RSA šifrování

5. a 6. přednáška z kryptografie

Obsah

RSA šifrování

- RSA šifrování je šifrování s veřejným klíčem.
- Bezpečnost se opírá o časovou složitost problému faktorizace na prvočísla.
- Autoři: Ronald Rivest, Adi Shamir, Leonard Adleman, USA, 1977. Patentováno 1983.
- Nezávisle též James Ellis, Clifford Cocks, Malcolm Williamson, Anglie, 1969-75. Zveřejněno až 1997.

Protokol RSA

Vytvoření klíče

Alice chce dostávat od Boba zašifrované zprávy. Přitom zprávy budou přirozená čísla menší než N.

Alice zvolí dvě různá prvočísla p, q tak, aby $n = pq > N$.
Spočte $\varphi(n) = (p-1)(q-1)$.
Dále Alice zvolí $e \in \mathbb{N}$ tak, aby $\gcd(e, \varphi(n)) = 1$.
Spočte $d = e^{-1}$ v $\mathbb{Z}_{\varphi(n)}$, který díky nesoudělnosti existuje.

- Veřejný klíč: (n, e) (ten Alice umístí pod své jméno do "telefonního seznamu")
- Soukromý klíč: (n, d) (ten Alice nikomu neprozradí)

Protokol RSA

Šifrování a dešifrování zpráv

- Bob chce Alici poslat zprávu $a < N < n$ (otevřená zpráva).
  Vezme Alicin veřejný klíč (n, e) a zašifruje takto:
  $a^e = b$ v $\mathbb{Z}_n$, kde $0 \leq b < n$.
  Bob pošle zprávu $b < n$ (šifrová zpráva).
- Alice použije k dešifrování svůj soukromý klíč (n, d) takto:
  $b^d = a$ v $\mathbb{Z}_n$, kde $0 \leq a < n$.

Názvy: n je modul, e je šifrovací (=encryption) exponent,
d je dešifrovací (=decryption) exponent protokolu RSA.

Protokol RSA

Protokol RSA šifrování funguje korektně:

Tvrzení

Nechť $n = pq$, kde $p \neq q$ jsou prvočísla,
a nechť $e, d \in \mathbb{N}$ splňují $ed = 1$ v $\mathbb{Z}_{\varphi(n)}$.
Potom pro každé $a \in \mathbb{Z}_n$ platí: $(a^e)^d = a$ v $\mathbb{Z}_n$.

Důkaz pro a nesoudělné s n plyne okamžitě z Euler-Fermatovy věty.
Pro a soudělné s n je třeba použít navíc Čínskou větu o zbytcích.

Protokol RSA

Tvrzení

Nechť $n = \prod_{i=1}^{k} p_i$, kde $p_1, \ldots, p_k$ jsou navzájem různá prvočísla,
a nechť $e, d \in \mathbb{N}$ splňují $ed = 1$ v $\mathbb{Z}_{\varphi(n)}$.
Potom pro každé $a \in \mathbb{Z}_n$ platí: $(a^e)^d = a$ v $\mathbb{Z}_n$.

Protokol RSA šifrování lze zobecnit pro libovolný "square free"
modul n a bude fungovat korektně.

Pokud by však $p^2 \mid n$ pro nějaké prvočíslo p, pak by se špatně
dešifrovaly zprávy dělitelné p, ale nedělitelné maximální mocninou
p^m obsaženou ve faktorizaci n.

Protokol RSA

Pro vygenerování klíče protokolu RSA potřebujeme:

- generovat náhodná velká prvočísla (používá se pravděpodobnostní Millerův-Rabinův test prvočíselnosti; vygenerování $l-$místného prvočísla trvá $O(l^4)$)
- spočítat inverzi exponentu e v $\mathbb{Z}_{\varphi(n)}$ (rozšířený Eukleidův algoritmus; potřebný čas je $O(\mathrm{len}(n)^2)$)

Používají se následující velikosti čísel (viz Shoup, 2008; dnes jsou běžná dvakrát delší čísla):

- obě prvočísla p, q mají zhruba 512 bitů, modul $n = pq$ má pak 1024 bitů
- soukromý exponent d má až 1024 bitů (aspoň 512 bitů)
- veřejný exponent e je výrazně kratší (ale aspoň 17 bitů)

Protokol RSA

Pro šifrování a děšifrování potřebujeme:

