Public-key systémy

(systémy s veřejným klíčem)

použití *jednosměrných* (trapdoor) *funkcí* - snadno vyčíslitelná funkce, jejíž inversní funkci lze efektivně počítat pouze se znalostí (malého) množství dodatečných informací. Nezávislé na zprávě.

každý uživatel vlastní pár klíčů:
veřejný (public) *klíč* - znám všem uživatelům systému, používá se k šifrování zpráv zasílaných tomuto uživateli
tajný, soukromý (private) *klíč* - uživatel uchovává v tajnosti, používá jej k dešifrování došlých zpráv
tajný klíč nelze efektivně odvodit ze znalosti odpovídajícího veřejného klíče

výhodou systémů s veřejnými klíči je relativně malé množství používaných klíčů, možnost vytváření veřejně ověřitelných elektronických podpisů, větší flexibilita správy klíčového materiálu

celá řada mechanismů:

- asymetrické šifry
- podepisovací schémata
- autentizační schémata
- distribuce klíčů
- generátory náhodných sekvencí
- …

Asymetrické šifry

tzn. šifrovací algoritmy založené na mechanismech s veřejným klíčem
veřejný klíč … šifrování, tajný klíč … dešifrování

Merkle-Hellman systém

založen na problému batohu, přenášená zpráva je chápána jako vektor řešení, přenášena je výsledná suma - "hmotnost batohu"

$a_1, a_2, \ldots a_n$ - posloupnost celých čísel, T cílová suma = hmotnost batohu, hledáme vektor $\boldsymbol{v}$ takový, aby

$$\sum_i a_i v_i = T$$

nechť posloupnost $a_1, a_2, \ldots a_n$ je superrostoucí, problém nalezení vektoru $\boldsymbol{v}$ je v tomto případě zvládnutelný v lineárním čase

Konstrukce systému

zvolíme superrostoucí posloupnost $s_1, s_2, \ldots s_n$, dále vybereme číslo w a modul m, w bereme nesoudělné s m, $m > s_n$.
Ze zvolených hodnot sestavíme již obecnou posloupnost

$$h_i = w * s_i \bmod m$$

Posloupnost $\{s_i\}_{i=1\ldots n}$ a čísla w a m utajíme, dále budou sloužit jako soukromý klíč.
Posloupnost $\{h_i\}_{i=1\ldots n}$ zveřejníme jakožto veřejný klíč.

Šifrování

Otevřený text P rozdělíme na bloky délky n bitů.
Každý blok P_j nahradíme sumou

$$Cj = \sum_i p_{j_i} h_i$$

Zašifrovaný text C odešleme

Dešifrování

Autorizovaný příjemce vypočítá w^{-1} - z vlastností w a m určitě existuje.
Pro každý blok C_j spočítá $C_j * w^{-1}$.
Vyřeší problém batohu se superrostoucí posloupností $\{s_i\}_{i=1\ldots n}$ pro všechny hodnoty získané v předchozím bodě.
Konkatenací řešení vznikne původní zpráva P.

Korektnost dešifrování

$$w^{-1}\mathbf{e}(p) \bmod \mathrm{m} = w^{-1}(p_1 h_1 + p_2 h_2 + \ldots + p_n h_n) \bmod \mathrm{m} =$$
$$= p_1 w^{-1} h_1 + p_2 w^{-1} h_2 + \ldots + p_n w^{-1} h_n \bmod \mathrm{m} =$$
$$= p_1 w^{-1} w s_1 + p_2 w^{-1} w s_2 + \ldots + p_n w^{-1} w s_n \bmod \mathrm{m} =$$
$$= p_1 s_1 + p_2 s_2 + \ldots + p_n s_n \bmod \mathrm{m}.$$

Poznámky k implementaci

Pro rozumné aplikace*:* m bývá voleno ve velikosti 100 až 200 číslic,
s_i mají délku 200 až 400 číslic,
batoh mívá přibližně 200 položek

možný způsob vytvoření superrosotoucího batohu:

vygenerujeme n náhodných čísel r_i z intervalu $\langle 0, 2^{200} \rangle$
$s_i = 2^{200+i-1} + r_i$

