Diskrétní matematika – 5. týden
Aplikace teorie čísel – Počítání s velkými čísly, kryptografie

Lukáš Vokřínek

Masarykova univerzita
Fakulta informatiky

podzim 2020

Obsah přednášky

1. Diofantické rovnice — „nové" oproti MB104 – bez teorie
2. ~~Výpočetní aspekty teorie čísel~~
3. Kryptografie s veřejným klíčem — v minulém semestru komentovaná cvičení

Doporučené zdroje

- Jan Slovák, Martin Panák, Michal Bulant
  Matematika drsně a svižně, *e*-text na
  www.math.muni.cz/Matematika_drsne_svizne.

Doporučené zdroje

- Jan Slovák, Martin Panák, Michal Bulant
  Matematika drsně a svižně, *e*-text na
  `www.math.muni.cz/Matematika_drsne_svizne`.
- V. Švábenský, **Sbírka příkladů** (a další zdroje),
  `https://is.muni.cz/auth/th/395868/fi_b/`
- Jiří Herman, Radan Kučera, Jaromír Šimša, **Metody řešení matematických úloh**. MU Brno, 2001.
- William Stein, **Elementary Number Theory: Primes, Congruences, and Secrets**, Springer, 2008. Dostupné na
  `http://wstein.org/ent/ent.pdf`

Plán přednášky

Diofantické rovnice

Příklad

Vyřešte diofantickou rovnici $x, y \in \mathbb{Z}$ nebo $\mathbb{N}$

$$72x + 100y = 16.$$

$72x + 100y = 16$

[řešitelné vzhledem k x ⟺ „zbytek" $72x = 16 - 100y$ je dělitelné 72 a řešení je pak jediné

⇝ přejdeme ke kongruenci mod 72

$$72x + 100y \equiv 16 \pmod{72}$$

$$72y \equiv 0$$

$$28y \equiv 16$$

$$16y \equiv -32$$

$$12y \equiv 48$$

$$4y \equiv -80 \equiv -8 \quad (72) \qquad \underline{\underline{y \equiv -2 \quad (18)}}$$

$$0y \equiv 72 \equiv 0$$

$$72x + 100y = 16$$

$$y \equiv -2 \quad (18)$$

$$\boxed{y = -2 + 18t}$$

$$72x + 100(-2 + 18t) = 16$$

$$72x - 200 + 1800t = 16$$

$$72x = -216 - 1800t$$

$$\boxed{x = -3 - 25t}$$

$\Rightarrow$ řešení $(x,y) = (-3-25t, -2+18t)$

$= (-3,-2) + t \cdot (-25, 18)$

$\Rightarrow$ řešení $x, y \in \mathbb{N}$ neex.

Diofantické rovnice

Příklad

Vyřešte diofantickou rovnici

$$72x + 100y = 16.$$

Příklad

Vyřešte diofantickou rovnici

$$72x + 100y + 45z = 1.$$

$$72x + 100y + 45z = 1$$

$\rightsquigarrow$ modulo nejv. spol. děl $(72, 100) = 4$

$$72x + 100y + 45z \equiv 1 \quad (4)$$

$$z \equiv 1 \quad (4) \longrightarrow z = 1 + 4t$$

$$72x + 100y + 45 + 180t = 1$$

$$72x + 100y = -44 - 180t$$

$\rightsquigarrow$ modulo 72:

$$72x + 100y \equiv -44 - 180t \quad (72)$$

$$28y \equiv 28 - 36t \quad (72)$$

$72y \equiv 0$

$28y \equiv 28 - 36t \quad (72)$

$16y \equiv 16$

$12y \equiv 12 - 36t$

$4y \equiv 4 + 36t \quad (72)$

$\Rightarrow y \equiv 1 + 9t \quad (18)$

$\underline{y = 1 + 9t + 18s}$

$\underline{z = 1 + 4t}$

$72x + 100y + 45z = 1$

$72x + 100(1 + 9t + 18s) + 45(1 + 4t) = 1$

$72x + 145 + 1080t + 1800s = 1$

$72x = -144 - 1080t - 1800s$

$\underline{x = -2 - 15t - 25s}$

$(x, y, z) =$
$= (-2, 1, 1)$
$+ t(-15, 9, 4)$
$+ s(-25, 18, 0)$

Plán přednášky

Základní úlohy výpočetní teorie čísel

V mnoha praktických úlohách využívajících výsledky teorie čísel je zapotřebí umět rychle provést jeden či více z následujících výpočtů:

1. běžné aritmetické operace (součet, součin, dělení se zbytkem) na celých číslech,

Základní úlohy výpočetní teorie čísel

V mnoha praktických úlohách využívajících výsledky teorie čísel je zapotřebí umět rychle provést jeden či více z následujících výpočtů:

1. běžné aritmetické operace (součet, součin, dělení se zbytkem) na celých číslech,
2. zbytek mocniny celého čísla a na přirozené číslo n po dělení daným m.

