Úvodem

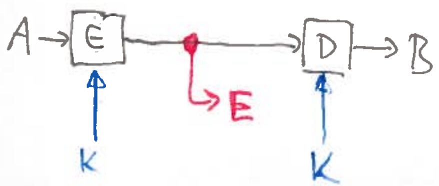
- Cíle:
  - přehled o kryptografii teoretické i praktické!
    → kryptografická primitiva
    → protokoly
    → implementační otázky
  - cílem je rozumět existujícím protokolům
    ... a vědět dost o návrhu vlastních, abychom to nechtěli dělat :)
- nebudeme budovat hlubokou teorii (→ Foundations of Theor. Crypt. & ledacos z MMIB)
  ... ale občas nějakou větu dokážeme :)
- o obtížnosti návrhu bezpečných systémů (a vývoje SW obecně)
- prerekvizity (tak trochu): algoritmy, architektura HW, algebra, složitost...

↓ náhodný uživatel vs. zlomyslný uživatel
↓ weakest links, attach trees

Primitiva

(1) Scénář: • Alice chce posílat Bobovi šifrované zprávy
→ to obecně nejsou konkrétní osoby, ale role v protokolu
  → posílání oběma směry (prohození rolí)
  → Bob může být třeba Alice v budoucnosti

- Eva poslouchá [< eavesdropping]
- Mallory data upravuje (ale o tom později)
  ← The Man in the Middle



- hodí se E a D parametrizovat klíčem
- Kerckhoffsův princip: tajný má být klíč, ne algoritmus!

Rationale: ② je-li šifra veřejně známá, bývá lépe otestovaná (2)
③ vyměnit kompromitovaný klíč je snazší než algoritmus
① dobrých šifer je málo a je těžké je vytvořit

→ symetrická šifra (E i D používají stejný klíč)

Šifra obvykle nemůže utajit délku zprávy.
↓
typ. sym. šifra délku zachovává
↓
formálně:
$E: \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$
$D:$ — " —
$\forall K \forall X \; D(X, E(X,K)) = X$
& pro náhodný klíč se $E(-,K)$ chová jako náhodná permutace na $\{0,1\}^n$

příklad: Caesarova šifra

② Asymetrická šifra
- odvodit šifrovací a dešifrovací klíč
- typické aplikace:
  → N lidí komunikujících navzájem
    - šifrovací klíč je veřejný
    - dešifrovací je tajný
    - problémy s distribucí klíčů!
  → digitální podpisy
    - šifrovací klíč tajný, dešif. veřejný
    - každý může podpis ověřit, ale jen 1 osoba vytvořit

③ Hešovací funkce: $\{0,1\}^* \to \{0,1\}^b$ (třeba $b = 256$)
"dostatečně náhodná"
Chceme:
- nemožnost inverze
- nemožnost nalezení kolize

- typické aplikace: → kompaktnější podpisy (nechceme kolize!)
  → Message Auth Code (symetrická verze podpisu)

④ Náhodné generátory
Chceme:
- nepredikovatelnost
- neovlivnitelnost

- aplikace: → hybridní šifra ze symetrické a asymetrické
  → challenge–response autentikace

Společné cvičení: protokol pro aukci (viz Smart)
- → padding
- → timestamps / seq. numbers (proti replayování)
- → nonce (proti porovnávání šifrovaných zpráv)
- → session ID (proti replayi jiné instance protokolu)

Modely útoku - proti komu se bráníme ← Dú: házení mincí po telefonu
- jak dlouho musí tajemství vydržet

↓ commitment pomocí haš. fcí

Typy útoků
- known ciphertext (chceme plaintext)
- known plaintext (chceme klíč)
- chosen plaintext } též chceme klíč
- chosen plaintext & ciphertext }
- rozlišovací útoky

Jak měřit obtížnost útoku? → security level

"Narozeninové" útoky

① Challenge-response autentifikace, n různých nonců
∞ kolik pokusů v průměru potřebujeme, než se nonce zopakuje?

