# Matematika IV – 6. přednáška
# Kódování a šifrování

Michal Bulant

Masarykova univerzita
Fakulta informatiky

6. 4. 2011

# Obsah přednášky

1. Úvod do kódování
   - $(n, k)$–kódy
   - Lineární kódy
2. Pár slov o šifrách

# Doporučené zdroje

- Martin Panák, Jan Slovák, **Drsná matematika**, *e*-text.
- *Předmětové záložky v IS MU*

# Doporučené zdroje


- Martin Panák, Jan Slovák, **Drsná matematika**, *e*-text.
- *Předmětové záložky v IS MU*
- R. B. Ash, Abstract algebra, `http://www.math.uiuc.edu/~r-ash/Algebra.html`.
- Jiří Rosický, *Algebra*, PřF MU, 2002.
- Peter J. Cameron. *Introduction to algebra*, Oxford University Press, 2001, 295 s. (Dostupné v knihovně PřF).
- Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone , *Handbook of Applied Cryptography*, CRC Press, 2001, 780 p., `http://www.cacr.math.uwaterloo.ca/hac/`
- Jan Paseka, *Kódování*, elektronický text, MU – `http://www.math.muni.cz/~paseka/ftp/lectures/kodovani.ps`.

# Plán přednášky

1. Úvod do kódování
   - $(n, k)$–kódy
   - Lineární kódy
2. Pár slov o šifrách

# Kódování

Při přenosu informace zpravidla dochází k její deformaci. Budeme pro jednoduchost pracovat s modelem, kdy jednotlivé částečky informace jsou buď nuly nebo jedničky (tj. prvky v $\mathbb{Z}_2$) a přenášíme slova o $k$ bitech. Obdobné postupy jsou možné i nad ostatními konečnými tělesy.

# Kódování

Při přenosu informace zpravidla dochází k její deformaci. Budeme pro jednoduchost pracovat s modelem, kdy jednotlivé částečky informace jsou buď nuly nebo jedničky (tj. prvky v $\mathbb{Z}_2$) a přenášíme slova o $k$ bitech. Obdobné postupy jsou možné i nad ostatními konečnými tělesy.
Přenosové chyby chceme

1. rozpoznávat
2. opravovat

a za tím účelem přidáváme dodatečných $n - k$ bitů informace pro pevně zvolené $n > k$.

# Kódování

Při přenosu informace zpravidla dochází k její deformaci. Budeme pro jednoduchost pracovat s modelem, kdy jednotlivé částečky informace jsou buď nuly nebo jedničky (tj. prvky v $\mathbb{Z}_2$) a přenášíme slova o $k$ bitech. Obdobné postupy jsou možné i nad ostatními konečnými tělesy.
Přenosové chyby chceme

1. rozpoznávat
2. opravovat

a za tím účelem přidáváme dodatečných $n-k$ bitů informace pro pevně zvolené $n > k$.
Všech slov o $k$ bitech je $2^k$ a každé z nich má jednoznačně určovat jedno **kódové slovo** z $2^n$ možných. Máme tedy ještě

