Stream Ciphers

Rohit Musti

CUNY - Hunter College

February 9, 2022

Overview

- We just introduced the concept of semantic security, but referred to it rather obliquely.

Overview

- We just introduced the concept of semantic security, but referred to it rather obliquely.
- While we feel comfortable with its mechanisms, we have not defined an actual cipher that meets its requirements using short keys.

Pseudo-Random Generators (PRG)

- So the primary challenge of using short keys of length ℓ is deriving longer keys that allow us to encrypt messages of length L.

Pseudo-Random Generators (PRG)

- So the primary challenge of using short keys of length ℓ is deriving longer keys that allow us to encrypt messages of length L.
- An PRG can be described as

$$G : \ell \rightarrow L$$

Pseudo-Random Generators (PRG)

- So the primary challenge of using short keys of length ℓ is deriving longer keys that allow us to encrypt messages of length L.
- An PRG can be described as

$$G : \ell \rightarrow L$$

- Modified OTP based on a secure PRG:

$$E(s, m) = G(s) \oplus m$$

Pseudo-Random Generators (PRG)

- So the primary challenge of using short keys of length ℓ is deriving longer keys that allow us to encrypt messages of length L.
- An PRG can be described as

$$G : \ell \rightarrow L$$

- Modified OTP based on a secure PRG:

$$E(s, m) = G(s) \oplus m$$

$$D(s, c) = G(s) \oplus c$$

Security of Modified OTP

- The security rests on whether or not $G(s)$ can be distinguished from k where $s \in \{0,1\}^\ell$ and $k \in \{0,1\}^L$

Security of Modified OTP

- The security rests on whether or not $G(s)$ can be distinguished from k where $s \in \{0,1\}^{\ell}$ and $k \in \{0,1\}^{L}$
- What is convenient is that if we prove this to be true, then we can inherit the semantic security from the original OTP!

PRG Security Game

- Let $S := \{0,1\}^\ell$ and $K := \{0,1\}^L$, s.t. $\ell < L$. Let our adversary A be computationally bounded.

PRG Security Game

- Let $S := \{0,1\}^{\ell}$ and $K := \{0,1\}^{L}$, s.t. $\ell < L$. Let our adversary A be computationally bounded.
- If experiment 0, challenger samples $s \overset{R}{\longleftarrow} S$ and generates $G(s) = k$ and sends to $\mathcal{A}$.

PRG Security Game

- Let $S := \{0,1\}^{\ell}$ and $K := \{0,1\}^{L}$, s.t. $\ell < L$. Let our adversary A be computationally bounded.
- If experiment 0, challenger samples $s \xleftarrow{R} S$ and generates $G(s) = k$ and sends to $\mathcal{A}$. If experiment 1, challenger samples $k \xleftarrow{R} K$ and sends to $\mathcal{A}$.

PRG Security Game

- Let $S := \{0,1\}^\ell$ and $K := \{0,1\}^L$, s.t. $\ell < L$. Let our adversary A be computationally bounded.
- If experiment 0, challenger samples $s \overset{R}{\leftarrow} S$ and generates $G(s) = k$ and sends to $\mathcal{A}$. If experiment 1, challenger samples $k \overset{R}{\leftarrow} K$ and sends to $\mathcal{A}$.
- $\mathcal{A}$ returns b, guessing which experiment the challenger conducted.

PRG Security Game

- Let $S := \{0,1\}^\ell$ and $K := \{0,1\}^L$, s.t. $\ell < L$. Let our adversary A be computationally bounded.
- If experiment 0, challenger samples $s \overset{R}{\leftarrow} S$ and generates $G(s) = k$ and sends to $\mathcal{A}$. If experiment 1, challenger samples $k \overset{R}{\leftarrow} K$ and sends to $\mathcal{A}$.
- $\mathcal{A}$ returns b, guessing which experiment the challenger conducted.

The PRG advantage of $\mathcal{A}$: $PRGA[\mathcal{A}, G] = |Pr[W_0] - Pr[W_1]|$ where W_b is the probability that $\mathcal{A}$ outputs 1 during experiment b.

Takeaways

- If we have a secure PRG, then we can encrypt messages efficiently given a key shorter than the message.

Takeaways

- If we have a secure PRG, then we can encrypt messages efficiently given a key shorter than the message.
- We don't actually know if PRGs exist. Proving that they exist would demonstrate that $P = NP$.
- This is the bulk of cryptography: we assume that certain problems are hard and use those hardness properties to develop secure protocols.