Analýza Merkle-Hellmanova systému

Známe-li m, je možné odvodit prvky superrostoucího batohu.
Položme

$$p = h_o / h_1 \bmod m$$

Pak ovšem platí

$$p = \left(w * s_o\right) / \left(w * s_1\right) \bmod m = s_o / s_1 \bmod m$$

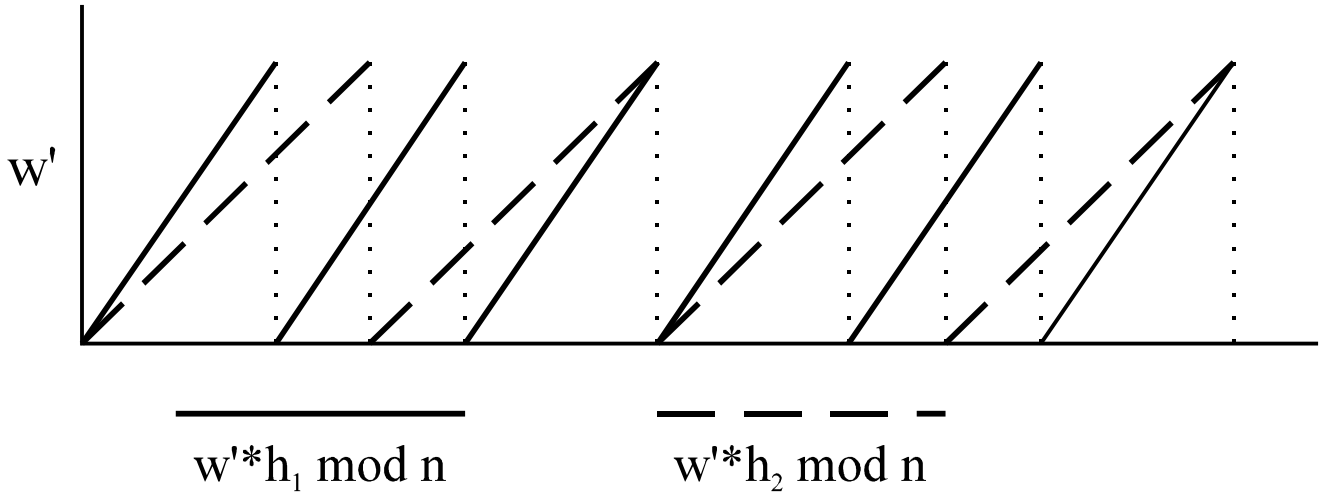
Spočítáme posloupnost

$$\mathsf{D} = \left\{ i * p \bmod m \right\}_{i=1}^{2^m}$$

Pro nějaké k ovšem nastane

$$k * p \bmod m = k * s_0 * 1/s_1 \bmod m = s_0$$

Lze očekávat, že s_0 bude nejmenším prvkem D. Známe-li s_0, lze spočítat w a tedy všechny s_i.
Hodnoty w a m je možné odhadovat pouze z posloupnosti $\{h_i\}_{i=1\ldots n}$. Hodnota m je větší než libovolné h_i. Budeme zkoušet různé hodnoty pro w.



Možné správné hodnoty *w* se nacházejí v překrývajících se bodech diskontinuity.

K prolomení Merkle-Hellmanova systému tedy není nutné vyřešit obecný problém batohu, ale stačí použít naznačeného postupu, který je daleko rychlejší. M-H systém tedy není vhodný k ochraně důležitých informací.

El Gamal kryptosystém

založen na obtížnosti výpočtu diskrétního logaritmu nad okruhem
randomizovaný kryptosystém

Konstrukce kryptosystému

Společný modul *q*, dále je zvoleno číslo *g* co nejvyššího řádu (nejlépe generátor).

Každý účastník *i* si zvolí tajný klíč y_i a vypočítá veřejný klíč $g^{y_i} \bmod q$

Šifrování

nechť uživatel *A* posílá zprávu *P* (< q) uživateli *B*
náhodně vybere číslo *k* a vypočítá:

$g^k \bmod q$; $P\left(g^{y_B}\right)^k \bmod q$
obě čísla zašle *B*

Dešifrování

uživatel *B* vypočítá

$\left(g^k\right)^{y_B} \bmod q$

a určí inverzní prvek. S jeho použitím z druhého čísla zpětně získá *P*.

Korektnost dešifrování

Zřejmě

$$P\left(g^{y_B}\right)^k\left(\left(g^k\right)^{y_B}\right)^{-1} = P\left(g^{y_B}\right)^k\left(\left(g^{y_B}\right)^k\right)^{-1} = P$$

Analýza El Gamalova kryptosystému

kryptosystém je považován za bezpečný,
nevýhodou je nutnost generování náhodných čísel *k* a zdvojnásobení objemu dat při šifrování, je relativně pomalý

Rivest-Shamir-Adelman kryptosystém

uveřejněný v roce 1977, někdy označován jako kód Herkules

Kryptoschéma je založeno na Eulerově formuli

$$a^{\varphi(n)} \equiv 1(\mathbf{mod}\, n) \quad (1)$$

kde $\phi(n)$ je počet čísel z intervalu 1,…, *n*, která jsou s *n* nesoudělná.
Platí:

$$\varphi(\mathrm{n}) = (\mathrm{p}_1 - 1)\mathrm{p}_1^{\mathrm{a}_1 - 1} \cdot (\mathrm{p}_2 - 1)\mathrm{p}_2^{\mathrm{a}_2 - 1} \ldots (\mathrm{p}_\mathrm{k} - 1)\mathrm{p}_\mathrm{k}^{\mathrm{a}_\mathrm{k} - 1} \quad (2)$$

kde

$$n = p_1^{a_1} \cdot p_2^{a_2} \cdot \ldots \cdot p_k^{a_k} \quad (3)$$

je prvočíselný rozklad čísla *n*.

Šifrování

je třeba znát číslo *n* a malé prvočíslo *e*.
Otevřený text *P* převedeme na posloupnost čísel modulo *n*.
Každý blok P_j zašifrujeme dle vzorce

$$C_j \equiv P_j^e \quad (\mathbf{mod}\, n) \quad (4)$$

Spojením výsledných bloků C_j vznikne zašifrovaný text.

