)
- proudové šifry představují rozdílný přístup k sym.šifrování oproti blokovým šifrám. Blokové šifry pracují na velkých blocích čísel s neměnnou transformací. Tento rozdíl není vždy tak jasný, v některých modech se může bloková šifra chovat stejně efektivně jako proudová (CFB,OFB). Proudové šifry jsou často mnohem rychlejší než blokové a mají menší hardwarové nároky. Nicméně proudové šifry mohou být náchylné na bezpečnostní problémy pokud jsou použity nekorektně – především, počáteční stav nesmí být nikdy použit dvakrát.

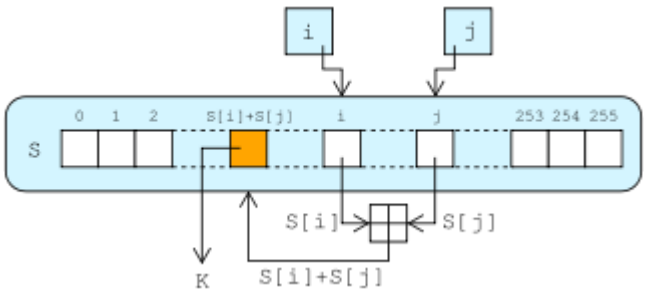
49. RC4

- proudová (tzn. symetrická) šifra od R. Rivesta (Rivest Cipher 4), používá se v SSL a WEP
- RC4 generuje pseudonáhodný proud bitů (keystream) který, pro šifrování, je kombinován s proudem plaintextu pomocí XOR; dešifrování se dělá stejně (bo XOR je symetrická operace)
- Nepoužívá inicializační vektor, čili klíč se musí pro každé spojení generovat nový a dopravit druhé straně pomocí nějaqké asymetrické metody. Šifra má volitelnou délku klíče, nejčastěji používané jsou klíče délky 40 a 128 bitů.
- Z klíče se vyrobí permutace na množině {0,..,255}, čili na množině všech bajtů. Inicializace (key-schedule):

```
for (i = 0; i < 256; i++) { S[i]=i }
j = 0;
for (i = 0; i < 256; i++) {
    j = (j + S[i] + key[i mod keylength]) mod 256;
    swap(&S[i],&S[j]);
}
```

- Pseudonáhodné generování keystreamu (PRGA)

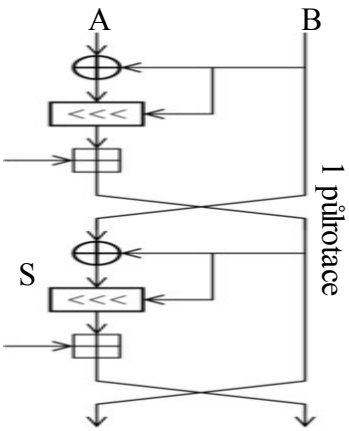
```
i = 0; j = 0;
while GeneratingOutput {
    i = (i + 1) mod 256;
    j = (j + S[i]) mod 256;
    swap(&S[i],&S[j]);
    output S[(S[i] + S[j]) mod 256];
}
```



- velmi jednoduchý a rychlý algoritmus, má ale své slabosti, zvláště když začátek výstupního keystreamu není vyřazen (špatné statistické vlastnosti prvních cca 100 bajtů), nenáhodné nebo podobné klíče použity nebo jeden keystream použit dvakrát - v těchto případech vede na velmi ne-bezpečné kryptosystémy jako je třeba WEP (bezdr.sítě)

50. RC5

- bloková (tzn. symetrická) Feistelova šifra, publikoval v roce 1994 R. Rivest
- velmi pružný algoritmus s celou řadou parametrů - délka šifrovacího klíče (0 až 255 bytů), počet kol šifrovacího procesu (0 až 255), z hodnot 16, 32, 64, ale i vyšších lze zvolit délku slova, bloky mají délku 2xslovo
- základem RC5 jsou rotace závislé na datech (nová myšlenka), skládá se také ze sčítání modulo délka slova a XORů, základem alg. je Feistelova síť aplikovaná na dvě části bloku A a B
- inicializace (key-schedule): rozšíříme klíč jednocestnou funkcí pomocí základu přirozeného logaritmu a zlatého řezu =>pole subklíčů S
- Jako rozumná se jeví volba délka slova 32 bitů, 12 kol, 16bajtový klíč, což krátce zapíšeme RC5-32/12/16.



