УДК 512.624.95

Рябухо О.М., Турка Т.В., Шерстюк О.С.

[1] *доцент кафедри алгебри, ДДПУ*

[2] *доцент кафедри алгебри, ДДПУ*

[3] *студентка 5 курсу фізико-математичного факультету, ДДПУ*

e-mail: ren_elena@mail.ru, tvturka@mail.ru

ПРАКТИЧНІ СХЕМИ РЕАЛІЗАЦІЇ ПРОТОКОЛІВ ЦИФРОВОГО ПІДПИСУ

Дана робота присвячена вивченню проблеми створення електронного цифрового підпису. Надано опис алгоритмів створення електронних підписів RSA, Ель Гамала, DSA та підтвердження їх автентичності. Проведений порівняльний аналіз описаних алгоритмів. Представлені приклади створення та верифікації цифрового підпису.

Ключові слова: *цифровий підпис, алгоритм підпису, алгоритм верифікації, криптографічні протоколи, дискретний логарифм*

Вступ

Із появою у 1976 році ідеології відкритого ключа криптографічна практика почала використовувати фундаментальні результати теорії чисел і сучасної алгебри. Деякі з асиметричних алгоритмів можуть використовуватися для генерування цифрового підпису.

Схема цифрового підпису — це метод підписування електронного документу, який надсилається через комп'ютерну мережу. Протокол цифрового підпису має дві складові: алгоритм підпису та алгоритм його верифікації.

Першою найбільш відомою в світі системою ЕЦП стала система RSA, математична схема якої була розроблена в 1977 році в Масачусетському технологічному інституті США і Роном Рівестом, Аді Шаміром і Леонардом Ейдельманом.

Схема підпису Ель Гамала була описана в [2], стандарт цифрового підпису (DSS) вперше був опублікований в серпні 1991 року, але офіційний статус стандарту одержав в грудні 1994 року [1]. В липневому номері Communications of the ACM з 1992 року містяться описи DSS.

Метою роботи є вивченні теоретико-числових алгоритмів для створення та реалізації електронного підпису документу.

1. Цифровий підпис

Цифровим підписом називають блок даних, згенерований з використанням деякого секретного ключа. При цьому за допомогою відкритого ключа можна перевірити, що дані дійсно згенеровані за допомогою цього секретного ключа. Алгоритм генерації цифрового підпису мусить бути таким, щоб без застосування секретного ключа було неможливо створити підпис, автентичність якого можна підтвердити.

Цифрові підписи використовуються для того, щоб підтвердити, що повідомлення прийшло дійсно від даного відправника (у припущенні, що лише відправник володіє секретним ключем, відповідним його відкритому ключу). Також підписи використовуються для створення штампу часу (timestamp) на документах: сторона, якій ми довіряємо, підписує документ з штампом часу за допомогою свого секретного ключа і, таким чином, підтверджує, що документ вже існував в момент, оголошення в штампі часу.

Цифрові підписи також можна використовувати для посвідчення (сертифікації — to certify) того, що документ належить певній особі. Це робиться так: відкритий ключ і інформація про той, кому він належить підписуються стороною, якої довіряємо. При цьому довіряти підписуючій стороні ми можемо на підставі того, що її ключ був підписаний третьою стороною. Таким чином виникає ієрархія довіри. Очевидно, що деякий ключ має бути коренем ієрархії (тобто йому ми довіряємо не тому, що він кимсь підписаний, а тому, що ми віримо a-priori, що йому можна довіряти). У централізованій інфраструктурі ключів є дуже невелика кількість кореневих ключів мережі (наприклад, наділені повноваженнями державні агентства; їх також називають сертифікаційними агентствами — certification authorities). У розподіленій інфраструктурі немає необхідності мати універсальні для всіх кореневі ключі, і кожна із сторін може довіряти своєму набору кореневих ключів (скажемо своєму власному ключу і ключам, нею підписаним). Ця концепція носить назву мережі довіри (web of trust).

Цифровий підпис документа зазвичай створюється в такий спосіб: з документа генерується так званий дайджест (message digest) і до нього додається інформація про той, хто підписує документ, штамп часу і інше. Рядок, що вийшов, далі зашифровується секретним ключем що підписує з використанням того або іншого алгоритму. Зашифрований набір біт, що вийшов, і є підписом. До підпису зазвичай прикладається відкритий ключ того, що підписує. Одержувач спочатку вирішує для себе чи довіряє він тому, що відкритий ключ належить саме тому, кому повинен належати (за допомогою мережі довіри або апріорного знання), і потім дешифрує підпис за допомогою від-

критого ключа. Якщо підпис нормально дешифрувався, і її вміст відповідає документу (дайджест і ін.), то повідомлення вважається підтвердженим.