Dešifrování

je třeba znát číslo *n*, a číslo *d*.
Každý z bloků C_j dešifrujeme takto:

$$P_j \equiv C_j^{\,d} \pmod{n} \quad \textbf{(5)}$$

Výpočet dešifrovacího klíče *d*

Musí platit

$$ed \equiv 1 \pmod{\varphi(n)} \quad \textbf{(6)}$$

Prvočíslo *e* nesmí dělit ϕ (*n*). *d* určíme z předchozího vztahu rozšířeným Euclidovým algoritmem.
Mimochodem, uvažte následující postup:
Nalezneme

$$r \equiv -\varphi(n)^{e-2} \pmod{e} \quad \textbf{(7)}$$

Ze (2) plyne $\varphi(e) = e - 1$, s použitím (1)

$$r\varphi(n) \equiv -\varphi(n)^{\varphi(e)} \equiv -1 \pmod{e} \quad \textbf{(8)}$$

Položíme tedy

$$d = \frac{r\varphi(n)+1}{e} \quad \textbf{(9)}$$

… tedy existuje více než jeden dešifrovací klíč
V praxi volíme *e* pevné (65535), pro každého účastníka nalezneme zvláštní *n* a dopočítáme dešifrovací klíč. *d* se počítá rozšířeným Euklidovým algoritmem.

Korektnost dešifrování

S použitím (1) a (6) postupně dostáváme

$$P_j^{\,ed} \equiv P_j^{\,ed \bmod \varphi(n)} \equiv P_j^{\,1} \equiv P_j \pmod{n}$$

Výběr klíčů, implementační poznámky

Veřejný klíč tvoří pár (*n, e*), soukromý klíč pár (*n, d*).
Číslo *n* musí být velmi velké, nesmí mít malé faktory. Pro reálné použití přibližně 100 až 200 bitů.
Nechť *n* je součinem prvočísel *p* a *q*. Klíč *e* volíme jako prvočíslo větší než (*p* - 1) a (*q* - 1).
Hranice bezpečnosti se posouvá od 1024 bitů modulu *n*, směrem k 1500 bitů, lépe 2048, klíče pro delší použití nebo vyšší stupeň bezpečnosti 4096 a více bitů
Nejlepším současným algoritmem pro faktorizaci velkých čísel je NFS (Number Field Sieve), které rozkládá čísla prakticky bez ohledu na strukturu

Při této volbě má nepřítel na výběr zhruba $\frac{\sqrt[2]{n}}{\ln n} - \frac{10^{50}}{100 \ln 10}$ možných prvočíselných činitelů.

Invertování čísla v okruhu

dělení čísel, případně výpočet inverzní hodnoty v rámci okruhu (tělesa) – nelze samozřejmě pro tzv. dělitele nuly
používá se rozšířený Euclidův algoritmus (viz. níže), uvažte, že $ax \equiv 1 \pmod d$

Rozšířený Euclidův algoritmus

vstup: nezáporná čísla *a* a *b*, a ≥
výstup: $d = \mathbf{gcd}(a, b)$ a celá čísla *x*, *y* tž. $ax + by = d$

if ($b = 0$) do
$d \leftarrow a$, $x \leftarrow 1$, $y \leftarrow 0$, return (d, x, y)
enddo
$x_2 \leftarrow 1$, $x_1 \leftarrow 0$, $y_2 \leftarrow 0$, $y_1 \leftarrow 1$
while $b > 0$ do
$q \leftarrow a/b$, $r \leftarrow a - qb$, $x \leftarrow x_2 - q\,x_1$, $y \leftarrow y_2 - q\,y_1$
$a \leftarrow b$, $b \leftarrow r$, $x_2 \leftarrow x_1$, $x_1 \leftarrow x$, $y_2 \leftarrow y_1$, $y_1 \leftarrow y$
enddo
$d \leftarrow a$, $x \leftarrow x_2$, $y \leftarrow y_2$
return (d, x, y)

takže položíme $b = \varphi(n)$, $x = e$ kde *e* nesoudělné s $\varphi(n)$ a počítáme

Volba prvočísel

Vygenerujeme náhodné liché číslo zvoleného řádu
Otestujeme prvočíselnost
Není-li prvočíslo, pokračujeme bodem 1.

Testy prvočíselnosti

Pro každé liché přirozené číslo n definujeme množinu $W(n) \subset \mathbf{Z}_n$:
pro $a \in \mathbf{Z}_n$ lze v polynomiálním čase ověřit, zda $a \in W(n)$
pokud je n prvočíslo, $W(n) = \theta$
pokud je n složené, $|W(n)| >= n/2$
Prvky množiny $W(n)$ nazýváme *svědky* toho, že číslo n je složené, ostatním číslům v $\mathbf{Z}_n$ - $W(n)$ říkáme *lháři*.