Základní úlohy výpočetní teorie čísel

V mnoha praktických úlohách využívajících výsledky teorie čísel je zapotřebí umět rychle provést jeden či více z následujících výpočtů:

1. běžné aritmetické operace (součet, součin, dělení se zbytkem) na celých číslech,
2. zbytek mocniny celého čísla a na přirozené číslo n po dělení daným m.
3. inverzi celého čísla a modulo $m \in \mathbb{N}$,

Základní úlohy výpočetní teorie čísel

V mnoha praktických úlohách využívajících výsledky teorie čísel je zapotřebí umět rychle provést jeden či více z následujících výpočtů:

1. běžné aritmetické operace (součet, součin, dělení se zbytkem) na celých číslech,
2. zbytek mocniny celého čísla a na přirozené číslo n po dělení daným m.
3. inverzi celého čísla a modulo $m \in \mathbb{N}$,
4. největší společný dělitel dvou celých čísel (a případně koeficienty do Bezoutovy rovnosti),

Základní úlohy výpočetní teorie čísel

V mnoha praktických úlohách využívajících výsledky teorie čísel je zapotřebí umět rychle provést jeden či více z následujících výpočtů:

1. běžné aritmetické operace (součet, součin, dělení se zbytkem) na celých číslech,
2. zbytek mocniny celého čísla a na přirozené číslo n po dělení daným m.
3. inverzi celého čísla a modulo $m \in \mathbb{N}$,
4. největší společný dělitel dvou celých čísel (a případně koeficienty do Bezoutovy rovnosti),
5. rozhodnout o daném čísle, je-li prvočíslo nebo složené,

Základní úlohy výpočetní teorie čísel

V mnoha praktických úlohách využívajících výsledky teorie čísel je zapotřebí umět rychle provést jeden či více z následujících výpočtů:

1. běžné aritmetické operace (součet, součin, dělení se zbytkem) na celých číslech,
2. zbytek mocniny celého čísla a na přirozené číslo n po dělení daným m.
3. inverzi celého čísla a modulo $m \in \mathbb{N}$,
4. největší společný dělitel dvou celých čísel (a případně koeficienty do Bezoutovy rovnosti),
5. rozhodnout o daném čísle, je-li prvočíslo nebo složené,
6. v případě složenosti rozložit dané číslo na součin prvočísel.

Základní aritmetické operace

Základní aritmetické operace se i na velkých číslech obvykle provádějí obdobně jako jsme se to učili na základní a střední škole, kdy umíme sčítat v *lineárním*, násobit a dělit se zbytkem v *kvadratickém* čase.

Základní aritmetické operace

Základní aritmetické operace se i na velkých číslech obvykle provádějí obdobně jako jsme se to učili na základní a střední škole, kdy umíme sčítat v *lineárním*, násobit a dělit se zbytkem v *kvadratickém* čase. Pro **násobení**, které je základem mnoha dalších operací, existují asymptoticky rychlejší algoritmy (typu *rozděl a panuj*) - např. první takový Karatsubův (1960) časové náročnosti $\Theta(n^{\log_2 3})$

Základní aritmetické operace

Základní aritmetické operace se i na velkých číslech obvykle provádějí obdobně jako jsme se to učili na základní a střední škole, kdy umíme sčítat v *lineárním*, násobit a dělit se zbytkem v *kvadratickém* čase. Pro **násobení**, které je základem mnoha dalších operací, existují asymptoticky rychlejší algoritmy (typu *rozděl a panuj*) - např. první takový Karatsubův (1960) časové náročnosti $\Theta(n^{\log_2 3})$ nebo algoritmus Schönhage-Strassenův (1971) časové náročnosti $\Theta(n \log n \log \log n)$, který využívá tzv. Fast Fourier Transform. Ten je ale přes svou asymptotickou převahu výhodný až pro násobení čísel majících alespoň desítky tisíc cifer (a používá se tak např. v GIMPS).

