Cryptography Notes

Rasmus Kirk Jakobsen Mikkel Skafsgaard Berg

January 17, 2024

Note that these notes are based on the 2023v3 version of the cryptography book.

- Curriculum
- Basic Facts from Probability Theory
- Information theory and Cryptography (Chapter 5)
  - Perfect Security
  - Entropy
  - Conditional Entropy
  - Entropy of Random Variables in Cryptography
  - Unicity Distance
- Symmetric (secret-key) cryptography (Chapter 4.1 + 6)
  - Disposition (Kirk)
  - Disposition (Berg)
  - Notes
    - Symmetric Cryptosystems
    - PRF Security
    - CPA Security
      - Proof: CBC
  - Appendix
    - DES (Data Encryption Standard)
    - The computations of the oracles during CPA proof
- Public-key cryptography from Factoring (Chapter 7 & 8)
  - RSA:
- Public-key cryptography based on discrete log and LWE (Chapter 9 & 10, definition of CPS security in chapter 8)
- Symmetric authentication and hash functions (Chapter 11)
  - Disposition (Kirk)
  - Disposition (Berg)
  - Notes
- Signature schemes (Chapter 12)
  - Disposition (Kirk)
  - Disposition (Berg)
  - Notes
  - Appendix

Curriculum

Background material:

Chapters 2+3 (preliminary probability theory and math), Section 5.4 (optimistic results on key exchange), Section 6.3 (diff and lin analysis), Proof of Theorem 7.8 (if you can get the secret exponent you can factor), Section 7.6.1 (factoring algorithms), Section 9.6 (discrete log algorithms).

Example Templates:

Here are some details on what you might for instance cover in each the exam subject. But do not misunderstand this in the sense that you have to follow the templates below, they really are just examples...

Information theory and Cryptography:

Definition of perfect secret security, why you need as many keys at plaintexts to have perfect security. Definition of entropy, and proof of some of the inequalities properties it satisfies. Unicity distance (but be careful, this may take a lot of time, so test this beforehand)

Symmetric (secret-key) cryptography:

What a crypto-system is (the three algorithms) You can describe DES or AES - but you can also just give a high-level description of what a block cipher is. Definitions of PRF and CPA security. Specification of CBC or CTR modes (or both), proofs of CPA security for CBC or CTR mode (or both). Perhaps a brief talk about stream ciphers and how to make one from a block cipher.

Public-key cryptography from Factoring:

What a public-key cryptosystem is. Basic spec of RSA, maybe proof that decryption works. Then some selection of the following: How to make RSA be CPA secure (the PCRSA scheme, and the result that computing the least significant bit is as hard as inverting RSA). How to generate keys and Miller-Rabin primality testing, how to get CCA security: OAEP and the intuition on why it works.

Public-key cryptography based on discrete log and LWE:

The DL, DH and DDH problems, and how they relate. The El Gamal cryptosystem and proof that it is secure if DDH is hard. Then some example of groups we can use, can be a subgroup of $\mathbb{Z}_p^*$, or you can talk about elliptic curves. You can also put less emphasis on El Gamal, for instance skip the example groups and go to LWE instead, define the problem and the cryptosystem and do the proof from the exercise that decryption works under a certain assumption about the noise distribution.

Symmetric authentication and hash functions:

Definition of collision-intractable hash functions. Then a selection of: construction from discrete log, proof that collision-intractable implies one-way, construction and proof that we can get any size input from fixed size input. Finally, MAC schemes, definition of CMA security, CBCMAC and EMAC security result for EMAC. Maybe a brief mention of HMAC.

Signature schemes:

Definition of signatures schemes and of CMA security. The Schnorr signature scheme, you can do many details here, such as the proof that you cannot cheat the underlying interactive game with better than 1/q probability, and the full story on how you derive the signature scheme from the interactive game. Or you can just do the spec of the scheme, giving you time for something else, such as RSA+hash signatures and the proof that secure hash + secure signature scheme is secure. Or you can do the one-time signatures based on hash functions and the proof that they are secure.