Solovay-Strassenův test prvočíselnosti

Základem je tzv. Eulerovo kritérium:
Pro liché prvočíslo p platí $r^{(p-1)/2} \equiv \mathbf{J}(r, p) (\mathrm{mod}\, n)$ pro všechna celá čísla r splňující **nsd**(p, r) = 1

Ze skutečnosti, že pro p liché existuje maximálně $\varphi(p)/2$ lhářů pro čísla 1, 2, ... p-1 lze odvodit následující algoritmus:

p číslo, které zkoumáme, r libv. číslo, pak nutně

nsd(p, r) = 1

a zároveň

$$\mathbf{J}(r, p) \equiv r^{p-1/2} \bmod p$$

kde **J**(r, p) je Jacobiho funkce, definovaná následovně

$$\mathbf{J}(r, p) \equiv \begin{cases} 1 & \text{pro } r = 1 \\ \mathbf{J}(r/2, p) * (-1)^{(p^2-1)/8} & \text{pro } r \text{ sudé} \\ \mathbf{J}(p \bmod r, r) * (-1)^{(r-1)(p-1)/4} & \text{pro } r \text{ liché}, r \neq 1 \end{cases}$$

zvolíme náhodně r tž. $2 \leq r \leq p-2$

spočítáme $n = r^{(p-1)/2} \bmod(p)$
pokud $n \neq 1$ a $n \neq p-1$ konec, je složené
spočítáme s = **J**(*r*, *p*), pokud není $n \equiv s$ konec, je složené
asi prvočíslo
Opakováním testu pro různé hodnoty *r* lze docílit požadované jistoty, že *p* je skutečně prvočíslo.

Miller-Rabinův test prvočíselnosti

Založen na následující skutečnosti:
Je-li *p* prvočíslo, potom jediná dvě řešení vztahu $x^2 \equiv 1 \ (\boldsymbol{mod}\ p)$ jsou 1 a -1.
Pro liché přirozené číslo *p* tž. *p* - 1 = $2^l s$, kde *s* je liché buď *a* celé číslo takové, že **nsd**(*a*,*p*) = 1. Potom $a^s \equiv 1 (\bmod p)$, nebo $a^{2^j s} \equiv -1 (\bmod p)$ pro nějaké *j* tž. $0 \leq j \leq l-1$ … pokud obě kongruence neplatí, je *a* silný svědek.
Lhářů je max $\varphi(p)/_4$, tzn. test mnohem rychleji konverguje.

Odtud odvodíme následující algoritmus:
Dáno testované číslo *p*.
Nechť *p* - 1 = $2^l s$, pro nějaké liché *s*
Náhodně zvolíme *a* ∈ { 2, … , p - 2}
Spočítáme $q \equiv a^s \bmod p$. Pokud $q \equiv 1 \bmod p$ konec, asi prvočíslo.
Počítáme q^2, q^{2^2}, … , $q^{2^l} = a^{p-1}$, vše **mod** *p*. Pokud není $a^{p-1} \equiv 1 \bmod p$, konec - nemůže být prvočíslo.
Nalezeme největší *k* tak, že $q^{2^k} \neq 1 \bmod n$. Pokud $q^{2^k} \equiv -1 \bmod p$, konec - asi prvočíslo, jinak nemůže být prvočíslo.

Analýza RSA

algoritmus je stále považován za bezpečný, za cenu neustále rostoucích parametrů (délka klíče)
problém faktorizace řeší kvantový Schorrův algoritmus
nutná správná implementace dlouhé aritmetiky pro zamezení statistických útoků na probíhající výpočet
slabostí je hypotetická možnost vytvořit elektronický podpis zprávy bez znalosti dešifrovacího klíče na základě zachycení vhodných předchozích zašifrovaných

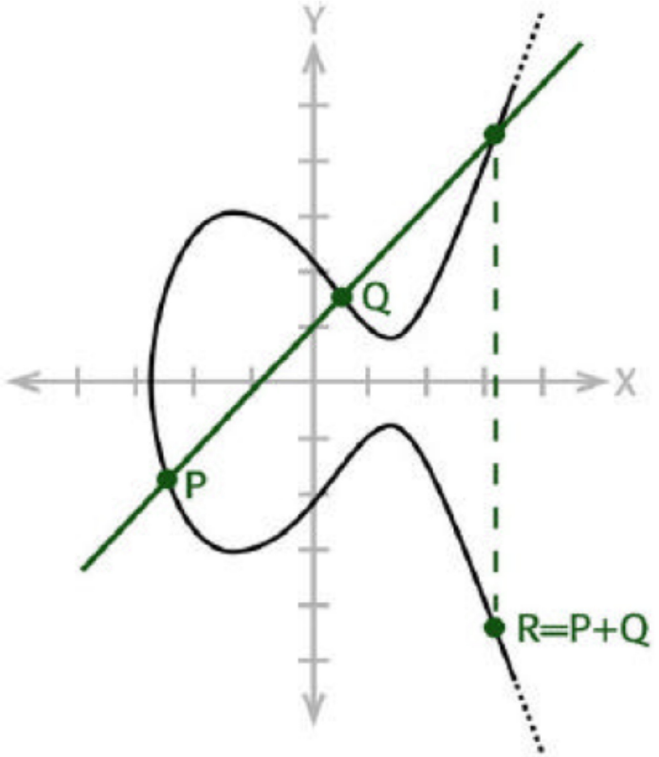
zpráv.