51. Rijndael (AES)

- produkční bloková (tzn. symetrická) šifra s proměnnou délkou bloku 16, 24 nebo 32 bajtů a klíčem o délce 128, 192, 256 bitů
- je založena na síti S-Boxů a P-Boxů =>jednoduše soft.i hardw. implementovatelná
- Rijndael pracuje na poli 4x(délka_bloku /32) bytů což určuje její stav (klíč je také pole 4x (délka_klíče /32))
- šifra je určena jako počet opakování tranformačních kol, které přemění plaintext na ciphertext, každé kolo se skládá z tranformačních kroků a jednom co záleží na klíči, posloupnost opačných kol se používá na dešifrování se stejným klíčem
- Zajímavé je, že celé kolo šifrovacího procesu lze na 32 bitovém procesoru implementovat jako 4 výběry z tabulky a 4 XOR operace
- Algoritmus byl podroben rozsáhlé analýze a zvolen jako nový standard AES, v současnosti není známa jakákoliv podstatná slabina

52. RSA

- asymetrický kryptosystém, jedná se o první algoritmus, který je vhodný jak pro podepisování, tak šifrování
- Kryptoschéma je založeno na Eulerově formuli
- Alice a Bob se veřejne dohodnou na hranici N a chtějí si vyměnovat tajné zprávy $0 \leq z < N$.
- **Tvorba klíčového páru:**
- Nejprve si bude Alice muset vyrobit pár veřejného a soukromého klíče:
  1. Zvolí dvě různá velká náhodná prvočísla p a q, tak aby $n = pq > N$.
  2. Alice spočte $\varphi(n) = (p - 1)(q - 1)$ a tajně zvolí v $Z_{\varphi(n)}$ invertibilní prvek d.
  3. Nalezne číslo e tak, aby platilo $de \equiv 1 \pmod{\varphi(n)}$.
- Veřejným klíčem je dvojice (n, e), soukromým klíčem je dvojice (n, d)
- **Šifrování:** Bob vyhledá Alicin verejný klíc (n, e) a spočte $x = z^e$ v Z_n. Bob verejne odešle Alici číslo x.
- **Dešifrování:** Alice přijme x a spočte $z = x^d$ v Z_n.
- Pro reálné použití čísla přibližně 100 až 200 bitů. Klíč e volíme jako prvočíslo větší než (p - 1) a (q – 1). Hranice bezpečnosti pro modul n je N = 1024 bitů, rozumné 1500 bitů, lépe 2048
- není známa metoda vedoucí k rozbití tohoto algoritmu
- slabostí je hypotetická možnost vytvořit elektronický podpis zprávy bez znalosti dešifrovacího klíče na základě zachycení vhodných předchozích zašifrovaných zpráv

53. RSA generátor

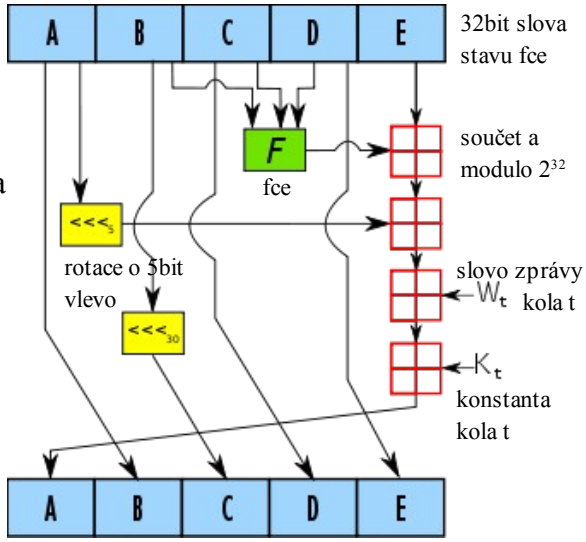
- Generátor pseudonáhodných sekvencí = posloupnost délky n, o které žádný deterministický p-time algoritmus není schopen s pravděpodobností větší než (1/n)+ε rozhodnout, zda se jedná o náhodnou posloupnost, či nikoliv.
  - všechny tyto generátory jsou deterministickými p-time algoritmy, které na vstupu přijmou náhodný řetězec a expandují jej do (obvykle) mnohem delší posloupnosti
- $Xi=X^{e}_{i-1} \bmod n$
- výsledkem nejnižší (nejméně významný) bit Xi. Bezpečnost ekvivalentní s bezpečností RSA

54. rysy návrhů kryptografických protokolů

- arbitrované protokoly - arbitrem rozumíme neutrální třetí stranu, zajišťující férovost, nevýhodou použití arbitra jsou zvýšené náklady na provoz, problémy s dostupností, časové prodlevy, potíže s důvěryhodností a výkonem
- rozhodované (adjudicated) protokoly - rozhodčí (adjudicator) je třetí strana, která je schopna rozhodnout, zda operace byla vykonána férově, případně kdo z účastníků porušuje pravidla, třetí strana je tedy používána pouze při sporu
- samozabezpečovací (self-enforcing) protokoly - samotný protokol zajišťuje vzájemnou ochranu účastníků