Basic Facts from Probability Theory

Theorem 2.4 (Jensen's inequality): Let $p_1, ..., p_n$ be a probability distribution, that is, $\sum_i p_i = 1$ and $0 \leq p_i \leq 1$. Then for any concave f and any $x_1, .., x_n$, we have:

$$\sum_{i=1}^{n} p_i f(x_i) \leq f(\sum_{i=1}^{n} p_i x_i)$$

Furthermore, if f is strictly concave, equality holds **iff** all the x_i's are equal.
TLDR: $f(\mathbb{E}[X]) \leq \mathbb{E}[f(X)]$

Note: *Concave* means that $\frac{f(a)+f(b)}{2} \leq \frac{a+b}{2}$.
TLDR: $f'' = 0$.

Note: *Strictly concave* means that $\frac{f(a)+f(b)}{2} < \frac{a+b}{2}$ for all $a \neq b$.
TLDR: This basically means that graph of the function always curves and is never linear.

Note: For this course, we only apply this to log() which is strictly concave. △

Information theory and Cryptography (Chapter 5)

Disposition (Kirk):

- Perfect Security
- Entropy
- Unicity Distance
  - H_L
  - Redundancy
  - Spurious Keys
  - Unicity Distance

Disposition (Berg):

Perfect Security

Below we have the definition for perfect security:

Definition 5.1: A cryptosystem has perfect security if for all $x \in \mathcal{P}$ and $y \in \mathcal{C}$, it holds that $P[x|y] = P[x]$.
TLDR: Information about the ciphertext gives you *no* information about the plaintext. △

Theorem - $|\mathcal{K}| \geq |\mathcal{C}| \geq |\mathcal{P}|$: If you have perfect security then $|\mathcal{K}| \geq |\mathcal{C}| \geq |\mathcal{P}|$.
TLDR: If you have perfect security your key can not be shorter than your ciphertext, which cannot be shorter than your plaintext. △

Proof:

- $|\mathcal{C}| \geq |\mathcal{P}|$: This is true for all crypto systems in order for decryption to function correctly.
- $|\mathcal{K}| \geq |\mathcal{C}|$: For a fixed plaintext x must be able to hit every ciphertext y, otherwise an adversary could conclude that $E(x) \neq y$ and therefore learn information from y.

Therefore, given perfect security, you have $|\mathcal{K}| \geq |\mathcal{C}| \geq |\mathcal{P}|$. □

Entropy

> *As a further illustration of the intuition behind entropy, consider the following thought experiment: suppose you get access to an oracle that will magically tell you if you will live to be more than 110 years old. One would naturally expect that the probability p of getting "yes" as the answer is very small, while the probability $1 - p$ of "no" is close to 1.*

Definition 5.6: Let X be a random variable that takes values $x_1, ..., x_n$ with probabilities $p_1, ..., p_n$. Then the entropy of X, written $H(X)$, is defined to be:

$$H(X) = \sum_{i=1}^{n} p_i \log_2(1/p_i)$$

TLDR: If an event A occurs with probability p and you are told that A occurred, then you have learned $\log_2(1/p)$ bits of information. **TLDR:** The entropy $H(X)$ can be described as:

- How many bits we need to send on average to communicate the value of X.
- The amount of uncertainty you have about X before you are told what the value is. △

Theorem 5.7: For a random variable X taking n possible values, it holds that $0 \leq H(X) \leq \log_2(n)$. Furthermore, $H(X) = 0$ **iff** one value X has probability 1 (and the others 0). $H(X) = log_2(n)$ **iff** it is uniformly distributed, i.e., all probabilities are $1/n$.
TLDR: If the entropy of X is 0 there is no uncertainty, meaning that we know the value of X. If the

entropy of X is 1 then the uncertainty of X is highest meaning that all possible values of X have the same probability. $\triangle$

Proof: We need to prove the following:

- $H(X) > 0$
  - $H(X)$ is defined as a product of positive sums, therefore $H(X)$ is also positive.
- $H(X) = 0$ **iff** a single $p_i = 1$ and all other $p_j = 0$
  - The function $f(p) = p\log(1/p)$ is only 0 if $p = 0 \vee p = 1$. This coupled with the fact that probabilities must sum up to one means that $H(X) = 0$ **iff** a single $p_i = 1$ and all other $p_j = 0$.
- $H(X) < \log_2(n)$
  - log is a concave function ($\log'' = 0$) therefore we can use Theorem 2.4 (Jensen's inequality):

$$H(X) = \sum_{i=1}^{n} p_i \log_2(1/p_i) \leq \log_2(\sum_{i=1}^{n} p_i \cdot 1/p_i) = \log_2(n)$$

    Thus $H(X) < \log_2(n)$.
- $H(X) = \log_2(n)$ **iff** X is uniformly distributed.
  - Since log_2 is strictly concave then from Theorem 2.4 we know that $H(X) = \log_2(n)$ **iff** all p_i are equal i.e. X is uniformly distributed. $\square$

Conditional Entropy

Definition 5.9: Given the above definition of $H(X \mid Y = y_j)$, we define the conditional entropy of X given Y to be:

$$H(X \mid Y) = \sum_j P[Y = y_j]H(X \mid Y = y_j)$$

Entropy of Random Variables in Cryptography

Theorem 5.11: For any cryptosystem with deterministic encryption function, it holds that:

$$H(K \mid C) = H(K) + H(P) - H(C)$$

TLDR: Answers how much uncertainty remains about the key given the ciphertext

Unicity Distance

Definition - Redundancy: Given a language L and a plaintext space $\mathcal{P}$, the *redundancy* of the language is the amount of superflous information is contained, on avarage in the language L.

$$R_L = \frac{\log(|\mathcal{P}|) - H_L}{\log(|\mathcal{P}|)} = 1 - \frac{H_L}{\log(|\mathcal{P}|)}$$

H_L is a measure of the number of bits of information each letter contains in the language L, on average. For English, we have that H_L is (very approximately) 1.25 bits per letter.

$$H_L = \lim_{n \mapsto \infty} H(P_n)/n$$

TLDR: A language contains redundancy, which is how much duplicate information there is on avarage in the language.
Example: The following sentance displays redundancy in english:

"cn y rd th fllwng sntnc, vn f t s wrttn wtht vcls?"

△

Definition - Spurious Keys: If an adversary has a ciphertext y that he wants to decrypt, he can try all keys and see if y decrypts to meaningful english. If y decrypts to meaningful english under the *wrong* key, then that key is said to be a *spurious key.*
TLDR: A spurious key is a key that *seems* to be the correct key for a ciphertext but is not. △

Definition - Number of Spurious Keys: The average number of spurious keys, taken over all choices of ciphertexts of length n:

$$sp_n = \sum_{\boldsymbol{y} \in \mathcal{C}^n} P[y](|K(\boldsymbol{y})| - 1) = \sum_{\boldsymbol{y} \in \mathcal{C}^n} P[y]|K(\boldsymbol{y})| - 1$$

Given a ciphertext $\boldsymbol{y}$, we use $K(\boldsymbol{y})$ to denote the set of keys that are possible given this ciphertext. More precisely, a key K is in this set if decryption of $\boldsymbol{y}$ under K yields a plaintext that could occur with non-zero probability:

$$K(\boldsymbol{y}) = \{K \in \mathcal{K} \mid P[D_K(\boldsymbol{y} > 0)]\}$$

TLDR: This formula for sp_n describes the average number of spurious keys of a ciphertext $\boldsymbol{y}$ of length n.

