Теорія чисел у криптографії

Нікіта Скибицький

Київський національний університет імені Тараса Шевченка

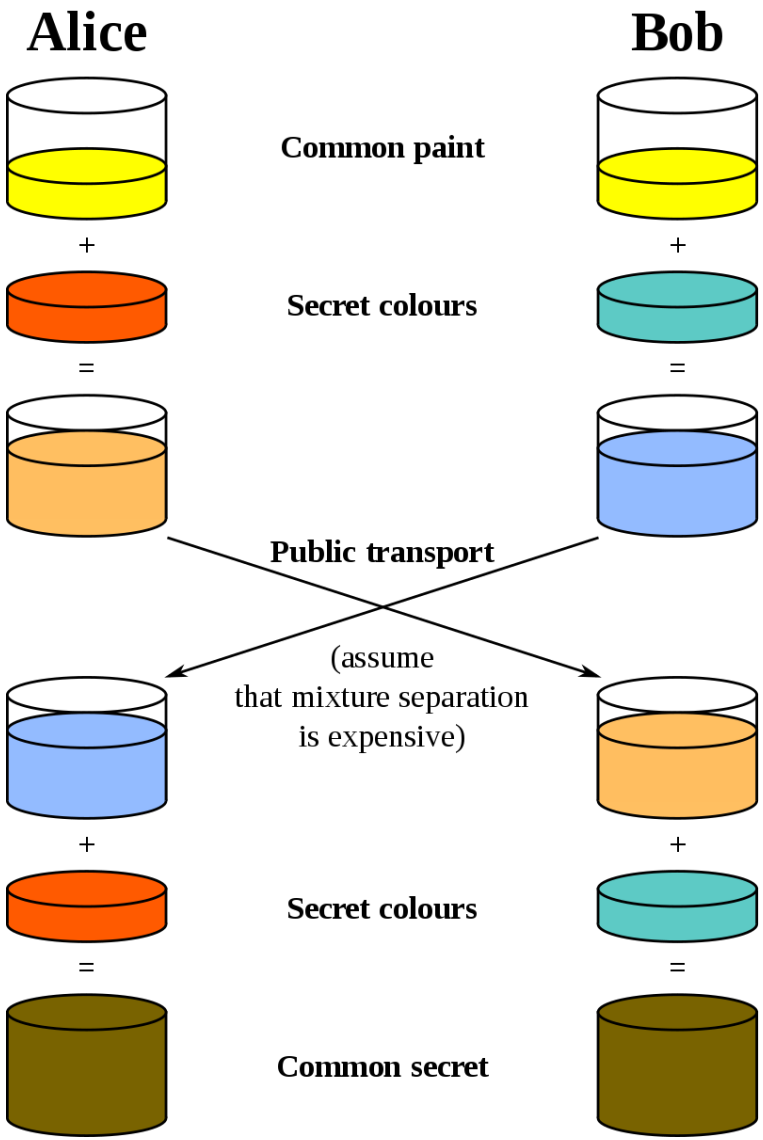
31 січня 2019 р.

Протокол Діффі-Геллмана

Ідея не симетричної крипто-системи з приватним і публічним ключами належить Діффі і Геллману, які запропонували відомий протокол генерації спільного секретного ключа.

Проілюструємо цей алгоритм наступною картинкою:

Ілюстрація


Alice
Bob
Common paint
+
+
Secret colours
=
=
Public transport
(assume
that mixture separation
is expensive)
+
+
Secret colours
=
=
Common secret

Mixture separation

Найважливішою вимогою протоколу Діффі-Геллмана є складність розділення кольору-суміші що передається через публічний канал.

Наведемо найпростішу математичну реалізацію запропоновану авторами і пояснимо чому вона, взагалі кажучи, не дуже безпечна.

Історично перша реалізація

Розглянемо мультиплікативну групу $(\mathbb{Z}/p\mathbb{Z})^\times$ лишків за модулем p, де $p \in \mathbb{P}$.

