

CS4236 Cryptography

Theory and Practice

Topic 4 - Private Key System Security:

Multi-encryption, CPA, PRF and PRG

Hugh Anderson


National University of Singapore
School of Computing

August, 2022



An oracle...

Consult the Oracle!



The mystic oracle awaits your question!

Ask her a yes or no question about the future, and ponder her answer with care...

Is the padding in this message correct?... 060606060606

CONSULT THE ORACLE!

[Close this window](#)

wavTones.com.unregis....wav audacity-macos-2.1.3.dmg wavTones.com.unregis....wav [Show All](#) ×

Outline

1 Private Key System Security

- Multi-encryption
- CPA attack
- Pseudo-random functions and generators

Help session on Saturday, meet-up today!

Or extra tutorial, or open house, or town square, or ...

On Saturdays, from 2:00 to 3:00, I am running a zoom session from my home. You can join at any time, and just yell out or something (I will leave the machine running in the living room, and try to keep an eye on it). Last Saturday a few people joined, and there was some extra discussion about

- Reduction proofs, and
- Probability (for Assignment 1)

If you have any questions, contact me somehow. Come to my room, email me, or come and talk to me via zoom on Saturday:

URL: `https://nus-sg.zoom.us/j/82466546798?pwd=Z1FXZnF6OWdCQnJNeFAyTDEzKzFkZz09`

Meeting ID: 824 6654 6798

Passcode: 182428

After the lecture today...

You should meet up with your project group members. I will try to help this process...

The story so far ... where are we?

The last 3 weeks: weekly steps we have taken

- 1 We had a historical introduction and introduced a **framework**.
- 2 We progressed from the traditional view of perfect secrecy, through to a game/experiment view: *perfect* indistinguishability.
- 3 *Perfectly* indistinguishable was relaxed to give *computationally* indistinguishable, where we assume our adversary (and our challenger) are PPT algorithms, and we admit a negligible probability of success. This form of secrecy is the one used by most modern encryption systems. In the mathematical model, the system (Π) is parameterized by n (typically a keysize) which can be increased so that the probability of a successful attack is so small that it can be discarded - it is negligible. The negl probability function was defined. An EAV-Secure system was constructed with a PRG.

Example: Symmetric encryption scheme

$$\text{ENC}_K = (\text{Gen}(1^n), \text{Enc}_K(m), \text{Dec}_K(c)) \quad (\text{ch1})$$

Properties	Defs	Constructions	Proofs
Perfect secrecy	2.3	(One time pad)	Thm 2.9 Lemma 2.6
↓			
Perfect indistinguishability	2.5	3.17 (PRG)	
↓			
EAV-secure	3.8		Thm 3.18 Proof given
↑			
EAV-Multi-encryption	3.19		
↑		3.30 (PRF)	
CPA-Multi-encryption	3.22		Thm 3.24
↓			Thm 3.31
CPA-secure	3.23		

Computational indistinguishability is not enough...

Relaxing requirements

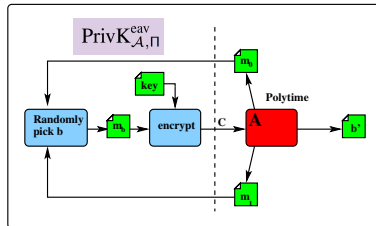
We saw that Perfect Secrecy is impossible in practice. In response, Perfect Secrecy is relaxed (polytime restriction on adversary + allowing negligible probability of failure) to EAV-security. The relaxation seems reasonable. We also saw a construction that is EAV-secure, on the condition that we have a pseudorandom generator (which seems to be the case).

Now, is it happily-ever-after? Is EAV-secure all we need? **No**. The relaxation opens up other practical attacks (consider why we have IVs). Today, we will first look into two different ideas:

- ① “Multi-encryption” security (Definition 3.19) and show that an EAV-secure scheme might fail under Multi-encryption.
- ② Motivated by a practical attack, we define another formulation where the adversary is given an encryption oracle: CPA - (Chosen Plaintext Attack) security (Definition 3.22).

We then formulate a version of CPA with multi-encryption (Definition 3.23), which turns out to be equivalent to Definition 3.22.