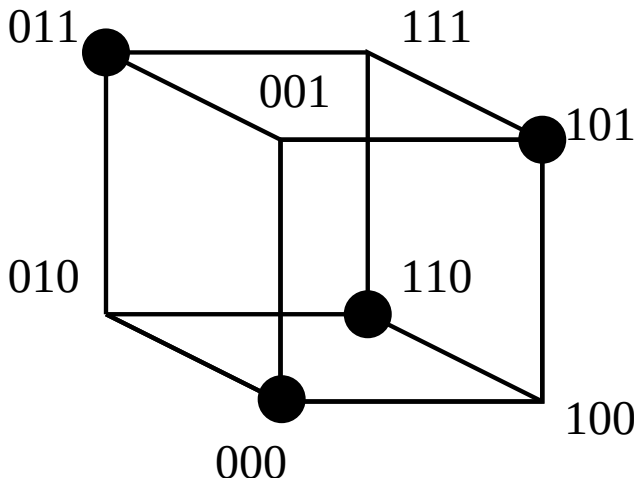
$$2^n - 2^k = 2^k(2^{n-k} - 1)$$

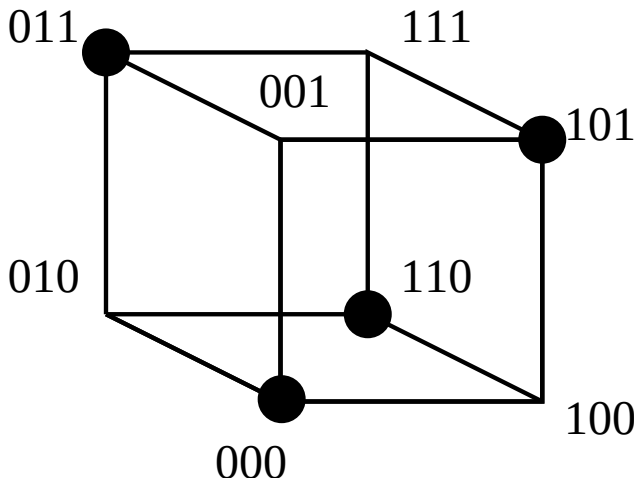
slov, která jsou chybová. Lze tedy tušit, že pro veliké $k$ nám i malý počet přidaných bitů (tj. $n-k$) dává hodně redundantní informace.

Úplně jednoduchým příkladem je **kód kontrolující paritu**. Kódové slovo o $k+1$ bitech je určené tak, aby přidáním prvního bitu byl zaručen sudý počet jedniček ve slově.

Úplně jednoduchým příkladem je **kód kontrolující paritu**. Kódové slovo o $k+1$ bitech je určené tak, aby přidáním prvního bitu byl zaručen sudý počet jedniček ve slově.
Pokud při přenosu dojde k lichému počtu chyb, přijdeme na to. Dvě různá kódová slova se při tomto kódu vždy liší alespoň ve dvou pozicích, chybové slovo se ale od dvou různých (vhodných) kódových slov liší pouze v pozici jedné. Nemůžeme proto umět chyby opravovat, ani kdybychom věděli, že došlo k právě jedné chybě.

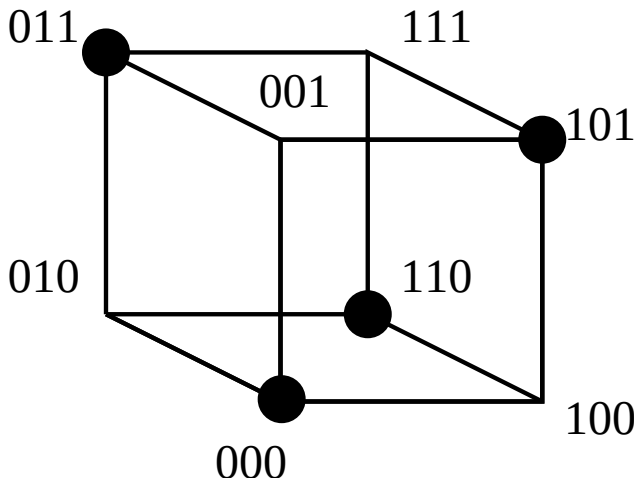
Úplně jednoduchým příkladem je **kód kontrolující paritu**. Kódové slovo o $k+1$ bitech je určené tak, aby přidáním prvního bitu byl zaručen sudý počet jedniček ve slově.
Pokud při přenosu dojde k lichému počtu chyb, přijdeme na to.
Dvě různá kódová slova se při tomto kódu vždy liší alespoň ve dvou pozicích, chybové slovo se ale od dvou různých (vhodných) kódových slov liší pouze v pozici jedné. Nemůžeme proto umět chyby opravovat, ani kdybychom věděli, že došlo k právě jedné chybě.
Navíc neumíme detekovat tak obvyklé chyby, jako je záměna dvou sousedních bitů ve slově.


011
111
001
101
010
110
100
000




## Definice

**Hammingova vzdálenost** dvou slov je rovna počtu bitů, ve kterých se liší.




## Definice

**Hammingova vzdálenost** dvou slov je rovna počtu bitů, ve kterých se liší.

## Věta

1. *Kód odhaluje $r$ a méně chyb právě, když je minimální Hammingova vzdálenost kódových slov právě $r + 1$.*
2. *Kód opravuje $r$ a méně chyb právě, když je minimální Hammingova vzdálenost kódových slov právě $2r + 1$.*

Jak konstruovat kódová slova, abychom je snadno rozpoznali? Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3, 1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.

Jak konstruovat kódová slova, abychom je snadno rozpoznali?
Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3, 1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.
Systematickou cestou je pak využití dělitelnosti polynomů. Zpráva $b_0 b_1 \dots b_{k-1}$ je reprezentována jako polynom
$m(x) = b_0 + b_1 x + \cdots + b_{k-1} x^{k-1}$.