△

Definition 5.12: The unicity distance n_0 of a cryptosystem is the minimal length of plaintexts such that $sp_{n_0} = 0$, if such a value exists, and ∞ otherwise.
TLDR: The unicity distance tells you how many times you can encrypt something where multiple keys seem to be valid keys. △

Theorem 5.13: Assume we have a cryptosystem with deterministic encryption function, where the plaintext and ciphertext alphabets have the same size ($|\mathcal{C}| = |\mathcal{P}|$), and where keys are uniformly chosen from $\mathcal{K}$. Assume we use the system to encrypt sequences of letters from language L. Then

$$n_0 \geq \frac{\log(|\mathcal{K}|)}{R_L \log(|\mathcal{P}|)}$$

TLDR: If we reuse keys, our unconditional security will always be gone, once we encrypt enough plaintext under the same key. The only exception is the case where $R_L = 0$ which leads to n_0 being ∞. Which makes sense, if every sequence of characters is a plaintext that can occur, the adversary can never exclude a key. △

Proof: We start by unfolding the definition of $H(K \mid C_n)$ using Definition 5.9:

$$H(K \mid C_n) = \sum_{\boldsymbol{y} \in \boldsymbol{\mathcal{C}_n}} P[C_n = \boldsymbol{y}] H(K \mid C_n = \boldsymbol{y})$$

First, note that given some ciphertext $\boldsymbol{y}$, the key K will have some conditional distribution, but of course only values in $K(\boldsymbol{y})$ can occur. Therefore $H(K|C_n = \boldsymbol{y}) \leq \log_2(|K(\boldsymbol{y})|)$:

$$\begin{aligned}
H(K \mid C_n) &\leq \sum_{\boldsymbol{y} \in \boldsymbol{\mathcal{C}_n}} P[C_n = \boldsymbol{y}] \log_2(|K(\boldsymbol{y})|) \\
&\leq \log_2 \left(\sum_{\boldsymbol{y} \in \boldsymbol{\mathcal{C}_n}} P[C_n = \boldsymbol{y}]|K(\boldsymbol{y})| \right) && \text{(Definition 2.4 - Jensen's Inequality)} \\
&\leq \log_2(sp_n + 1) && \text{(Definition - Number of Spurious Keys)}
\end{aligned}$$

Now we want to simplify $H(K \mid C_n)$. We start by applying Theorem 5.11:

$$H(K \mid C_n) = H(K) + H(P_n) - H(C_n)$$

Observe that $H(C_n) \geq log(|C|^n) = n \log(|\mathcal{P}|)$. Moreover, recalling the definition on H_L, let us assume that we take n large enough so that $H(P_n) \approx nH_L$.

$$\begin{aligned} H(P_n) &\approx nH_L \\ &\approx n(\log(|\mathcal{P}|)(1 - R_L)) \quad \text{(Definition - Number of Spurious Keys)} \end{aligned}$$

Now we try to find $H(K \mid C_n)$

$$\begin{aligned} H(K \mid C_n) &= H(K) + H(P_n) - H(C_n) \\ &\geq H(K) + H(P_n) - n \log(|\mathcal{P}|) && \text{(From our observation of } H(C_n)) \\ &\approx H(K) + n \log(|\mathcal{P}|)(1 - R_L) - n \log(|\mathcal{P}|) && \text{(From our estimate of } H(P_n)) \\ &= H(K) + n \log(|\mathcal{P}|) - n \log(|\mathcal{P}|)R_L - n \log(|\mathcal{P}|) \\ &= H(K) - n \log(|\mathcal{P}|)R_L \\ &= \log(|\mathcal{K}|) - n \log(|\mathcal{P}|)R_L && \text{(Theorem 5.7, } K \text{ is uniform)} \\ H(K \mid C_n) &\geq \log(|\mathcal{K}|) - n \log(|\mathcal{P}|)R_L \end{aligned}$$

Combining our equations, setting $sp_n = 0$ and solving for n:

$$\begin{aligned} \log(|\mathcal{K}|) - n \log(|\mathcal{P}|)R_L &\leq \log_2(sp_n + 1) \\ \log(|\mathcal{K}|) - n \log(|\mathcal{P}|)R_L &\leq \log_2(0 + 1) \\ n \log(|\mathcal{P}|)R_L &\leq \log(|\mathcal{K}|) \\ n &\leq \frac{\log(|\mathcal{K}|)}{\log(|\mathcal{P}|)R_L} \end{aligned}$$