The old game: $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}}$



Game definition elements - the adversary chooses two

... messages : m_0 and m_1 . The challenger then chooses b , and encrypts, to obtain

$$C = \text{Enc}_k(m_b)$$

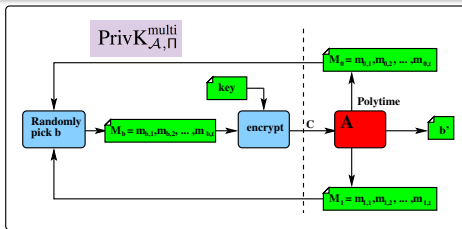
The adversary succeeds if $b' = b$, and game output is $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}} = 1$

Definition 3.8 - EAV-secure

An encryption scheme Π has indistinguishable encryption in the presence of an eavesdropper if, for any \mathcal{A} , there is a negl , s.t.

$$\Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$$

A new game: $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{multi}}$



Game definition elements - the adversary chooses two

... sequences of messages: M_0 and M_1 . The challenger then chooses b , and encrypts, to obtain

$$C = (\text{Enc}_k(m_{b,1}), \dots, \text{Enc}_k(m_{b,t}))$$

The adversary succeeds if $b' = b$, and game output is $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{multi}} = 1$

Definition 3.19 - Multi-secure

An encryption scheme Π has **indistinguishable multi-encryption** in the presence of an eavesdropper if for any \mathcal{A} , there is a negl , s.t.

$$\Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{multi}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$$

Multi-encryption is “stronger”...

Multi-encryption-secure is stronger than EAV-secure

There exists a scheme that is EAV-secure, but not secure under multi-encryption (Definition 3.19) - it is easy to construct...

Multi-secure (3.19) implies EAV-secure (3.8). This is easy to show by defining $\text{Multi-Secure}(\Pi, t)$ over a system Π as in Definition 3.19 (t is just the number of messages), and similarly for $\text{EAV-Secure}(\Pi)$. Then we have that

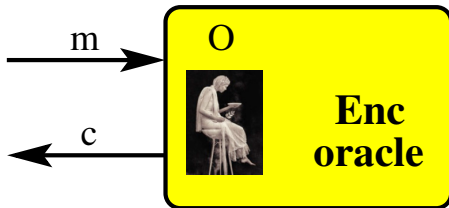
$$\begin{array}{llll} \forall \Pi & \text{Multi-Secure}(\Pi, t) & \longrightarrow & \text{Multi-Secure}(\Pi, 1) \quad (\text{True for all } t) \\ \text{but} & \text{Multi-Secure}(\Pi, 1) & = & \text{EAV-Secure}(\Pi) \quad \mathcal{A}_{\text{multi}} \equiv \mathcal{A}_{\text{eav}} \\ \text{so } \forall \Pi & \text{Multi-Secure}(\Pi, t) & \longrightarrow & \text{EAV-Secure}(\Pi) \end{array}$$

Note that we could also do “Suppose Not”.

The attack scenarios are practical. Eg. the “penguin” example in ECB:



CPA attack uses an encryption “oracle”



Adversary has an encryption oracle

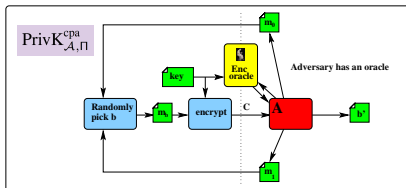
An encryption oracle takes in a message, and outputs the ciphertext, encrypted with some pre-determined key. The key is always the same. The steps of the game indicate how the keys are chosen.

There are two main variants: batch or adaptive. Unless otherwise specified, we assume adaptive.

Batch: Adversary selects multiple queries (messages) and sends to the oracle in a batch.

Adaptive: Adversary sends a query, obtains the result, and decides the next query. That is, adaptive to the outcome of previous queries.

Consider this game: $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cpa}}$



Game definition

The challenger generates a key k , the adversary has multi-access to an oracle $\text{Enc}_k()$ and can choose m_0, m_1 . The challenger chooses b , computes

$$C = \text{Enc}_k(m_b)$$

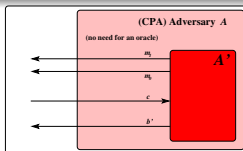
Adversary has C and continues to have access to the encryption Oracle.

Definition 3.22

An encryption scheme Π has indistinguishable encryption under the chosen plaintext attack (or is CPA-secure) if, for any \mathcal{A} , there is a negl , s.t.

$$\Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cpa}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$$

Proof by contrapositive



CPA-secure is stronger than EAV-secure

There exists a scheme that is EAV-secure, but not CPA-secure.

