Chacha and Salsa Stream Ciphers - QSC Assignment I

Eleena Gupta and Vidhu Catherine Antony

October 3, 2023

Recap : Stream Cipher

- **Plaintext** is encrypted one bit at a time
- **Pseudorandom Key** generated by a PRG and XOR it with the plaintext
- Take a smaller key as a seed to generate a pseudorandom bit sequence of the length of the plaintext

Building Pseudorandom Generators [1]

- **Security of PRG:**
  - Unpredictability
  - Indistinguishability
- **PRGs:**
  - **Salsa20/*r***: The family of 256-bit fast stream ciphers designed by Dan Bernstein in 2005.
  - **ChaCha*r***: its variant with improved diffusion per round

[1] , Daniel J. "ChaCha, a variant of Salsa20." Workshop record of SASC. Vol. 8. No. 1. 2008.

PRGs of Salsa and ChaCha

- **Inputs:**
  - 256-bit seed (secret key) s
  - 64-bit nonce n
- **Output:**
  - 512-bit pseudorandom block
- **Components:**
  - Padding function: Pad(s,j,n) 512 bit block (j : counter)
- Fixed public permutation π :

PRGs of Salsa and ChaCha: Algorithm

Use these components to output L¡$2^{6}4suchblocksof512biteach$.

Algorithm:

```
**input**: seed $s \in \{0,1\}^{256}$
for $j \leftarrow 0$ to $L-1$
    $h_j \leftarrow \mathrm{pad}(s, j, 0) \in \{0,1\}^{512}$
    $r_j \leftarrow \pi\left(h_j\right) \oplus h_j$
**output**: $(r_0, \ldots, r_{L-1})$.
```

We get L such 512-bit blocks as the PRG output.

PRGs of Salsa and ChaCha

The Schematic

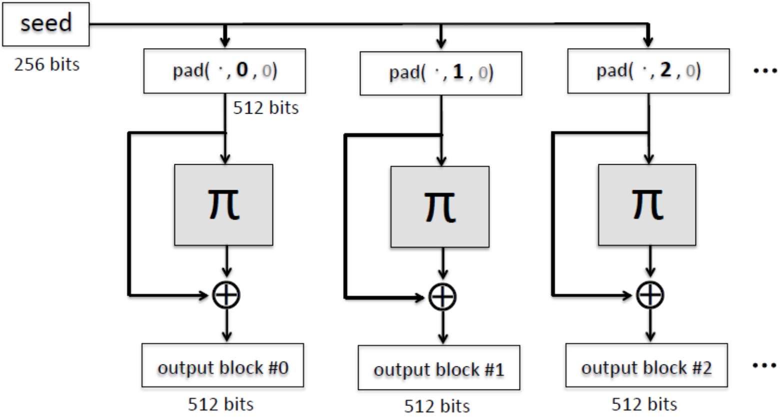


Figure: A schematic of the Salsa and ChaCha PRGs

Invocations of QuarterRound

First Four

$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \qquad \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix}$$

$QuarterRound(x_0, x_4, x_8, x_{12})$ $QuarterRound(x_1, x_5, x_9, x_{13})$

$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \qquad \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix}$$

$QuarterRound(x_2, x_6, x_{10}, x_{14})$ $QuarterRound(x_3, x_7, x_{11}, x_{15})$

Figure: Invocations of Quarter Round

Invocations of QuarterRound

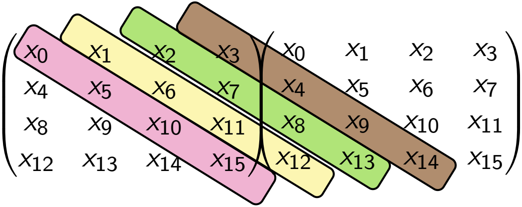
Next Four

$$QuarterRound(x_0, x_5, x_{10}, x_{15})$$

$$QuarterRound(x_1, x_6, x_{11}, x_{12})$$

$$QuarterRound(x_2, x_7, x_8, x_{13})$$

$$QuarterRound(x_3, x_4, x_9, x_{14})$$



$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix}$$

Figure: Invocations of QuarterRound - Broken Diagonals

Padding Function

Matrix Representation of Inputs and Outputs

The output is arranged in a 4×4 matrix of 32 -bit words as follows:

$$\begin{pmatrix} x_0 & x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 & x_7 \\ x_8 & x_9 & x_{10} & x_{11} \\ x_{12} & x_{13} & x_{14} & x_{15} \end{pmatrix} \longleftarrow \begin{pmatrix} c_0 & c_1 & c_2 & c_3 \\ s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ j_0 & j_1 & n_0 & n_1 \end{pmatrix}$$

The Input Matrix

Constants, Seed, Counter and Nonce

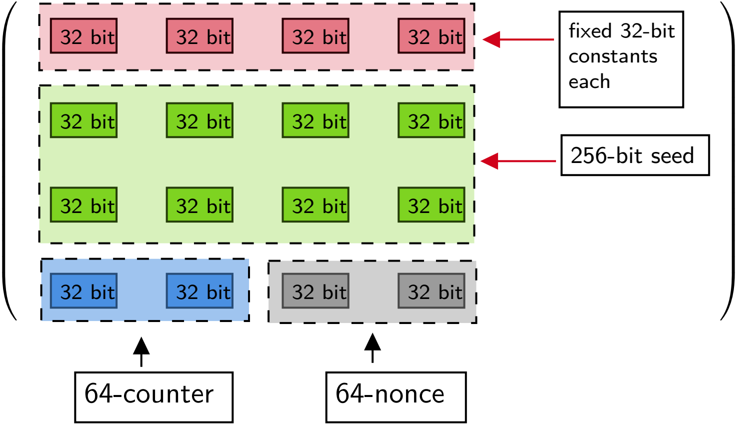


Figure: Input Matrix

Permutation π

The Fixed Public Permutation

- The permutation $\pi : \{0,1\}^{512} \to \{0,1\}^{512}$ is constructed by iterating a simple permutation a fixed number of times.
- The 512-bit input to π is treated as a 4×4 array of 32-bit words denoted by $x_0, \ldots, x_{15}$.
- In ChaCha20 the function π is implemented by repeating the following sequence of steps ten times:
  1. QuarterRound (x_0, x_4, x_8, x_{12})
  2. QuarterRound (x_1, x_5, x_9, x_{13})
  3. QuarterRound $(x_2, x_6, x_{10}, x_{14})$
  4. QuarterRound $(x_3, x_7, x_{11}, x_{15})$
  5. QuarterRound $(x_0, x_5, x_{10}, x_{15})$
  6. QuarterRound $(x_1, x_6, x_{11}, x_{12})$
  7. QuarterRound (x_2, x_7, x_8, x_{13})
  8. QuarterRound (x_3, x_4, x_9, x_{14})

C code for Quarter Round

Chacha20

A macro ROTL (a, b) that rotates left a 32-bit word a by b bits:

c define ROTL(a,b) (((a) « (b)) — ((a) » (32 - (b)))) a += b; d $^{=}a; ROTL(d, 16); c+ =$
$d; b^{=}c; ROTL(b, 12); a+ = b; d^{=}a; ROTL(d, 8); c+ = d; b^{=}c; ROTL(b, 7);$