- rychle umocňovat v $\mathbb{Z}_n$ (algoritmus opakovaných čtverců; potřebný čas je $O(\mathrm{len}(n)^3)$)
- při dešifrování můžeme počítat reziduálně v $\mathbb{Z}_p$ a $\mathbb{Z}_q$ (použijeme navíc Euler-Fermatovu větu a Čínskou větu o zbytcích; potřebný čas je $O(2(\frac{\mathrm{len}(n)}{2})^3) = O(\frac{1}{4}\,\mathrm{len}(n)^3)$, budeme tedy 4−krát rychlejší)

V praxi se RSA šifrování používá většinou jen k výměně klíčů pro symetrické šifrování. Zprávy se pak šifrují symetrickou šifrou (DES, AES), která je mnohem rychlejší (stokrát až tisíckrát rychlejší).

Bezpečnost protokolu RSA

Problém faktorizace n

Bezpečnost protokolu RSA se opírá o časovou složitost problému faktorizace na prvočísla.

Pokud by někdo rozložil modul n na $n = pq$, pak si z veřejného klíče dopočítá soukromý klíč a dešifruje. To je tzv. *útok hrubou silou*.

Problém faktorizace n na prvočísla je exponenciální, resp. subexponenciální problém:

- dělení všemi prvočísly do $\sqrt{n}$ trvá $O(2^{\frac{1}{2}\,\mathrm{len}(n)})$
- zatím nejrychlejší algoritmus (General Number Field Sieve) vyžaduje čas $O(2^{(c+o(1))\,\mathrm{len}(n)^{1/3}\,\mathrm{len}(\mathrm{len}(n))^{2/3}})$, kde $c < 2$

Bezpečnost protokolu RSA

Diskrétní odmocnina

Funkce diskrétní mocniny $\rho : \mathbb{Z}_n \to \mathbb{Z}_n : x \mapsto x^e$ je tzv. *jednosměrná funkce*.
Chceme-li počítat diskrétní e−tou odmocninu, musíme znát faktorizaci n, resp. znát d, pro něž $ed = 1$ v $\mathbb{Z}_{\varphi(n)}$.
Exponent d je "padacím mostem" či *"zadními vrátky"* k invertování této funkce.

Bez znalosti d lze počítat e−tou odmocninu z prvku b pouze *hrubou silou*, tj. postupným umocňováním všech $a \in \mathbb{Z}_n$, dokud nebude $a^e = b$ v $\mathbb{Z}_n$.
To vyžaduje opět exponenciální čas $O(2^{\mathrm{len}(n)})$.

Bezpečnost protokolu RSA

Nejde $\varphi(n)$ nebo d zjistit bez rozkladu n na prvočísla?

Tvrzení

Nechť n je součinem dvou různých prvočísel. Znalost těchto prvočísel je ekvivalentní znalosti $\varphi(n)$.

Důkaz: $n = pq$, $\varphi(n) = (p-1)(q-1) = n - (p+q) + 1$.
Známe-li n a $\varphi(n)$, pak známe součin π a součet σ dvou neznámých čísel p, q.
Tato jsou tudíž řešením kvadratické rovnice $x^2 - \sigma x + \pi = 0$.

Poznámka

Znalost $\varphi(n)$ stačí k efektivní faktorizaci libovolného n. Algoritmus, který to dokáže v polynomiálním čase, ukážeme později.

Bezpečnost protokolu RSA

Tvrzení

Nechť (n, e) je veřejný klíč protokolu RSA.
Znalost soukromého exponentu d umožní efektivně faktorizovat n.

Lemma

Pokud nalezneme v $\mathbb{Z}_n$ netriviální druhou odmocninu z 1
(tj. $b \neq \pm 1$, pro něž $b^2 = 1$ v $\mathbb{Z}_n$), tak umíme faktorizovat n.

K důkazu tvrzení: Víme, že $ed = 1$ v $\mathbb{Z}_{\varphi(n)}$. Odtud
$ed - 1 = k\varphi(n) = k(p-1)(q-1)$, kde $p-1$, $q-1$ jsou sudá čísla.
Zvolme libovolně $a \in \mathbb{Z}_n^*$, pak $a^{\varphi(n)} = 1$ v $\mathbb{Z}_n$ (Euler-Fermat).
Jelikož $a^{ed-1} = 1$ v $\mathbb{Z}_n$, bude $b = a^{(ed-1)/2}$ splňovat $b^2 = 1$ v $\mathbb{Z}_n$
(přitom exponent $\frac{ed-1}{2}$ je přirozené číslo). Je-li $b \neq \pm 1$,
faktorizujeme n. V opačném případě zkusíme jiné $a \in \mathbb{Z}_n^*$.