2. Електронний підпис у системі RSA

Нагадаємо, що в системі RSA кожен абонент X має пару ключів — загальновідомий відкритий (n_X, e_X) і таємний d_X, який знає лише X і ніхто інший. Таким чином, будь-хто може скористатися алгоритмом шифрування E_X абонента X, але тільки він сам володіє алгоритмом дешифрування D_X. Важливим є виконання таких співвідношень для довільного повідомлення M:

$$D_X(E_X(M)) = E_X(D_X(M)) = M. \tag{1}$$

Ці співвідношення зводяться до рівностей $(M^{eX})^{dX} = (M^{dX})^{eX} = M$ в Z_{nX}, і виражають той факт, що шифруюче відображення E_X та дешифруюче D_X є взаємно оберненими.

Припустимо, що Аліса хоче послати повідомлення M Бобу і переконати його, що це повідомлення дійсно послане Алісою, а не зловмисником, який маскується під Алісу. Для цього пропонується такий протокол, в якому (E_A, D_A) та (E_B, D_B) — алгоритми шифрування та дешифрування Аліси та Боба.

- Аліса обчислює $C = E_B(D_A(M))$ і посилає C Бобові.
- Боб, отримавши C, обчислює $M = E_A(D_B(C))$.

Коректність протоколу зводиться до рівності

$$E_A(D_B(E_B(D_A(M)))) = M,$$

яка випливає із співвідношень (1).

Ефективність. Аліса та Боб використовують ефективні алгоритми шифрування та дешифрування криптосистеми RSA. Зауважимо, що Аліса використовує свій приватний алгоритм D_A та відомий всім алгоритм E_B. Теж саме із Бобом – він використовує особистий алгоритм D_B і загальновідомий алгоритм E_A.

Конфіденційність. Під конфіденційністю цього протоколу ми розуміємо його надійність як звичайної криптосистеми для пересилання повідомлень. Перед зловмисником, який підслухав розмову і одержав криптотекст C, стоїть завдання визначити M із $E_B(D_A(M))$, тобто зламати криптосистему RSA.

Достовірність. Під достовірністю ми розуміємо таку властивість протоколу підпису: не тільки Боб, а й хто завгодно може впевнитися, що відправником повідомлення є саме Аліса. Що стосується описаного протоколу, то Боб, який успішно розшифрував криптотекст C і прочитав повідомлення M, дійсно має вагомі підстави вважати, що воно послане Алісою.

Справді, криптотекст має структуру $C = E_B(D_A(M))$, тому інтуїтивно переконливим є висновок, що особа, яка послала C, мала би знати алгоритм D_A. Але алгоритм D_A відомий лише Алісі (якщо тільки зловмиснику не вдалося зламати систему RSA).

Недоліки алгоритму цифрового підпису RSA.

При обчисленнях в системі цифрового підпису RSA необхідно перевіряти велику кількість додаткових умов, що створює великі труднощі. у разі невиконання якоїсь з цих умов дає можливість зловмиснику сфальшувати цифровий підпис.

Для забезпечення криптостійкості цифрового підпису RSA на рівні національних стандартів необхідно використовувати при обчисленнях цілі числа не менші за 2512 (або 10154), а це збільшує обчислювальні затрати на 20...30% у порівнянні з другими алгоритмами цифрового підпису для забезпеченні того ж рівня криптостійкості.

Цифровий підпис RSA уразливий до так званої мультиплікативної атаки. Алгоритм цифрового підпису RSA дозволяє зловмиснику без знання секретного ключа D сформувати підписи під тими документами, у яких результат хешування можна обчислити як добуток результатів хешування вже підписаних документів.

3. Система цифрового підпису Ель Гамала

Більш надійний і зручний для реалізації на персональних комп'ютерах алгоритм цифрового підпису був розроблений в 1984 році американцем арабського походження Тахером Ель Гамалем. В 1991 році Національний інститут стандартів і технологій (NIST) США обґрунтував перед комісією Конгреса США вибір алгоритму цифрового підписи Ель Гамала в якості основи для національного стандарту.

Назва EGSA походить від слів EI GamaI Signature Algorithm (алгоритм цифрового підпису Эль Гамаля). Ідея EGSA полягає на використанні задачі дискретного логарифмування.

Як і криптосистеми для забезпечення конфіденційності повідомлень, системи цифрового підпису допускають ймовірнісну модифікацію. у ймовірнісних системах підпису алгоритм $SIGN$ є не детермінованим, а ймовірнісним.

Генерування ключів. Вибирають велике просте p, а також число g, $1 < g < p-1$, яке має в мультиплікативній групі Z великий порядок. В ідеальному випадку g є первісним коренем за модулем p. Числа p і g не є таємницею і перебувають в загальному користуванні. Кожен абонент вибирає собі випадкове число a у проміжку від 1 до $p-1$, і обчислює $h = g^a \bmod p$.