Нехай також g – первісний корінь за модулем p, тобто таке число, що $g, g^2, \ldots, g^{p-1}$ дають різні залишки при діленні на p.

Як показав Гаусс у 1801 р. таке g існує для довільного простого p (а також для $n = 2$, 4, p^k, $2 \cdot p^k$ і тільки для них, де p – непарне просте, $k \in \mathbb{N}$).

Історично перша реалізація

Нехай у Аліси є приватний ключ a, а у Боба приватний ключ b, і вони обидвоє знають публічні ключі p, g, тоді алгоритм наступний:

1. Аліса обчислює $A = g^a \pmod{p}$ і передає його Бобу.
2. Боб обчислює $B = g^b \pmod{p}$ і передає його Алісі.
3. Аліса обчислює $s_1 = B^a \pmod{p}$ і запам'ятовує його.
4. Боб обчислює $s_2 = A^b \pmod{p}$ і запам'ятовує його.

$A^b \pmod{p} = g^{ab} \pmod{p} = g^{ba} \pmod{p} = B^a \pmod{p}$, тому $s_1 = s_2 = s$, спільний приватний ключ.

Приклад роботи протоколу

Позначатимемо приватні значення **червоним**, а публічні – *синім*.
Нехай попередньо було вибрано $p = 23$, $g = 5$, $\mathbf{a} = \mathbf{4}$, $\mathbf{b} = \mathbf{3}$.

1. Аліса обчислює $5^{\mathbf{4}} \pmod{23} = 4$ і передає Бобу.
2. Боб обчислює $5^{\mathbf{3}} \pmod{23} = 10$ і передає Алісі.
3. Аліса обчислює $10^{\mathbf{4}} \pmod{23} = \mathbf{18}$ і запам'ятовує.
4. Боб обчислює $4^{\mathbf{3}} \pmod{23} = \mathbf{18}$ і запам'ятовує.

Знаючи лише модуль p, первісний корінь g, публічно передані числа $A = g^a \pmod{p}$ та $B = g^b \pmod{p}$ взагалі кажучи складно відновити приватні ключі a та b (ця задача носить назву дискретного логарифмування) або хоча б парний приватний ключ $s = g^{ab} \pmod{p}$.

Складність

Зауважимо, що у реальності p – дуже велике просте число, що містить кілька сотень цифр. Здавалося б, що тоді протокол стає неприпустимо повільним, адже необхідно обчислювати $g^a \pmod p$, де усі числа мають сотні цифр.

Але, на щастя, існує такий алгоритм як двійкове піднесення до степеню, який працює за $O(\ln a)$ (а не за $O(a)$ як наївний алгоритм), і базується він на наступній ідеї:

$$x^{1024} = (x^{512})^2 = \ldots = (\ldots(x^2)^2\ldots)^2,$$

і аналогічно для інших a. Для детальніших пояснень див. також наступну статтю.

Бонус: секретний мульти-чат

Зауважимо, що використовуючи цю ж ідею можна згенерувати (залишаємо це як вправу для читача) секретний ключ для чату з n людей за $O(\ln^2 n \cdot \ln p)$ замість $O(n^2 \cdot \ln p)$.

Остання $O(\cdot)$-шка – це складність циклічного обчислення $(\ldots(g^{a_1})^{a_2}\ldots)^{a_n}$, $(\ldots(g^{a_2})^{a_3}\ldots)^{a_1}$ і так далі аж до $(\ldots(g^{a_n})^{a_1}\ldots)^{a_{n-1}}$.

Зауваження: можливо я помилився і асимптотики мають бути $O(\ln n \cdot \ln p)$ і $O(n \cdot \ln p)$ відповідно.

Автентифікація

В оригінальному описі протокол Діффі-Геллмана не надає засобу автентифікації, тобто користувач не може знати напевне з ким він встановлює зв'язок.