Building Pseudorandom Generators

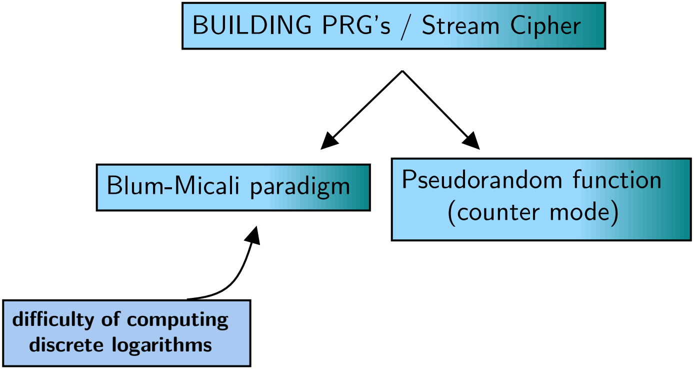

BUILDING PRG's / Stream Cipher
Blum-Micali paradigm
Pseudorandom function (counter mode)
difficulty of computing discrete logarithms

Figure: Building PRGs in Practice

eStream Portfolio

Profile 1 (SW)	Profile 2 (HW)
HC-128	Grain v1
Rabbit	MICKEY 2.0
Salsa20/12	Trivium
SOSEMANUK	

