1 ŠIFROVANIE S VEREJNÝM KĽÚČOM

1.1 ÚVOD

Algoritmy šifrovania s verejným kľúčom patria v súčasnosti medzi najčastejšie využívané kryptografické algoritmy. Využívajú skutočnosť, že je možné vytvoriť **odlišný šifrovací** a **dešifrovací kľúč**, pričom zo znalosti jedného je **výpočtovo veľmi náročné** odvodiť druhý. Odbornej verejnosti[1] sú tieto algoritmy známe od roku 1977, kedy Diffie a Hellman zverejnili na Národnej počítačovej konferencii svoj spôsob šifrovania s verejným kľúčom. Algoritmy s verejným kľúčom využívajú rôzne ťažko riešiteľné matematické problémy. V rámci cvičenia si precvičíme algoritmus založený na **batožinovom (ruksakovom) probléme** a známy **algoritmus RSA**. Opíšeme tiež algoritmus na výmenu kľúčov **KEA** (Key Exchange Algorithm, ktorý je modifikáciou Diffieho-Helmanovho algoritmu na výmenu kľúčov a je optimalizovaný pre algoritmus **Skipjack**.

1.2 BATOŽINOVÝ (RUKSAKOVÝ) PROBLÉM

Podstata batožinového problému je jednoduchá. Majme množinu predmetov, pričom každý má určitú hmotnosť. Našou úlohou je z tejto množiny vybrať takú podmnožinu, aby batožina po ich naplnení mala predpísanú hmotnosť. Formálne vyjadrené:

Je daná množina hodnôt $M_1, M_2, \ldots, M_n$ a súčet S a je potrebné určiť množinu koeficientov b_i tak, aby platilo

$$S = b_1 M_1 + b_2 M_2 + \ldots + b_n M_n \tag{1.1}$$

Koeficienty b_i môžu mať buď hodnotu 0 (príslušné M_i nie je súčasťou batožiny) alebo 1 (M_i je súčasťou batožiny).

Šifrovanie je po zverejnení hodnôt M_i veľmi jednoduché.

Príklad

Zašifrujte binárny text

1 1 0 1 0 1 1 1 1 0 0 1 0 0 0 1 0 1 ....

ak množina $\{M_i\} = \{1, 5, 6, 11, 14, 20\}$.

[1] Predpokladá sa, že NSA poznalo tento spôsob šifrovania už podstatne skôr.

Dešifrovanie je už vo všeobecnosti podstatne náročnejšia úloha a čas pre riešenie tohto problému rastie vo všeobecnom prípade exponenciálne s počtom položiek n a je pre veľké hodnoty (už niekoľko stoviek) extrémne výpočtovo zložitý.

Dešifrovanie je však extrémne jednoduché v prípade tzv. **superrastúcej batožiny**, t.j. ak pre prvky množiny $\{M_i\}$ platí

$$\sum_{k=1}^{i} M_k < M_{i+1} \tag{1.2}$$

Príklad

Zvoľte superrastúcu množinu $\{M_i\}$ a ukážte, že dešifrovací algoritmus je veľmi jednoduchý.

Podstata algoritmu spočíva v možnosti transformovať superrastúcu postupnosť $\{M_i\}$ - **privátny kľúč** na všeobecnú postupnosť $\{M_i^{'}\}$ -**verejný kľúč**. Súčasťou privátneho kľúča sú aj dve čísla v, m pre ktoré platí

$$m > \sum_{i=1}^{n} M_i \tag{1.3}$$

$$GCD(v,m) = 1 \tag{1.4}$$

a transformáciu je možné realizovať pomocou vzťahu

$$M_i^{'} = (v * M_i) \bmod m \tag{1.5}$$

pričom šifrovanie sa realizuje klasickým spôsobom

$$S^{'} = b_1 M_1^{'} + b_2 M_2^{'} + \ldots + b_n M_n^{'} \tag{1.6}$$

Príklad

Na základe predchádzajúcich vzťahov vytvorte šifrovací systém s verejným kľúčom.

Dešifrovanie je možné realizovať na základe nasledujúcich vzťahov:

$$S = (v^{-1} * S^{'}) \bmod m \tag{1.7}$$

pričom

$$v^{-1} * v \equiv 1 (\bmod m) \tag{1.8}$$

a číslo S dešifrovať pre superrastúcu postupnosť $\{M_i\}$ - privátny kľúč.