Content Scrambling System: DVDs

- Idea: approximate PRG by using Linear Feedback Shift Registers (LFSR)

Content Scrambling System: DVDs

- Idea: approximate PRG by using Linear Feedback Shift Registers (LFSR)
- This a very weak approximation

Content Scrambling System: DVDs

- Idea: approximate PRG by using Linear Feedback Shift Registers (LFSR)
- This a very weak approximation
- This disadvantage was compounded by the fact that it was illegal in the United States for manufacturers to export cryptographic systems with keys exceeding 40 bits.

Content Scrambling System: DVDs

- Idea: approximate PRG by using Linear Feedback Shift Registers (LFSR)
- This a very weak approximation
- This disadvantage was compounded by the fact that it was illegal in the United States for manufacturers to export cryptographic systems with keys exceeding 40 bits.
- I wonder which US government actor is interested in people not having strong encryption... hmm... beats me ;)

Breaking CSS

- Idea: use the stream output bytes and the outputs of one of the smaller LFSRs to obtain the output of another one of the LFSRs.

Breaking CSS

- Idea: use the stream output bytes and the outputs of one of the smaller LFSRs to obtain the output of another one of the LFSRs.
- Mechanism: you brute force guess the smaller LFSR and each guess forces the state of the larger LFSRs, you can check if the output matches to vaildate.

Breaking CSS

- Idea: use the stream output bytes and the outputs of one of the smaller LFSRs to obtain the output of another one of the LFSRs.
- Mechanism: you brute force guess the smaller LFSR and each guess forces the state of the larger LFSRs, you can check if the output matches to vaildate.
- This is better than brute forcing the whole key 2^{40},

Breaking CSS

- Idea: use the stream output bytes and the outputs of one of the smaller LFSRs to obtain the output of another one of the LFSRs.
- Mechanism: you brute force guess the smaller LFSR and each guess forces the state of the larger LFSRs, you can check if the output matches to vaildate.
- This is better than brute forcing the whole key 2^{40}, by 1999 the full key-recovery attack ran in 18 seconds.

Breaking CSS

- Idea: use the stream output bytes and the outputs of one of the smaller LFSRs to obtain the output of another one of the LFSRs.
- Mechanism: you brute force guess the smaller LFSR and each guess forces the state of the larger LFSRs, you can check if the output matches to vaildate.
- This is better than brute forcing the whole key 2^{40}, by 1999 the full key-recovery attack ran in 18 seconds.
- Great article in WIRED about this click here

Other Examples

- GSM encryption (uses 2 LSFRs), they tried to keep this design private but it was eventually reverse engineered and attacks were found

Other Examples

- GSM encryption (uses 2 LSFRs), they tried to keep this design private but it was eventually reverse engineered and attacks were found a key lesson here is to never rely on security through obscurity.

Other Examples

- GSM encryption (uses 2 LSFRs), they tried to keep this design private but it was eventually reverse engineered and attacks were found a key lesson here is to never rely on security through obscurity.
- The Snowden documents revealed that the NSA can process data encrypted with A5/1 (uses 3 LSFRs)

Other Examples

- GSM encryption (uses 2 LSFRs), they tried to keep this design private but it was eventually reverse engineered and attacks were found a key lesson here is to never rely on security through obscurity.
- The Snowden documents revealed that the NSA can process data encrypted with A5/1 (uses 3 LSFRs) (important to know that there were known attacks on this protocol well before the NSA was known to have cracked it)

Other Examples

- GSM encryption (uses 2 LSFRs), they tried to keep this design private but it was eventually reverse engineered and attacks were found a key lesson here is to never rely on security through obscurity.
- The Snowden documents revealed that the NSA can process data encrypted with A5/1 (uses 3 LSFRs) (important to know that there were known attacks on this protocol well before the NSA was known to have cracked it)
- Bluetooth E0 (uses 4 LSFRs) has been broken

Other Examples

- GSM encryption (uses 2 LSFRs), they tried to keep this design private but it was eventually reverse engineered and attacks were found a key lesson here is to never rely on security through obscurity.
- The Snowden documents revealed that the NSA can process data encrypted with A5/1 (uses 3 LSFRs) (important to know that there were known attacks on this protocol well before the NSA was known to have cracked it)
- Bluetooth E0 (uses 4 LSFRs) has been broken
- RC4 cypher, used in SSL/TLS is also broken (though not as badly)