Systémy nad eliptickými křivkami

Problémem „klasického“ počítání kryptografických algoritmů nad $\mathbf{Z}_n$ je značná existence relativně rychlých faktorizačních či logaritmujících algoritmů
trikem je přenést počítání známých algoritmů do algebraických struktur, kde by tyto kryptoanalytické metody nefungovaly

Mějme $q = p^r$, $p \geq 5$ a vhodné a a $b \in \mathrm{F}_q$. Eliptickou křivkou nad okruhem F_q rozumíme množinu bodů

$$E(\mathrm{F}_q) = \{(x, y) \in \mathrm{F}_q^2 \mid y^2 = x^3 + ax + b\} \cup \{\Theta\}$$



Θ nazýváme *bod v nekonečnu*. Plní úlohu nulového prvku.

Nechť $P = (x, y)$ je bod křivky. Zavedeme $-P = (x, -y)$, $P + Q$ je průsečík křivky s přímkou definovanou body P a Q, pokud $P = Q$, bereme tangentu. Označení nP používáme pro $\underbrace{P + P + \cdots + P}_{n\text{-krát}}$

Obdobou umocňování přirozených čísel je zde právě uvedené násobení. Problém hledání diskrétního logaritmu zde má podobu:
Pro dané P, Q nalézt n takové, že

Q = nP

uvedený popis problému diskrétního logaritmu nad eliptickou křivkou přímo umožňuje implementovat El-Gamal kryptosystém, nebo D-H.
podobně je možné zavést problém faktorizace a definovat Eulerovu funkci $\varphi(n)$, což umožňuje implementovat RSA

Konstrukce kryptosystému

nad takto definovanou grupou můžeme používat obvyklé šifrovací algoritmy, jako El-Gamalův kryptosystém, RSA, Diffie-Hellmanův systém výměny klíčů.
běžné umocňování pouze nahradíme sčítáním

Analýza systémů nad eliptickými křivkami

Obecně se má za to, že použití eliptických křivek přináší zvýšení bezpečnosti 8(1)500]TJ ET Q Q q 4503 236.668 33 1236.668 33 6.6 6-85 dí ž ů

Dešifrování

spočítáme vektor

$$\hat{c} = cP^{-1}$$

ten dekódujeme kódem K na vektor

$$\hat{m} = Dek_K(\hat{c})$$

a odsud už lze získat původní zprávu

$$m = \hat{m}S^{-1}$$

Korektnost dešifrování

$$\hat{c} = cP^{-1} = \left(m\hat{G} + z\right)P^{-1} = (mSGP + z)P^{-1} = mSG + zP^{-1}$$

teď uvažme, že $\hat{m} = mS$ a položme $\hat{z} = zP^{-1}$, potom

$$\hat{c} = \hat{m}G + \hat{z}$$

v tomto bodě lze korektně dekódovat, $\hat{z}$ je jen permutací původního chybového vektoru, takže $Dek_K(\hat{c}) = \hat{m}$ a dosazením za $\hat{m}$ získáme žádaný výsledek.

Implementace

obvykle se používají binární Goppovy kódy
bezpčnost algoritmu závisí na délce kódu a redundanci, v současné době bychom volili přinejmenším $n = 2048$, $k = 1278$, $d = 81$

Podepisovací schémata

digitální podpis asociuje zprávu a (jejího) odesílatele
obecně v rámci podepisovacího procesu se nejprve provede mapování prvku *prostoru zpráv* do tzv. *podepisovacího prostoru* (zpravidla přidáním redundance, paddingem, hashováním, ...), odkud jej podepisovací schéma (na základě tajného klíče) mapuje do prostoru podpisů

klasifikace podepisovacích schémat

- s příponou (dig. signature with appendix) – potřebují původní zprávu jako vstup verifikačního procesu
- s obnovou zprávy (dig. signature with message recovery) – původní zpráva je rekonstruována z dat vlastního podpisu

v závislosti na tom, zda existuje pouze jedno mapování (bijekce) z prostoru zpráv do podepisovacího prostoru rozdělujeme podepisovací schémata na

- randomizovaná
- deterministická

Obecný postup podepisování s příponou

zvolíme mapování *k* zajišťující redundanci

- spočítáme $\tilde{m} = \langle m \rangle$
- podpisem je $s = S_{A,k}(\tilde{m})$, kde $S_{A,k}$ je podepisovací algoritmus závislý na tajném klíči entity *A* a konkrétním algoritmu pro přidání redundance *k*

pro hashování se volí vhodná CRHF

pro verifikaci je třeba podpis *s* a původní zpráva *m*

- spočítáme $\tilde{m} = \langle m \rangle$ a $u = V_A(\tilde{m}, s)$
- podpis je přijat pokud *u* je true

Obecný postup podepisování s obnovou zprávy

- zvolíme mapování *k* zajišťující redundanci
- spočítáme $\tilde{m} = R(m)$
- podpisem je $s = S_{A,k}(\tilde{m})$, kde $S_{A,k}$ je podepisovací algoritmus závislý na tajném klíči entity *A* a konkrétním algoritmu pro přidání redundance *k*

funkce pro doplnění redundance *R* musí být invertibilní a je veřejně známa, podepisovací prostor, do kterého mapuje prostor zpráv musí být podstatně větší, jinak bude schéma náchylné na existenciální podvržení, tj. bude možné sestavovat páry zpráva-podpis bez znalosti tajného klíče (byť bez možnosti kontrolovat obsah zprávy)

pro verifikaci je třeba podpis *s* a původní zpráva *m*

- spočítáme a $\tilde{m} = V_A(s)$
- podpis je přijat pokud $\tilde{m}$ je prvkem obrazu prostoru zpráv v podepisovacím prostoru
- rekonstruujeme původní zprávu $m = R^{-1}(\tilde{m})$