Základní aritmetické operace

Základní aritmetické operace se i na velkých číslech obvykle provádějí obdobně jako jsme se to učili na základní a střední škole, kdy umíme sčítat v *lineárním*, násobit a dělit se zbytkem v *kvadratickém* čase. Pro **násobení**, které je základem mnoha dalších operací, existují asymptoticky rychlejší algoritmy (typu *rozděl a panuj*) - např. první takový Karatsubův (1960) časové náročnosti $\Theta(n^{\log_2 3})$ nebo algoritmus Schönhage-Strassenův (1971) časové náročnosti $\Theta(n \log n \log \log n)$, který využívá tzv. Fast Fourier Transform. Ten je ale přes svou asymptotickou převahu výhodný až pro násobení čísel majících alespoň desítky tisíc cifer (a používá se tak např. v GIMPS).

Pěkný přehled je např. na `http://en.wikipedia.org/wiki/Computational_complexity_of_mathematical_operations`

GCD a modulární inverze

Jak už jsme ukazovali dříve, výpočet řešení kongruence $a \cdot x \equiv 1 \pmod{m}$ s neznámou x lze snadno (díky Bezoutově větě) převést na výpočet největšího společného dělitele čísel a a m a na hledání koeficientů k, l do Bezoutovy rovnosti $k \cdot a + l \cdot m = 1$ (nalezené k je pak onou hledanou inverzí a modulo m).

GCD a modulární inverze

Jak už jsme ukazovali dříve, výpočet řešení kongruence $a \cdot x \equiv 1 \pmod{m}$ s neznámou x lze snadno (díky Bezoutově větě) převést na výpočet největšího společného dělitele čísel a a m a na hledání koeficientů k, l do Bezoutovy rovnosti $k \cdot a + l \cdot m = 1$ (nalezené k je pak onou hledanou inverzí a modulo m).

```
function extended_gcd(a, m)
    if m == 0
        return (1, 0)
    else
        (q, r) := divide (a, m)
        (k, l) := extended_gcd(m, r)
        return (l, k - q * l)
```

Podrobná analýza (viz např. [Knuth] nebo [Wiki]) ukazuje, že tento algoritmus je **kvadratické** časové složitosti.

Modulární umocňování

Modulární umocňování je, jak jsme již viděli dříve, velmi využívaná operace mj. při ověřování, zda je dané číslo prvočíslo nebo číslo složené. Jedním z efektivních algoritmů je tzv. **modulární umocňování zprava doleva**:

Modulární umocňování

Modulární umocňování je, jak jsme již viděli dříve, velmi využívaná operace mj. při ověřování, zda je dané číslo prvočíslo nebo číslo složené. Jedním z efektivních algoritmů je tzv. **modulární umocňování zprava doleva**:

```
function modular_pow(base, exponent, modulus)
    result := 1
    while exponent > 0
      if (exponent mod 2 == 1):
        result := (result * base) mod modulus
      exponent := exponent >> 1
      base = (base * base) mod modulus
    return result
```

Algoritmus modulárního umocňování je založen na myšlence, že např. při počítání $2^{64} \pmod{1000}$

- není třeba nejprve počítat 2^{64} a poté jej vydělit se zbytkem číslem 1000, ale lépe je postupně násobit „dvojky“ a kdykoliv je výsledek větší než 1000, provést redukci modulo 1000, Algoritmus modulárního umocňování je založen na myšlence, že např. při počítání $2^{64} \pmod{1000}$

- není třeba nejprve počítat 2^{64} a poté jej vydělit se zbytkem číslem 1000, ale lépe je postupně násobit „dvojky“ a kdykoliv je výsledek větší než 1000, provést redukci modulo 1000,
- ale zejména, že není třeba provádět takové množství násobení (v tomto případě 63 naivních násobení je možné nahradit pouze šesti umocněními na druhou, neboť

$$2^{64} = (((((2^2)^2)^2)^2)^2)^2.$$

Příklad (Ukázka průběhu algoritmu)

Vypočtěme $2^{560} \pmod{561}$.