So $n_0 \leq \frac{\log(|\mathcal{K}|)}{\log(|P|)R_L}$. □

Symmetric (secret-key) cryptography (Chapter 4.1 + 6)

Disposition (Kirk)

- Symmetric Cryptosystems
- CBC
- PRF
- CPA
- CPA security proof for CBC/CTR

Symmetric Cryptosystems

For a symmetric cryptosystem, we need 3 finite sets which will define all possible values of the system:

- The key space $\mathcal{K}$
- The plaintext space $\mathcal{P}$
- The ciphertext space $\mathcal{C}$

To generate values from these sets, we must have 3 corresponding algorithms, each responsible for outputting values of a set:

- $G \to K \in \mathcal{K}$ (Generates keys): Probabilistic. Usually uniform in $\mathcal{K}$.
- $E : E_K(x) = y \in \mathcal{C}$ (Encrypts plaintexts): (Probabilistic). Ciphertext's probability distribution is determined by K and x, typically uniform in some subset of the ciphertexts.
- $D : D_K(y) = x \in \mathcal{P}$ (Decrypts ciphertexts): (Probabilistic)

This triple of algorithms (G, E, D) constitutes the given cryptosystem.

We always require the following basic relationship between (G, E, D): For any $x \in \mathcal{P}$, $x = D_K(E_K(x))$ This says nothing about security however.

PRF Security

In terms of security, we would like the encryption schemes of our system to be *pseudo*-randomly secure, meaning that it must act like a pseudo-random function (PRF). In general, a PRF is a function that is inherently *deterministic* but behaves like a *random* one. In order to model this, we say that some adversary plays the following game: *draw PRF Game*

We want the $Adv_A(O_{real}, O_{ideal}) \leq \epsilon$. If this is the case, we say that the probability of him succeeding in an attack on our system is *negligible*, meaning we deem it infeasible in practice.

More formally, we want our PRF's to be secure as given by the following definition:
Definition - PRF Security: $\left\{ f_K \mid K \in \{0,1\}^k \right\}$ is (t, q, ϵ) PRF-secure if $Adv_A\left(O_{\text{Real}}, O_{\text{Ideal}}\right) \leq \epsilon$
(We say that $\left\{ f_K \mid K \in \{0,1\}^k \right\}$ is (t, q, ϵ) PRF-secure, if any adversary A that runs in time at most t and makes at most q queries to the oracle, satisfies $Adv_A\left(O_{\text{Real}}, O_{\text{Ideal}}\right) \leq \epsilon$.) Where $\left\{ f_K \mid K \in \{0,1\}^k \right\}$ denotes a family of functions mapping $\mathcal{P} \to \mathcal{C}$. For a symmetric and deterministic encryption scheme f_K is replaced by E_K. (Just like the set of all DES functions. Each $f_K : \{0,1\}^n \to \{0,1\}^m$. The advantage is defined as $Adv_A\left(O_{\text{Real}}, O_{\text{Ideal}}\right) = |p(A,0) - p(A,1)|$.)