Jak konstruovat kódová slova, abychom je snadno rozpoznali? Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3, 1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.
Systematickou cestou je pak využití dělitelnosti polynomů. Zpráva $b_0 b_1 \dots b_{k-1}$ je reprezentována jako polynom $m(x) = b_0 + b_1 x + \cdots + b_{k-1} x^{k-1}$.

**Definice**

Nechť $p(x) = a_0 + a_1 x + \cdots + a_{n-k} x^{n-k} \in \mathbb{Z}_2[x]$ je polynom s $a_0 = 1$, $a_{n-k} = 1$. **Polynomiální kód generovaný polynomem** $p(x)$ je $(n, k)$–kód jehož slova jsou polynomy stupně menšího než $n$ dělitelné $p(x)$.

Jak konstruovat kódová slova, abychom je snadno rozpoznali? Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3, 1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.
Systematickou cestou je pak využití dělitelnosti polynomů. Zpráva $b_0 b_1 \dots b_{k-1}$ je reprezentována jako polynom $m(x) = b_0 + b_1 x + \cdots + b_{k-1} x^{k-1}$.

**Definice**

Nechť $p(x) = a_0 + a_1 x + \cdots + a_{n-k} x^{n-k} \in \mathbb{Z}_2[x]$ je polynom s $a_0 = 1$, $a_{n-k} = 1$. **Polynomiální kód generovaný polynomem** $p(x)$ je $(n, k)$–kód jehož slova jsou polynomy stupně menšího než $n$ dělitelné $p(x)$.
Zpráva $m(x)$ je zakódována jako $v(x) = r(x) + x^{n-k} m(x)$, kde $r(x)$ je zbytek po dělení polynomu $x^{n-k} m(x)$ polynomem $p(x)$.

Z definice víme

$$v(x) = x^{n-k} m(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x).$$

Budou tedy všechna kódová slova dělitelná $p(x)$.

Z definice víme

$$v(x) = x^{n-k}m(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x).$$

Budou tedy všechna kódová slova dělitelná $p(x)$.
Původní zpráva je obsažena přímo v polynomu $v(x)$, takže dekódování správného slova je snadné.

Z definice víme

$$v(x) = x^{n-k}m(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x).$$

Budou tedy všechna kódová slova dělitelná $p(x)$.
Původní zpráva je obsažena přímo v polynomu $v(x)$, takže dekódování správného slova je snadné.

**Příklad**

1. Polynom $p(x) = 1 + x$ generuje $(n, n-1)$–kód kontroly parity pro všechna $n \geq 3$.
2. Polynom $p(x) = 1 + x + x^2$ generuje $(3, 1)$–kód opakování bitů.

První tvrzení plyne z toho, že $1 + x$ dělí polynom $v(x)$ tehdy a jen tehdy, když $v(1) = 0$ a to nastane tehdy, když je ve $v(x)$ sudý počet nenulových koeficientů. Druhé je zřejmé.

Přenos slova $v \in \mathbb{Z}_2^n$ dopadne příjmem polynomu

$$u(x) = v(x) + e(x)$$

kde $e(x)$ je tzv. **chybový polynom** reprezentující vektor chyby přenosu.

Přenos slova $v \in \mathbb{Z}_2^n$ dopadne příjmem polynomu

$$u(x) = v(x) + e(x)$$

kde $e(x)$ je tzv. **chybový polynom** reprezentující vektor chyby přenosu.
Chyba je rozpoznatelná pouze, když generátor kódu $p(x)$ nedělí $e(x)$. Máme proto zájem o polynomy, které nevystupují jako dělitelé zbytečně často.

Přenos slova $v \in \mathbb{Z}_2^n$ dopadne příjmem polynomu

$$u(x) = v(x) + e(x)$$

kde $e(x)$ je tzv. **chybový polynom** reprezentující vektor chyby přenosu.
Chyba je rozpoznatelná pouze, když generátor kódu $p(x)$ nedělí $e(x)$. Máme proto zájem o polynomy, které nevystupují jako dělitelé zbytečně často.

**Definice**

Ireducibilní polynom $p(x) \in \mathbb{Z}_2[x]$ stupně $m$ se nazývá **primitivní**, jestliže $p(x)$ dělí polynom $(x^k - 1)$ pro $k = 2^m - 1$ ale nedělí jej pro žádná menší $k$.

