MA-53G Intr. a la Criptografía **Semestre Primavera, 1996**

Cátedra No. 10 : 4 de Septiembre

Transcriptor: Ricardo Valenzuela

1 Teoría de Cuerpos Finitos

Sea F un cuerpo finito, denotamos por F^* al conjunto $F^* = F \backslash \{0\}$.

Definición 1 *Sea $\alpha \in F^*$, el menor entero positivo k tal que $\alpha^k = 1$, se denomina el orden de α, y se denota $ord(\alpha)$.*

Notar que existe una potencia finita de α igual a 1, ya que las potencias de α en cuerpo finito F_q^* no pueden ser todas distintas, y tal como tenemos $\alpha^i = \alpha^j$ para $j<i$, tambien tenemos $\alpha^{j-i} = 1$.

Hay $q-1$ elementos no nulos en F_q. Por definición de cuerpo estos elementos forman un grupo abeliano con respecto a la multiplicación, esto significa que: la multiplicación es cerrada, la multiplicación es asociativa, la multiplicación es conmutativa, existe neutro multiplicativo, todo elemento tiene inverso.

Teorema 1 (Lagrange) *Todo $\alpha \in F_q^*$ tiene un orden que divide a $q-1$*

Dem: (Primera demostración) Sea $\alpha^d = 1$, donde d es el orden de α.
Sea $S = \{1, \alpha, \alpha^2, ..., \alpha^{d-1}\}$ el conjunto de todas las potencias distintas de α, y para algún $b \in F_q^*$ sea bS el "coconjunto" de los elementos de la forma $b\alpha^j$ (por ejemplo, $1S = S$). Es fácil ver que dos coconjuntos cualesquiera son idénticos o bien distintos (nominalmente : si algún $b_1\alpha^i$ está en b_1S, $b_2\alpha^j$ está en b_2S, y $b_1\alpha^i = b_2\alpha^j$ entonces $b_1 = b_2\alpha^{j-i}$ de donde es fácil deducir que b_1S es subconjunto de b_2S y b_2S es subconjunto de b_1S lo que implica que $b_1S = b_2S$). Y cada coconjunto contiene exactamente d elementos. La unión de todos los coconjuntos abarca F_q^* (de $q-1$ elementos), esto significa que F_q^* es una unión disjunta de conjuntos de d elementos. Por lo tanto d divide a $q-1$.

$\square$

Dem: (Segunda demostración) Notar que $f : F_q^* \longrightarrow F_q^*$, donde $f(x) = \alpha x$ es una función biyectiva, ya que al multiplicar cada elemento de F_q^* por α obtenemos una permutación de F_q^*, porque cada elemento distinto a otro, seguira siendo distinto luego de la multiplicación. Luego:

$$\begin{aligned}\prod_{\beta\in F_q^*}\beta &= \prod_{\beta\in F_q^*} f(\beta)\\ \prod_{\beta\in F_q^*}\alpha\beta &= \alpha^{q-1}\prod_{\beta\in F_q^*}\beta.\end{aligned}$$

Cancelando, se concluye que $\alpha^{q-1}=1$.
Supongamos que el orden de α es d. Si d no divide a $q-1$, entonces $r=(q-1)\mathrm{mod} d$ es no nulo y menor que d. Además $q-1=td+r$, para algún t que pertenece a los naturales. Luego,

$$1=\alpha^{q-1} = (\alpha^d)^t\alpha^r.$$

Pero el orden de α es d, entonces $\alpha^r=1$. Lo que contradice la minimalidad de d. □

Definición 2 *Un cuerpo G se dice cíclico si existe un $g\in G$ tal que: $G=\{g^i : i$ entero $\}$, y a g se le llama generador.*

Luego, un generador de F_q^* es un elemento de orden $q-1$.
El siguiente lema nos dice, en particular, que los elementos no nulos de F_q forman un grupo cíclico.

Lema 1 *Todo cuerpo finito tiene un generador. Si g es un generador de F_q^*, entonces g^j es un generador de F_q^* si y solo si j y $q-1$ son primos relativos. En particular hay $\varphi(q-1)$ generadores de F_q^*.*

Dem: Sea d el orden de $\alpha\in F_q^*$, es decir, $\alpha^d=1$ y $\alpha,\alpha^2,...,\alpha^d$ son todas distintas y son soluciones de la ecuación: $x^d=1$. Por el teorema fundamental del álgebra, la última ecuación tiene a lo más d soluciones, y todo elemento $\beta\in F_q^*$ de orden d satisface esta ecuación. Luego, todo elemento $\beta\in F_q^*$ de orden d es tal que $\beta\in\{\alpha,\alpha^2,...,\alpha^d\}$. Veamos ahora que α^j es de orden d si y solo si $\mathrm{mcd}(j,d)=1$ (*). En efecto, si $mcd(j,d)=d'>1$, sigue que $1=(\alpha^d)^{\frac{j}{d'}}=(\alpha^j)^{\frac{d}{d'}}$ es decir, α tiene orden $\frac{d}{d'}$. Contradicción.