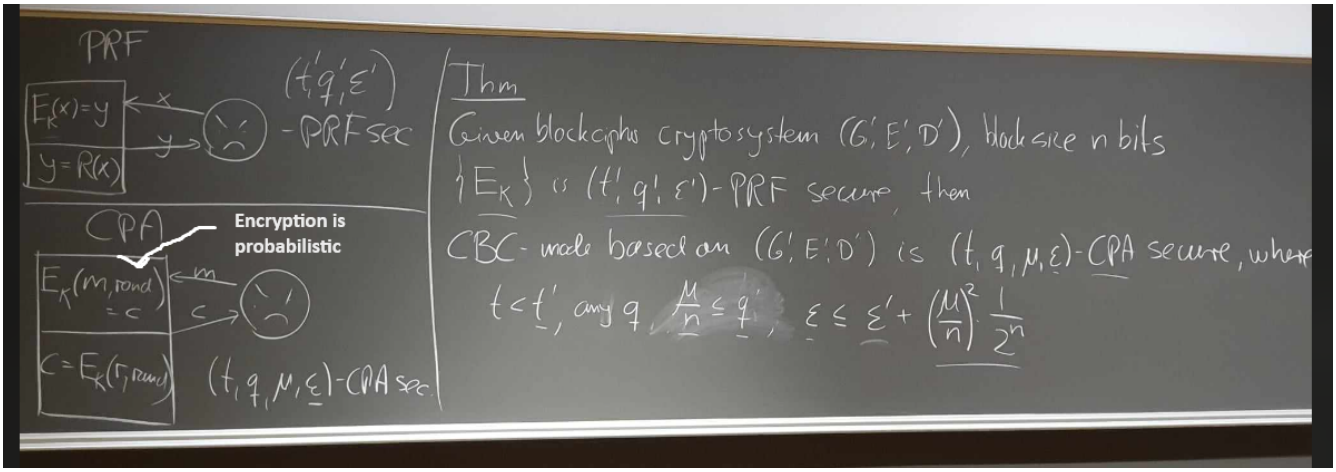
Examples of PRF-secure cryptosystems which we've seen in the course are DES and AES, which are two types of block ciphers. They have the following properties: G outputs a *fixed* length key, chosen uniformly at random, takes as input a bitstring of *fixed* length and outputs a ciphertext of the *same* length.

CPA Security

Unfortunately, despite a cryptosystem being PRF-secure, it still suffers from information leakage because an adversary can easily detect duplicate messages, *if we're using the same key*, since each input maps to the same output, once the key is fixed. Thus, we would like introduce some notion of randomness in E. For such probabilistic encryption schemes we require that the adversary *cannot* tell the difference between between a real encryption of a message x he chooses. Thus, duplicate messages can no longer be detected.

In other words, we want our encryption schemes to be secure against a chosen-plaintext-attack (CPA): **Definition - Chosen-Plaintext Attack(CPA)-security:** (G, E, D) is (t, q, μ, ϵ) CPA-secure if $\mathrm{Adv}_A\left(O_{\text{Real}}, O_{\text{Ideal}}\right) \leq \epsilon$ (We say the cryptosystem (G, E, D) is (t, q, μ, ϵ) CPA-secure, if for any adversary A that runs in time at most t, and makes at most q queries to the oracle, with plaintexts consisting of a total of μ bits, it holds that $\mathrm{Adv}_A\left(O_{\text{Real}}, O_{\text{Ideal}}\right) \leq \epsilon$.)

Here the difference in security-parameters is μ, which denotes the number of bits an adversary encrypts. We need to take this into account, because systems that are CPA secure, may also handle variable length input, which we will see shortly. Under this stronger notion of security, we need to update our model, where the adversary instead plays this variation of the PRF-game: *draw CPA game*



A way to inject this randomness and achieve CPA security for e.g. DES is by using a so-called *mode of operation*, such as Cipher Block Chaining (CBC). The way that CBC mode works is by constructing a new cryptosystem (G, E, D) from the PRF-secure system (G', E', D'), where $G = G'$. As an added bonus, this system can handle variable length input. For simplicity $\mathcal{P}$ for (G', E', D') will be all strings divisble by n, the blocksize of (G, E, D). *draw CBC mode*

In turns out that in using CBC, we achieve a greater level of security, namely one that is secure against the previously mentioned, chosen-plaintext-attack (CPA). This property is encapsulated in the following theorem:

Theorem: If (G, E, D) is (t, q, ϵ) PRF-secure then (G', E', D') using CBC is $(t', q'\mu, \epsilon')$ CPA-secure for any q', and for

$$\epsilon' = \epsilon + \left(\frac{\mu}{n}\right)^2 \cdot \frac{1}{2^n} = \epsilon + \frac{\mu^2}{n^2 \cdot 2^n}$$

provided that

$$t' \leq t, \quad \frac{\mu}{n} \leq q$$

Intuitively, what this results says is: as long as CBC encryption using (G, E, D) is attacked by an adversary who is no more powerful[1] than what (G, E, D) can handle, the probability of the attack being successful will be no better than $\epsilon' = \epsilon + \left(\frac{\mu}{n}\right)^2 \cdot \frac{1}{2^n}$. We'll refer to *blocks* as bit strings of length n, which is the block-size of the original system. We note that $(\frac{\mu}{n})^2$ is the number of blocks that are encrypted during CBC, squared. ϵ' and ϵ will then be roughly equal as long as the number of blocks is much less than 2^n, since the fraction would then go toward 0. A heuristic given in the book is $(\frac{\mu}{n})^2 << 2^{n/2} = \sqrt{2^n}$.

Proof: CBC

Let's now prove the theorem:

We start by introducing the *hybrid* oracle to the game *draw hybrid* (Does normal CBC, except E_K is replaced by R, where R only takes and outputs bit strings of length n)

Right off the bat, since the *only* difference between the hybrid and real game is that E_K is replaced with R, we must have:

$$Adv_A(O_{real}, O_{hybrid}) = |p(A, real) - p(A, hybrid)| \leq \epsilon$$

If this was not the case, A could be used to distinguish between E_K and a random function with advantage greater than ϵ, contradicting our assumption that (G, E, D) was PRF-secure.

Now note that if we are in the ideal case, the oracle does *not* use CBC, but simply outputs $N + 1$ blocks, where N is the number of blocks in the input. It should now be difficult for A to distinguish between the ideal and hybrid case, since the hybrid case outputs a concatenation of random blocks, also yielding $N + 1$ random blocks, *UNLESS* a certain bad event happens. We define BAD as; if at any point during the hybrid game, the function R receives an input that it has received before in this game. In this case we will have an input collision, which will yield a repeated block. This could hint A that he is in the hybrid case. Therefore, his advantage in distinguishing hybrid from ideal must be bounded by:

$$|p(A, hybrid) - p(A, ideal)| \leq Pr(BAD)$$

If we add our two inequalities, we get:

$$\begin{aligned} |p(A, real) - p(A, ideal)| &\leq |p(A, real) - p(A, hybrid)| + |p(A, hybrid) - p(A, ideal)| \\ Adv_A(O_{real}, O_{ideal}) &\leq \epsilon + Pr(BAD) \end{aligned}$$

So now, we just have to estimate $Pr(BAD)$ by bounding it. Let M_j be the event that a collision occurs after j calls to R. Clearly $P(M1) = 0$. Using the Law of Total Probability, we have that:

$$\begin{aligned} P[M_j] &= P[M_j|M_{j-1}]P[M_{j-1}] + P[M_j|\neg M_{j-1}]P[\neg M_{j-1}] \quad \text{(Law of Total Probability)} \\ &\leq P[M_{j-1}] + P[M_j|\neg M_{j-1}] \\ &= P[M_{j-1}] + \frac{(j-1)}{2^n} \end{aligned}$$

The last probability on the right hand side is equal to $\frac{(j-1)}{2^n}$: First, since M_{j-1} did not occur we have seen $j - 1$ different inputs before. Second, the new input nr. j is the XOR of some message block and an independently chosen random block (either a y0-value chosen by the oracle or an output from R), it is therefore uniformly chosen. We conclude that in fact

$$P[M_j] \leq (1 + 2 + \ldots + (j-1)) \leq \frac{j^2}{2^n}$$

[1](t,q) are a measure of the adversaries computational power. ϵ is a measure of the probability of success for an attack.