Podepisovací schéma RSA

deterministické podepisovací schéma s obnovou zprávy
založeno na obtížnosti faktorizace velkých čísel

Inicializace – generování klíčů

stejně jako v případě RSA šifrování zvolíme dvě velká prvočísla *p* a *q*

spočítáme $n = pq$ a $\phi = (p - 1)(q - 1)$
zvolíme e nesoudělné s ϕ a spočítáme d tž. $ed \equiv 1 (\mathrm{mod}\, \phi)$
veřejným klíčem je dvojice (n, e), tajným klíčem d

Podpis

- spočítáme $\tilde{m} = R(m)$
- a následně podpis $s = \tilde{m}^d \bmod n$

Ověření podpisu

- spočítáme $\tilde{m} = s^e \bmod n$ a ověříme, že není poškozena redundance
- obnovíme původní zprávu $m = R^{-1}(\tilde{m})$

Bezpečnost podp. schématu RSA
schéma trpí vlastností multiplikativnosti (či. homomorfismu), tj. pokud znám podpis dvou zpráv, mohu bez znalosti klíče sestavit podpis třetí zprávy, která je jejich součinem, pokud by funkce pro přidání redundance byla sama multiplikativní
volba parametrů odpovídá volbě pro RSA šifrování

Pozn: Pozor, tímto způsobem zpravidla RSA nepoužíváte, v běžných knihovnách se RSA používá jako schéma s příponou.

Rabinovo podepisovací schéma

podobné RSA, ale používá sudý pevně stanovený veřejný exponent e
podepisovací prostor je prostorem kvadratických reziduí mod n

Inicializace – generování klíčů
každý účastník vygeneruje dvě velká prvočísla p a q a spočítá $n = pq$
n je veřejným klíčem, dvojice (p, q) tajným klíčem

Podpis

- spočítáme $\tilde{m} = R(m)$
- podpisem je $s = \sqrt[e]{\tilde{m}} (\mathrm{mod}\, n)$

obvykle se e volí 2

není jisté, že výsledné $\tilde{m}$ je skutečně kvadratickým reziduem, existuje modifikace schématu, která to zajistí, případně je možné přidat ke zprávě část náhodných dat, jejichž změnou docílíme residuosity (v průměru 2 pokusy)

Ověření podpisu

- spočítáme $\tilde{m} = s^e \pmod n$
- ověříme, že není poškozena redundance v $\tilde{m}$
- obnovíme původní zprávu $m = R^{-1}(\tilde{m})$

Bezpečnost Rabinova podepisovacího schématu
bezpečnost zavisí na kvalitě funkce přidávající redundanci

Podepisovací schéma ElGamal

randomizované podepisovací schéma s příponou
je zobecněním principu DSA

Inicilizace – generování klíčů
každý účastník zvolí náhodně prvočíslo p a generátor α multiplikativní grupy Z_p^*
dále vybere náhodné číslo a, $1 \le a \le p-2$
a spočítá $y = \alpha^a \bmod p$
veřejným klíčem je trojice (p, α, y), tajným klíčem je a

Podepisování

- zvolíme náhodné celé číslo k, $1 \le k \le p-2$ nesoudělné s $p-1$
- spočítáme $r = \alpha^k \bmod p$ a $s = k^{-1}(\langle m \rangle - ar) \bmod (p-1)$

podpisem je dvojice (r, s)

Ověření podpisu

- ověřovatel verifikuje, že $1 \le r \le p-1$
- a spočítá $v_1 = y^r r^s \bmod p$ a $v_2 = \alpha^{\langle m \rangle}$

podpis je přijat pokud $v_1 = v_2$ a platí shora uvedené požadavky na r

Bezpečnost
schéma je bezpečné pokud zůstává těžký problém diskrétního logaritmu

je nutné volit k náhodně pro každou podepisovanou zprávu, v opačném případě je možné s velkou pravděpodobností k zjistit a následně dopočítat tajný parametr a, neboť $k = \frac{\langle m_1 \rangle - \langle m_2 \rangle}{s_1 - s_2} \bmod (p-1)$

pro volbu velikosti parametrů platí přibližně totéž, co pro RSA

DSA – data signature algorithm

založen na problému diskrétního logaritmu
podepisovací schéma s příponou (appendix), pro hashování se používá SHA-1
standardizováno jako FIPS186 (DSS)

Inicializace – generování klíčů

- každý účastník zvolí náhodně prvočísla q a p t.ž. $q \mid (p-1)$
- a generátor $\alpha = g^{(p-1)/q} \bmod p$ pro libovolně zvolené g aby $\alpha \neq 1$
- dále zvolí náhodně a t.ž. $1 \leq a \leq q-1$
- a spočítá $y = \alpha^a \bmod p$

veřejným klíčem je čtveřice (p, q, α, y), tajným klíčem je a.