**Věta**

*Je-li $p(x)$ primitivní polynom stupně $m$, pak pro všechna $n \leq 2^m - 1$ rozpoznává příslušný $(n, n-m)$–kód všechny jednoduché a dvojité chyby.*

**Věta**

*Je-li $p(x)$ primitivní polynom stupně $m$, pak pro všechna $n \leq 2^m - 1$ rozpoznává příslušný $(n, n-m)$–kód všechny jednoduché a dvojité chyby.*

**Důsledek**

*Je-li $q(x)$ primitivní polynom stupně $m$, pak pro všechna $n \leq 2^m - 1$ rozpoznává $(n, n-m-1)$–kód generovaný polynomem $p(x) = q(x)(1+x)$ všechny dvojité chyby a všechna slova s lichým počtem chyb.*

Tabulka dává o informace o výsledcích předchozích dvou vět pro několik polynomů:

Tabulka dává o informace o výsledcích předchozích dvou vět pro několik polynomů:

| primitivní polynom | kontrolní bity | délka slova |
|---:|---:|---:|
| $1 + x + x^2$ | 2 | 3 |
| $1 + x + x^3$ | 3 | 7 |
| $1 + x + x^4$ | 4 | 15 |
| $1 + x^2 + x^5$ | 5 | 31 |
| $1 + x + x^6$ | 6 | 63 |
| $1 + x^3 + x^7$ | 7 | 127 |
| $1 + x^2 + x^3 + x^4 + x^8$ | 8 | 255 |
| $1 + x^4 + x^9$ | 9 | 511 |
| $1 + x^3 + x^{10}$ | 10 | 1023 |

**Definice**

**Lineární kód** je injektivní lineární zobrazení $g : (\mathbb{Z}_2)^k \to (\mathbb{Z}_2)^n$. Matice $G$ typu $k/n$ reprezentující toto zobrazení v standardních bazích se nazývá generující **matice kódu**.

**Definice**

**Lineární kód** je injektivní lineární zobrazení $g : (\mathbb{Z}_2)^k \to (\mathbb{Z}_2)^n$. Matice $G$ typu $k/n$ reprezentující toto zobrazení v standardních bazích se nazývá generující **matice kódu**.

Pro každé slovo $v$ je

$$u = G \cdot v$$

příslušné kódové slovo.

**Věta**

*Každý polynomiální $(n, k)$–kód je lineární kód.*

**Věta**

*Každý polynomiální $(n, k)$–kód je lineární kód.*

Generující matice $(7, 4)$ kódu příslušná k polynomu $p(x) = 1 + x^2 + x^3$ je

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

**Věta**

*Je-li $g$ lineární kódování s maticí*

$$G = \begin{pmatrix} P \\ \mathbb{E}_k \end{pmatrix},$$

*potom zobrazení $h : (\mathbb{Z}_2)^n \to (\mathbb{Z}_2)^k$ s maticí*

$$H = \begin{pmatrix} \mathbb{E}_{n-k} & P \end{pmatrix}$$

*má následující vlastnosti*

1. $\operatorname{Ker} h = \operatorname{Im} g$, *tj.*
2. *přijaté slovo $u$ je kódové slovo právě, když je $H \cdot u = 0$*

**Věta**

*Je-li $g$ lineární kódování s maticí*

$$G = \begin{pmatrix} P \\ \mathbb{E}_k \end{pmatrix},$$

*potom zobrazení $h : (\mathbb{Z}_2)^n \to (\mathbb{Z}_2)^k$ s maticí*

$$H = \begin{pmatrix} \mathbb{E}_{n-k} & P \end{pmatrix}$$

*má následující vlastnosti*

1. $\operatorname{Ker} h = \operatorname{Im} g$, *tj.*
2. *přijaté slovo $u$ je kódové slovo právě, když je $H \cdot u = 0$*

Matici $H$ z věty se říká **matice kontroly parity** příslušného $(n, k)$–kódu.

# Samoopravné kódy

Jak jsme viděli, přenos zprávy $u$ dává výsledek

$$v = u + e.$$

To je ale nad $\mathbb{Z}_2$ ekvivalentní s $e = u + v$.

# Samoopravné kódy

Jak jsme viděli, přenos zprávy $u$ dává výsledek

$$v = u + e.$$

To je ale nad $\mathbb{Z}_2$ ekvivalentní s $e = u + v$.
Pokud tedy známe podprostor $V \subset (\mathbb{Z}_2)^n$ správných kódových slov, víme u každého výsledku, že správné slovo (s opravenými případnými chybami) je ve třídě rozkladu $v + V$ v prostoru $(\mathbb{Z}_2)^n / V$.