Bezpečnost protokolu RSA

Algoritmus na faktorizaci modulu n ze znalosti veřejného i soukromého exponentu

Vstup: veřejný klíč (n, e) a soukromý klíč (n, d) protokolu RSA
Výstup: prvočíslo p, které je faktorem modulu n

- spočti r, pro něž $ed - 1 = 2^r l$, kde l je liché
- repeat
  - zvol $a \in \mathbb{Z}_n$ náhodně
  - $d \leftarrow \gcd(a, n)$
  - if $d > 1$ then output d (a skonči)
  - $c \leftarrow a^l$ v $\mathbb{Z}_n$ (nyní je $a \in \mathbb{Z}_n^*$)
  - while $c^2 \neq 1$ do $c \leftarrow c^2$ v $\mathbb{Z}_n$ enddo (víme, že $c^{2^r} = 1$)
- until $c \neq \pm 1$
- output $\gcd(c - 1, n)$

Bezpečnost protokolu RSA

Časová složitost algoritmu

Pravděpodobnost, že náhodnou volbou $a \in \mathbb{Z}_n^*$ najdeme netriviální druhou odmocninu z 1 je v $\mathbb{Z}_n$, kde $n = pq$, rovna $\frac{1}{2}$. Průměrně budou potřeba dva repeat-cykly.

Dále se používá Eukleidův algoritmus a algoritmus opakovaných čtverců, kde exponent $ed - 1$ předpokládáme velikosti $O(n)$.

Očekávaný čas běhu algoritmu je $O(2\,\mathrm{len}(n)^3)$.

Shrnutí

Ukázali jsme, že získání jakékoliv informace k dešifrování - ať $\varphi(n)$ nebo d - je ekvivalentní znalosti faktorizace modulu n. Vyžaduje tedy (aspoň doposud) exponenciální či subexponenciální čas.

Útoky na RSA při sdíleném modulu

Útok insidera

Správce přidělí k účastníkům klíče se stejným modulem n,
označme klíče $i-$tého účastníka (n, e_i), (n, d_i) pro $1 \leq i \leq k$.

Libovolný účastník může díky znalosti svého soukromého klíče d_i faktorizovat n předchozím algoritmem.
Pak může dopočíst soukromé klíče ostatních účastníků a dešifrovat jim určené zprávy.

Obrana

Vždy generovat nový (privátní) modul n.

Útoky na RSA při sdíleném modulu

Útok outsidera

Správce přidělí k účastníkům klíče se stejným modulem n, označme klíče $i-$tého účastníka (n, e_i), (n, d_i) pro $1 \leq i \leq k$.

Eva zachytila zprávy posílané dvěma (resp. $s \leq k$) účastníkům, jejichž veřejné klíče jsou nesoudělné. Eva navíc ví, že se jedná o šifrové zprávy vzniklé ze stejné otevřené zprávy:
Eva zná $b_1, \ldots, b_s$, kde $b_i = a^{e_i}$ v $\mathbb{Z}_n$, a chce zjistit a.

Z Bezoutovy věty je $1 = \gcd(e_1, \ldots, e_s) = t_1 e_1 + \ldots + t_s e_s$ pro vhodná $t_i \in \mathbb{Z}$.
Eva spočte (pokud lze) $b_1^{t_1} \cdot \ldots \cdot b_s^{t_s} = a^{t_1 e_1 + \ldots + t_s e_s} = a^1 = a$ v $\mathbb{Z}_n$.

Útoky na RSA při sdíleném modulu

Útok outsidera

Poznámka: Některé koeficienty t_i budou určitě záporné, pro $t_i < 0$ je $b_i^{t_i} = (b_i^{-1})^{|t_i|}$ v $\mathbb{Z}_n$. Šifrová zpráva b_i však nemusí být invertivilní v $\mathbb{Z}_n$! Pak ale $\gcd(b_i, n) = p$.

Eva faktorizuje $n = pq$ a dopočte jakýkoliv soukromý klíč (n, d_j).

Obrana

Vždy generovat nový (privátní) modul n.

Útoky na RSA při sdíleném veřejném exponentu

Hastadův útok (Hastad's broadcast attack)

Klíče se stejným malým veřejným exponentem e má k účastníků, kde $e \leq k$. Označme veřejný klíč $i-$tého účastníka (n_i, e).