Відкритий ключ: p, g, h.

Таємний ключ: a.

Підписування. Аліса виробляє свій підпис S на повідомленні M таким чином. Вона

- вибирає випадкове число $r \in Z_{p-1}^*$;
- обчислює $s_1 = g^r \bmod p$;
- обчислює $r' = r^{-1} \bmod (p-1)$;
- обчислює $s_2 = (M - as_1)r' \bmod (p-1)$;
- покладає $S = (s_1, s_2)$.

Підтвердження підпису.

- Боб перевіряє, чи $g^M \equiv h^{s_1} s_1^{s_2} (\bmod p)$.

Коректність. З правил обчислення r' і s_2 випливає, що $M \equiv as_1 + rs_2 (\bmod p-1)$. Звідси з використанням теореми Ойлера отримуємо

$$g^M = g^{as_1 + rs_2 + k(p-1)} \equiv (g^a)^{s_1} (g^r)_2^s \equiv h^{s_1} s_1^{s_2} (\bmod p).$$

Приклад підписів Ель Гамала.

Приклад 1. *Нехай* $p = 467, \alpha = 2$ *і* $\beta = 132$. *Припустимо Оскар вибере* $i = 99$, *а також* $j = 179$; *тоді* $j^{-1} mod\ (p-1) = 151$. *Зараз Оскар виконує наступні обчислення:*

$$\gamma = 2^{99} 132^{179} mod\ 467 = 117$$

$$\delta = -117 \cdot 151 mod\ 466 = 41$$

$$x = 99 \cdot 41 mod\ 466 = 331.$$

Пара $(117, 41)$ *є справжнім підписом для відомості* 331, *що перевіримо через наступне обчислення:*

$$132^{117} 117^{41} \equiv 303 (mod\ 467),$$

а також

$$2^{331} \equiv 303 (mod\ 467).$$

Отже підпис є автентичний.

Схема цифрового підпису Ель Гамала має ряд переваг порівняно зі схемою цифрового підпису RSA.

При заданому рівні стійкості алгоритму цифрового підпису цілі числа, що застосовуються в обчисленнях, має запис на 25% коротший, що зменшує складність обчислень майже вдвічі і дозволяє значно скоротити об'єм використаної пам'яті.

При виборі модуля p достатньо перевірити його простоту і наявність у числа $(p-1)$ великого простого множника (тобто всього дві умови, які порівняно просто перевіряються).

Процедура формування підпису за схемою Ель Гамала не дозволяє обчислювати цифрові підписи під новими повідомленнями без знання секретного ключа на відміну від RSA.

Однак алгоритм цифрового підпису Ель Гамала має і деякі недоліки порівняно зі схемою RSA. А саме, довжина цифрового підпису в 1,5 рази довша, а це в свою чергу збільшує час обчислення.

4. Алгоритм цифрового підпису DSA

У 1991 році NIST запропонував для алгоритму цифрового підпису DSA (Digital Signature Algorithm) стандарт DSS (Digital Signature Standard), в основу якого покладено алгоритми Ель Гамала і RSA.

Генерування ключів. Вибирають велике просте число p таке, що $p-1$ має досить великий простий дільник q. Стандарт вимагає, щоб $2^{512} < p < 2^{1024}$ і $q > 2^{216}$. Далі вибирають у групі Z_p^* довільний елемент h порядку p. Параметри p, q, h не становлять таємниці і є спільними для всіх абонентів мережі.

Аліса вибирає випадкове число a в діапазоні від 0 до $q-1$ і обчислює число $b = h^a \bmod p$. Її ключі формуються так.

Відкритий ключ: b таке, що $b = h^a \bmod p$.

Таємний ключ: a.

Підписування. Алгоритм підписування використовується вкорочуючу функцію f, в якості якої DSS пропонує функцію SHA з довжиною вкорочення 160 бітів. Щоб виробити свій підпис S для повідомлення M, Аліса

- вибирає випадкове число r в діапазоні від 0 до $q-1$;
- обчислює $r' = r^{-1} \bmod q$;
- обчислює $s_1 = (h^r \bmod p) \bmod q$;
- обчислює $s_2 = (r'(f(M) + as_1)) \bmod q$;
- формує підпис $S = (s_1, s_2)$.

Підтвердження підпису. Отримавши повідомлення M із підписом $S = (s_1, s_2)$, Боб

- обчислює $s' = s_2^{-1} \bmod q$;
- обчислює $u_1 = (f(M)s') \bmod q$;
- обчислює $u_2 = (s_1, s') \bmod q$;
- обчислює $t = (h^{u_1} b^{u_2} \bmod p) \bmod q$;
- перевіряє рівність $t = s_1$.