# Samoopravné kódy

Jak jsme viděli, přenos zprávy $u$ dává výsledek

$$v = u + e.$$

To je ale nad $\mathbb{Z}_2$ ekvivalentní s $e = u + v$.
Pokud tedy známe podprostor $V \subset (\mathbb{Z}_2)^n$ správných kódových slov, víme u každého výsledku, že správné slovo (s opravenými případnými chybami) je ve třídě rozkladu $v + V$ v prostoru $(\mathbb{Z}_2)^n/V$.
Zobrazení (homomorfismus grup) $h : (\mathbb{Z}_2)^n \to (\mathbb{Z}_2)^{n-k}$ má $V$ za jádro, proto indukuje injektivní lineární zobrazení $h : (\mathbb{Z}_2)^n/V \to (\mathbb{Z}_2)^{n-k}$. Jeho hodnoty jsou jednoznačně určeny hodnotami $H \cdot u$.

**Definice**

Hodnota $H \cdot u$ se nazývá **syndrom** slova $u$.

**Definice**

Hodnota $H \cdot u$ se nazývá **syndrom** slova $u$.

**Věta**

*Dvě slova jsou ve stejné třídě rozkladu právě tehdy, když mají týž syndrom.*

Samoopravné kódy lze konstruovat tak, že pro každý syndrom určíme prvek v příslušné třídě, který je nejvhodnějším slovem.

## Příklad

Buď dán $(6, 3)$ kód nad $\mathbb{Z}_2$ generovaný polynomem $x^3 + x^2 + 1$.

1. Určete jeho generující matici a matici kontroly parity.
2. Dekódujte zprávu 111101 předpokládáte-li, že při přenosu došlo k nejmenšímu možnému počtu chyb.

## Příklad

Buď dán $(6, 3)$ kód nad $\mathbb{Z}_2$ generovaný polynomem $x^3 + x^2 + 1$.

1. Určete jeho generující matici a matici kontroly parity.
2. Dekódujte zprávu 111101 předpokládáte-li, že při přenosu došlo k nejmenšímu možnému počtu chyb.

## Řešení

Protože se jedná o lineární kód, stačí určit jak se zakódují bázové vektory 1, $x$ a $x^2$, tedy určit zbytky polynomů $x^3$, $x^4$ a $x^5$ po dělení polynomem $x^3 + x^2 + 1$. Máme

$$\begin{aligned} x^3 &\equiv x^2 + 1 \pmod{x^3 + x^2 + 1} \\ x^4 &= x(x^3) \equiv x(x^2 + 1) \equiv x^2 + x + 1 \pmod{x^3 + x^2 + 1} \\ x^5 &= x(x^4) \equiv x(x^2 + x + 1) \equiv x + 1 \pmod{x^3 + x^2 + 1} \end{aligned}$$

## Řešení (pokr.)

Bázové vektory (zprávy) 100, 010 a 001 se tedy zakódují do vektorů (kódů) 101100, 111010 a 110001, generující matice, resp. matice kontroly parity kódu jsou tedy

$$G = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{resp.} \quad \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

## Řešení (pokr.)

Bázové vektory (zprávy) 100, 010 a 001 se tedy zakódují do vektorů (kódů) 101100, 111010 a 110001, generující matice, resp. matice kontroly parity kódu jsou tedy

$$G = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{resp.} \quad \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

Vynásobíme-li přijatou zprávu 111101 maticí kontroly parity, dostáváme syndrom 100 a víme, že při přenosu došlo k chybě. Sestavme tabulku všech syndromů a jim odpovídajících kódových slov.

| Syndrom | Kódová slova s daným syndromem | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 000 | **000000** | 110001 | 111010 | 101100 | 010110 | 001011 | 011101 | 100111 |
| 001 | **001000** | 111001 | 110010 | 100100 | 011110 | 000011 | 010101 | 101111 |
| 010 | **010000** | 100001 | 101010 | 111100 | 000110 | 011011 | 001101 | 110111 |
| 100 | **100000** | 010001 | 011010 | 001100 | 110110 | 101011 | 111101 | 000111 |
| 011 | **011000** | 101001 | 100010 | 110100 | 001110 | 010011 | **000101** | 111111 |
| 101 | 101000 | 011001 | 010010 | **000100** | 111110 | 100011 | 110101 | 001111 |
| 110 | 110000 | **000001** | 001010 | 011100 | 100110 | 111011 | 101101 | 010111 |
| 111 | 111000 | 001001 | **000010** | 010100 | 101110 | 110011 | 100101 | 011111 |