Podpis

pro podpis zprávy m:

- zvolíme náhodně k, $0 < k < q$
- spočítáme $r = (\alpha^k \bmod p) \bmod q$, $s = k^{-1}(\langle m \rangle + ar) \bmod q$

podpisem je pár (r, s)

Ověření podpisu

- ověřovatel verifikuje, že $0 < r < q$ a $0 < s < q$
- spočítá $u_1 = s^{-1} \cdot \langle m \rangle \bmod q$ a $u_2 = s^{-1} r \bmod q$
- a následně $v = (\alpha^{u_1} y^{u_2} \bmod p) \bmod q$

podpis je přijat pokud $v = r$ a platí shora uvedené požadavky na r a s

Bezpečnost DSA

q se volí ve velikosti 160 bitů, zatímco p má délku násobku 64 mezi 512 a 1024 bity, doporučuje se alespoň 768 bitů

bezpečnost se opírá o obtížnost počítání diskrétního logaritmu v Z_p^* a jeho cyklické podgrupě o řádu q
bezpečnostní vlastnosti jsou podobné jako v případě El-Gamalova podepisovacího schématu.

Podepisovací schéma Merkle pro jednorázové podpisy

umožňuje s daným tajným klíčem podepsání právě jedné zprávy
při podepsání další zprávy je možná fabrikace podpisu
je nezbytná důvěryhodná třetí strana na validaci parametrů algoritmu

Inicializace

zvolíme $t = n + \lg n + 1$ náhodných řetězců k_1, k_2, ... k_t, každý o délce l a uchováme je v tajnosti
spočítáme $v_i = \langle k_i \rangle$ pro $1 \le i \le t$ pomocí vhodné CRHF
veřejným klíčem je t-tice (v_1, v_2, ... v_t,), tajným (k_1, k_2, ... k_t,)

Podpis

pro podpis zprávy m o délce n:
- spočítáme c .... počet nul ve zprávě m
- a sestavíme $w = m|c = (a_1, a_2, ... a_t,)$
- podpisem je výběr (s_1, s_2, ... s_u,), který vznikne z (k_1, k_2, ... k_t,) vybráním těch k_i, kde a_i=1

Ověření podpisu

- spočítáme c .... počet nul ve zprávě m
- a sestavíme $w = m|c = (a_1, a_2, ... a_t,)$
- ověříme, že $v_{i_j} = \langle s_j \rangle$ pro všechny pozice, kde a_i=1

Bezpečnost Merkelova schématu

pokud je použita kvalitní CRHF, je schéma bezpečné

Neodmítnutelné (Undeniable) podpisy

... k ověření podpisu je nezbytná spolupráce podepisujícího

Podepisovací schéma Chaum-van Antwerpen

neodmítnutelné podepisovací schéma

Inicializace – generování klíčů

- každý účastník zvolí náhodně prvočíslo $p = 2q + 1$ pro nějaké prvočíslo q
- náhodně zvolí β v Z_p^* a spočítá $\alpha = \beta^{(p-1)/q} \bmod p$ tak, aby $\alpha \neq 0$
- dále náhodně vybere $0 < a <$ q a spočítá $y = \alpha^a \bmod p$

veřejným klíčem je trojice (p, α, y), tajným klíčem a

Podpis

pro podpis zprávy m podepisující

spočítá podpis $s = m^a \bmod p$

Ověření podpisu

- ověřovatel zvolí náhodná čísla x_1, x_2, tž. $0 < x_i <$ q
- spočítá $z = s^{x_1} y^{x_2} \bmod p$ a výsledek zašle podepisujícímu
- podepisující zašle ověřujícímu $w = (z)^{a^{-1}} \bmod p$, kde $aa^{-1} \equiv 1 \bmod q$
- ověřující spočítá $w' = m^{x_1} \alpha^{x_2} \bmod p$
- podpis je přijat pokud w = w′

Odmítnutí podpisu

používá se pro ověření, zda podepisovatel odmítá potvrdit platný podpis, či zda podpis je podvrhem

$c' = \left(w' \alpha^{-x_2'}\right)^{x_1} \bmod p$

- ověřovatel zvolí náhodná čísla x_1, x_2, tž. $0 < x_i <$ q a spočítá $z = s^{x_1} y^{x_2} \bmod p$ a výsledek zašle podepisujícímu
- podepisující zašle ověřujícímu $w = (z)^{a^{-1}} \bmod p$, kde $aa^{-1} \equiv 1 \bmod q$
- pokud $w = m^{x_1} \alpha^{x_2} \bmod p$ ověřovatel akceptuje a ukončí protokol
- ověřovatel zvolí náhodná čísla x'_1, x'_2, tž. $0 < x_i <$ q a spočítá $z' = s^{x_1'} y^{x_2'} \bmod p$ a výsledek zašle podepisujícímu
- podepisující zašle ověřujícímu $w' = (z')^{a^{-1}} \bmod p$, kde $aa^{-1} \equiv 1 \bmod q$
- pokud $w' = m^{x_1'} \alpha^{x_2'} \bmod p$ ověřovatel akceptuje a ukončí protokol
- ověřovatel spočítá $c = \left(w \alpha^{-x_2}\right)^{x_1'} \bmod p$ a

- pokud $c = c'$, ověřovatel potvrdí, že podpis je podvrhem, v opačném případě se domnívá, že podepisovatel odmítá potvrdit platný podpis