Tatáž otevřená zpráva $a < n_i$ (pro všechna $1 \leq i \leq k$) byla rozeslána všem účastníkům. Eva zachytila tyto šifrové zprávy $b_1, \ldots, b_k$, kde $b_i = a^e$ v $\mathbb{Z}_{n_i}$.

Eva tedy zná zbytky čísla a^e modulo n_i pro sadu čísel $n_1, \ldots, n_k$. Lze předpokládat, že tato čísla jsou po dvou nesoudělná: To Eva snadno otestuje Eukleidovým algoritmem, v případě soudělnosti najde faktor nějakého n_i a dopočte soukromý klíč (n_i, d_i).

Z Čínské věty o zbytcích Eva spočte a^e v $\mathbb{Z}_n$, kde $n = \prod_{i=1}^{k} n_i$. Protože $a^e < n$, může Eva použít $e-$tou odmocninu v $\mathbb{Z}$.

Útoky na RSA při sdíleném veřejném exponentu

Obrana vůči Hastadově útoku

1) Nepoužívat příliš malý veřejný exponent e, aby nebylo $e \leq k$.

Kvůli jiným útokům využívajícím malého veřejného exponentu se doporučuje $e > 2^{16}$, tedy aspoň $17-$ti bitový exponent. To už zaručí bezpečnost a zároveň umožní rychlejší zašifrování zpráv.

2) Ke zprávě a před každým zašifrováním přidat náhodné bity, aneb šifruje se pokaždé jiná zpráva a_i.

Přidávání náhodných bitů je nutná běžná praxe u všech deterministických šifrovacích algoritmů. Bez něj by šifrování nemělo tzv. sémantickou bezpečnost = pokud si Eva dokáže tipnout, jakou zprávu Bob Alici posílá, tak si snadno ověří, zda si tipla správně.

Útoky na RSA při malém soukromém exponentu

Tvrzení (Wienerův útok)

Nechť je pro RSA protokol zvoleno $n = pq$, kde $q < p < cq$ pro malé $c \in \mathbb{N}$, $e < \varphi(n)$, $d < \frac{1}{c+1} n^{\frac{1}{4}}$. Pak lze z veřejného klíče (n, e) efektivně spočítat soukromý klíč (n, d).

Důkaz využívá representaci čísel pomocí řetězových zlomků, kterou lze spočítat pomocí Eukleidova algoritmu.

$$\frac{a}{b} = q_1 + \cfrac{1}{q_2 + \cfrac{1}{\ddots \cfrac{}{q_{\lambda-1} + \frac{1}{q_\lambda}}}}$$

Pro racionální číslo $\frac{a}{b}$ je řetězový zlomek konečný a má délku $\lambda < 2 \min\{\mathrm{len}(a), \mathrm{len}(b)\}$.

Útoky na RSA při malém soukromém exponentu

Označme řetězový zlomek pro $\frac{a}{b}$ jako $(q_1; q_2, \ldots, q_\lambda)$. Každá podposloupnost $(q_1; q_2, \ldots, q_i)$ se nazývá $i-$tou konvergentou čísla $\frac{a}{b}$. (Pro nekonečný řetězový zlomek neracionálního čísla r posloupnost konvergent racionálně aproximuje číslo r a konverguje k číslu r.)

Z rovnosti $ed - 1 = k\varphi(n)$, přičemž k, d ani $\varphi(n)$ neznáme, lze díky předpokladům tvrzení odvodit: $|\frac{e}{n} - \frac{k}{d}| < \frac{1}{2d^2}$, $\gcd(k, d) = 1$. Nerovnost zaručuje, že zlomek $\frac{k}{d}$ v základním tvaru už musí být nějakou konvergentou racionálního čísla $\frac{e}{n}$ (což je hlubší výsledek).

Zbývá spočítat $O(\mathrm{len}(n))$ konvergent čísla $\frac{e}{n}$ a pro každou zkusit, zda jmenovatel může být hledaným soukromým exponentem. Např. spočítat $\varphi(n)$ z rovnosti výše a použít ho k faktorizaci n, nebo zkoušet zašifrovat a dešifrovat náhodné zprávy.

Útoky na RSA při malém soukromém exponentu

Obrana vůči Wienerovu útoku

Nepoužívat příliš malý soukromý exponent d.

Ukazuje se, že Wienerova mez není pevná. Je pravděpodobné, že při $d < n^{\frac{1}{2}}$ půjde efektivně dopočítat d z veřejného klíče e. Doporučuje se při n o 1024 bitech volit d aspoň o 512 bitech, bežně se používá d o 1024 bitech.

