Ing. Sistemas Computacionales

Lenguaje Ensamblador

Rubén Hernández Torres

Implementación de algoritmo de salsa20

Documentación de salsa20

Einar Naim Aguilar Santana

Daniel Alejandro Barbosa Ayala

Axel Daniel Macias Heredia

Sebastián Rodríguez Hernández

Jesús Ruvalcaba Lozano

Fecha de entrega:

15 de junio de 2023

Algoritmo de salsa20

El algoritmo de Salsa20 es un algoritmo de cifrado de flujo diseñado para proporcionar confidencialidad y seguridad en las comunicaciones. Fue creado por Daniel J. Bernstein en 2005 como parte de la familia de cifradores de flujo Salsa.

Las funciones principales del algoritmo de Salsa20 son el cifrado y el descifrado de datos.

Lo que el docente pidió fueron las siguientes: el Quarterround, Rowround, Columnround y Doubleround.

La función Quarterround opera en un estado interno de 4 palabras de 32 bits y se repite varias veces para generar una secuencia de bytes cifrados.

$$\begin{aligned}
&\mathrm{quarterround}(\texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000000}) \\
&\qquad = (\texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000000}). \\
&\mathrm{quarterround}(\texttt{0x00000001}, \texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000000}) \\
&\qquad = (\texttt{0x08008145}, \texttt{0x00000080}, \texttt{0x00010200}, \texttt{0x20500000}). \\
&\mathrm{quarterround}(\texttt{0x00000000}, \texttt{0x00000001}, \texttt{0x00000000}, \texttt{0x00000000}) \\
&\qquad = (\texttt{0x88000100}, \texttt{0x00000001}, \texttt{0x00000200}, \texttt{0x00402000}). \\
&\mathrm{quarterround}(\texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000001}, \texttt{0x00000000}) \\
&\qquad = (\texttt{0x80040000}, \texttt{0x00000000}, \texttt{0x00000001}, \texttt{0x00002000}). \\
&\mathrm{quarterround}(\texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000000}, \texttt{0x00000001}) \\
&\qquad = (\texttt{0x00048044}, \texttt{0x00000080}, \texttt{0x00010000}, \texttt{0x20100001}). \\
&\mathrm{quarterround}(\texttt{0xe7e8c006}, \texttt{0xc4f9417d}, \texttt{0x6479b4b2}, \texttt{0x68c67137}) \\
&\qquad = (\texttt{0xe876d72b}, \texttt{0x9361dfd5}, \texttt{0xf1460244}, \texttt{0x948541a3}). \\
&\mathrm{quarterround}(\texttt{0xd3917c5b}, \texttt{0x55f1c407}, \texttt{0x52a58a7a}, \texttt{0x8f887a3b}) \\
&\qquad = (\texttt{0x3e2f308c}, \texttt{0xd90a8f36}, \texttt{0x6ab2a923}, \texttt{0x2883524c}).
\end{aligned}$$

Se puede visualizar la función "quarterround" como una modificación de "y" en su lugar: primero "y1" cambia a "z1", luego "y2" cambia a "z2", luego "y3" cambia a "z3" y finalmente "y0" cambia a "z0". Cada modificación es invertible, por lo que la función completa es invertible.

La función "rowround" modifica las filas de la matriz en paralelo al alimentar una permutación de cada fila a través de la función "quarterround". En la primera fila, la función "rowround" modifica y1,

luego y2, luego y3 y finalmente y0; en la segunda fila, la función "rowround" modifica y6, luego y7, luego y4 y finalmente y5; en la tercera fila, la función "rowround" modifica y11, luego y8, luego y9 y finalmente y10; en la cuarta fila, la función "rowround" modifica y12, luego y13, luego y14 y finalmente y15.

La función "columnround" es, desde esta perspectiva, simplemente la transpuesta de la función "rowround": modifica las columnas de la matriz en paralelo al alimentar una permutación de cada columna a través de la función "quarterround".

Una "double round" es una "columnround" seguida de una "rowround": doubleround(x) = rowround(columnround(x)). Se puede visualizar una "double round" como la modificación de las columnas de la entrada en paralelo, seguida de la modificación de las filas en paralelo. Cada palabra se modifica dos veces.

Durante la implementación de este proyecto, hubo ciertas dudas con respecto a lo que nos pide el docente, ya que el equipo tenía la idea que el proyecto debía encriptarse (como funciona todo completo). Después de una revisión el docente nos dijo que no se tiene que encriptar, simplemente se tiene que realizar las funciones de Quarterround, el Doubleround y el ColumnRound.

Después de dicha revisión se tuvo modificar el objetivo y las prioridades, para así terminar el código en tiempo y forma.

Finalmente logramos implementar este algoritmo de cifrado en lenguaje ensamblador para la arquitectura x86-i386.

El algoritmo que elaboramos lo que hace es una matriz de 16 variables doubleword, pasa por 2 procesos, que son el columnround y posteriormente el rowround, que la base de estos 2 procesos es el quarterround, ya anteriormente explicados.

El archivo que nos facilito mucho y sobretodo nos aclaro muchas dudas fue el spec.pdf que se encuentra en el GITHUB del proyecto.