Seguridad de las funciones resumen

Este documento no pretende ser un curso exhaustivo, en el mejor de los casos, únicamente puede considerarse como un conjunto de notas complementarias en alguna asignatura. Por supuesto, todo documento es susceptible de mejora, cualquier sugerencia o comunicación de error u omisión será bienvenida.

En este documento se presenta el resultado básico que permite acotar inferiormente el tamaño de una función resumen para que esta se considere segura. Este resultado muestra que no es necesario acercarse al número total de posibles resúmenes que puede obtener una función hash para encontrar una colisión.

En este documento se recoge también el esquema básico de un proceso que conduciría a encontrar dos mensajes con el mismo resumen (Algoritmo de Yuval). Notamos que, pese a todo, el ataque a una función de resumen de tamaño medio, supone un esfuerzo de cómputo que puede ser importante.

Seguridad de las funciones resumen

Consideraremos una función resumen $h : X \to Z$. Denotamos con m el conjunto posible de mensajes (esto es $m = |X|$) y con n el conjunto de posibles resúmenes (esto es $n = |Z|$). Como condición mínima consideramos que, al menos $|X| \geq 2|Z|$ o lo que es lo mismo que existirán al menos n colisiones y que estas se distribuyen de manera uniforme (no hay resúmenes más comunes que otros).

A partir de este marco, buscamos calcular la probabilidad de, dados k mensajes al azar $(x_1, x_2, \ldots, x_k)$, no se produzca colisión entre estos mensajes.

Teniendo en cuenta que la probabilidad de que el resumen de un mensaje x_i suponga una colisión o no es independiente de la probabilidad de que el resumen de otro mensaje suponga colisión, calcularemos el producto de las probabilidades de colision para cada uno de los mensajes.

Obviamente el primer mensaje no puede provocar colisión porque no hay otros resúmenes con los que colisionar. El resumen del segundo mensaje sólo puede colisionar con el del primero, por lo que la probabilidad de colisión es $1/n$ (casos favorables o de colisión partido casos posibles o resúmenes posibles), así, la probabilidad de no colisión considerando dos mensajes es:

$$\left(1 - \frac{1}{n}\right)$$

Si consideramos un tercer mensaje, la probabilidad de que el resumen de este colisione con los dos anteriores es $2/n$, por lo que la probabilidad de no colisión considerando tres

mensajes es:

$$\left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)$$

por lo que considerando k mensajes obtenemos que la probabilidad de no colisión de sus resúmenes es:

$$\left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\ldots\left(1-\frac{k-1}{n}\right) \;=\; \prod_{i=1}^{k-1}\left(1-\frac{i}{n}\right) \tag{1}$$

Aunque parezca desconectado, el cálculo de este productorio podemos aproximarlo teniendo en cuenta que el desarrollo de e^{-x} como la suma de la serie infinita:

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!}\ldots$$

ya que, para el caso en que x sea un valor que se aproxima a cero, los distintos términos $\frac{x^j}{j!}$ sse aproximan aún más a cero, por lo que:

$$e^{-x} \approx 1 - x$$

En una función resumen clásica, el número de resúmenes posible (el valor n) es muy grande, por lo que en la fórmula (1), que calculaba la probabilidad de no colisión de los resúmenes de k mensajes, podemos ver los términos i/n como valores que tienden a cero, por lo que podemos aproximar el productorio como:

$$\prod_{i=1}^{k-1}\left(1-\frac{i}{n}\right) \;\approx\; \prod_{i=1}^{k-1} e^{\frac{-i}{n}} \;=\; e^{\frac{-k(k-1)}{2n}} \tag{2}$$

con lo que disponemos una forma de aproximar la probabilidad de no colisión de los resúmenes de k mensajes.

Paradoja del cumpleaños

La busqueda del número de mensajes necesario para que, dada una función resumen cualquiera, encontrar una colisión puede ejemplificarse como:

> ¿Cuantas personas hace falta agrupar para que, en ese grupo y con probabilidad ϵ, haya dos con la misma fecha de cumpleaños?

En esta reducción la fecha de nacimiento es una función resumen de 365 valores distintos y las personas hacen el papel de mensajes. Es importante destacar que el cálculo incluye la probabilidad que deseamos que tenga la colisión.