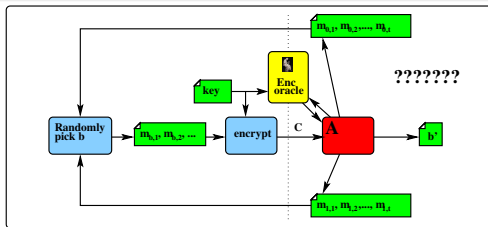
CPA-secure (3.22) implies EAV-secure (3.8). We want to prove $p \rightarrow q$, so we assume $\neg q$, and prove $\neg q \rightarrow \neg p$. We have $\text{CPA-Secure}(\Pi)$ with adversary \mathcal{A} , and $\text{EAV-Secure}(\Pi)$ with adversary \mathcal{A}' . Our assumption:

$$\forall \mathcal{A} \exists \Pi \text{ CPA-Secure}(\Pi) \rightarrow \neg \text{EAV-Secure}(\Pi)$$

The RHS, $\neg \text{EAV-Secure}(\Pi)$ means that there is an \mathcal{A}' that can break EAV - for some m_0, m_1 it can predict $b = b'$ with more than negl . So we construct an adversary \mathcal{A} , for $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cpa}}$, which uses \mathcal{A}' as a subroutine. So this now means that there exists an \mathcal{A} which will break CPA for Π , and so:

$$\begin{aligned} \neg \text{EAV-Secure}(\Pi) &\rightarrow \neg \text{CPA-Secure}(\Pi) \\ \text{so } \text{CPA-Secure}(\Pi) &\rightarrow \text{EAV-Secure}(\Pi) \end{aligned}$$

Multi-encryption and CPA?



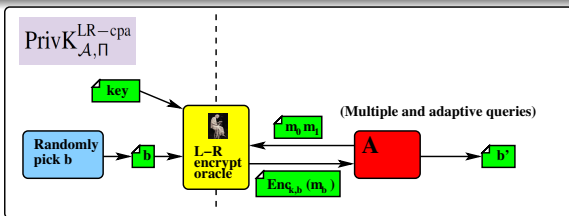
A straight combination?

A combination of them would include an oracle, and multi-encryption, but has new issues. Do we do adaptive (i.e. changing between each of m_0, m_1, \dots or not?). The problem is that the definition starts getting more difficult. With batch mode, we just duplicate $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cpa}}$, but with multi-messages as above. However, with adaptive mode it is more complex, so....

Consider if the adversary has access to a "left-right" encryption oracle. LR-cpa is a more concise game, which can be driven as either CPA or multi.

- 1 Some key k and a bit b is pre-determined.
- 2 A query consists of two messages (m_0, m_1) , the oracle outputs $\text{Enc}_{k,b}(m_b)$.

Consider this game: $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{LR-cpa}}$



Game LR-cpa: Multi-encryption under CPA

The challenger generates a key k and a bit b . The adversary has multi-access to an L-R encryption oracle $\text{Enc}_{k,b}()$, outputs b' , and succeeds if $b = b'$ (in which case, the output of the game is 1).

Definition 3.23

An encryption scheme Π has indistinguishable multi-encryption under the chosen plaintext attack (or is CPA-secure for multi-encryption) if, for any \mathcal{A} , there is a negl , s.t.

$$\Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{LR-cpa}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$$

CPA with multi-encryption...

Note that with game $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{LR-cpa}}$

We can “customize” the game for multiple encryption.

We can “customize” the game for CPA.

Discussion

Is the combined version stronger? ... CPA-secure is equivalent to CPA-secure for multiple encryption.

Theorem 3.24: A scheme that is CPA-secure is also CPA-secure for multiple encryption

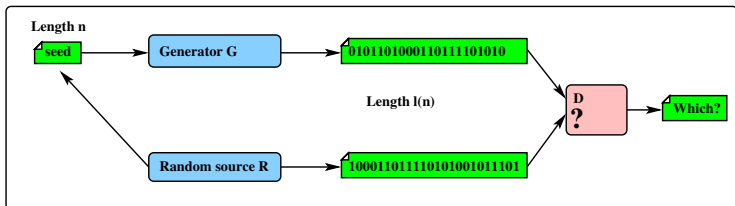
The proof is quite involved. The book delays it until the chapter on public keys.

Pseudo-random functions and generators

Note that...

We use a pseudorandom function to get CPA-secure.

