Estudio y comparación de métodos de tokenización

Daniel Ayala Zamorano, Laura Natalia Borbolla Palacios, Ricardo Quezada Figueroa

Escuela Superior de Cómputo, Instituto Politécnico Nacional
daniel@ejemplo.com, laura@ejemplo.com, qf7.ricardo@gmail.com

Resumen La tokenización consiste en el reemplazo de información sensible por valores sustitutos, llamados tokens, en donde el camino de regreso, del token a la información sensible, no es factible. En los últimos años este proceso se ha vuelto muy popular entre los comercios en línea, pues les permite descargar parte de las responsabilidades de seguridad adquiridas al manejar números de tarjetas de crédito en un tercero, proveedor de servicios de tokenización. Lamentablemente, existe una gran cantidad de desinformación alrededor de cómo generar los tokens, principalmente producida por las estrategias publicitarias de las empresas tokenizadoras, en donde cada una intenta convencer al comprador de que su sistema es el mejor, sin explicar realmente qué es lo que hacen para generar tokens. Uno de los mensajes más comunes entre la publicidad es que la criptografía y la tokenización son cosas distintas, y la segunda es mucho más segura. En este trabajo se explica a detalle en qué consiste la tokenización y cuál es su relación con la criptografía; se revisan y comparan los desempeños de los métodos más comunes para tokenizar; para terminar se concluye con una discusión alrededor de las ventajas y desventajas de cada uno.

1. Introducción

2. Preliminares

2.1. Notación

Denotaremos a todas las cadenas de bits de longitud n como $\{0,1\}^n$. Un algoritmo tokenizador es una función $E : \mathcal{X} \to \mathcal{Y}$ en donde los conjuntos X y Y son el espacio de números de tarjetas y el de tokens, respectivamente.

2.2. Algoritmo de Luhn

2.3. Estructura de un número de tarjeta de crédito

2.4. Cifrado por bloques

Un cifrado por bloques es un cifrado simétrico que se define por la función $E : \mathcal{M} \times \mathcal{K} \to \mathcal{C}$ en donde $\mathcal{M}$ es el espacio de textos en claro, $\mathcal{K}$ es el espacio de llaves y $\mathcal{C}$ es el espacio de mensajes cifrados. Tanto los mensajes en claro como los cifrados tienen una misma longitud n, que representa el tamaño del bloque [**?**].

Los cifrados por bloque son un elemento de construcción fundamental para otras primitivas criptográficas. Muchos de los algoritmos tokenizadores que se presentan en este trabajo los ocupan de alguna forma. Las definiciones de los algoritmos son flexibles en el sentido de que permiten instanciar cada implementación con el cifrado por bloques que se quiera; en el caso de las implementaciones hechas para este trabajo se ocupó AES (*Advanced Encryption Standard*) en la mayoría de los casos.

2.5. Cifrado que preserva el formato

Un cifrado que preserva el formato puede ser visto como un cifrado simétrico en donde el mensaje en claro y el mensaje cifrado mantienen un formato en común. Formalmente, de acuerdo a lo definido en [**?**], se trata de una función $E : \mathcal{K} \times \mathcal{N} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X}$, en donde los conjuntos $\mathcal{K}$, $\mathcal{N}$, $\mathcal{T}$, $\mathcal{X}$ corresponden al espacio de llaves, espacio de formatos, espacio de *tweaks* y el dominio, respectivamente. El proceso de cifrado de un elemento del dominio con respecto a una llave K, un formato N y un *tweak* T se escribe como $E_K^{N,T}(X)$. El proceso inverso es también una función $D : \mathcal{K} \times \mathcal{N} \times \mathcal{T} \times \mathcal{X} \to \mathcal{X}$, en donde $D_K^{N,T}(E_K^{N,T}(X)) = X$.

Para lo que a este trabajo respecta, el formato usado es el de las tarjetas de crédito: una cadena de entre 12 y 19 dígitos decimales. Esto es $N = \{0, 1, \ldots, 9\}^n$ en donde $12 \leq n \leq 19$.

3. Resultados de comparaciones de desempeño

Todos los resultados presentados en esta sección se llevaron a cabo en una computadora con las siguientes características:

Procesador: Intel i5-7200U (2.5 GHz) de 4 núcleos.
Sistema operativo: Arch Linux, kernel 4.17.
Base de datos: MariaDB 10.1.
Compilador: GCC 8.1.1

Algoritmo	Tokenización (μs)	Detokenización (μs)
FFX	81	61
BPS	169	87
TKR	11014	203
AHR	4692	832
DRBG	18325	399

Tabla 1. Comparación de tiempos de tokenización.

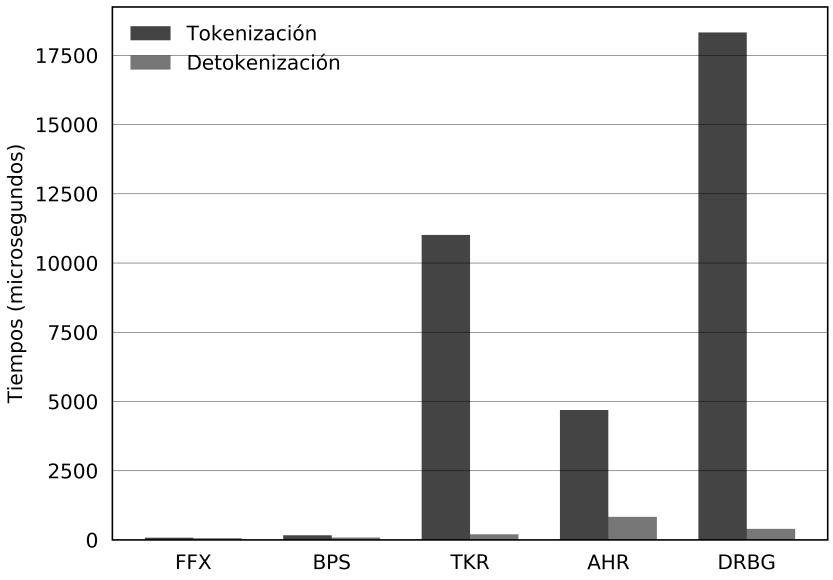

Figura 1. Tiempos de tokenización generales.