Aprovechando la fórmula (2) para el cálculo de no colisión , si consideramos ϵ la probabilidad deseada de colisión, tenemos que:

$$e^{\frac{-k(k-1)}{2n}} \approx 1 - \epsilon$$

aplicando logaritmos a ambas partes y operando mínimamente se tiene:

$$\frac{-k(k-1)}{2n} \approx \ln 1 - \epsilon$$
$$\frac{k(k-1)}{2n} \approx -\ln 1 - \epsilon$$

aplicando propiedades de los logaritmos e intentando despejar k:

$$\frac{k(k-1)}{2n} \approx \ln \frac{1}{1-\epsilon}$$
$$k^2 - k \approx 2n \ln \frac{1}{1-\epsilon}$$

a partir de este punto podemos despejar directamente k^2 haciendo un poco peor la aproximación (aunque suficientemente buena) como:

$$k^2 \approx 2n \ln \frac{1}{1-\epsilon}$$

a partir de la cual puede obtenerse que:

$$k \approx \sqrt{2n \ln \frac{1}{1-\epsilon}}$$

Recordando el objetivo, esta expresión permite calcular el número k de mensajes necesarios para que dos de ellos (no se indica cuales) coincidan en su resumen con probabilidad ϵ.

Volviendo al escenario de personas y la fecha de nacimiento como resumen (n igual a 365), si $\epsilon = 0{,}5$ entonces bastan $k \approx \sqrt{n(2 \ln 2)}$ personas para que dos coincidan en fecha de aniversario, esto es, aproximadamente 23. Si deseamos casi absoluta certeza de que se va a dar la coincidencia, tomando $\epsilon = 0{,}95$, entonces el número k de personas sería de 47.

Esto se debe al peso determinante de $\sqrt{n}$ en el cálculo, siendo este valor el que establece la cota inferior en el tamaño de las funciones resumen para que su uso en criptografía se considere seguro.

Algoritmo de Yuval

La paradoja del cumpleaños es la base para un algoritmo que permite la búsqueda de mensajes con el mismo resumen.

Para intentar aportar intuición acerca de las implicaciones en la seguridad que puede tener que la búsqueda de colisiones sea computacionalmente asequible señalar que, en procesos de firma digital, por distintas razones[1] habitualmente no se firma un documento sino el resumen de este. Esto supone que si dispusieramos de dos mensajes contrapuestos pero

[1] Para reducir las necesidades computacionales pero también como vía de garantizar la integridad

con el mismo resumen, la firma que pudiera hacerse sobre uno de ellos podría utilizarse para firmar el segundo.

El el Algoritmo 1 se describe el esquema de Yuval para la búsqueda de mensajes con el mismo resumen. En el esquema que se muestra se habla de mensajes legítimos e ilegítimos para poner de manifiesto el riesgo antes expuesto. En cualquier caso el algoritmo, para un par de mensajes distintos x_l y x_i, establece un proceso de búsqueda que obtiene como resultado dos mensajes x'_l y x'_i probablemente distintos pero con la misma semántica y tales que $h(x'_l) = h(x'_i)$ para una función resumen h dada.

Algoritmo 1 Algoritmo de Yuval

Entrada: Par de mensajes x_l y x_i
// podría hablarse de un mensaje inocente y otro
// agresivo o de un mensajes legítimo y otro ilegítimo
Entrada: Función hash h de m bits
Salida: Mensajes x'_l y x'_i (con cambios menores respecto la entrada),
tales que $h(x'_l) = h(x'_i)$

1: Generar $t = 2^{m/2}$ modificaciones menores de x_l
2: Computar el resumen y almacenar los pares $(x'_l, h(x'_l))$
3: **Mientras** no se encuentre colisión **hacer**
4: Generar x'_i modificación menor de x_i y computar $h(x'_i)$
5: Buscar si existe x'_l tal que $h(x'_l) = h(x'_i)$
// la colisión será probable después de t intentos
6: **FinMientras**
7: **Devolver** $(x'_l,\ x'_i)$

En una primera fase, el algoritmo de Yuval busca obtener un conjunto de mensajes *distintos pero equivalentes* al mensaje x_l. El tamaño de este conjunto está determinado por la paradoja del cumpleaños y se establece, para una función resumen de m bits en $2^{m/2}$, esto es, la raiz del número posible de resúmenes.

Una vez registrados los resúmenes de cada uno de los mensajes generados, el algoritmo itera buscando mensajes equivalentes a x_i que tengan un resumen igual a uno de los mensajes considerados en la primera fase. De nuevo la paradoja del cumpleaños establece que después de $2^{m/2}$ iteraciones es cada vez más probable encontrar la colisión.

Pese a este algoritmo, para el caso particular de SHA-256, sería necesario considerar $2^{128} \approx 10^{43}$ mensajes distintos (junto a sus resúmenes) e iterar 10^{43} veces para tener probabilidad de encontrar una colisión... en efecto, dada la capacidad computacional actual no parece un proceso que pueda acabar en un plazo corto de tiempo.