$\Pr[\text{náhodná } f \text{ z } [m] \text{ do } [n] \text{ je prostá}]$ (m: pokusy, n: nonce)

$= \frac{\#\text{prostých fcí}}{\#\text{všech fcí}} = \frac{n^{\underline{m}}}{n^m} = 1 \cdot \left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right) \cdot \_\_ \cdot \left(1-\frac{m-1}{n}\right)$

Jelikož $1-x \approx e^{-x}$, aproximujeme $1 \cdot e^{-\frac{1}{n}} \cdot e^{-\frac{2}{n}} \cdot \_\_ \cdot e^{-\frac{m-1}{n}}$

$= e^{-\frac{1+2+\dots+m-1}{n}} = e^{-\frac{m(m-1)}{2n}}$

Chceme $\Pr[\text{kolize}] = \frac{1}{2} \rightarrow e^{-\frac{m(m-1)}{2n}} = \frac{1}{2} \Rightarrow \frac{m(m-1)}{n} = -2\ln\frac{1}{2} \doteq 1.38$ ($\ln\frac{1}{2} \doteq -0.7$)

$\Rightarrow$ přibližně $m \approx \sqrt{n}$

(2) Protokol: • A zvolí náhodný klíč (a pošle ho zašifrovaným šifrou) (4)
• A pošle uvítací zprávu podepsanou klíčem
• další zprávy podepsané stejně

z množiny velké n

Útok: Eva si předpočítá podpisy uvítací zprávy pro a náh. klíčů
Pak poslouchá m relací a čeká, až se objeví předpočítaný klíč

Model:

$$Pr[f \text{ se strefí do } A] = \left(1-\frac{a}{n}\right)^m \approx e^{-\frac{am}{n}}$$

... to je konstanta pro $n \approx am$.

→ trade-off mezi časem na předvýpočet a délkou útoku.

! Pozor, security level je někdy $2\times$ menší, než bychom čekali !

Jednorázové klíče – Vernamova šifra

- zpráva $x \in \{0,1\}^n$, klíč náhodný $k \in_R \{0,1\}^n$ → $x \oplus k \in \{0,1\}^n$ $E(x,k)$
  - E a D jsou tatáž funkce
  - výsledek je posloupnost n nezávislých náhodných bitů!
    ... očistěm korelovaných s klíčem
- Jiná podobná konstrukce: $x \in \mathbb{Z}_{2^n}$, $k \in \mathbb{Z}_{2^n}$, $E(x,k) = x+k$, $D(y,k) = y-k$ v $\mathbb{Z}_{2^n}$

∀y ∀k ∃!x: $E(x,k)=y$

⇒ $Pr_k[D(y,k)=x]$ je pro všechna x stejná

⇒ y nenese žádnou informaci o x (kromě délky)

Def. perfektní bezpečnosti

→ Čemu je to dobré? → code books

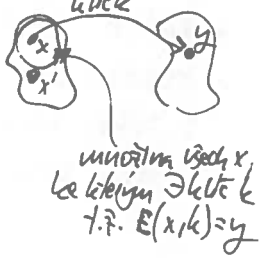
Ale pozor!

- nesmíme nikdy zopakovat klíč (viz Sověti ve WW2)
- útočník může zprávu triviálně měnit

Věta: Pokud #klíčů < #zpráv, šifra není perfektně bezpečná.

Dk: Nechť $y \in \{0,1\}^n$

Pak $\exists x, x' \in \{0,1\}^n : \exists k : E(x,k) = y$
ale $\forall k' : E(x',k') \neq y$



Proto $Pr_k[D(y,k) = x] > 0$,
ale $Pr_k[D(y,k) = x'] = 0$
$\Rightarrow$ rozdělení není rovnoměrné.

Dělení tajemství (aneb o silnějších generátorech)

① $x \to x^1, x^2$ t.ž. samotné x^i mi neřekne nic o x (kromě délky), ale x^1, x^2 dohromady určí x jednoznačně.

Řešení: x^1 náhodné, $x^2 := x \oplus x^1$

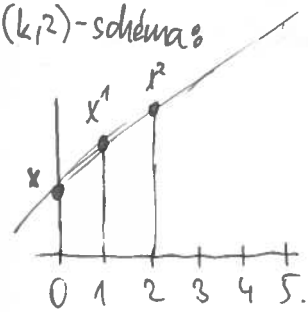
② $x \to x^1, \dots, x^k$ t.ž. všech k částí určí x jednoznačně, žádných $k-1$ nic neprozradí.

Řešení: $x^1 \dots x^{k-1}$ náhodné, $x^k := x \oplus \bigoplus_{i=1}^{k-1} x^i$.

Obecně: (k,l)-prahové schéma rozdělí zprávu na k částí tak, že
- libovolných l částí určí celé x,
- žádných $l-1$ nic neprozradí.

$\Rightarrow$ pomocí ③ sestrojíme (k,k)-schéma.

③ $(k,2)$-schéma:



Hledám $f(t) = at + b$
t.ž. $f(0) = x$
$f(1)$ je náhodné
} tahle f existuje právě 1

a pak volím $x^1 = f(1), \dots, x^k = f(k)$

(aby to bylo dobře def., počítám v konečném tělese (dost velkém))

libovolná dvě x^i, x^j jednoznačně určí f, ale pokud znám jen x^i, všechna x jsou stejně pravděpodobná (každému odpovídá právě jedna f).