## Řešení (dokončení)

Syndrom 000 mají všechna kódová slova. Počínaje druhým řádkem, je každý řádek tabulky afinním prostorem jehož zaměřením je vektorový prostor daný prvním řádkem. Zejména je tedy rozdíl každých dvou slov ve stejném řádku nějakým kódovým slovem. Všechna slova s daným syndromem dostaneme přičtením syndromu (doplněného nulami na délku kódového slova) ke všem kódovým slovům. Tzv. vedoucím reprezentantem třídy (řádku, afinního prostoru) odpovídající danému syndromu je slovo s nejmenším počtem jedniček v řádku, a tedy i nejmenším počtem bitových změn, jež je třeba provést, abychom dostali kódové slovo – v našem případě jde o slovo 100000 a jeho odečtením od obdrženého slova dostaneme platné kódové slovo 011101. Je to platné kódové slovo s nejmenší Hammingovou vzdáleností od obdrženého slova . Odeslaná zpráva tedy byla 101.

# Plán přednášky

1 Úvod do kódování
- $(n, k)$–kódy
- Lineární kódy

2 Pár slov o šifrách

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$
- generování klíčů: zvolí dvě velká prvočísla $p, q$, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [$n$ je veřejné, ale $\varphi(n)$ nelze snadno spočítat ]

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$
- generování klíčů: zvolí dvě velká prvočísla $p, q$, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [$n$ je veřejné, ale $\varphi(n)$ nelze snadno spočítat ]
- zvolí **veřejný klíč** $e$ a ověří, že $(e, \varphi(n)) = 1$

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$
- generování klíčů: zvolí dvě velká prvočísla $p, q$, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [$n$ je veřejné, ale $\varphi(n)$ nelze snadno spočítat ]
- zvolí **veřejný klíč** $e$ a ověří, že $(e, \varphi(n)) = 1$
- např. pomocí Euklidova algoritmu spočítá **tajný klíč** $d$ tak, aby $e \cdot d \equiv 1 \pmod{\varphi(n)}$

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$
- generování klíčů: zvolí dvě velká prvočísla $p, q$, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [$n$ je veřejné, ale $\varphi(n)$ nelze snadno spočítat ]
- zvolí **veřejný klíč** $e$ a ověří, že $(e, \varphi(n)) = 1$
- např. pomocí Euklidova algoritmu spočítá **tajný klíč** $d$ tak, aby $e \cdot d \equiv 1 \pmod{\varphi(n)}$
- zašifrování numerického *kódu* zprávy $M$: $C = C_e(M) \equiv M^e \pmod{n}$

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$
- generování klíčů: zvolí dvě velká prvočísla $p, q$, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [$n$ je veřejné, ale $\varphi(n)$ nelze snadno spočítat ]
- zvolí **veřejný klíč** $e$ a ověří, že $(e, \varphi(n)) = 1$
- např. pomocí Euklidova algoritmu spočítá **tajný klíč** $d$ tak, aby $e \cdot d \equiv 1 \pmod{\varphi(n)}$
- zašifrování numerického *kódu* zprávy $M$: $C = C_e(M) \equiv M^e \pmod{n}$
- dešifrování šifry $C$: $OT = D_d(C) \equiv C^d \pmod{n}$

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$
- generování klíčů: zvolí dvě velká prvočísla $p, q$, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [$n$ je veřejné, ale $\varphi(n)$ nelze snadno spočítat ]
- zvolí **veřejný klíč** $e$ a ověří, že $(e, \varphi(n)) = 1$
- např. pomocí Euklidova algoritmu spočítá **tajný klíč** $d$ tak, aby $e \cdot d \equiv 1 \pmod{\varphi(n)}$
- zašifrování numerického *kódu* zprávy $M$: $C = C_e(M) \equiv M^e \pmod{n}$
- dešifrování šifry $C$: $OT = D_d(C) \equiv C^d \pmod{n}$

# RSA

*Ron Rivest, Adi Shamir, Leonard Adleman* (1977; C. Cocks,GCHQ – 1973)

- každý účastník $A$ potřebuje dvojici klíčů – veřejný $V_A$ a soukromý $S_A$
- generování klíčů: zvolí dvě velká prvočísla $p, q$, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [$n$ je veřejné, ale $\varphi(n)$ nelze snadno spočítat ]
- zvolí **veřejný klíč** $e$ a ověří, že $(e, \varphi(n)) = 1$
- např. pomocí Euklidova algoritmu spočítá **tajný klíč** $d$ tak, aby $e \cdot d \equiv 1 \pmod{\varphi(n)}$
- zašifrování numerického *kódu* zprávy $M$: $C = C_e(M) \equiv M^e \pmod{n}$
- dešifrování šifry $C$: $OT = D_d(C) \equiv C^d \pmod{n}$

**Důkaz.**

Fermatova, resp. Eulerova věta. □

## Poznámka

- Korektní naprogramování bez postranních kanálů není triviální (viz např. PKCS#1, RFC 3447).
- Analogicky podepisování (hashů) zpráv (viz např. DSA)
- Viz RSA factoring challenge (např. rozklad 212 ciferného čísla RSA-704 vynese 30 000 USD).

