Báo cáo BT nhóm
Tìm hiểu mật mã dòng trong họ eStream

Thành viên:
Hoàng Văn Hưng 20205333
Nguyễn Thành Luân 20200375
Đỗ Tuấn Minh 20200389

Giới thiệu Mật mã dòng eStream

SALSA20

- Mật mã dòng (stream cipher) Salsa20 là một thuật toán mật mã dòng được thiết kế bởi Daniel J. Bernstein vào năm 2005.

- Nó đã trở thành một trong những thuật toán mật mã dòng phổ biến và an toàn trong nhiều ứng dụng bảo mật trên internet và trong các hệ thống khác.

Salsa20 có một loạt ưu điểm quan trọng, bao gồm:

1. Hiệu suất cao: Salsa20 được thiết kế để hoạt động nhanh chóng và hiệu quả trên nhiều nền tảng phần cứng và phần mềm. Điều này làm cho nó trở thành lựa chọn lý tưởng cho các ứng dụng đòi hỏi xử lý nhanh và hiệu suất cao, như mã hóa dữ liệu trong thời gian thực.

2. Bảo mật mạnh: Salsa20 được chấp nhận rộng rãi trong cộng đồng mật mã học vì tính bảo mật mạnh của nó. Nó đã trải qua nhiều cuộc kiểm tra và đánh giá bởi các chuyên gia an ninh và không có lỗ hổng lớn nào được phát hiện.

3. Dễ triển khai và hiểu: Salsa20 có cấu trúc đơn giản, dễ hiểu, và có thể triển khai dễ dàng trong nhiều ngôn ngữ và môi trường khác nhau. Điều này làm cho nó phù hợp cho việc sử dụng trong các ứng dụng khác nhau mà không đòi hỏi kiến thức mật mã sâu rộng.

4. Loạt biến thể và mở rộng: Salsa20 có nhiều phiên bản và biến thể, cho phép tùy chỉnh cấu hình theo nhu cầu cụ thể của ứng dụng. Điều này bao gồm Salsa20/20 và XSalsa20, mang lại sự linh hoạt cho việc đảm bảo tính bảo mật và hiệu suất theo yêu cầu.

5. Ứng dụng rộng rãi: Salsa20 được sử dụng rộng rãi trong các ứng dụng bảo mật, bao gồm HTTPS, SSH, VPN, và nhiều giao thức mạng khác. Điều này chứng tỏ tính phổ biến và đáng tin cậy của thuật toán này.

Tóm lại, Salsa20 là một thuật toán mật mã dòng mạnh mẽ, hiệu suất cao, và phù hợp cho nhiều ứng dụng bảo mật.

Giới thiệu

Salsa20 là một hàm băm với đầu vào là 64 byte và đầu ra là 64 byte. Hàm băm này được sử dụng trong chế độ tính toán như một mã dòng: **Salsa20** thực hiện mã hóa một khối plaintext 64 byte bằng cách áp dụng phép băm cho khóa, số nonce và số block, sau đó thực hiện phép XOR với plaintext.

word

- Một word là một phần tử trong tập $\{0, 1, \dots, 2^{32} - 1\}$.
  Mỗi word kích thước 4 byte.

- Phép quay trái c-bit của một word u kí hiệu là $u <<< c$.

Hàm quarter_round

- Input: chuỗi 4 word $\{y_0, y_1, y_2, y_3\}$
- Output: chuỗi 4 word $\{z_0, z_1, z_2, z_3\}$
- $z1 = y1 \oplus \big((y0 + y3) <<< 7\big)$,
- $z2 = y2 \oplus \big((z1 + y0) <<< 9\big)$,
- $z3 = y3 \oplus \big((z2 + z1) <<< 13\big)$,
- $z0 = y0 \oplus ((z3 + z2) <<< 18)$.

Hàm row_round

- Input: $\{y_0, y_1, y_2, \dots, y_{15}\}$
- Output: $\{z_0, z_1, z_2, \dots, z_{15}\}$
- $(z0, z1, z2, z3) = quarter\_round(y0, y1, y2, y3)$,
- $(z5, z6, z7, z4) = quarter\_round(y5, y6, y7, y4)$,
- $(z10, z11, z8, z9) = quarter\_round(y10, y11, y8, y9)$,
- $(z15, z12, z13, z14) = quarter\_round(y15, y12, y13, y14)$.

Hàm column_round

- Input: $\{x_0, x_1, x_2, \dots, x_{15}\}$
- Output: $\{y_0, y_1, y_2, \dots, y_{15}\}$
- $(y0, y4, y8, y12) = quarter\_round(x0, x4, x8, x12)$,
- $(y5, y9, y13, y1) = quarter\_round(x5, x9, x13, x1)$,
- $(y10, y14, y2, y6) = quarter\_round(x10, x14, x2, x6)$,
- $(y15, y3, y7, y11) = quarter\_round(x15, x3, x7, x11)$.

Hàm double_round

- Input: chuỗi 16 word
- Output: chuỗi 16 word
- $doubleround(x) = row\_round(column\_round(x))$

Hàm little_endian

- Input: chuỗi 4 byte b = (b0, b1, b2, b3)
- Output: một word
- $little\_endian(b) = b0 + 2^8.b1 + 2^{16}.b2 + 2^{24}.b3$

Hàm Salsa20 hash

- Input: chuỗi 64 byte $x = (x[0], x[1], \dots, x[63])$
- Output: chuỗi 64 byte
- $x0 = little\_endian(x[0], x[1], x[2], x[3])$,
- $x1 = little\_endian(x[4], x[5], x[6], x[7])$,
- $x2 = little\_endian(x[8], x[9], x[10], x[11]), \dots$
- $x15 = little\_endian(x[60], x[61], x[62], x[63])$.