55. SHA-1

- message digest funkce transformující zprávu libovolné délky na 160-bitový digest (otisk zprávy)
- Pracuje ve třech krocích
  - inicializace – zarovnání vstupu, příprava interních datových struktur – nejprve se doplní zpráva do násobku 512: doplní se 1, potom tolik nul, aby zpráva byla o 64 kratší než nejbližší vyšší násobek 512 a na konec do 64 bitů zapsaná původní délka zprávy
  - iterace kompresní funkce – iterativní aplikace kompresní funkce na bloky zprávy, akumulace historie výpočtu
  - dokončení – konstrukce finálního výsledku z akumulovaných vnitřních stavů



- SHA 256 (SHA-384, SHA-512) - principiálně stejné algoritmy jako SHA-1, jednotliví členové rodiny SHA-X se od sebe liší délkou slova a počtem kol
- Na konci roku 2005 byl oznámen útok spočívající v nalezení kolize se složitostí 2^{63} operací (Wang, Yao, Yao)

56. šifrovací klíče

- Šifrovací klíč je informace, která určuje průběh kryptografického algoritmu. Při šifrování, klíč specifikuje transformaci zprávy do šifrovaného textu, při dešifrování je tomu naopak. Klíče se používají také v digitálních podpisových schématech a hašovacích funkcích (také MAC - message authentication code), často používaných na autentizaci
- V bezpečných algoritmech, šifrováním zprávy pomocí různých klíčů dostaneme kompletně různé šifry a také dešifrování nesprávným klíčem dá náhodně vypadající text (nicméně existují také kryptosystémy, kde dešifrování různými klíči může dát různé rozumně vypadající zprávy).
- V praxi je užitečné předpokládat, že kryptografický algoritmus je útočníkovi znám a spoléhat se jenom na bezpečnost klíče, protože je většinou jednodušší uchovat v tajnosti relativně malý klíč, nežli detaily algoritmu. Tento princip se nazývá Kerckhoffovo pravidlo — „iba bezpečnost klíče zaručí bezpečnost systému„, nebo “nepřítel zná tvůj systém“.
- Aby se ztížilo případné uhádnutí klíče, měl by být generován náhodně a mít dostatečnou entropii, což v

podstate znamená, že je náhodně vybrán z dostatečně velkého počtu možností. Na generování náhodnosti se často používá hardwarový šum, v případě potřeby jenom malého množství však postačí i obyčejná (vyvážená) hrací kostka.

57. šifrování vysokokapacitního spoje (např. real-video)

•**navrhnut reseni, vlastnosti**: no ja som v tej ulohe napisal len obecne kecy viac menej a bohate mu to stacilo. konkretne, pouzijem hybridne sifrovanie, na vymenu klucov pouzijem DH-kryptosystem, a potom som popisal RC5, v mode OFB, a popisal som tu sifru...

•**co sa stane pri chybe-vypadek**: samotny rezim OFB zarucuje, ze aj pri chybe sa vysielanie zotavi rychlo...

58. srovnání symetrických a asymetrických šifer

- Symmetric-key algorithms are generally much less computationally intensive than asymmetric key algorithms. In practice, this means that a quality asymmetric key algorithm is hundreds or thousands of times slower than a quality symmetric key algorithm.

59. statistické testy generátorů

- **Frequency test** - zkoumá počet 0 a 1 v generované posloupnosti
- **Serial test** – zkoumá počet dvojic bitů (00, 01, 10, 11) v posloupnosti
- **Runs test** – série 1 za sebou se nazývá blok, séria 0 za sebou se nazývá díra, zkoumá počet bloků a děr nějaké délky
- **Poker test** – posloupnost rozdělíme na k úseků délky i, zkoumá počet úseků typu j (0000, 0001 aj.)
- **Autocorelation test**
- **Ticket/Coupon Collector Test** – zkoumá, po jaké době se ve zkoumaném vzorku od začátku objeví všechny možné permutace dané délky

60. symetrický keymanagement

- text + klic> sifra / sifra + klic> text
- vždy je nutné provést nějakou část distribuce klíčů manuálně (např. distribuce master klíče, ostatní se pak distribuují zašifrované masterem) např. si vymenit klíče pomocí D-H-kryptosystému
- většinou musí mít každá dvojice, která se chce komunikovat, mezi sebou vlastní klíč -> uživatel musí spravovat a držet v tajnosti velké množství klíčů

61. útok na generátor

- **Choosen input atttack** - Útočník ovládá nebo alespoň všechny, nebo některé zdroje entropie generátoru, pokud má přístup k výstupu, může provádět adaptivní útok
- **Kryptoanalytický útok** - Hledání charakteristiky výstupní posloupnosti, predikovatelnosti způsobené nesprávným návrhem generátoru, odpovídá lámání proudových šifer
- **Iterative guessing** - Pokud při překlíčování generátor nezahrne dostatečné množství nové entropie, je změna vnitřního stavu generátoru „příliš malá“. Pokud útočník znal stav generátoru před změnou, dokáže nalézt stav po změně a nadále predikovat výstup.