Príklad
Pre šifrovací systém s verejným kľúčom vytvorený v predchádzajúcom príklade a na základe predchádzajúcich vzťahov overte šifrovací a dešifrovací postup.

1.3 ALGORITMUS RSA

Algoritmus RSA (pomenovaný po tvorcoch R. **R**ivestovi, A. **S**hamirovi a L. **A**dlemanovi) patrí medzi najznámejšie algoritmy s verejným kľúčom. Bezpečnosť algoritmu RSA je založená na zložitosti faktorizácie (t.j. rozklade na prvočíselné súčinitele) veľkých čísel (rádovo 100 až 200 ciferných dekadických čísel).

Šifrovacie kľúče (privátny a verejný) sa vypočítajú[2] z dvoch veľkých prvočísel[3] p a q. Po ich zvolení sa určí modul

$$n = p * q \tag{1.9}$$

Potom sa náhodne zvolí **šifrovací kľúč** e tak, aby platilo

$$GCD\left(e,\left(p-1\right)*\left(q-1\right)\right)=1 \tag{1.10}$$

Veľmi často sa v praxi využívajú hodnoty[4] $e = F_2 = 2^4 + 1 = 17 = 0x11$, resp. $e = F_4 = 2^{16} + 1 = 65537 = 0x10001$

Príklad
Akou metódou je možné určiť číslo e*?*

Dešifrovací kľúč d sa vypočíta tak, aby platilo

$$e * d \equiv 1\left(\operatorname{mod}\left(p-1\right)*\left(q-1\right)\right) \tag{1.11}$$

t.j. platí

$$d = e^{-1} \operatorname{mod}\left(\left(p-1\right)*\left(q-1\right)\right) \tag{1.12}$$

Čísla $\left(e,n\right)$ tvoria **verejný kľúč** a číslo d tvorí **privátny kľúč**. Šifruje sa podľa vzťahu

$$c_i = m_i^e \operatorname{mod} n \tag{1.13}$$

a dešifruje podľa vzťahu

$$m_i = c_i^d \operatorname{mod} n \tag{1.14}$$

[2] V praktických aplikáciách sa často používajú pravdepodobnostné metódy generovania prvočísel priemyselnej kvality, ktoré boli popísané v predchádzajúcich cvičeniach.

[3] Z dôvodu maximálnej bezpečnosti sa veľkosť čísel volí rádovo rovnaká.

[4] Sú to špeciálne prvočísla – tzv. druhé a štvrté Fermatove číslo. Tieto čísla umožňujú výrazné zrýchlenie šifrovania (pretože exponent je malé číslo) bez zníženia bezpečnosti celého algoritmu. Dešifrovanie pomocou privátneho (ktorý má vo všeobecnosti veľkosť porovnateľnú s n) exponentu je tak výrazne pomalšie.

Príklad

Pre čísla $p = 47$ *a* $q = 71$ *zvolíme* $e = 79$. *Je táto voľba prípustná? Zašifrujte správu 6882326879666683. Čo vidno zo zašifrovanej správy?*

Príklad

Pre čísla $p = 47$ *a* $q = 71$ *zvolíme* $e = 79$. *Určite hodnotu dešifrovacieho exponentu d.*

Existuje aj **rýchlejšia metóda dešifrovania**[5], ktorá využíva znalosť čísel p a q a umožňuje dosiahnuť 4 až 8 násobné zrýchlenie dešifrovania[6], pričom využíva nasledujúce rovnice (index $_i$ pre c_i je v nasledujúcich vzťahoch pre jednoduchosť vynechaný)

$$y_1 = c \bmod p \tag{1.15}$$

$$y_2 = c \bmod q \tag{1.16}$$

$$d_1 = d \bmod (p-1) \tag{1.17}$$

$$d_2 = d \bmod (q-1) \tag{1.18}$$

$$x_1 = y_1^{d_1} \bmod p \tag{1.19}$$

$$x_2 = y_2^{d_2} \bmod q \tag{1.20}$$

Pre čísla p a q $(p < q)$ nájdeme číslo A pre ktoré platí

$$A * p \equiv 1 \bmod q \qquad 0 < A \le q-1 \tag{1.21}$$

a dešifrovanie realizujeme podľa vzťahu

$$m_i = \left[\left(\left(x_2 - x_1 + q\right) * A\right) \bmod q\right] * p + x_1 \tag{1.22}$$

Príklad

Overte rýchly dešifrovací algoritmus na údajoch z predchádzajúcich príkladov.