SignCryption

… tedy česky podpifrování
algoritmy provádí zároveň podepsání zprávy a její ochranu šifrováním
výsledná zpráva je kratší než při samostatné aplikaci podpisu a následném šifrování, sníží se i celková výpočetní náročnost

ElGamal signcription

pro konstrukci schématu je potřeba předem zvolit vhodnou hashovací funkci (budeme používat klíčované hashování, tj. $\langle m \rangle_k = \langle k, m \rangle = kh$) a symetrický šifrovací algoritmus
schéma zvládne kombinaci podpis + šifrování za 58% času a výsledná zpráva zabírá cca 70% prostoru, než kdyby bylo použita kombiance ELGamal šifrování DSS podpis

Inicializace
zvolíme společné prvočíslo p a q tak aby $q \mid p - 1$ a celé číslo g , které má řád q mod p v [1 … p-1]
každý účastník i si zvolí vlastní pár (x_i, y_i), *t.ž.* $y_i = (g)^{x_i} \bmod p$

Podpiso-šifrování
chce-li odesilatel a odeslat zprávu příjemci b, zvolí náhodně x a spočítá
$k = \langle y_b{}^x \rangle \bmod p$
dále rozdělí k na k_1 a k_2 a spočítá
$r = \langle m \rangle_{k_2}$
$s = x / r + x_a \bmod q$
$c = \{m\}_{k_1}$
a výsledek (r, s, c) zašle příjemci b

Ověř-dešifrování
příjemce spočítá

$$k' = \left\langle \left(y_a g^r \right)^{s x_b} \bmod p \right\rangle$$

rozdělí k' na k'_1 a k'_2 a spočítá

$$m' = \{c\}_{k'_1}^{-1}$$

zprávu m přijme pouze tehdy, pokud $r = \langle m' \rangle_{k_2}$

… a další kryptografické triky

Pravděpodobnostní šifrování

zajišťuje, že stejný plaintext je při opakovaném použití stejného klíče šifrován na jiný zašifrovaný text

Kryptosystém Blum – Goldwasser

založen na složitosti faktorizace celých čísel
jádrem BBS generátor náhodných čísel

inicializace

každý účastník zvolí dvě prvočísla p a q kongruentní s 3 mod 4
$n = pq$ ... tzn. n je Blumovo číslo
pomocí rozšířeného Euklidova algoritmu určíme a a b tž. $ap + bq = 1$

n je veřejný klíč, (p, q, a, b) je tajný klíč

šifrování

při inicializaci šifrování zvolíme náhodně r a spočítáme $x_0 = r^2 \bmod n$
tzn. x_0 je kvadratické reziduum mod n

i-tý blok plaintextu p_i šifrujeme takto:

$$x_i = x_{i-1}^2 \bmod n$$

$$c_i = p_i \otimes x_i$$

výsledkem šifrování zpráva $(c_1, c_2, \ldots, c_t, x_{t+1})$

dešifrování

spočítáme

$u = x_{t+1}^{((p+1)/4)^{t+1} \bmod p-1} \bmod p$ a $v = x_{t+1}^{((q+1)/4)^{t+1} \bmod q-1} \bmod q$

odsud

$x_0 = vap + ubq \bmod n$

dále obdobně jako v případě šifrování spočítáme x_i pro dešifrování i-tého bloku

korektnost dešifrování

uvažme

$x_{t+1}^{(p+1)/4} \equiv \left(x_t^2\right)^{(p+1)/4} \equiv x_t^{(p-1)/2} x_t \equiv x_t (\bmod p)$, neboť $x_t^{(p+1)/2} \equiv 1 (\bmod p)$ (x_t je kvadratické reziduum), opakováním dostaneme

$u \equiv x_{t+1}^{((p+1)/4)^{t+1}} \equiv x_0 (\bmod p)$

obdobně lze dovodit

$v \equiv x_{t+1}^{((q+1)/4)^{t+1}} \equiv x_0 (\bmod q)$

protože

$ap + bq = 1$, $vap + ubq \equiv x_0 (\bmod p)$ a $vap + ubq \equiv x_0 (\bmod q)$

nutně $x_0 = vap + ubq \bmod n$ je zkonstruováno správně

Bezpečnost

algoritmus je ekvivalentní s problémem faktorizace velkých čísel

velikost n volit obdobně jako v případě RSA

náchylný na choosen-ciphertext attack

Srovnání symetrických a asymetrických šifer

Vlastnost	**Symetrické**	**Asymetrické**
Rychlost (instrukcí/byte)	x *1	x * $10^3 \div 10^4$
Počet klíčů	o(n^2)	o(n)
Autentizace protistrany	nativně	nutno explicitně
Anonymita příjemce	protokol	protokol/nativně
Anonymita odesílatele	protokol	nativně
Nepopiratelnost	nelze	nativně
Veřejně ověřitelné operace	s arbitrem	nativně
Implementace	triviální	dlouhá aritmetika

Klíče	obvykle triviání spec. vlastnosti
Bezpečnost	definice parametry
Malý prostor zpráv	bez problémů nešifrují

Hybridní šifra

(téměř) ideální kompromis

$$C = \{k\}_{Kpbl_a} \mid \{P\}_k$$

kde k se volí náhodně pro každou zprávu