Příklad (Ukázka průběhu algoritmu)

Vypočtěme $2^{560} \pmod{561}$. Protože $560 = (1000110000)_2$, dostaneme uvedeným algoritmem

$2^{560} = (((2^2)^2)^2)^3)^5$

exponent	base	result	exp's last digit
560	2	1	0
280	4	1	0
140	16	1	0
70	256	1	0
35	460	1	1
17	103	460	1
8	511	256	0
4	256	256	0
2	460	256	0
1	103	256	1
0	511	1	0

Příklad (Ukázka průběhu algoritmu)

Vypočtěme $2^{560} \pmod{561}$. Protože $560 = (1000110000)_2$, dostaneme uvedeným algoritmem

exponent	base	result	exp's last digit
560	2	1	0
280	4	1	0
140	16	1	0
70	256	1	0
35	460	1	1
17	103	460	1
8	511	256	0
4	256	256	0
2	460	256	0
1	103	256	1
0	511	1	0

A tedy $2^{560} \equiv 1 \pmod{561}$.

Efektivita modulárního umocňování

V průběhu algoritmu se pro každou binární číslici exponentu provede umocnění základu na druhou modulo n (což je operace proveditelná v nejhůře kvadratickém čase), a pro každou „jedničku" v binárním zápisu navíc provede jedno násobení. Celkově jsme tedy schopni provést modulární umocňování nejhůře v **kubickém** čase.

Testování prvočíselnosti, rozklad složených čísel

Toto je téma na samostatnou přednášku, nebudeme zde uvádět, v učebnici lze mnohé najít v odstavcích 10.38-47.

Plán přednášky

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

A $M \to C$ - - - - B $C \to M$

K

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

$$\begin{tikzcd}
A & & B \\
M \arrow[d] & & M \arrow[ll] \arrow[d] \\
H_M = C & & H_M \arrow[d] \\
N \arrow[u] & & \text{Dešifruje} \arrow[ll]
\end{tikzcd}$$

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

Nejčastěji používané systémy PKC:

- RSA (šifrování) a odvozený systém pro podepisování zpráv

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

Nejčastěji používané systémy PKC:

- RSA (šifrování) a odvozený systém pro podepisování zpráv
- Digital signature algorithm (DSA) a varianta založená na eliptických křivkách (ECDSA)

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

Nejčastěji používané systémy PKC:

- RSA (šifrování) a odvozený systém pro podepisování zpráv
- Digital signature algorithm (DSA) a varianta založená na eliptických křivkách (ECDSA)
- Rabinův kryptosystém (a podepisování)

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

Nejčastěji používané systémy PKC:

- RSA (šifrování) a odvozený systém pro podepisování zpráv
- Digital signature algorithm (DSA) a varianta založená na eliptických křivkách (ECDSA)
- Rabinův kryptosystém (a podepisování)
- ElGamal kryptosystém (a podepisování)

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

Nejčastěji používané systémy PKC:

- RSA (šifrování) a odvozený systém pro podepisování zpráv
- Digital signature algorithm (DSA) a varianta založená na eliptických křivkách (ECDSA)
- Rabinův kryptosystém (a podepisování)
- ElGamal kryptosystém (a podepisování)
- Kryptografie eliptických křivek (ECC)

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

Nejčastěji používané systémy PKC:

- RSA (šifrování) a odvozený systém pro podepisování zpráv
- Digital signature algorithm (DSA) a varianta založená na eliptických křivkách (ECDSA)
- Rabinův kryptosystém (a podepisování)
- ElGamal kryptosystém (a podepisování)
- Kryptografie eliptických křivek (ECC)
- Diffie-Hellmanův protokol na výměnu klíčů (DH)

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]

(veřejný) součin prvočísel — veřejná!

$n = p \cdot q$

$$\varphi(n) = (p-1)(q-1)$$
$$= pq + 1 - (p+q)$$
$$= n + 1 - (p+q)$$

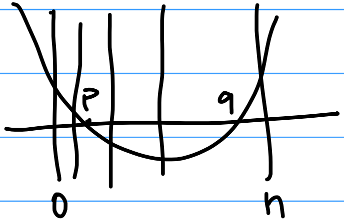
$$n + 1 - \varphi(n) = p + q$$

$n = p \cdot q \qquad n+1-\varphi(n) = p+q$

p, q jsou kořeny polynomu

$$(x-p)(x-q) = x^2 - (p+q)\cdot x + pq$$
$$= x^2 - (n+1-\varphi(n)) + n$$

$\rightarrow$ kořeny lze spočítat $\qquad \dfrac{-b \pm \sqrt{b^2-4ac}}{2a}$



$\log n$

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 $(e, \varphi(n)) = 1$

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 $e \cdot d \equiv 1 \pmod{\varphi(n)}$