[5] Táto metóda využíva Čínsku vetu o zvyškoch.

[6] Znižuje dĺžku čísel s ktorými sa počíta na polovicu.

1.4 ALGORITMUS KEA

KEA je asymetrický šifrovací algoritmus určený (a **optimalizovaný**!) pre výmenu 80-bitových šifrovacích kľúčov pre algoritmus Skipjack a bol zverejnený spolu s algoritmom Skipjack [4]. Algoritmus KEA naviac používa algoritmus Skipjack na redukciu 1024-bitovej premennej (**w**) na 80-bitový kľúč (**Key**). Algoritmus KEA využíva **problém diskrétneho algoritmu** (ktorý vo svojom algoritme využili Diffie a Hellman).

Diffieho-Hellmanov problém patrí medzi ťažko riešiteľné problémy a je ho možné sformulovať takto:

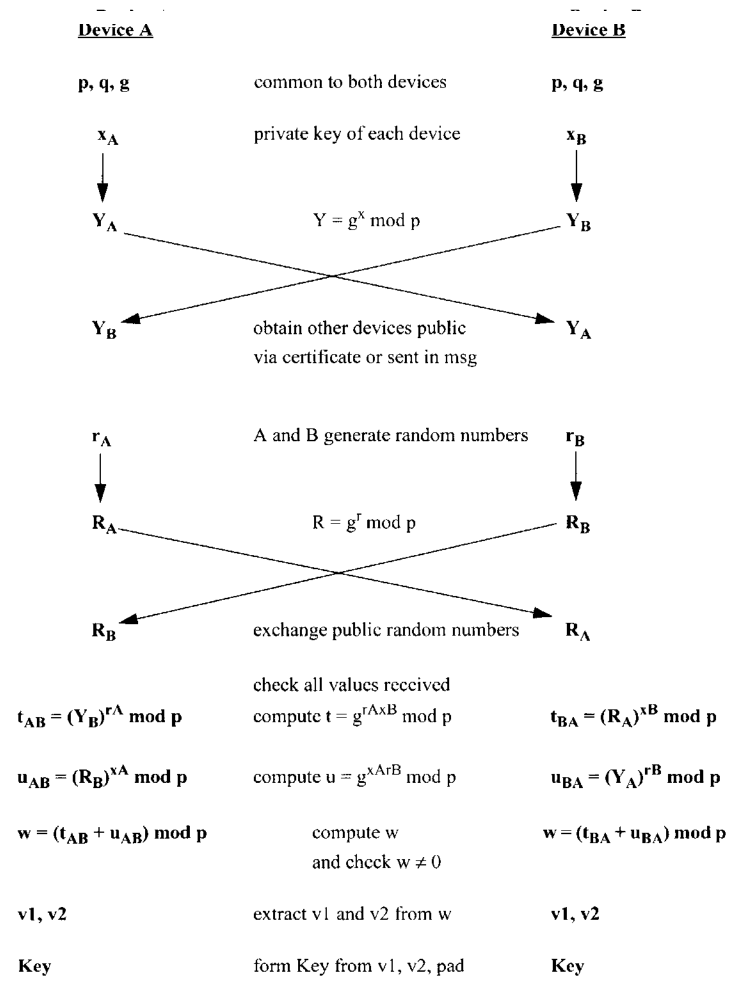
Majme prvočíslo p a číslo α, ktoré je generátorom telesa Z_p (t.j. všetky prvky telesa Z_p je možné vyjadriť ako mocninu čísla α). Ak poznáme prvky $\alpha^a \bmod p$ a $\alpha^b \bmod p$, hľadáme prvok $\alpha^{ab} \bmod p$.

Tento problém je základom praktických kryptografických protokolov na výmenu kľúčov cez nezabezpečené kanály. Algoritmus KEA je praktickou realizáciou tohto protokolu a v ďalšej časti ho stručne opíšeme.

