

Cryptography

This is my note of [Cryptography 1](#) by **Dan Boneh**

- [Cryptography](#)
 - [Overview](#)
 - [Crypto core](#)
 - [Three steps in cryptography](#)
 - [History](#)
 - [Substitution Cipher](#)
 - [Vigener Cipher](#)
 - [Roter Machines](#)
 - [Data Encryption Standard](#)
 - [Randomized algorithm](#)
 - [An important property of XOR](#)
 - [The birthday paradox](#)
 - [Stream Cipher](#)
 - [The One Time Pad](#)
 - [Information Theoretic Security](#)
 - [Pseudorandom Generators](#)
 - [Attacks on OTP and stream ciphers](#)
 - [Attack 1: two time pad is insecure](#)
 - [Attack 2: no integrity](#)
 - [Real-world stream ciphers](#)
 - [RC4](#)
 - [CSS](#)
 - [eStream](#)
 - [PRG Security Defs](#)
 - [Statistical Tests](#)
 - [Advantage](#)
 - [Semantic Security](#)
 - [Stream ciphers are semantically secure](#)
 - [Block Cipher](#)
 - [PRFs and PRPs](#)
 - [Example PRPs\(also block ciphers\):](#)
 - [Secure PRFs and PRPs](#)
 - [Secure PRFs](#)

- Secure PRPs
- An easy application: PRF \Rightarrow PRG
- DES
 - Core idea : Feistel Network
- DES: 16 round Feistel Network

Overview

Crypto core

- Secret key establishment
- Secure communication
 - confidentially
 - integrity
- Digital signature
- Anonymous communication
- Anonymous **digital** cash
 - Can I spend a **digital coin** without anyone knowing who I am?
 - How to prevent double spending?
- Secure multi-party computation
 - **Thm:** anything that can be done with trusted authority can also be done without trusted authority
- Privately outsourcing computation
 - Get search result from Google while Google doesn't know what you search for
- **Zero knowledge**
 - one party (the prover) can prove to another party (the verifier) that they know a value x , **without conveying any information** apart from the fact that they know the value x

Three steps in cryptography

1. Precisely specify threat model
 - What an attacker can do
 - What an attacker's goal is
2. Propose a construction
3. Prove that breaking construction under threat model will solve an underlying hard problem

History

Substitution Cipher

- Use frequency of English letters and pairs of letters to easily break it

Vigener Cipher

- The key is a word
- Just break it as substitution cipher

Roter Machines

- Early example: the Hebern machine (single rotor)
- Most famous: the Enigma (3-5 rotors)
 - Designed to defend frequency attack (statistic attack)
 - keys = $26^4 = 2^{18}$
 - Still can't defend the ciphertext only attack

Data Encryption Standard

- DES: #keys = 2^{56} , block size = 64 bits
- Today: AES(2001), Salsa20(2008) (and many others)

Randomized algorithm

- Deterministic algorithm: $y \leftarrow A(m)$
 - output is a deterministic value
- Randomized algorithm: $y \leftarrow A(m; r)$ where $r \xleftarrow{R} \{0, 1\}^n$
 - output is a random variable $y \xleftarrow{R} A(m)$

An important property of XOR

Thm: Y a rand. var. on $\{0, 1\}^n$, X an indep. **uniform** var. on $\{0, 1\}^n$, Then $Z = Y \oplus X$ is a **uniform** var. on $\{0, 1\}^n$ s

Proof: Just consider $n = 1$

Y	Pr.
0	p_0
1	p_1

X	Pr.
---	-----

X	Pr.
0	$\frac{1}{2}$
1	$\frac{1}{2}$

X	Y	Pr.
0	0	$\frac{p_0}{2}$
0	1	$\frac{p_1}{2}$
1	0	$\frac{p_0}{2}$
1	1	$\frac{p_1}{2}$

$$\begin{aligned}
 P\{Z = 0\} &= P\{ (x, y) = (0, 0) \cup (x, y) = (1, 1) \} \\
 &= p_0/2 + p_1/2 \\
 &= 1/2
 \end{aligned}$$

The birthday paradox

Let $r_1, \dots, r_n \in U$ be indep. identically distributed random vars.

Thm: when $n = 1.2 \times |U|^{1/2}$ then

$$P\{\exists i \neq j : r_i = r_j\} \geq 1/2$$

Stream Cipher

The One Time Pad

- First example of a **secure** cipher
- key = (random bit string as long the message)

$$E(k, m) = k \oplus m$$

$$D(k, c) = k \oplus c$$

- Very fast enc/dec
- but long keys as long as PT

Information Theoretic Security

- Shannon 1949
- **Def:** A cipher (E, D) over (K, M, C) has **perfect secrecy** if

$$\forall m_0, m_1 \in M (|m_0| = |m_1|), \forall c \in C$$

$$P\{E(k, m_0) = c\} = P\{E(k, m_1) = c\}$$

- **No CT only attack**
- **Lemma:** OTP has **perfect secrecy**
 - **Proof:** $\forall m \in M, c \in C$, There is exactly one key $(m \oplus c)$ maps m to c
- **Thm:** perfect secrecy $\Rightarrow |K| \geq |M|$
- Hard to use in practice!