## Příklad

- **Generování klíče.** Alice vybere prvočísla $p = 2357, q = 2551$, and vypočte $n = p \cdot q = 6012707$ a $\varphi(n) = (p-1)(q-1) = 6007800$. Alice zvolí $e = 3674911$ a pomocí Euklidova algoritmu vypočte $d = 422191$ ($e \cdot d \equiv 1 \pmod{\varphi(n)}$). Soukromý klíč Alice je $d$, veřejný pak $(n, e)$.

## Příklad

- **Generování klíče.** Alice vybere prvočísla $p = 2357, q = 2551$, and vypočte $n = p \cdot q = 6012707$ a $\varphi(n) = (p-1)(q-1) = 6007800$. Alice zvolí $e = 3674911$ a pomocí Euklidova algoritmu vypočte $d = 422191$ ($e \cdot d \equiv 1 \pmod{\varphi(n)}$). Soukromý klíč Alice je $d$, veřejný pak $(n, e)$.
- Chce-li Bob poslat Alici zprávu $m = 5234673$, pomocí modulárního umocňování vypočte

$$c \equiv m^e \equiv 5234673^{3674911} \equiv 3650502 \pmod{n},$$

a tu odešle Alici.

## Příklad

- **Generování klíče.** Alice vybere prvočísla $p = 2357, q = 2551$, and vypočte $n = p \cdot q = 6012707$ a $\varphi(n) = (p-1)(q-1) = 6007800$. Alice zvolí $e = 3674911$ a pomocí Euklidova algoritmu vypočte $d = 422191$ ($e \cdot d \equiv 1 \pmod{\varphi(n)}$). Soukromý klíč Alice je $d$, veřejný pak $(n, e)$.
- Chce-li Bob poslat Alici zprávu $m = 5234673$, pomocí modulárního umocňování vypočte

$$c \equiv m^e \equiv 5234673^{3674911} \equiv 3650502 \pmod{n},$$

  a tu odešle Alici.
- Alice zprávu dešifruje díky výpočtu

$$c^d \equiv 3650502^{422191} \equiv 5234673.$$

# Diffie-Hellman key exchange, ElGamal

*Whitfield Diffie, Martin Hellman* (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýru s kufříky, . . . ).

# Diffie-Hellman key exchange, ElGamal

*Whitfield Diffie, Martin Hellman* (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýru s kufříky, . . . ).

- Dohoda stran na **cyklické grupě** $G$ a jejím generátoru $g$ (veřejné)

# Diffie-Hellman key exchange, ElGamal

*Whitfield Diffie, Martin Hellman* (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýru s kufříky, . . . ).

- Dohoda stran na **cyklické grupě** $G$ a jejím generátoru $g$ (veřejné)
- Alice vybere náhodné $a$ a pošle $g^a$

# Diffie-Hellman key exchange, ElGamal

*Whitfield Diffie, Martin Hellman* (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýru s kufříky, . . . ).

- Dohoda stran na **cyklické grupě** $G$ a jejím generátoru $g$ (veřejné)
- Alice vybere náhodné $a$ a pošle $g^a$
- Bob vybere náhodné $b$ a pošle $g^b$

# Diffie-Hellman key exchange, ElGamal

*Whitfield Diffie, Martin Hellman* (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýru s kufříky, . . . ).

- Dohoda stran na **cyklické grupě** $G$ a jejím generátoru $g$ (veřejné)
- Alice vybere náhodné $a$ a pošle $g^a$
- Bob vybere náhodné $b$ a pošle $g^b$
- Společným klíčem pro komunikaci je $g^{ab}$.

# Diffie-Hellman key exchange, ElGamal

*Whitfield Diffie, Martin Hellman* (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýru s kufříky, . . . ).

