

# Pseudorandom Generator

Chanathip Namprempre

Computer Science  
Reed College

# Agenda: Pseudorandom Generator

1. What is a PRG?
2. PRG-based stream cipher
3. Next-bit unpredictability
4. PRG security notion
5. Examples of PRGs
  - 5.1 Toy
  - 5.2 MS-PPTP
  - 5.3 802.11b WEP
  - 5.4 eStream
6. PRG security vs. unpredictability

# Pseudorandom Generator

Let  $n > s$ .

$$G : \{0, 1\}^s \longrightarrow \{0, 1\}^n$$

Use PRG to approximate OTP.

We call this a **PRG-based stream cipher**.

$$C \leftarrow G(K) \oplus M$$

# Unpredictability is important

Sendmail: fixed format e.g. email messages begin with "From:"

1. Snoop ciphertext  $C$
2.  $X \leftarrow C \oplus \text{"From:"}$
3.  $X$  is the first part of the output of  $G(K)$



Bottom line: If  $G$  is predictable, then a small prefix reveals entire message.

# PRG security notion

## Definition (PRG)

Let  $s, n$  be positive integers.

### Subroutines

Subroutine Initialize

$b \xleftarrow{\$} \{0, 1\}$

If  $b = 1$

then  $x \xleftarrow{\$} \{0, 1\}^s$ ;  $y \leftarrow G(x)$

else  $y \xleftarrow{\$} \{0, 1\}^n$

Return  $y$

Subroutine Finalize( $d$ )

Return ( $d = b$ )

### Experiment

Experiment  $\text{Exp}_G^{\text{prg}}(A)$

$y \xleftarrow{\$} \text{Initialize}$

$d \xleftarrow{\$} A(y)$

Return Finalize( $d$ )

We define the **prg advantage** of an adversary  $A$  attacking  $G$  as

$$\text{Adv}_G^{\text{prg}}(A) = 2 \cdot \Pr \left[ \text{Exp}_G^{\text{prg}}(A) \Rightarrow \text{true} \right] - 1.$$

# Next-Bit Unpredictability

## Definition (Unpredictability)

Let  $s, n, i$  be positive integers.

### Subroutines

Subroutine Initialize

$x \xleftarrow{\$} \{0, 1\}^s ; y \leftarrow G(x)$   
Return  $y[1, \dots, i]$

Subroutine Finalize( $z$ )

Return  $(z = y[i+1])$

### Experiment

Experiment  $\text{Exp}_G^{\text{unP}}(A)$

$y' \xleftarrow{\$} \text{Initialize}$   
 $z \xleftarrow{\$} A(y')$   
Return Finalize( $z$ )

We define the **next-bit unpredictability advantage** of an adversary  $A$  attacking  $G$  as

$$\text{Adv}_G^{\text{unP}}(A) = 2 \cdot \Pr \left[ \text{Exp}_G^{\text{unP}}(A) \Rightarrow \text{true} \right] - 1 .$$

# Examples

1. A generator  $G$  such that, for all  $K$ ,

$$\text{XOR}(G(K)) = 1 .$$

2. MS-PPTP in Windows NT
3. 802.11b WEP
4. eStream

## Toy Example

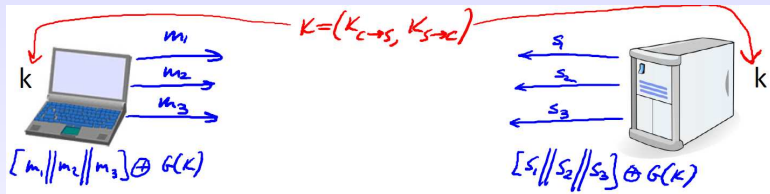
Consider a generator  $G$  such that, for all  $K$ ,

$$\text{XOR}(G(K)) = 1 .$$

Is  $G$  secure under the PRG notion? Prove your answer!



# Two-Time Pad is insecure



Bottom line: The secret key is being used twice, one for each direction. This is a two-time pad.

Source: Dan Boneh Coursera

## Two-Time pad spelled out

Let  $G : \{0, 1\}^s \rightarrow \{0, 1\}^n$  be a PRG. We define a symmetric encryption scheme  $(\mathcal{K}, \mathcal{E}, \mathcal{D})$  as follows.

$\mathcal{K} : S \xleftarrow{\$} \{0, 1\}^s ; \text{ return } K.$

$\mathcal{E}_S(M) :$

if  $(|M| > n \text{ or } |M| = 0)$

return  $\perp$

$K \leftarrow \text{first } |M| \text{ bits of } G(S)$

return  $K \oplus M$

$\mathcal{D}_S(C) :$

if  $(|C| > n \text{ or } |C| = 0)$

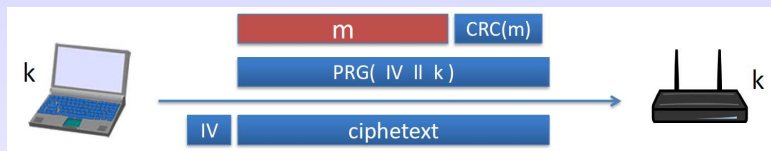
return  $\perp$

$K \leftarrow \text{first } |C| \text{ bits of } G(S)$

return  $K \oplus C$

Can you prove that this encryption scheme is insecure under IND-CPA?