Hàm Salsa20 hash

- $(z0, z1, \ldots, z15) = double\_round^{10}\ (x0, x1, \ldots, x15)$
- Khi đó , $\boldsymbol{Salsa20(x)}$ là nối của:
- $littleendian^{-1}(z0 + x0),$
- $littleendian^{-1}(z1 + x1),$
- $littleendian^{-1}(z2 + x2), \ldots$
- $littleendian^{-1}(z15 + x15).$

Hàm Salsa20 expansion

- k là một chuỗi 16 byte hoặc 32 byte $(k0, k1)$
- n là một chuỗi 16 byte
- Khi đó , $\boldsymbol{Salsa20_k(n)}$ là một chuỗi 64 byte.
- Định nghĩa

$$\sigma 0 = (101,120,112,97), \qquad \sigma 1 = (110,100,32,51),$$

$$\sigma 2 = (50,45,98,121), \qquad \sigma 3 = (116,101,32,107).$$

- Nếu $k0, k1$ là chuỗi 16 byte:

$$Salsa20_{k0,k1}(n) = Salsa20(\sigma 0, k0, \sigma 1, n, \sigma 2, k1, \sigma 3).$$

Hàm Salsa20 expansion

- Định nghĩa

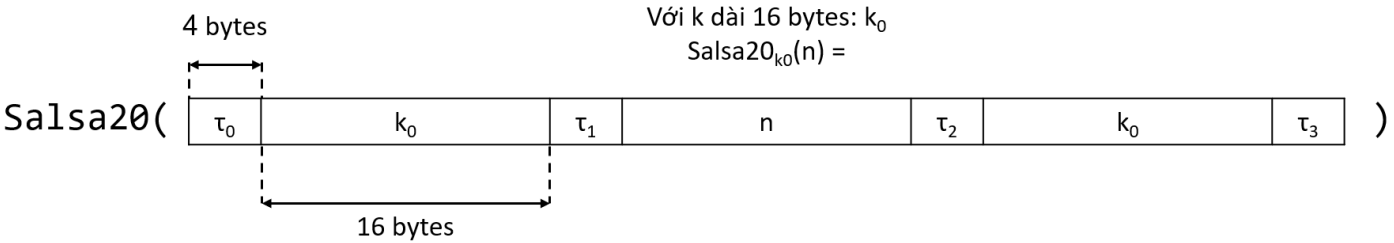
$$\tau 0 = (101,120,112,97), \qquad \tau 1 = (110,100,32,49),$$

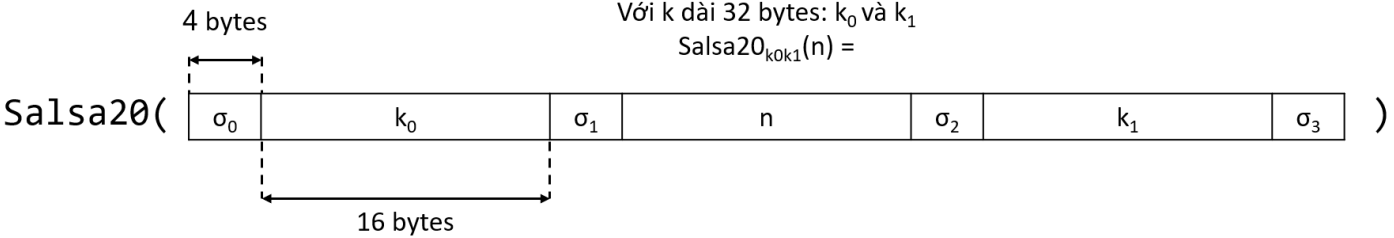
$$\tau 2 = (54,45,98,121), \qquad \tau 3 = (116,101,32,107).$$

- Nếu k là 16-byte thì

$$Salsa20_k(n) = Salsa20(\tau 0, k, \tau 1, n, \tau 2, k, \tau 3).$$

Hàm Salsa20 expansion





Hàm Salsa20 encryption

- k là chuỗi 32-byte hoặc 16-byte.
- v là chuỗi 8-byte.
- m là chuỗi l byte $\in \{0,1,\dots,2^{70}\}$.
- $Salsa20\ encryption$ của $\boldsymbol{m}$ với nonce $\boldsymbol{v}$ và khóa $\boldsymbol{k}$, được kí hiệu là $Salsa20_k(v) \oplus m$, là một chuỗi l byte.

Hàm Salsa20 encryption

- $\boldsymbol{Salsa20_k(v)} = Salsa20_k(v, \bar{0}), Salsa20_k(v, \bar{1}), Salsa20_k(v, \bar{2})$

  $\ldots Salsa20_k(v, \overline{2^{64}-1})$

- Trong đó $\bar{\imath}$ là chuỗi 8 bit $(i0, i1, \ldots i7)$ thỏa mãn

  $$i = i0 + 2^8.i1 + 2^{16}.i2 + \cdots + 2^{56}.i7$$

- Cắt ngắn $Salsa20_k(v)$ cho có cùng chiều dài với m. Tức là:

  $$Salsa20_k(v) \oplus (m[0], m[1], \ldots, m[-1]) = (c[0], c[1], \ldots, c[-1])$$

  trong đó

  $$c[i] = m[i] \oplus Salsa20_k(v, [i/64])[i \bmod 64].$$

THANK YOU FOR LISTENING