We use a pseudorandom generator to get EVA-secure.



The main idea

A pseudorandom generator outputs a sequence that is computationally indistinguishable from a truly random sequence.

Consider a generator that outputs a function. This function, to an observer who has “oracle” access to this function, should be computationally indistinguishable from a uniformly chosen function.

Pseudo-random functions

How to think of these functions...

Let Func_n be the set of all functions with n -bit input and n -bit output, i.e.

$$f : \{0, 1\}^n \longrightarrow \{0, 1\}^n$$

It is useful to visualize a function f in Func_n to be a 2^n sequence of $f(0), f(1), f(2), \dots$ where $0, 1, 2, \dots$ are binary representations of the respective integers $0, 1, 2, \dots$. Each such function can be represented by a bit string of length $n \times 2^n$. Hence $|\text{Func}_n|$, the size of all possible such sequences is 2^{n2^n} . We can randomly choose a function from Func_n .

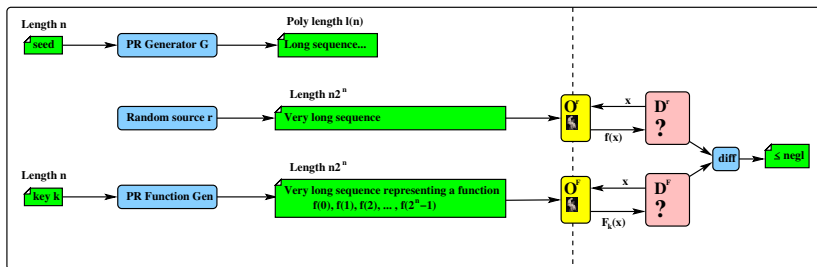
Consider a keyed function F that takes in two inputs, the first input is the key k , the next input is the message m , and the function outputs a ciphertext. So, we have the math notation

$$F : \{0, 1\}^* \times \{0, 1\}^* \longrightarrow \{0, 1\}^*$$

For a F and a fixed key k , for the purpose of crypto, let us (re)define $F_k : \{0, 1\}^* \longrightarrow \{0, 1\}^*$ and use the notation

$$F_k(x) \equiv F(k, x)$$

Pseudo-random functions



Definition 3.25

Let $F : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}^*$ be an efficient length-preserving keyed function. F is a pseudorandom function if for all PPT distinguishers D , there is a negl s.t.

$$\left| \Pr[D^{F_k(\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot)}(1^n) = 1] \right| \leq \text{negl}(n)$$

The first probability is taken over choice of the key k , randomness of D . The second probability is taken over choice of f from Func_n .

(Here, the size of the key is the same as the message. In general, length-preserving only refers to the size of message and ciphertext.)

Pseudo-random functions

The distinguisher D^F and D^f

For short, we wrote D^F and D^f for the two distinguishers. Both D^F and D^f have much the same algorithm. The difference is this: Both have access to an oracle, but the oracles in D^F and D^f are different.

- D^F has access to an oracle O^F where D^F first uniformly chooses a key k , fixed for subsequent accesses. On input x , O^F outputs $F_k(x)$.
- D^f has access to a function f from Func_n , fixed for subsequent accesses. On input x , O^f outputs $f(x)$.

Pseudorandom function vs pseudorandom sequence

In the definition, the distinguisher is PPT and thus can't handle the exponentially long sequence. It is replaced by access to an Oracle, that on input x , outputs $f(x)$. That is, the distinguisher can at most adaptively sample the function poly times.

Relationship, functions with generators

From pseudorandom function, we can construct a pseudorandom generator.
From pseudorandom generator, we can construct a pseudorandom function.

An interesting question...

How can we construct a PRF?

According to Wikipedia,

The guarantee of a PRG is that a single output appears random if the input was chosen at random. On the other hand, the guarantee of a PRF is that all its outputs appear random, regardless of how the corresponding inputs were chosen, as long as the function was drawn at random from the PRF family.

The rand() function found in many languages is not cryptographically secure, but Crypto libraries normally have PRGs that are cryptographically secure.

A PRF can be constructed using a PRG. The GGM (Goldreich, Goldwasser, Micali) technique involves repeated applications of a PRG $G()$. For an n -bit function call $f_k(i)$, there would be only n applications of $G()$, to get the i^{th} element of $f_k()$.