# WEP is insecure



- ▶ For WEP-40,  $|IV| = 24$  bits.  $|k| = 40$  bits.
- ▶ For WEP-104,  $|IV| = 24$  bits.  $|k| = 104$  bits.
- ▶ For WEP-104, PRG:  $\{0, 1\}^{128} \rightarrow 2048$  is RC4.
- ▶ IV is incremented by 1 per frame.
- ▶ IV repeats after  $2^{24} \approx 16M$  frames.
- ▶ On some 802.11 cards, IV resets to 0 after power cycle.

Bottom line:

**Problem #1:** The pad is being used twice whenever IV is reset.

# WEP issues

Actually, the picture is misleading. An 802.11 packet is usually 2000 bytes long. So, WEP breaks up each packet into frames.

| frame # | key stream                  |
|---------|-----------------------------|
| 1       | $\text{PRG}(1 \parallel k)$ |
| 2       | $\text{PRG}(2 \parallel k)$ |
| 3       | $\text{PRG}(3 \parallel k)$ |
| ...     | ...                         |

- ▶ **Problem #2:** The seeds are very much related. This **breaks security assumption** underlying PRG security definition.
- ▶ Attacks exploiting this using 1M frames were found in 2001.
- ▶ Recent attacks use about 40K frames.

The Big Question remains: How do we use a PRG to encrypt a long stream of data?

# Modern PRGs

Question: How do we use a PRG to encrypt a long stream of data?

Answer: Cheat! Add a “nonce.”

Old:

$$\text{PRG} : \{0, 1\}^s \rightarrow \{0, 1\}^n$$

New:

$$\text{PRG} : \{0, 1\}^s \times \mathcal{N} \rightarrow \{0, 1\}^L$$

- ▶  $\mathcal{N}$  is the nonce space, e.g.,  $\{0, 1\}^{64}$  (Salsa and Chacha).
- ▶  $L$  is much larger than  $n$ , e.g.,  $2^{73}$  bits (Salsa and Chacha).

**Restriction:** Nonces are not allowed to repeat.

## Nonce-Based Stream Cipher

$$\text{Enc}'(K, N, M) = M \oplus \text{PRG}'(K, N)$$

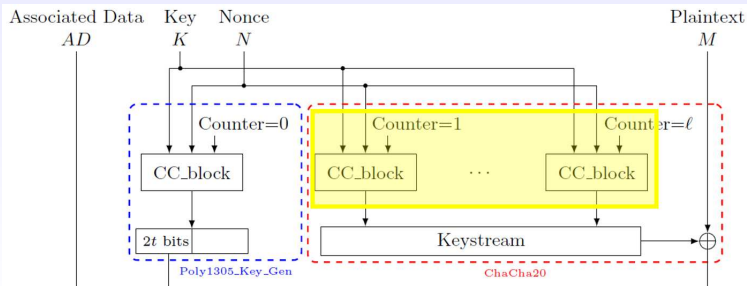
$$\text{Dec}'(K, N, C) = C \oplus \text{PRG}'(K, N)$$

# Chacha20-Poly1305 AEAD

## Nonce-Based Stream Cipher

$$\text{Enc}'(K, N, M) = M \oplus \text{PRG}'(K, N)$$

$$\text{Dec}'(K, N, C) = C \oplus \text{PRG}'(K, N)$$



# Compare old-style PRG vs. new-style PRG for building stream cipher

## Typical Stream Cipher: SE

$$\text{Enc}(K, M) = M \oplus \text{PRG}(K)$$

$$\text{Dec}(K, C) = C \oplus \text{PRG}(K)$$

This construction is **insecure** against IND-CPA!

## Nonce-Based Stream Cipher: SE'

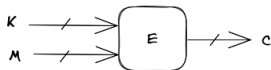
$$\text{Enc}'(K, N, M) = M \oplus \text{PRG}'(K, N)$$

$$\text{Dec}'(K, N, C) = C \oplus \text{PRG}'(K, N)$$

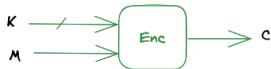
IND-CPA security proofs for this construction model  $\text{PRG}'$  as a **PRF**!

# Pattern emerging

## BLOCK CIPHER



## ENCRYPTION SCHEME



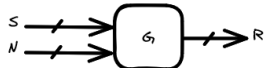
## NONCE-BASED ENCRYPTION SCHEME



## PRG



## MODERN PRG



## STREAM CIPHER



## MODERN STREAM CIPHER





# PRG security vs. Unpredictability

They are equivalent!

## Theorem

Let  $G$  be a PRG. Then, it is secure if and only if it is unpredictable.

$[ \implies ]$  Easy.

$[ \impliedby ]$  Hybrid proof, e.g., Boneh-Shoup Theorem 3.6.