Це призводить до можливості атаки man-in-the-middle, коли третя сторона встановлює два секретних канали зв'язку: один з Алісою (для якої третя сторона представляється Бобом) і один з Бобом (для якої третя сторона представляється Алісою).

Справді, такий підхід дозволяє їй отримувати повідомлення Боба, дешифрувати їх ключем Боба, читати, шифрувати їх ключем Аліси і відправляти Алісі, і навпаки.

RSA: теорія

Розглянемо $n = p \cdot q$, де $p, q \in \mathbb{P}$.

Розглянемо також функцію Ейлера $\phi(n)$ яка позначає кількість взаємно-простих з n чисел менших за n.

Можна показати, що якщо $n = \prod_{i=1}^{k} p_i^{\alpha_i}$ – розклад числа n на прості множники, то

$$\phi(n) = \prod_{i=1}^{k} (p_i - 1) \cdot p_i^{\alpha_i - 1} = n \cdot \prod_{i=1}^{k} \left(1 - \frac{1}{p_i}\right).$$

Зокрема $\phi(p \cdot q) = (p - 1) \cdot (q - 1)$.

Відомо, що якщо число e взаємно-просте з $\phi(n)$, то існує число d обернене до e за модулем $\phi(n)$, тобто $d = e^{-1} \pmod{\phi(n)}$.

RSA: алгоритм

Публічний ключ Аліси – пара (n, e), приватний – число d.

Шифрування: Боб отримує (публічно) публічний ключ Аліси, шифрує своє повідомлення m як $c = m^e \pmod{n}$, (публічно) передає його Алісі.

Дешифрування: Аліса обчислює $m = c^d \pmod{n}$.

$(m^e)^d = m \pmod{n}$ завдяки тому, що $ed = 1 \pmod{\phi(n)}$.

RSA: автентифікація

RSA безпечніший за оригінальний протокол Діффі-Геллмана за рахунок того, що публічний і приватний ключі зв'язані (через $\phi(n)$).

Зрозуміло, що тоді третя сторона може представитися Алісою на етапі шифрування (для цього потрібно знати лише публічний ключ), але не зможе дешифрувати повідомлення без приватного ключа Аліси (який є тільки у Аліси), тобто класична атака man-in-the-middle не працює.

Задача знаходження $\phi(n)$ (майже) рівносильна задачі факторизації n, а для неї (на сьогоднішній день) не існує поліноміальних (відносно $\ln n$) алгоритмів розв'язання. Декілька відомих (у тому числі суб-експоненціальних) алгоритмів наведені за посиланням.

Бонус: PGP Shredder

Шифрування корисне не лише для обміну даних з кимось, але також і для захисту власних даних. Уявіть собі що ви іноземний шпигун який зібрав і вже передав за кордон купу секретної інформації, і вам необхідно її терміново знищити (наприклад тому що у ваші двері ломиться спецназ).

У фільмах зображуються різні способи це зробити, включаючи розчинення жорсткого диску в кислоті (або надто повільно або вас посадять за синтез небезпечної кислоти, тобто біологічної зброї), використання мікрохвильової пічки (на практиці пічка вибухне до того як знищить суттєву частину даних) та навіть банальне видалення з кошика (соромно навіть пояснювати що при цьому у кращому випадку затирається заголовок файлу але аж ніяк не його зміст, особливо якщо файл кілька Гб).

Бонус: PGP Shredder

Існує простіший спосіб: просто зашифруйте всі дані і знищіть (дуже короткий) приватний ключ.

Навряд хтось подумає що ви можете його пам'ятати (адже це буде просто "випадкове" число з кількасот а то і кількома тисячами цифр), тому ніхто нічого з вас не випитає (а в ідеалі навіть не намагатиметься).

У свою чергу відновлення, приватного ключа за публічним займе щонайменше кілька років, до того часу інформація буде вже неактуальною.