Algoritmus využíva nasledujúce premenné:

p – 1024-bitový prvočíselný modul definujúci pole (presnejšie multiplikatívnu grupu) v ktorom (ej) sa vykonávajú výpočty, pričom $p = p_{1023}p_{1022}\dots p_0$
q – 160-bitový prvočíselný deliteľ čísla $p-1$, $q = q_{159}q_{158}\dots q_0$
g – 1024-bitový základ pre umocňovanie $g = g_{1023}g_{1022}\dots g_0$, ktorý je prvkom rádu q v multiplikatívnej grupe $\bmod p$ (matematicky $g^q \equiv 1 \bmod p$)
x – 160-bitový tajný kľúč $x = x_{159}x_{158}\dots x_0$ pre ktorý platí $0 < x < q$
Y – 1024-bitový verejný kľúč zodpovedajúci privátnemu kľúču x pre ktorý platí $Y = g^x \bmod p = Y_{1023}Y_{1022}\dots Y_0$
pad – 80-bitová konštanta $pad_{79}pad_{78}\dots pad_0 = 72F1A87E92824198AB0B$
r – náhodne vygenerované 160 bitové číslo $r = r_{159}r_{158}\dots r_0$

Protokol výmeny kľúča je znázornený na nasledujúcom obrázku

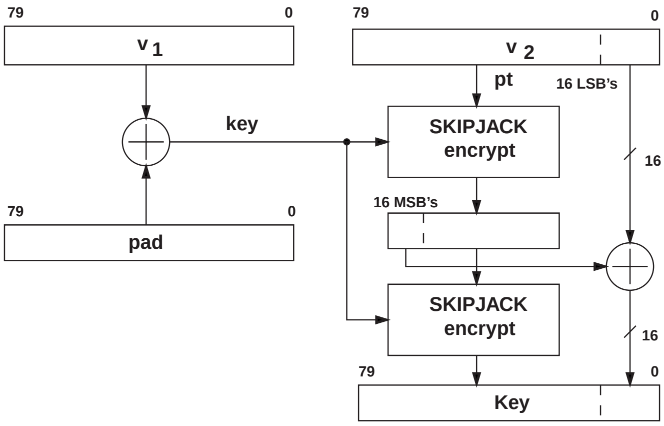


A summary of a full KEA exchange between devices A and B is as follows:

Device A		Device B
p, q, g	common to both devices	p, q, g
x_A	private key of each device	x_B
Y_A	$Y = g^x \bmod p$	Y_B
Y_B	obtain other devices public via certificate or sent in msg	Y_A
r_A	A and B generate random numbers	r_B
R_A	$R = g^r \bmod p$	R_B
R_B	exchange public random numbers	R_A
$t_{AB} = (Y_B)^{rA} \bmod p$	check all values received compute $t = g^{rAxB} \bmod p$	$t_{BA} = (R_A)^{xB} \bmod p$
$u_{AB} = (R_B)^{xA} \bmod p$	compute $u = g^{xArB} \bmod p$	$u_{BA} = (Y_A)^{rB} \bmod p$
$w = (t_{AB} + u_{AB}) \bmod p$	compute w and check $w \neq 0$	$w = (t_{BA} + u_{BA}) \bmod p$
v1, v2	extract v1 and v2 from w	v1, v2
Key	form Key from v1, v2, pad	Key

Page 12 of 23

pričom vytvorenie 80-bitového kľúča **Key** z 1024-bitovej premennej **w** sa realizuje pomocou zapojenia na nasledujúcom obrázku



a platí

$$v_1 = \left(\frac{w}{2^{(1024-80)}} \right) \bmod 2^{80} \tag{1.23}$$

$$v_2 = \left(\frac{w}{2^{(1024-160)}} \right) \bmod 2^{80} \tag{1.24}$$

$$Key = 2^{16} \left[E_{v_1 \oplus pad} \left(E_{v_1 \oplus pad} \left[\frac{v_2}{2^{16}} \bmod 2^{64} \right] \right) \right] \oplus \tag{1.25}$$

$$\oplus \left[\left(\frac{E_{v_1 \oplus pad} \left[\frac{v_2}{2^{16}} \bmod 2^{64} \right]}{2^{48}} \right) \oplus \left(v_2 \bmod 2^{16} \right) \right]$$

LITERATÚRA

[1] Přibil, J. – Kodl, J.: Ochrana dat v iformatice. Vydavatelství ČVUT, Praha 1996, ISBN 80-01-01664-1.

[2] Grošek, O. – Porubský, Š.: Šifrovanie – algoritmy, metódy, prax. Grada, 1992, ISBN 80-85424-62-2.

[3] Adámek, J.: Kódovaní. Matematika pro vysoké školy – sešit XXXI. SNTL Praha, 1989.

[4] Skipjack and KEA Algorithm Specifications. Version 2.0, 29 May 1998. Dostupné v elektronickej forme **– Skipjack_KEA.pdf**