Pseudorandom Generators

- **Stream Ciphers:** making OTP practical
 - Idea: replace **random** key by "pseudorandom" key
 - Goal: decrease the length of key
- Stream ciphers **cannot** have perfect secrecy
 - Need a different definition of security
 - Security will depend on specific PRG
- **WEAK** PRG: glibc random():

$$r[i] \leftarrow (r[i - 3] + r[i - 31]) \% 2^{32}$$

$$\text{return } r[i] \gg 1$$

- PRG **MUST** be unpredictable
- We say that $G : K \rightarrow \{0, 1\}^n$ is **predictable** if:

$$\exists \text{ alg. } A, \exists 0 \leq i \leq n - 1$$

$$P\{A(G(k))|_{1..i} = G(k)|_{i+1}\} > 1/2 + \varepsilon$$

- **Def:** PRG is **unpredictable** if it is not predictable
- $\forall i$, no **efficient** alg. can predict bit_{i+1} for **non-negligible** ε
- Negligible
 - In practice: ε is a scalar
 - non-neg: $\varepsilon \geq 1/2^{30}$ (likely to happen over 1 GB of data)
 - negligible: $\varepsilon \leq 1/2^{80}$ (won't happen over life of key)

- In theory: ε is a function $\mathbb{Z}^{\geq 0} \rightarrow \mathbb{R}^{\geq 0}$
 - non-neg: $\exists d : \varepsilon(\lambda) \geq 1/\lambda^d$ inf. often
 - negligible: $\forall d, \lambda \geq \lambda_d : \varepsilon(\lambda) < 1/\lambda^d$

Attacks on OTP and stream ciphers

Attack 1: two time pad is insecure

Never use stream cipher key more than once

$$c_1 \leftarrow m_1 \oplus PRG(k)$$

$$c_2 \leftarrow m_2 \oplus PRG(k)$$

Then Adv. does:

$$c_1 \oplus c_2 \rightarrow m_1 \oplus m_2$$

Enough redundancy in English and ASCII encoding that:

$$m_1 \oplus m_2 \rightarrow m_1, m_2$$

- 802.11b WEP

$$c = m \oplus PRG(IV || k)$$

1. IV increases by one every frame, but length of IV is only 24 bits. After $2^{24} \approx 16M$ frames it repeats.
2. keys are related (only 24 of 1048 bits are different), And PRG used in WEP(RC4) is not secure when you use related keys.

A better construction

Also use $PRG(k)$ to generate new keys so that each frame has a pseudorandom key.

- Disk encryption
When the file changes, it's easy to tell where the change occurred instantly. That leaks information that attackers shouldn't actually know. Essentially it's another example of two time pad.

Typically do not use a stream cipher in disk encryption

Attack 2: no integrity

OTP is malleable. Modifications to CT are undetected and have **predictable** impact on PT.

$$D(E(m, k) \oplus p, k) = m \oplus p$$

Attackers can choose p to modify PT.

Real-world stream ciphers

RC4

Software cipher

$$k(128 \text{ bits}) \rightarrow k'(2048 \text{ bits}) \rightsquigarrow 1 \text{ byte per round}$$

- used in HTTPS and WEP
- Weaknesses:
 - Bias in initial output: $P\{2^{nd} \text{ byte} = 0\} = 2/256 > 1/256$
 - $P\{(0, 0)\} = 1/256^2 + 1/256^3 > 1/256^2$
 - Related key attacks

CSS

Hardware cipher **(badly broken)**

Using [Linear-feedback shift register \(LFSR\)](#)

eStream

a new kind of PRG : $\{0, 1\}^s \times \text{Nonce} \rightarrow \{0, 1\}^n$

Nonce: a non-repeating value for a given key.

$$E(k, m; r) = m \oplus PRG(k; r)$$

The pair (k, r) is never used more than once.

Nonce is designed to reuse the key more than once.

A famous and successful example: [Salsa 20](#)

PRG Security Defs

Let $G: K \rightarrow \{0, 1\}^n$ be a PRG

Goal: define what it means that

$$k \xleftarrow{R} K, \text{output } G(k)$$

is **indistinguishable** from

$$r \xleftarrow{R} R, \text{output } r$$

Statistical Tests

Def: an alg. A s.t. $A(x)$ outputs "0" or "1"

Advantage

Let $G:K \rightarrow \{0,1\}^n$ be a PRG , $\{0,1\}^n \rightarrow r$ and A a **statistical test** on $\{0,1\}^n$

Def:

$$Adv_{PRG}[A, G] = |Pr[A(G(k)) = 1] - Pr[A(r) = 1]| \in [0, 1]$$

Adv close to 1 \Rightarrow A can dist. G from random

Adv close to 0 \Rightarrow A cannot dist. G from random

Def: We say that $G : K \rightarrow \{0,1\}^n$ is a **secure PRG** if \forall "eff" stat. tests A : $Adv_{PRG}[A, G]$ is **negligible**.