④ obecné (k,ℓ) - schéma

- f bude polynom stupně menšího než ℓ nad konečným tělesem
- $f(0)=x$, $f(1)$ až $f(\ell-1)$ volím náhodně } to jednoznačně určí f (& všechny f jsou stejně pravděpodobné)
- rozdám části $f(1)$ až $f(k)$

→ pokud zradím ℓ částí, určím jednoznačně f a najdu $f(0)$

→ pokud zradím $c<\ell$ částí: pokud liberálně nastavím dalších $\ell-c-1$ částí, každá volba x určí právě jeden f
→ všechna x jsou stejně pravděpodobná

Lemma: Pokud p je polynom s kořeny $\alpha_1 - \alpha_t$, pak
$p(x) = (x-\alpha_1)\cdot \_ \cdot (x-\alpha_t)\cdot q(x)$ pro nějaký polynom q bez kořenů.

Věta: Polynom stupně d má nejvýše d kořenů.
(nenulový)

Důsledek: Pokud p,q jsou polynomy stupně menšího než d a $p(x_i)=q(x_i)$ pro navzájem různá (nejaká) $x_1 - x_d$, pak $p=q$.

Věta (Lagrange): $\forall$ $x_1 - x_d$ navzájem různá $\forall$ $y_1 - y_d$
$\exists p$ polynom stupně $<d$ t.ž. $\forall i$ $p(x_i)=y_i$.

(z předchozího víme, že je jednoznačný.
⇒ máme bijekci mezi polynomy stupně $<d$ a vektory $(f(x_1), \_, f(x_d))$ pro libovolné čísla navzájem různé $x_1 - x_d$.

SYMETRICKÉ ŠIFRY

Dva základní druhy:

- **proudové** (stream ciphers) — generují keystream, se kterým se data xorují

(vlastně Vernamova šifra s pseudonáhodným generátorem)

- D = E (inverzní sama k sobě)
- nesmíme opakovat nonci
- zneguvání y_i zneguje x_i
- E_K i $E_{K'}$ komutuje

více poradí.

- **blokové** (block ciphers)
  - šifrují bloky pevné délky b
    $E: \{0,1\}^b \times \{0,1\}^k \to \{0,1\}^b$
    často značíme $E_K: \{0,1\}^b \to \{0,1\}^b$
  - E_K musí být invertibilní: je to permutace na $\{0,1\}^b$
  - delší zprávy šifrujeme po blocích (TODO)

Triviální příklady

- Caesarova šifra má 1znakové bloky, permutace je cyklický posun abecedy o k dč.
  - Bug #1: málo klíčů → triviální brute-force útok
  - Bug #2: krátké bloky, nulová interakce mezi nimi → frekvenční analýza
- Vigenèrova šifra: víceznakové bloky, opět přičítám k dč. → na tohle se dá dívat i jako na proudové šifry
- obecné permutace abecedy nebo větších bloků

Bezpečnost blok. šifer

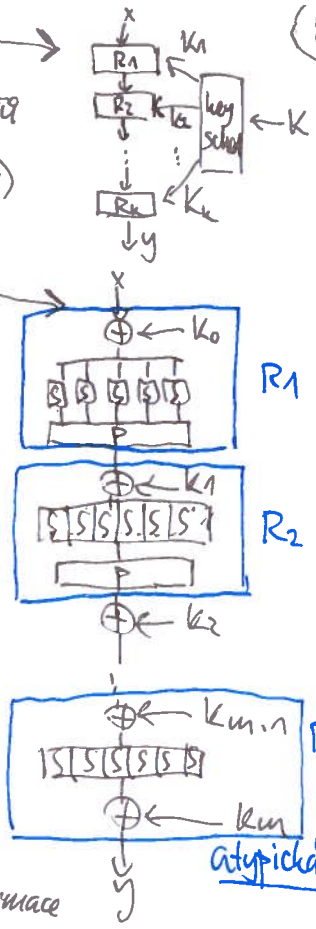
- Těžké definovat formálně (buď to umí útoky obejít, nebo definici nesplňuje žádná rozumná šifra)
- Idea: šifru nelze efektivně rozlišit od náhodné permutace
  - verifikátor dostane orákulum buď s E_K pro náhodný K, nebo s náhodnou permutací
  - má odpovědět, které orákulum dostal
  - může položádat více dotazů
  - chceme, aby nešla dosáhnout Pr úspěchu $\geq 2/3$ s lepší složitostí než $\sim 2^{\text{security level}}$
    ? co to je?
- tohle nepokrývá chosen-key/related-key útoky !
- časem prostudujeme další algoritmy

