Heidi's Cryptographic Commerce

by Robert R. Enderlein

Easy Task

Time travel opens up a fantastic array of business opportunities. Just imagine how much money one could make by selling today's technology in the past!

Unfortunately, Heidi is totally unaware how to build fast computers or synthesise any of the fancy nanotech-materials her time machine is built with. In fact, she barely remembers most of the algorithms she learnt at school, and regrettably forgot to take her textbooks with her on her chronological cruise.

One thing she (vaguely) remembers is this fancy public-key encryption scheme. It was invented by some Egyptian named Elgimli (or something similar, she never was good at remembering names) and it has the major advantage that two persons can communicate without the need to exchange secret keys: Heidi simply needs to create a public and secret key pair, and publish her public key. Anybody can now send Heidi a message by using her public key and her program, but only Heidi is then able to decrypt the message using her (carefully guarded) private key.

In fact, she now has found several people who would pay almost any price for this algorithm. However, she can only remember the key generation and encryption algorithms and needs your help to write the corresponding decryption algorithm (and of course, she can't sell her system without it).

To generate her keys, Heidi proceeds as follows:

- Heidi selects a (large) odd prime number, let's call it p
- Heidi selects a number g at random in the interval $[2, p-1]$
- Heidi selects a number x at random in the interval $[0, p-2]$
- Heidi computes $y = g^x (mod p)$ where mod stands for modulo (i.e., she selects the only integer y in $[0, p-1]$ such that $y = g^x - N \times p$ for some integer N)
- Heidi's public key is now (p, g, y), and she distributes it to all her friends.
- Heidi's secret key is (x), and she closely guards it.

To encrypt a message m (an integer in the interval $[0, p-1]$), Heidi's friends proceed as follows:

- They select a number r at random in the interval $[0, p-2]$
- They then compute $E = m \times y^r (mod p)$
- And they compute $F = g^{-r} (mod p)$ (The last statement is equivalent to $F = g^{(p-1-r)} (mod p)$ according to Fermat's little theorem)
- They then send (E, F) to Heidi

Now Heidi needs your help to write the decryption algorithm that given a ciphertext (E, F), her public key (p, g, y) and her private key (x) recovers the message that her friend sent her. The prime p will be between 3 and 2^{104}, and there will be slightly more than hundred testcases.

Heidi has provided you with the source code of the encryption algorithm. C/C++ users: you may use the GNU GMP library, which provides methods to handle large integers. You can find the documentation of the library on . Be careful with memory management!

Heidi et son Commerce Cryptographique

par Robert R. Enderlein

Tâche Facile

Le voyage dans le temps offre une fantastique palette d'opportunités pour faire du business. Imaginez seulement combien d'argent on pourrait gagner en vendant les technologies d'aujourd'hui dans le passé !

Malheureusement, Heidi ne sait absolument pas comment construire les calculateurs puissants ou synthétiser les matériaux nanotechnologiques qui sont nécessaires pour construire sa machine à voyager dans le temps. En effet, elle se rappelle à peine de la plupart des algorithmes qu'elle a appris à l'école et elle a, regrettablement, oublié de prendre ses livres de cours lorsqu'elle est partie pour son voyage chronologique.

Une chose dont elle se souvient (vaguement) est qu'il existe un mécanisme sophistiqué de chiffrement à clef publique. Il a été inventé par un Egyptien du nom de Elgimli (ou un truc du genre, elle n'a jamais été douée pour se rappeler des noms) et qu'il a le grand avantage que deux personnes peuvent communiquer sans besoin de s'échanger des clefs secrètes : Heidi doit simplement créer une paire de clefs, une publique et l'autre secrète, et publier sa clef publique. Quiconque peut maintenant envoyer un message à Heidi en utilisant sa clef publique et son programme, mais uniquement Heidi est en mesure de déchiffrer le message en utilisant sa clef privée.

De fait, cet algorithme suscite beaucoup d'intérêt et Heidi a trouvé plusieurs personnes souhaitant l'avoir à n'importe quel prix. Toutefois, elle se souvient uniquement de l'algorithme de génération de la clef et de chiffrement. Elle a donc grand besoin de votre aide pour écrire l'algorithme de déchiffrement, sans quoi elle ne pourra pas vendre le système !

Pour générer les clefs, Heidi exécute les operations suivantes :

- Heidi choisit un (grand) nombre premier impair, appelons-le p
- Heidi choisit un nombre g aléatoirement dans l'interval $[2, p-1]$
- Heidi choisit un nombre x aléatoirement dans l'interval $[0, p-2]$
- Heidi calcule $y = g^x (mod p)$ où mod est le modulo (càd, elle choisit le seul entier y dans $[0, p-1]$ tel que $y = g^x - N \times p$ pour un certain entier N)
- La clef publique de Heidi est maintenant (p, g, y), et elle la distribue à tous ses amis.
- Sa clef privée est (x), et elle la met à l'abri des regards indiscrets.