šifrování

dešifrování

$(M^e)^d \equiv M^{e \cdot d} \equiv M \quad (n)$

$M \ (n) \longrightarrow M^e \ (n)$

$M^e \equiv C \ (n)$

$C^d \ (n)$

čísla vody

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: $OT = D_d(C) \equiv C^d \pmod{n}$

příklady

Rabinův kryptosystém

Prvním veřejným kryptosystémem, k jehož prolomení je prokazatelně potřeba faktorizovat modul, je **Rabinův kryptosystém**, který si uvedeme ve zjednodušené verzi:

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

$n = p \cdot q$ $\qquad$ $p, q \equiv 3 \ (4)$

$M \pmod n$ $\qquad$ $C \pmod n$

$$M \longmapsto M^2$$
$$\sqrt{C} \longleftarrow C$$

čtyři

Rabinův kryptosystém

Prvním veřejným kryptosystémem, k jehož prolomení je prokazatelně potřeba faktorizovat modul, je **Rabinův kryptosystém**, který si uvedeme ve zjednodušené verzi:

- každý účastník A potřebuje dvojici klíčů – veřejný V_A a soukromý S_A
- generování klíčů: A zvolí dvě podobně velká prvočísla $p, q \equiv 3 \pmod 4$, vypočte $n = pq$.

Rabinův kryptosystém

Prvním veřejným kryptosystémem, k jehož prolomení je prokazatelně potřeba faktorizovat modul, je **Rabinův kryptosystém**, který si uvedeme ve zjednodušené verzi:

- každý účastník A potřebuje dvojici klíčů – veřejný V_A a soukromý S_A
- generování klíčů: A zvolí dvě podobně velká prvočísla $p, q \equiv 3 \pmod 4$, vypočte $n = pq$.
- $V_A = n$, $S_A = (p, q)$

Rabinův kryptosystém

Prvním veřejným kryptosystémem, k jehož prolomení je prokazatelně potřeba faktorizovat modul, je **Rabinův kryptosystém**, který si uvedeme ve zjednodušené verzi:

- každý účastník A potřebuje dvojici klíčů – veřejný V_A a soukromý S_A
- generování klíčů: A zvolí dvě podobně velká prvočísla $p, q \equiv 3 \pmod 4$, vypočte $n = pq$.
- $V_A = n$, $S_A = (p, q)$
- zašifrování numerického *kódu* zprávy M: $C = C_e(M) \equiv M^2 \pmod n$

Rabinův kryptosystém

Prvním veřejným kryptosystémem, k jehož prolomení je prokazatelně potřeba faktorizovat modul, je **Rabinův kryptosystém**, který si uvedeme ve zjednodušené verzi:

- každý účastník A potřebuje dvojici klíčů – veřejný V_A a soukromý S_A
- generování klíčů: A zvolí dvě podobně velká prvočísla $p, q \equiv 3 \pmod 4$, vypočte $n = pq$.
- $V_A = n$, $S_A = (p, q)$
- zašifrování numerického *kódu* zprávy M: $C = C_e(M) \equiv M^2 \pmod n$
- dešifrování šifry C: vypočtou se (čtyři) odmocniny z C modulo n a snadno se otestuje, která z nich byla původní zprávou.

$p, q \equiv 3 \pmod 4$

$\sqrt{c} \equiv\ ? \pmod{p \cdot q}$

tj. m t.ż. $m^2 \equiv c \pmod{p \cdot q}$

$\iff \begin{cases} m^2 \equiv c \pmod p \\ m^2 \equiv c \pmod q \end{cases}$

$\rightsquigarrow$ spoczątkowe odwrotności: $m \equiv \pm c^{\frac{p+1}{4}} \pmod p$

$m \equiv \pm c^{\frac{q+1}{4}} \pmod q$

Pomocą CRT spoczątkowe 4 odwrotności:

$\begin{cases} m \equiv + c^{\frac{p+1}{4}} \ (p) \\ m \equiv + c^{\frac{q+1}{4}} \ (q) \end{cases}$ $\begin{cases} m \equiv + c^{\frac{p+1}{4}} \ (p) \\ m \equiv - c^{\frac{q+1}{4}} \ (q) \end{cases}$ ...