GGM - a PRF $F_k(i)$ from a PRG $G(k)$

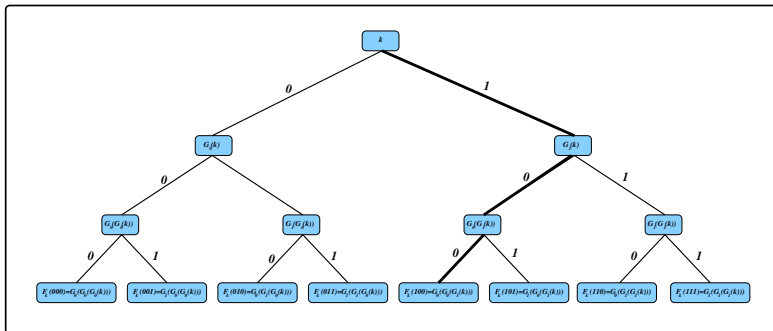
Length doubling PRG $G : \{0, 1\}^n \mapsto \{0, 1\}^{2n}$, and a tree...

Assume we can get bits $0 \dots n-1$, and $n \dots 2n-1$ using functions G_0, G_1 :

$$G_0(s) = G(s)_{0 \dots n-1}$$

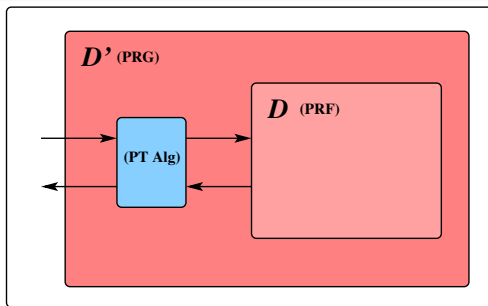
$$G_1(s) = G(s)_{n \dots 2n-1}$$

Given key k we can compute the i^{th} element of the PRF from the bits of i .



This can be proved to be a PRF, given $G()$ is a PRG, and it is PT!

Proof that GGM is a PRF construction



Proof sketch: if G is a PRG, then F is a PRF...

Suppose not, then from Def 3.25 we have a distinguisher D with

$$\left| \Pr \left[D^{F_k(\cdot)}(1^n) = 1 \right] - \Pr \left[D^{f(\cdot)}(1^n) = 1 \right] \right| = \varepsilon(n) \quad \left(\geq \frac{1}{p(n)} \right)$$

For any n -bit PRF (try with 1-bit first), if D can distinguish the PRF, then a distinguisher D' using it can distinguish that level of the PRG. There are n levels, hence D' is still a polynomial time distinguisher. G cannot be a PRG!

Hence F must be a PRF.

Practical PRFs

Some Message Authentication Codes (covered later)...

In the same way that PRGs may or may not exist, but we believe they do, some MACs are provably PRFs, but based on assumptions that seem likely.

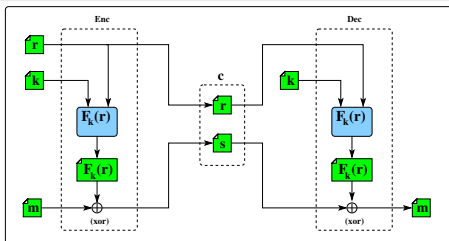
For example:

- HMAC (RFC2104, and <https://en.wikipedia.org/wiki/HMAC>) is provably a PRF if the compression function (again to be covered later) is a PRF.
 - CMAC (RFC4615, and <https://en.wikipedia.org/wiki/CMAC>) is provably a PRF if the underlying block cipher is a PRP
-

All these can be coded using standard libraries - for example Python:

```
from Crypto.Hash import CMAC
from Crypto.Cipher import AES
k = b'Sixteen byte key'
prf = CMAC.new(k, ciphermod=AES)
prf.update(b'An index i')
print(prf.hexdigest())
```

Construction of CPA-secure encryption scheme



Construction 3.30: CPA-secure from PRF

Given a pseudorandom function F construct the following private encryption scheme with length n :

$\text{Gen}(1^n)$: on input 1^n , output a uniform key k of size n .

$\text{Enc}_k(m)$: on input k and message m choose uniform r (the salt) of size n , output $c = \langle r, F_k(r) \oplus m \rangle$.

$\text{Dec}_k(c)$: on input k and ciphertext $c = \langle r, s \rangle$, output $m = F_k(r) \oplus s$.

Theorem 3.31

If F is a pseudorandom function, then Construction 3.30 is a CPA-secure private-key encryption for messages of length n .