Коректність. Припустимо, що підпис $S = (s_1, s_2)$ отримано згідно з алгоритмом підписування. Досить перевірити, що $h^r \equiv h^{u_1} b^{u_2} (\mathrm{mod}\ p)$. Оскільки $b = h^a \mathrm{mod}\ p$, то остання конґруенція рівносильна такій: $h^r \equiv h^{u_1 = u_2} (\mathrm{mod}\ p)$. Підставивши у праву частину вирази для u_1 та u_2, отримаємо

$$r^{'}(u_1 + au_2) \equiv r^{'}(f(M)s^{'} + as_1 s^{'}) \equiv r^{'}(f(M) + as_1)s^{'} \equiv s_2 s^{'} \equiv 1 (\mathrm{mod}\ q),$$

що завершує перевірку коректності.

Приклад підпису стандарту.

Приклад 2. *Приймемо* $q = 101$ *і* $p = 78q + 1 = 7879$. *Число* 3 *є елемент первісний в* Z_{7879}, *потім візьмемо*

$$\alpha = 3^{78} mod\ 7879 = 170.$$

Хай $a = 75$. *Тоді* $\beta = \alpha^a mod\ 7879 = 4567$.

Припустимо зараз, Боб хоче підписати відомість $x = 22$ *і вибирає випадково величину* $k = 50$, *з чого отримує*

$$k^{-1} mod\ 101 = 99.$$

Тоді

$$\gamma = (170^{50 mod\ 7879}) mod\ 10,1; \quad \gamma = 2518 mod\ 101; \quad \gamma = 94,$$

а також

$$\delta = (22 + 75 \cdot 94) 99 mod\ 101; \quad \delta = 97.$$

За допомогою наступних обчислень верифікує підпис $(94, 97)$ *під відомістю* 22*:*

$$\delta^{-1} = 97^{-1} mod\ 101 = 25,$$

$$e_1 = 22 \cdot 25 mod\ 101 = 45,$$

$$e_2 = 94 \cdot 25 mod\ 101 = 27,$$

$$(170^{45} 4567^{27} mod\ 7879) mod\ 101 = 2518 mod\ 101 = 94.$$

Отже підпис є справжній.

Порівняно з алгоритмом цифрового підпису Ель Гамала алгоритм DSA має певні переваги.

При будь-якому допустимому рівні стійкості, тобто при будь-якій парі чисел p и h (від 512 до 1024 біт), числа q, a, s_1, s_2 мають довжину 160 біт, скорочуючи довжину підпису до 320 біт.

Більшість операцій з числами r, a, s_1, s_2 при обчисленнях виконуються за модулем числа h довжиною 160 біт, що скорочує час обчислення підпису. При перевірці підпису більшість операцій з числами u_1, u_2, s', t виконуються за модулем числа q довжиною 160 біт, що скорочує об'єм пам'яті і час обчислення. Недоліком алгоритму DSA є те, що при підписуванні і при перевірці підпису необхідно виконувати складні операції ділення за модулем q, що не дозволяє одержувати максимальну швидкість дій.

Висновки

Як бачимо, кожна з схем цифрового підпису має свої недоліки та переваги.

Хоча на практиці в алгоритмах цифрових підписів використовуються великі числа, для ілюстрації описаних алгоритмів наведені приклади з невеличкими числами.

Слід зазначити, що реальне виконання алгоритму DSA може бути прискорене за допомогою проведення попередніх обчислень. Значення s_1 не залежить від повідомлення та його хеш-значення $f(M)$. Можна наперед створити рядок випадкових значень r і потім для кожного з цих значень обчислити значення s_1. Можна також заздалегідь обчислити обернені значення r'. Ці попередні обчислення значно прискорять роботу алгоритму DSA.

Отже, кожна з схем має свої слабкі та сильні сторони, тому вибір оптимального алгоритму залежить від обставин, при яких він буде використовуватись: обчислювальні можливості, доступні об'єми пам'яті, існування чи відсутність безпечних каналів зв'язку та кількість користувачів. Лише оцінивши всі ці параметри можна обрати зручну та ефективну схему цифрового підпису.

Література

1. Digital signature standart. — National Bureau of standarts FIPS Publication, 186. — 1994.
2. *ElGamal T.* A public key cryptosystem and a signature scheme based on discrete logarithms. — IEEE Transactions on Information Theory. — 1985. — 31. — P. 469–472.
3. *Koblitz N.* A Course in Number Theory and Cryptography. — Springer–Verlag, New York, Inc., 1994. — P. 235.
4. *Salomaa A.* Public–Key Cryptography. — Springer–Verlag, 2 ed., 1996, P. 271.
5. *Stinson D.R.* Kryptografia. Theory and Practice. — Copyright CRC Press LLC, 1995. — P. 437.