- Dohoda stran na **cyklické grupě** $G$ a jejím generátoru $g$ (veřejné)
- Alice vybere náhodné $a$ a pošle $g^a$
- Bob vybere náhodné $b$ a pošle $g^b$
- Společným klíčem pro komunikaci je $g^{ab}$.

# Diffie-Hellman key exchange, ElGamal

*Whitfield Diffie, Martin Hellman* (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýru s kufříky, . . . ).

- Dohoda stran na **cyklické grupě** $G$ a jejím generátoru $g$ (veřejné)
- Alice vybere náhodné $a$ a pošle $g^a$
- Bob vybere náhodné $b$ a pošle $g^b$
- Společným klíčem pro komunikaci je $g^{ab}$.

**Poznámka**

- Problém diskrétního logaritmu (DLP)
- Nezbytná autentizace (*man in the middle attack*)

Z protokolu DH na výměnu klíčů odvozen šifrovací algoritmus ElGamal:

- Alice zvolí cyklickou grupu $G$ spolu s generátorem $g$
- Alice zvolí **tajný klíč** $x$, spočítá $h = g^x$ a zveřejní **veřejný klíč** $(G, g, h)$
- šifrování zprávy $M$: Bob zvolí náhodné $y$ a vypočte $C_1 = g^y$ a $C_2 = M \cdot h^y$ a pošle $(C_1, C_2)$
- dešifrování zprávy: $OT = C_2 / C_1^x$

Z protokolu DH na výměnu klíčů odvozen šifrovací algoritmus ElGamal:

- Alice zvolí cyklickou grupu $G$ spolu s generátorem $g$
- Alice zvolí **tajný klíč** $x$, spočítá $h = g^x$ a zveřejní **veřejný klíč** $(G, g, h)$
- šifrování zprávy $M$: Bob zvolí náhodné $y$ a vypočte $C_1 = g^y$ a $C_2 = M \cdot h^y$ a pošle $(C_1, C_2)$
- dešifrování zprávy: $OT = C_2 / C_1^x$

**Poznámka**

Opět lze odvodit podepisování.

# Eliptické křivky

Eliptické křivky jsou rovinné křivky o rovnici tvaru $y^2 = x^3 + ax + b$ a zajímavé jsou tím, že na jejich bodech lze definovat operace tak, že výslednou strukturou bude komutativní grupa.
Přitom uvedené operace lze efektivně provádět a navíc se ukazuje, že mají (nejen) pro kryprografii zajímavé vlastnosti – srovnatelné bezpečnosti jako RSA lze dosáhnout již s podstatně kratšími klíči. Výhodou je rovněž velké množství použitelných eliptických křivek (a tedy grup různé struktury) podle volby parametru $a, b$ .

# Eliptické křivky

Eliptické křivky jsou rovinné křivky o rovnici tvaru $y^2 = x^3 + ax + b$ a zajímavé jsou tím, že na jejich bodech lze definovat operace tak, že výslednou strukturou bude komutativní grupa.
Přitom uvedené operace lze efektivně provádět a navíc se ukazuje, že mají (nejen) pro kryprografii zajímavé vlastnosti – srovnatelné bezpečnosti jako RSA lze dosáhnout již s podstatně kratšími klíči.
Výhodou je rovněž velké množství použitelných eliptických křivek (a tedy grup různé struktury) podle volby parametru $a, b$ .
Protokoly:

- ECDH - přímá varianta DH na eliptické křívce (jen místo generátoru se vybere *vhodný* bod na křivce)
- ECDSA - digitální podpis pomocí eliptických křivek.

# Eliptické křivky

Eliptické křivky jsou rovinné křivky o rovnici tvaru $y^2 = x^3 + ax + b$ a zajímavé jsou tím, že na jejich bodech lze definovat operace tak, že výslednou strukturou bude komutativní grupa.
Přitom uvedené operace lze efektivně provádět a navíc se ukazuje, že mají (nejen) pro kryprografii zajímavé vlastnosti – srovnatelné bezpečnosti jako RSA lze dosáhnout již s podstatně kratšími klíči.
Výhodou je rovněž velké množství použitelných eliptických křivek (a tedy grup různé struktury) podle volby parametru $a, b$ .
Protokoly:

- ECDH - přímá varianta DH na eliptické křívce (jen místo generátoru se vybere *vhodný* bod na křivce)
- ECDSA - digitální podpis pomocí eliptických křivek.

**Poznámka**

Problém diskrétního logaritmu (ECDLP).
Navíc se ukazuje, že eliptické křivky jsou velmi dobře použitelné při faktorizaci prvočísel.