Pour chiffrer le message m (un entier dans l'interval $[0, p-1]$), les amis de Heidi peuvent procéder ainsi :

- Ils choisissent un nombre r aléatoirement dans l'interval $[0, p-2]$
- Ils calculent ensuite $E = m \times y^r (mod p)$
- Et ils calculent $F = g^{-r} (mod p)$ (Ce dernier point est équivalent à $F = g^{(p-1-r)} (mod p)$ selon le petit théorème de Fermat)
- Ensuite, ils envoient (E, F) à Heidi

Maintenant Heidi a besoin de ton aide pour écrire l'algorithme de déchiffrage qui, à partir du texte chiffré (E, F), de sa clef publique (p, g, y) et de sa clef privée (x), récupère le message que son ami lui a envoyé. Le nombre premier p sera entre 3 et 2^{104}, et il y aura un peu plus de cent cas de test.

Heidi t'as fourni le code source de son algorithme de chiffrement. Utilisateurs C/C++ : vous pouvez utiliser la librairie GNU GMP, qui vous fournit des méthodes pour travailler avec des grands entiers. Vous pouvez trouver la documentation pour cette librairie à doc.hc2.ch. Faites attention à la gestion de mémoire !

Heidis kryptographisches Gewerbe

von Robert R. Enderlein

Einfache Aufgabe

Zeitreisen ermöglichen fantastische Geschäftsmöglichkeiten. Stellt euch nur mal vor, wieviel Umsatz sich mit dem Verkauf heutiger Technologie in der Vergangenheit machen liesse!

Leider ist Heidi nicht in der Lage, die schnellen Rechner oder die faszinierenden Nanomaterialien ihrer Zeitmaschine zu verstehen. Um's auf den Punkt zu bringen, räumen wir ein, dass Heidi nur wage Erinnerungen an die Algorithmen ihrer Schulzeit hat. Hätte sie bloss ihre Lehrbücher mit auf die Zeitreise genommen!

Nur an ein Verschlüsslungsschema mit öffentlichem Schlüssel kann sie sich noch in so etwa erinnern, erfunden von einem Ägypter namens Elgimli (oder so ähnlich, Namen waren nie ihre Stärke). Damit können sich zwei Personen verständigen, ohne zuvor geheime Schlüssel auszutauschen: Heidi erstellt ein öffentliches und privates Schlüsselpaar, und gibt dann ihren öffentlichen Schlüssel bekannt. Nun kann jederman ihr eine Nachricht senden, verschlüsselt mit Hilfe des öffentlichen Schlüssels und ihres Programmes. Einzig und alleine Heidi kann die Nachricht mit Hilfe ihres (sicher aufbewahrten) privaten Schlüssels entziffern.

Heidi hat mittlerweile interessierte Käufer ausfindig gemacht, welche bereit wären alles Geld der Welt für ihren Algorithms zu zahlen. Heidi erinnert sich jedoch nur an die Verfahren zum Erstellen der Schlüssel und zum Verschlüsseln, nicht an die Entschlüsslungsmethode. Könnt ihr Heidi helfen, einen Entschlüsslungsalgorithmus zu erstellen, damit sie das System verkaufen kann?

Die Schlüssel werden folgendermassen generiert:

- Heidi wählt eine (grosse) ungerade Primzahl, nennen wir diese p
- Dann wählt sie eine zufällige Zahl g im Intervall $[2, p-1]$
- Dann wählt sie eine zufällige Zahl x im Intervall $[0, p-2]$
- Heidi berechnet $y = g^x (mod p)$. Mod steht für den Rest der Division durch p (d. h., sie wählt die einzige ganze Zahl y in $[0, p-1]$ so dass $y = g^x - N \times p$ für eine ganze Zahl N gilt)
- Heidis öffentlicher Schlüssel ist jetzt (p, g, y). Sie verteilt diesen an alle ihre Freunde.
- Heidis geheimer Schlüssel ist (x). Sie wird ihn sicher verwahren.

Um eine Nachricht m (dargestellt durch eine ganze Zahl im Intervall $[0, p-1]$) zu verschlüsseln, tun Heidis Freunde folgendes:

- Sie wählen eine zufällige Zahl r im Intervall $[0, p-2]$
- Und errechnen dann $E = m \times y^r (mod p)$
- Und danach $F = g^{-r} (mod p)$ (was äquivalent ist zu $F = g^{(p-1-r)} (mod p)$ laut Fermats kleinem Theorem)
- Dann schicken sie (E, F) an Heidi

Nun braucht Heidi eure Hilfe, um den Entschlüsslungsalgorithmus zu schreiben. Als Eingabe erhält er den verschlüsselten Text (E, F), ihren öffentlichen Schlüssel (p, g, y), sowie ihren privaten Schlüssel (x). Der Algorithmus soll dann die ursprüngliche Nachricht berechnen. Die Primzahl p liegt zwischen 3 und 2^{104}, und es gibt etwas mehr als einhundert Testfälle.

Heidi stellt euch den Verschlüsslungsquellcode zur Verfügung. C/C++ Benutzer: Gerne dürft ihr die GNU GMP Bibliothek benützen, welche Methoden zur Verarbeitung grosser Zahlen enthält. Eine Beschreibung der Bibliothek ist unter erhältlich. Seid achtsam bei der Speicherverwaltung!