Při dešifrování velký exponent nevadí, protože počítáme reziduálně v $\mathbb{Z}_p$ a $\mathbb{Z}_q$ a exponent zmenšíme modulo $p - 1$, resp. $q - 1$. Ale ani tyto zbytkové exponenty d_p, d_q nesmí být příliš malé, neboť existuje útok, který umožňuje faktorizovat n v čase $O(\min\{\sqrt{d_p}, \sqrt{d_q}\}) = O(2^{\frac{1}{2}\min\{\mathrm{len}(d_p), \mathrm{len}(d_q)\}})$.

Útoky na implementaci RSA

První dva útoky předpokládají vpašování čipu do Alicina počítače.

- Kocher: Měření času výpočtu dešifrování. (Algoritmus opakovaných čtverců je řízen bity exponentu d, z měření časů lze rekonstruovat d bit po bitu. Používá se zde teorie pravděpodobnosti a statistika.)
- Náhodné chyby při dešifrování. (Pokud při reziduálním počítání vznikne chyba jen v jednom prvočíselném modulu, umožní to faktorizaci n.)
- Chybové hlášky při příjmu šifrové zprávy v nestandardním tvaru. (Eva posílá zachycenou šifrovou zprávu b Alici k dešifrování několikrát, a to ve tvaru cb pro různá $c \in \mathbb{Z}_n$. Podle toho, kdy obdrží chybovou hlášku, je schopna zjistit otevřenou zprávu a.)

Útoky na protokol RSA

Závěr

Veškeré útoky poukazují pouze, na co je třeba dát si pozor při provozu šifrování RSA. Doposud nebyl nalezen útok, který by protokol RSA znehodnotil.

Je-li RSA šifrování implementováno správně, zaručuje naprostou bezpečnost v digitální komunikaci.

Autentizace zpráv

Šifrování s veřejným klíčem otvírá následující otázky:

- Může si být Bob jistý, že Alicin veřejný klíč dala do "telefonního seznamu" skutečně Alice? Co když ho tam vpašovala Eva, aby mohla číst Bobovy zprávy pro Alici? Alicin veřejný klíč musí mít *certifikát* (= certifikační autorita se sešla s Alicí a ta jí potvrdila, že tento klíč je její).
- Může si být Alice jistá, že zprávu jí poslal skutečně Bob? Co když ji poslala Eva, která se vydává za Boba? Bob může ke své zprávě připojit *digitální podpis*. Podpis má zaručit, že Bob je autorem zprávy a že zpráva nebyla cestou změněna.

Autentizace zpráv

Digitální podpis pro RSA šifrování

Bob může svou otevřenou zprávu podepsat svým soukromým klíčem a pak ji teprve zašifrovat Aliciným veřejným klíčem:

$a \longrightarrow a^{d_B} \ v \ \mathbb{Z}_{n_B} \longrightarrow (a^{d_B})^{e_A} = b \ v \ \mathbb{Z}_{n_A}$

Alice dešifruje svým soukromým klíčem a poté odstraní podpis Bobovým veřejným klíčem:

$b \longrightarrow c^{d_A} \ v \ \mathbb{Z}_{n_A} \longrightarrow (c^{d_A})^{e_B} = a \ v \ \mathbb{Z}_{n_B}$

Vznikle-li smysluplná zpráva, musel ji napsat Bob.

Pozn.: Nejprve je nutno zprávu podepsat, potom teprve zašifrovat. Jinak by mohla Eva podpis odstranit (Bobovým veřejným klíčem) a podepsat zprávu sama.

Autentizace zpráv

Digitální podpis pro RSA šifrování

Podepisuje se většinou pouze "hash" z otevřené zprávy.
Je to rychlejší a odolné vůči útokům na podpis.

Alice s Bobem si domluví hašovací funkci $h(x)$ (zde je symetrie).

Bob podepíše hash ze své otevřené zprávy svým soukromým klíčem a pak zprávu i podpis zašifruje Aliciným veřejným klíčem:

$a \longrightarrow (a, h(a)^{d_B}) \ v \ \mathbb{Z}_{n_B} \longrightarrow (a^{e_A}, (h(a)^{d_B})^{e_A}) = (b, c) \ v \ \mathbb{Z}_{n_A}$

Alice dešifruje obě části svým soukromým klíčem, a odstaraní Bobův podpis z hashe Bobovým veřejným klíčem.