$$M_1 \equiv C^{\frac{p+1}{4}} \quad (p) \qquad\qquad M_2 \equiv C^{\frac{p+1}{4}} \quad (p)$$

$$M_1 \equiv C^{\frac{q+1}{4}} \quad (q) \qquad\qquad M_2 \equiv -C^{\frac{q+1}{4}} \quad (q)$$

$$M_1 - M_2 \equiv 0 \quad (p) \qquad\qquad M_1 - M_2 \not\equiv 0 \quad (q)$$

$$p \mid M_1 - M_2 \qquad\qquad q \nmid M_1 - M_2$$

$$(M_1 - M_2, \underset{p \cdot q}{\underset{\shortparallel}{n}}) = p$$

$M \longrightarrow C = M^2 \longrightarrow$ nějaká z odmocnin z M^2

$\frac{1}{4}: M$, $\frac{1}{4}: -M$, $\frac{1}{4}$

Výpočet druhé odmocniny z C modulo $n = pq$, kde $p \equiv q \equiv 3 \pmod 4$

- vypočti $r = C^{(p+1)/4} \pmod p$ a $s = C^{(q+1)/4} \pmod q$

---

[a] Uvědomte si, že jde vlastně o aplikaci Čínské zbytkové věty!

Výpočet druhé odmocniny z C modulo $n = pq$, kde $p \equiv q \equiv 3 \pmod 4$

- vypočti $r = C^{(p+1)/4} \pmod p$ a $s = C^{(q+1)/4} \pmod q$
- vypočti a, b tak, že $ap + bq = 1$

[a] Uvědomte si, že jde vlastně o aplikaci Čínské zbytkové věty!

Výpočet druhé odmocniny z C modulo $n = pq$, kde $p \equiv q \equiv 3 \pmod 4$

- vypočti $r = C^{(p+1)/4} \pmod p$ a $s = C^{(q+1)/4} \pmod q$
- vypočti a, b tak, že $ap + bq = 1$
- polož[a] $x = (aps + bqr) \pmod n$, $y = (aps - bqr) \pmod n$

[a] Uvědomte si, že jde vlastně o aplikaci Čínské zbytkové věty!

Výpočet druhé odmocniny z C modulo $n = pq$, kde $p \equiv q \equiv 3 \pmod 4$

- vypočti $r = C^{(p+1)/4} \pmod p$ a $s = C^{(q+1)/4} \pmod q$
- vypočti a, b tak, že $ap + bq = 1$
- polož[a] $x = (aps + bqr) \pmod n$, $y = (aps - bqr) \pmod n$
- druhými odmocninami z C modulo n jsou $\pm x$, $\pm y$.

[a] Uvědomte si, že jde vlastně o aplikaci Čínské zbytkové věty!

Příklad

V Rabinově kryptosystému Alice zvolila za svůj soukromý klíč $p = 23, q = 31$, veřejným klíčem je pak $n = pq = 713$. Zašifrujte zprávu $m = 327$ pro Alici a ukažte, jak bude Alice tuto zprávu dešifrovat.

Příklad

V Rabinově kryptosystému Alice zvolila za svůj soukromý klíč $p = 23, q = 31$, veřejným klíčem je pak $n = pq = 713$. Zašifrujte zprávu $m = 327$ pro Alici a ukažte, jak bude Alice tuto zprávu dešifrovat.

Řešení

$c = 692$, kandidáti původní zprávy jsou $\pm 4 \cdot 23 \cdot 14 \pm 3 \cdot 31 \cdot 18 \pmod{713}$.

Princip digitálního podpisu

Podepisování

M

1. Vygeneruje se otisk (hash) H_M zprávy pevně stanovené délky (např. 160 nebo 256 bitů).

   Rabin ... ne vše je druhou mocnina

$S_A(H_M)$

2. Podpis zprávy $S_A(H_M)$ je vytvořen (pomocí dešifrování) z tohoto hashe s nutností znalosti soukromého klíče podepisujícího.
3. Zpráva M (případně zašifrovaná veřejným klíčem příjemce) je spolu s podpisem odeslána.

Princip digitálního podpisu

Podepisování

1. Vygeneruje se otisk (hash) H_M zprávy pevně stanovené délky (např. 160 nebo 256 bitů).
2. Podpis zprávy $S_A(H_M)$ je vytvořen (pomocí dešifrování) z tohoto hashe s nutností znalosti soukromého klíče podepisujícího.
3. Zpráva M (případně zašifrovaná veřejným klíčem příjemce) je spolu s podpisem odeslána.