62. Vernamova šifra

- proudová(tzn.symetrická) šifra (okolo1920), která XORuje plaintext s náhodným nebo pseudonáhodným streamem stejné délky
- pokud jsou data opravdu náhodná a použitá pouze jednou jde o one-time pad (šifr.alg. kde plaintext je kombinován s tajným nahodným klíčem který je použit pouze jednou)

- příkladem Vernamovy šifry je RC4

63. vyrobte hašovací funkci z šifry

- využije se libovolná bloková šifra (např. DES), bere se jeden blok za druhým, pro zašifrování každého se jako klíč použije výsledek z předchozího kroku (tj. předchozí blok již v zašifrované podobě)
- mezikrokem může být funkce, která mapuje prostor zašifrovaných bloků do prostoru klíčů
- příklady: MDC-2, MDC-4

64. vyrobte šifru z hašovací funkce

- There are several methods to use a block cipher to build a cryptographic hash function. The methods resemble the block cipher modes of operation usually used for encryption. Using a block cipher as a hash function is usually much slower than using a specially designed hash function. But, in some cases it is easier because a single implementation of a block cipher can be used for both block cipher and a hash function. It can also save code space in very tiny embedded systems like for instance smart cards or nodes in cars or other machines. In fact, all the existing hash functions are based either on block ciphers or on stream ciphers. MDx, SHA, Whirlpool, etc. can all be used as block ciphers without the final addition of the initial state to the output. Hash functions can be constructed based on stream ciphers as well (VEST), although certain properties are required for it to work - the stream cipher must be able to accept variable-length IVs and must process them bijectively.

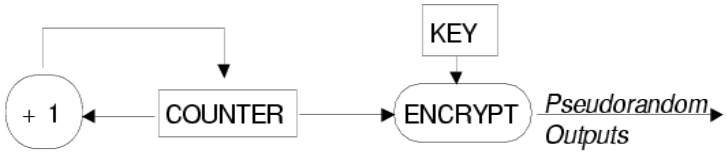
65. XOR z pohledu kryptologa

- exkluzivní disjunkce
- **kde se používá:** Používá se ve většině Feistelových šifer(blokové šifry) (Blowfish, DES, IDEA,RC5), Rijndael, MD5 (v kompresní funkci), LFSR (proudové šifry)
- **špatné vlastnosti:** symetrická operace?
- **další operace:** substituce (S-Box), modulo, diskrétní logaritmus, permutace (P-Box)

66. Yarrow

- příklad komplexní konstrukce náhodného generátoru, název znamená kytku řebříček která byla ve staré číně používána pro randomizaci
- cílem útočníka je získat vnitřní stav generátoru, pokud uspěje, může predikovat výstup generátoru, nebo rekonstruovat dřívější části výstupní posloupnosti
- kvalita generátoru závisí na kvalitě vstupu, častým problémem nesprávný odhad míry entropie zdrojů
- generátor se skládá ze čtyř hlavních komponent:

- akumulátor entropie (entropy accumulator) - používají se dva pooly – rychlý a pomalý; každý pool je realizován jako kontext hashovací funkce (zde SHA-1), do kterého se přimíchávají vstupy „tak jak přicházejí“
- mechanismus změny klíče (reseed mechanism) - na pokyn řízení změny klíče počítá novou hodnotu klíče K
  - změna klíče na základě rychlého poolu využívá stávající klíč + hash všech vstupů v rychlém poolu od poslední změny klíče
  - změna klíče na základě pomalého poolu využívá stávající klíč, hash všech vstupů v rychlém poolu od poslední změny klíče a hash všech vstupů v pomalém poolu od poslední změny klíče
  - po změně klíče jsou vynulovány čítače entropie všech dotčených poolů
- mechanismus generování (generation mechanism)



- využívá se symetrická šifra (3DES) běžící v tzv. counter módu

- řízení změny klíče (reseed control)
  - s každým zdrojem entropie je v poolu udržován čítač nashromážděných náhodných bitů
  - změna klíče z rychlého poolu se provede pokud aspoň jeden čítač přesáhne práh
  - změna klíče z pomalého poolu se provede pokud aspoň k čítačů přesáhne práh
  - práh pro rychlý pool bývá 100, pro pomalý 160
- cílem je kvalitní akumulace entropie ze všech zdrojů, zajištění dostatečně častých (nealgoritmických !) změn klíče a občasná změna klíče, která bude pro útočníka nepřekonatelná i v případě, že u některého zdroje došlo k nadhodnocení obsahu entropie