$(b, c) \longrightarrow (b^{d_A}, c^{d_A}) = (a, m) \ v \ \mathbb{Z}_{n_A} \longrightarrow (a, m^{e_B}) \ v \ \mathbb{Z}_{n_B}$

Pak spočte Alice hash z dešifrované zprávy.
Pokud $h(a) = m^{e_B} \ v \ \mathbb{Z}_{n_B}$, tak zprávu poslal Bob.

Autentizace zpráv

Digitální podpis pro symetrické šifrování

Digitální podpis pro symetrické šifry používá svůj symetrický klíč. Tento klíč r určuje parametry použité hašovací funkce $h_r(x)$.

Podpis a šifrování: $a \longrightarrow (a, h_r(a)) \longrightarrow (b, c)$
Dešifrování: $(b, c) \longrightarrow (a, m)$, ověření podpisu: $h_r(a) = m$

Obě strany musí znát klíč pro šifrování a klíč pro podpis.
Message Authentication Codes (MACs) používají různé sady hašovacích funkcí.

Autentizace zpráv

Hašovací funkce

Sada hašovacích funkcí $h_r(x)$, $r \in R$, musí splňovat:

- $h_r(x)$ musí zprávu výrazně zkrátit na konstantní délku;
- každý hash má zhruba stejně vzorů a ty jsou rozházeny v množině zpráv tak, že malá změna zprávy způsobí velkou změnu hashe (z hashe nelze o zprávě nic uhodnout);
- každé zprávě může být přiřazen (nejlépe se stejnou pravděpodobností) jakýkoliv hash, použijeme-li různé klíče, (aneb Eva nemůže pod svou zprávu padělat Bobův podpis, nezná-li klíč r).

Autentizace zpráv

Hašovací funkce

Příklady hašovacích funkcí (viz Shoup):

- Klíč $r = (r_0, r_1, \ldots, r_k) \in \mathbb{Z}_p^{\times(k+1)}$, p prvočíslo, $k \in \mathbb{N}$.
  $h_r : \mathbb{Z}_p^{\times k} \to \mathbb{Z}_p : (x_1, \ldots, x_k) \mapsto r_0 + r_1 x_1 + \cdots + r_k x_k$
- Klíč $r = (r_0, r_1) \in \mathbb{Z}_p \times \mathbb{Z}_p^*$, p prvočíslo, $m \in \mathbb{N}$, $m < p$.
  $\tilde{h}_r : \mathbb{Z}_p \to \mathbb{Z}_m : x \mapsto (r_0 + r_1 x) \bmod m$, kde operace *mod* vrací zbytek po dělení.
- Klíč $r \in \mathbb{Z}_p$, p prvočíslo, $k \in \mathbb{N}$.
  $\bar{h}_r : \mathbb{Z}_p^{\times(k+1)} \to \mathbb{Z}_p : (x_0, x_1, \ldots, x_k) \mapsto x_0 + x_1 r + x_2 r^2 \cdots + x_k r^k$

h_r má dlouhý klíč, ale malé p (odpovídající délce hashe).
$\tilde{h}_r$ má krátký klíč, ale velké p (odpovídající délce zprávy).
$\bar{h}_r$ má obě výhody, ale trochu slabší parametry.

Autentizace zpráv

Hašovací funkce

Jak pomocí uvedených funkcí hašovat binární zprávu?

Například: Zprávu $a < n$, kde n má 1024 bitů, chceme hašovat na 160−bitový hash.
Zvolíme prvočíslo p o 161 bitech, tj. $2^{160} < p < 2^{161}$ (takové prvočíslo existuje podle Bertrandova postulátu).
Zprávu a rozdělíme na zprávy $a_1, \ldots, a_7$ délky 160 bitů, pak $a_i < p$.
Použijeme funkci $h_r(x)$, kde klíč r bude obsahovat 8 čísel ze $\mathbb{Z}_p$.

RSA šifrování

Literatura

- Velebil: Diskrétní matematika. Kapitoly 3.5-6 a příloha A.
  ftp://math.feld.cvut.cz/pub/velebil/y01dma/dma-notes.pdf
- Boneh: Twenty Years of Attacks on the RSA Cryptosystem.
  https://crypto.stanford.edu/ dabo/papers/RSA-survey.pdf
- Shoup: A Computational Introduction to Number Theory and Algebra. Kapitola 4.7, 8.7.
  http://shoup.net/ntb/