$M, S_A(H_M)$

Ověření podpisu

1. K přijaté zprávě M se (po jejím případném dešifrování) vygeneruje otisk H'_M
2. S pomocí veřejného klíče (deklarovaného) odesílatele zprávy se rekonstruuje původní otisk zprávy $V_A(S_A(H_M)) = H_M$.
3. Oba otisky se porovnají $H_M = H'_M$?.

Diffie-Hellman key exchange

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

Diffie-Hellman key exchange

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

- Dohoda stran na **prvočísle** p a primitivním kořenu g modulo p (veřejné)

(p, g) veřejné

Diffie-Hellman key exchange

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

- Dohoda stran na **prvočísle** p a primitivním kořenu g modulo p (veřejné)
- Alice vybere náhodné a a pošle $g^a \pmod{p}$

Diffie-Hellman key exchange

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

- Dohoda stran na **prvočísle** p a primitivním kořenu g modulo p (veřejné)
- Alice vybere náhodné a a pošle $g^a \pmod p$
- Bob vybere náhodné b a pošle $g^b \pmod p$

$A \quad \underline{\underline{g^b}} \longleftarrow b \quad B$

$a \longrightarrow \underline{\underline{g^a}}$

každý z nich umí spočítat

$(g^a)^b = \underline{\underline{g^{ab}}} = (g^b)^a$

Diffie-Hellman key exchange

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

- Dohoda stran na **prvočísle** p a primitivním kořenu g modulo p (veřejné)
- Alice vybere náhodné a a pošle $g^a \pmod p$
- Bob vybere náhodné b a pošle $g^b \pmod p$
- Společným klíčem pro komunikaci je $g^{ab} \pmod p$.

Diffie-Hellman key exchange

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

- Dohoda stran na **prvočísle** p a primitivním kořenu g modulo p (veřejné)
- Alice vybere náhodné a a pošle $g^a \pmod p$
- Bob vybere náhodné b a pošle $g^b \pmod p$
- Společným klíčem pro komunikaci je $g^{ab} \pmod p$.

Diffie-Hellman key exchange

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

- Dohoda stran na **prvočísle** p a primitivním kořenu g modulo p (veřejné)
- Alice vybere náhodné a a pošle g^a (mod p)
- Bob vybere náhodné b a pošle g^b (mod p)
- Společným klíčem pro komunikaci je g^{ab} (mod p).

? spočítat $a = \log_g g^a$

$b = - \cdot -$

g^{ab}

Poznámka

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

$a \rightsquigarrow g^a \qquad\qquad g^b \longleftarrow b$

A $\qquad\qquad\qquad$ C $\qquad\qquad\qquad$ B

$g^c \leftsquigarrow c \longrightarrow g^c$

$\rightarrow$ AC mají domluvený společný klíč

BC ———— // ————

Kryptosystém ElGamal

Z protokolu DH na výměnu klíčů odvozen šifrovací algoritmus ElGamal:

- Alice zvolí prvočíslo p spolu s primitivním kořenem g
- Alice zvolí **tajný klíč** x, spočítá $h = g^x \pmod{p}$ a zveřejní **veřejný klíč** (p, g, h)
- šifrování zprávy M: Bob zvolí náhodné y a vypočte $C_1 = g^y \pmod{p}$ a $C_2 = M \cdot h^y \pmod{p}$ a pošle (C_1, C_2)
- dešifrování zprávy: $OT = C_2 / C_1^x$

$A \longleftarrow B$

x $\quad$ y

g^{xy}

$g^y + M \cdot g^{xy} \longleftarrow M$

obojí modulo p

$\underline{(p,g) + g^x}$

(p,g,g^x)

soukr. y, $\quad M \mapsto (g^y, M \cdot g^{xy})$

Kryptosystém ElGamal

Z protokolu DH na výměnu klíčů odvozen šifrovací algoritmus ElGamal:

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

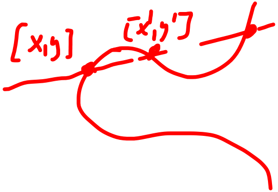
Poznámka

Analogicky jako v případě RSA 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 .

$g^x = g \cdot g \cdots g$



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 .

! video
z j2020
z MB104

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.