! Reálné šifry jsou prakticky vždy sudé permutace

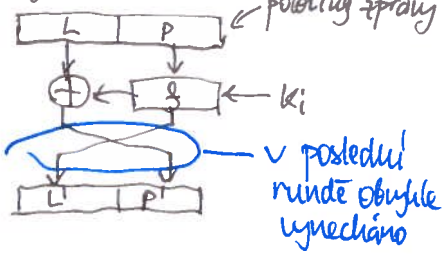
DES (Digital Encryption Standard)

Odbočka: o způsobech konstrukce blok. šifer

- iterované šifry, rundy, rozvrh klíčů
- substitučně-permutační sítě (SPN)
  - S-boxy: malé tabulky, musí být invertibilní
  - počáteční a koncový XOR: whitening (omezuje kontrolu útočníka nad vstupy do S-boxů
  - P-box: obecná permutace na pozicích v bloku
  - díky atypické R_m je inverze k SPN zase SPN (někdy volíme S, P jako involuce → tatáž SPN, jen obrátíme rozvrh klíčů)
  - confusion vs. diffusion
  - upgrade: kromě P povolit invertibilní lin. transformace



- Feistelovy sítě
  - konstrukce s neinvertibilními S-boxy
  - runda obecně vypadá takto:
  - $f(P, k_i)$ může být libovolná funkce (typ. postavená z S/P-boxů)



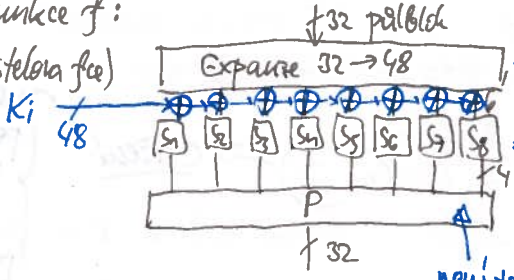
Historie DESu:

- vyvinut začátkem 70. let v IBM na zakázku NBS (Nat. Bureau for Standards), do vývoje mluvila i NSA
- 56-bitový klíč (technicky 64, ale v byte má paritní bit)
  - původní verze byla silnější
- 64-bitové bloky
- NSA na poslední chvíli vynechala S-boxy – krajně podezřelé?
  - dnes už víme, že tím šifru zeslabila

Struktura DESu:

- Feistelova síť s 16 rundami pracujícími s 32-bitovými půlblocky
- Navíc počáteční a koncový P-box (zcela zbytečné)
- Funkce f:

(Feistelova fce)

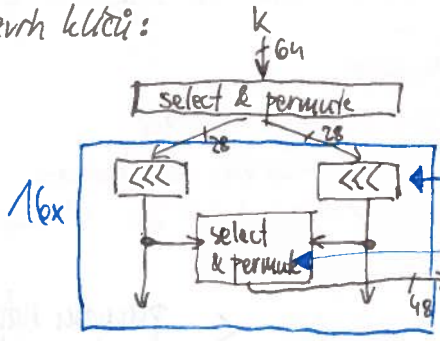


∀ 4-bitový blok si vezme krajní bit z ∀ souseda (cyklicky) → 6 bitů výstupu

8 S-boxů 6→4 bity (různých)

není to involuce

- Rozvrh klíčů:



rotace o 1 nebo 2 pozice (pro různá i různě)

pro všechna i stejné

Kritika DESu

- Pokud $K=0^{56}$, všechny k_i jsou $0^{48} \rightarrow E_K = D_K$ } 4 tzv. slabé klíče
  podobně pro $K=1^{56}$ a 2 další klíče.
- 6 dvojic klíčů (k, k'), takové, že $K \rightarrow (k_1, k_2, k_1, k_2, \ldots)$ a $k' \rightarrow (k_2, k_1, k_2, k_1, \ldots)$
  Pak: $E_k(E_{k'}(x)) = x$ pro všechna x.
- $E_{\bar{k}}(\bar{x}) = \overline{E_k(x)}$ } komplementárnost
- Příliš krátké klíče?
  - už v roce 1977 se odhadovalo, že za 20 M$ jde postavit stroj, který vyřeší všechny klíče za 1 den
  - 1997: RSA Security Inc. DES Challenge – cena 10k$
    → crackнuto distrib. výpočtem v idle čase 78k počítačů
  - ⋮
  - 2012: deska s 48 FPGA prohledá celý prostor za 26 hodin (pronajímají jako službu?)