Thm: a secure PRG is unpredictable.

Proof: Just show a predictable is insecure.

Thm: an unpredictable PRG is secure.

Proof: https://en.wikipedia.org/wiki/Yao's_test

More generally, let P_1 and P_2 be two distributions over $\{0,1\}^n$

Def: We say that P_1 and P_2 are **computationally indistinguishable** if no **eff.** stat. tests that can distinguish P_1 and P_2 (denoted $P_1 \approx_p P_2$)

Semantic Security

Adv. A gives Chal. two message m_0 and m_1 , Chal. returns $c=E(k, m_0)$ or $c=E(k, m_1)$

Def: E is **semantically secure** if for all efficient A :

$$Adv_{SS}[A, E] = |Pr[c = E(k, m_0)] - Pr[c = E(k, m_1)]| < negligible$$

\Rightarrow for all explicit $m_0, m_1 \in M, k \leftarrow K : \{E(k, m_0)\} \approx_p \{E(k, m_1)\}$

Stream ciphers are semantically secure

Thm: $G : K \rightarrow \{0, 1\}^n$ is a secure $PRG \Rightarrow$ stream cipher E derived from G is semantically secure

Proof: Just to prove: \forall sem. sec. adversary A , \exists a PRG adversary B s.t.

$$Adv_{SS}[A, E] \leq 2Adv_{PRG}[B, G]$$

Block Cipher

PRFs and PRPs

Def: Pseudo Random Function (**PRF**) defined over (K, X, Y) :

$$F : K \times X \rightarrow Y$$

such that exists **efficient** algorithm to evaluate $F(k, x)$

Def: Pseudo Random Permutation (**PRP**) defined over (K, X)

$$E : K \times X \rightarrow X$$

such that:

1. Exist **efficient** algorithm to evaluate $E(k, x)$
2. The function $E(k, \cdot)$ is one-to-one
3. Exists **efficient** inversion algorithm $D(k, y)$

Block cipher is actually a PRP

Example PRPs(also block ciphers):

AES: $K \times X \rightarrow X$ where $K = X = \{0, 1\}^{128}$

3DES: $K \times X \rightarrow X$ where $X = \{0, 1\}^{64}, K = \{0, 1\}^{168}$

Secure PRFs and PRPs

Secure PRFs

Let $F : K \times X \rightarrow Y$ be a PRF

- S_U : the set of all functions from $|X|$ to $|Y|$, $|S_U| = |Y|^{|X|}$

- $S_F = \{F(k, \cdot), k \in K\} \subseteq S_U, |S_F| = |K|$

Def: a PRF is **secure** if a random function in S_U is **indistinguishable** from a random function in S_F

Secure PRPs

If we replace S_U with the set of all **one to one** functions from X to X , then we get **def.** of secure PRP, and actually we also get secure block cipher.

An easy application: PRF \Rightarrow PRG

Let $F : K \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a secure PRF.

Then following $G : K \rightarrow \{0, 1\}^{nt}$ is a secure PRG:

$$G(k) = F(k, 0) || F(k, 1) || \dots || F(k, n-1)$$

It's security is easy to prove from the security of PRF.

DES

Core idea : Feistel Network

Given functions $f_1, \dots, f_d : \{0, 1\}^n \rightarrow \{0, 1\}^n$

Goal: build **invertible** function $F : \{0, 1\}^{2n} \rightarrow \{0, 1\}^{2n}$

Feistel Network uses f as round functions. For each round $i = 0, 1, \dots, d$

1. Split the text block into two equal pieces L_i, R_i
2. compute

$$\begin{cases} R_i = f_i(R_{i-1}) \oplus L_{i-1} \\ L_i = R_{i-1} \end{cases}$$

Then $L_d || R_d$ is the ciphertext.

Decryption is just to replace the second step with

$$\begin{cases} L_{i-1} = f_i(L_i) \oplus R_i \\ R_{i-1} = L_i \end{cases}$$

Then $L_0 || R_0$ is the plaintext.

Thm(Luby-Rackoff '85): If $f : K \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ is a secure PRF, Then 3-round Feistel $F : K^3 \times \{0, 1\}^{2n}$ is a secure PRP.

Now we have a method to build a secure from a secure PRP from a secure PRF. Essentially we have an efficient method to construct invertible functions from normal functions.

DES: 16 round Feistel Network