PRGs underlying Salsa and ChaCha stream ciphers

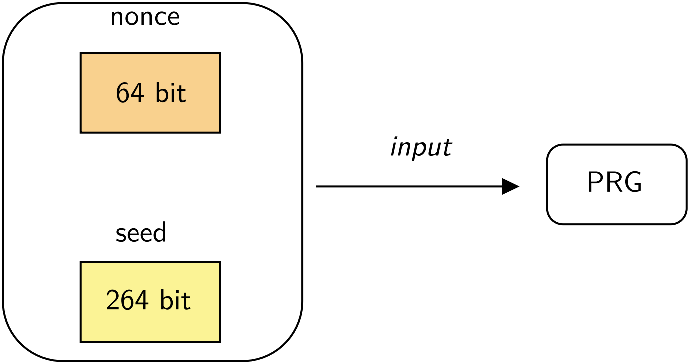


Figure: Inputs to the PRG

PRGs underlying Salsa and ChaCha stream ciphers

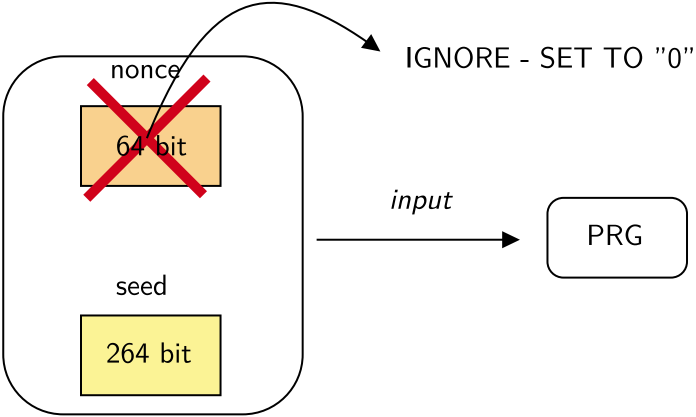


Figure: Inputs to the PRG

High Level Structure

For Salsa and Chacha families

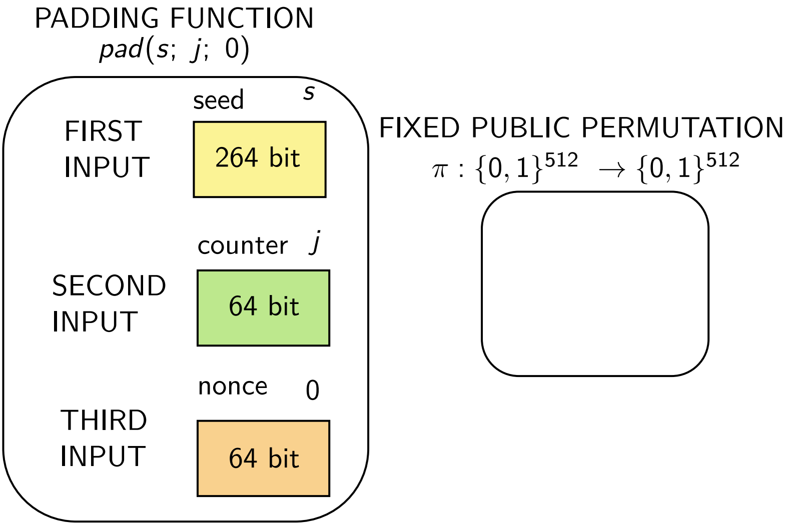


Figure: Salsa and ChaCha PRGs - high-level structure

PRGs underlying Salsa and ChaCha stream ciphers

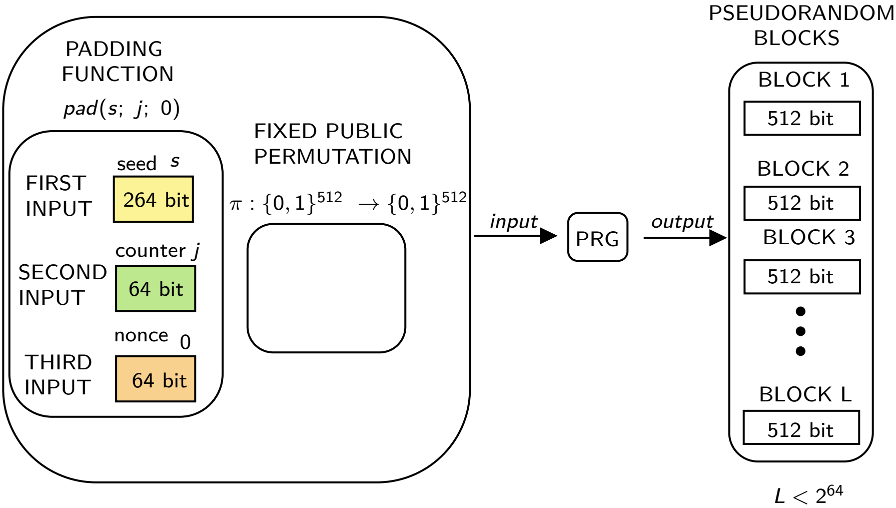


Figure: Inputs and Outputs to the PRG - High Level Structure

PRGs underlying Salsa and ChaCha - Algorithm

```
input: seed $s \in \{0,1\}^{256}$
1  for $j \leftarrow 0$ to $L-1$
2  $h_j \leftarrow \text{pad}(s, j, 0) \in \{0,1\}^{512}$
3  $r_j \leftarrow \pi(h_j) \oplus h_j$ (The Hard Part)
output: $(r_0, \ldots, r_{L-1})$.
```

References

1. Boneh, Dan, and Victor Shoup. ”A graduate course in applied cryptography.” Draft 0.5 (2020).
2. Bernstein, Daniel J. ”The Salsa20 family of stream ciphers.” New stream cipher designs: the eSTREAM finalists (2008): 84-97.
3. Bernstein, Daniel J. ”ChaCha, a variant of Salsa20.” Workshop record of SASC. Vol. 8. No. 1. 2008.

Difference between Chacha and Salsa

Salsa	Chacha
Older stream cipher designed to be secure and fast on embedded devices with limited computing resources	variant of Salsa with a number of modifications and improvements, such as increased security, increased speed, and reduced code size.
64-bit block size and operates on 8x8 matrix of bytes, and has a fixed 20-round structure.	128-bit block size and can have a variable number of rounds, with the default being 20 rounds.
simple and efficient stream cipher	more flexible design and can be used with a broader range of algorithms, including encrypting data for transport layer security (TLS) and internet protocols

Definitions

- **nonce** - In normal terms means something that is used only once. In cryptography, A nonce is a random or semi-random number generated for a specific use. The term means "number used once" or "number once".
- **Transport Layer Security (TLS)** encrypts data sent over the Internet to ensure that eavesdroppers and hackers cannot see what you transmit, which is particularly useful for private and sensitive information such as passwords, credit card numbers, and personal correspondence.
- **SSH or Secure Shell** is a network communication protocol that enables two computers to communicate and share data.