- Krátké bloky – kolize bloků jednou za 2^{32} bloků !
- Útoky na strukturu:
  - diferenciální kryptoanalýza: stačí 2^{47} chosen plaintextů
  - lineární kryptoanalýza: stačí 2^{43} známých plaintextů
  - → dost na to, abychom šifru považovali za rozbitou

Pokusy o záchranu DESu (90. léta)

- 2-DES – nepomůže ! → sec. level jen 57 → cvičení
- 3-DES – $E_{K_3}(D_{K_2}(E_{K_1}(x)))$ → 168-bit. klíč, sec. level 112
  - někdy se potká varianta s $K_1 = K_3$, pozor, má sec. level jen cca 80

nebezpečí: permutace mohou tvořit grupu [DES ji netvoří]

AES – Advanced Encryption Standard

- 1997 – NIST (nástupce NBS) vypisuje otevřenou soutěž
  - 15 návrhů šifer, několik kol veřejného hodnocení
  - kriteria: bezpečnost, rychlost + snadnost SW i HW implementací
- 2001 – šifra Rijndael prohlášena za AES
- 128-bit. bloky, klíč 128, 192 nebo 256 bitů ← původní návrh uměl i delší klíče a větší bloky

Struktura:

↓ 10 rund ↓ 12 ↓ 14 rund

- není to feistelovská šifra, ale SPN s lineární transformací navíc
- bajtově orientovaná (pro efektivní implementaci v SW)
  ↳ s bajty zacházíme jako s prvky $GF(2^8)$
- Stav šifry (předávaný mezi rundami) je matice 4×4 bytů, rundový klíč má stejný tvar.
- Runda:
  - ByteSub — bajty stavu prožene me identickými S-boxy $8 \to 8$ [S-box je inverze v $GF(2^8)$ + afinní transf. z rotací a XORů]
  - ShiftRow — 1-tý řádek rotujeme o i bytů doleva
  - MixColumn — na ∀ sloupec (coby 4D vektor) aplikujeme stejnou invertibilní lin. transformaci
  - AddRoundKey — XOR s klíčem

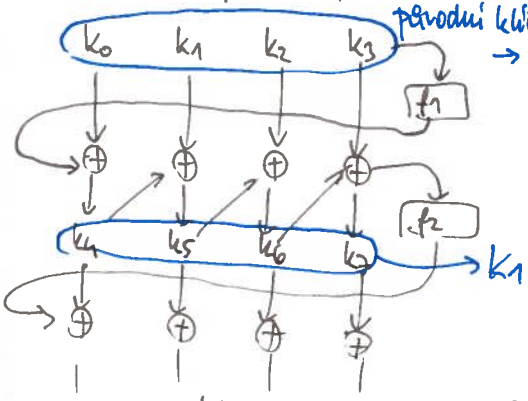
  v poslední rundě není

- Před 1. rundou AddRoundKey

- Inverzní runda:
  - AddRoundKey (K_i) } komutuje, pokud K_i nahradíme jeho mixem
  - Inv Mix Column
  - Inv Shift Row } komutuje
  - Inv Byte Sub

} tohle prohodíme kvůli posunu rund → D_K vypadá skoro jako E_K, jen máme jiné S-boxy a jiné mixování

- Rozvrh klíčů: pracuje po 32-bit. slovech



f_i je postavena z S-boxu (téhož jako v rundě), rotace o 1 byte a přimíchání rundové konstanty

verze pro 128 bit. klíč → pro 192 bit. je to jen širší, pro 256 bit. je na prostředních $\oplus k_i$ ještě jedna nelinearita (aplikace S-boxu)

- Vylepšení implementace na 32-bit. CPU
  - 4 tabulky $8 \to 32$ kombinující S-box s částí Mix Columns (+ sloupec je XOR 4 hodnot v tabulkách) } 4 kB

Kritika

- jednoduchá algebraická struktura (útok řešením rovnic ... zatím se neumí)
- příliš malá rezerva v # rund
- zarovnaná na bajty

→ ale zatím se žádný zajímavý útok neumí. [kromě implementačních — viz později]

→ nějaké related-key útoky, ale stejně mají velkou složitost a tolik rel. keys se těžko shodí

- 128-bit. klíč není bezpečný proti kvantovým počítačům (Grover's alg.)
- 128-bit. bloky hrozí kolizními útoky po 2^{64} blocích
  → obejdeme změnami klíče po $\sim 2^{32}$ blocích (v protokolu)