Para el converso, notemos que $mcd(j,d) = 1$, implica que existe u, v enteros tales que $uj + vd = 1$. Luego, si α^i tuviese orden d' menor que d, sigue que $(\alpha^j)^{d'} = 1$ y $(\alpha^d)^{d'} = 1$, luego $\alpha^{jd'u} = 1$ y $\alpha^{dd'v} = 1$.
Por lo tanto $\alpha^{jd'u+dd'v} = 1$, es decir, $\alpha^{d'} = 1$, lo que contradice la minimalidad de d.
Luego, si existe un elemento de orden d, entonces hay $\varphi(d)$ elementos de orden d.
Si d no divide a $q-1$ no hay elementos de orden d (por Lagrange). Si d divide a $q-1$ hay 0 o $\varphi(d)$ elementos de orden d.
Finalmente:

$$q - 1 \;=\; \sum_{d|q-1} |\{\alpha : ord(\alpha) = d\}| \leq \sum_{d|q-1} \varphi(d) = q - 1$$

es decir $|\{\alpha : ord(\alpha) = d\}| = \varphi(d)$, para cualquier d que divide a $q-1$.

En particular, hay $\varphi(q-1)$ elementos de orden $q-1$, es decir generadores.
Si g es generador de F_q^*, g^j es un generador para todo $j \in \{1, ..., q-1\}$, tal que mcd$(j, q-1) = 1$. (Por afirmación (*) probada anteriormente.) $\square$

Corolario 1 *Para todo primo p, existe un $g \in \mathbf{Z}_p^*$ cuyas potencias generan $\mathbf{Z}_p^*$.*

2 Implementación de RSA

Existen muchas situaciones donde se quiere saber si un número n grande es primo. Por ejemplo, en la implementación de RSA y en varios criptosistemas, necesitamos encontrar un primo "aleatorio" grande.
Un test de primalidad es una prueba para saber si un número es o no primo. Si un número pasa la prueba, éste puede ser primo. Si un número pasa una gran cantidad de estos test, entonces es muy probable que sea primo. Y por lo tanto si un número no pasa la prueba, éste es definitivamente no primo. Empíricamente, se sabe que es más costoso (en tiempo) factorizar un número grande no primo (fallo en el test de primalidad) que encontrar uno primo del mismo orden de magnitud.
La seguridad de RSA se basa en el hecho de que es mucho más fácil encontrar dos primos extremadamente grandes p y q que, conociendo n, encontrar los dos factores p y q no conocidos tal que $n = pq$.

Tenemos los siguientes problemas de implementación :

1. Elegir p y q primos grandes.
2. Calcular $n = pq$ y $\varphi(n) = (p-1)(q-1)$.
3. Elegir un exponente de encriptación a tal que $0 < a < \varphi(n)$ y donde a sea primo relativo con $\varphi(n)$.
4. Encontrar $x \in$ { 1 ... φ(n)-1 } tal que: $xa \equiv 1(\text{mod } \varphi(n))$
5. Publicar (n, a) (No publicar en cualquier Newsgroup.)
6. Calcular x^c mod n, donde c es a o b.

Las partes 2. y 3. sabemos como implementar eficientemente.
Para implementar 6. utilizamos el método de los cuadrados repetidos, es decir, el siguiente algoritmo:

$z = 1$
for $i = l$ downto 0
  $z = z^2$
  if $c_i = 1$ then $z = zx$ mod n

Donde $c = c_1 ... c_l$ es la representación binaria de c.

Todos los números a multiplicar tienen $O(logn)$ bits, además en cada iteración tenemos dos multiplicaciones, por lo tanto, con l iteraciones tenemos que el algoritmo toma $O(2l(logn)^2)$ operaciones binarias.
Como ya sabemos RSA requiere elegir primos grandes p y q y no demasiado similares. Hoy en día sabemos factorizar números de hasta 130 dígitos. Se recomienda usar números p y q de alrededor de 100 dígitos para que n quede de alrededor de 200 dígitos. Las implementaciones en hardware de RSA que existen utilizan un módulo n de 512 bits ($\approx$ 154 dígitos). Estas implementaciones logran encriptar 600 Kbits/seg. Por otro lado DES implementado a nivel de hardware logra encriptar 1 Gbit/seg.