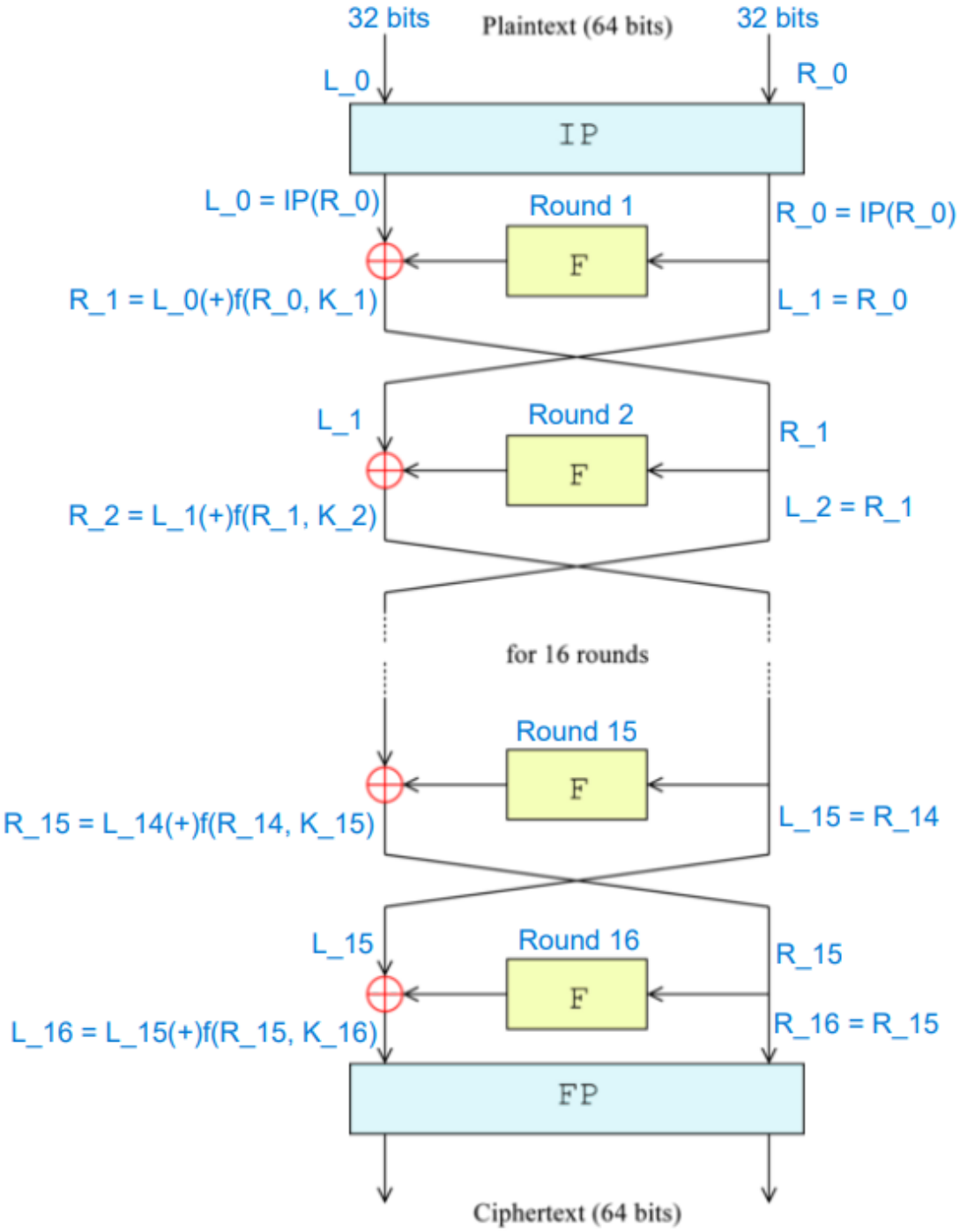
Now we've provided a bound for all j's (calls to R). Since the total number of calls is at most μ/n, we can replace j with μ/n. Thus it follows that $P(BAD) \leq \frac{\mu^2}{n^2 \cdot 2^n}$ and we are done.

Appendix

DES (Data Encryption Standard)

DES is a block cipher: G outputs a *fixed* length key, chosen uniformly at random, takes as input a bitstring of *fixed* length and outputs a ciphertext of the *same* length. Furthermore, it is deterministic, such that under a fixed key, any unique input maps to a unique output.

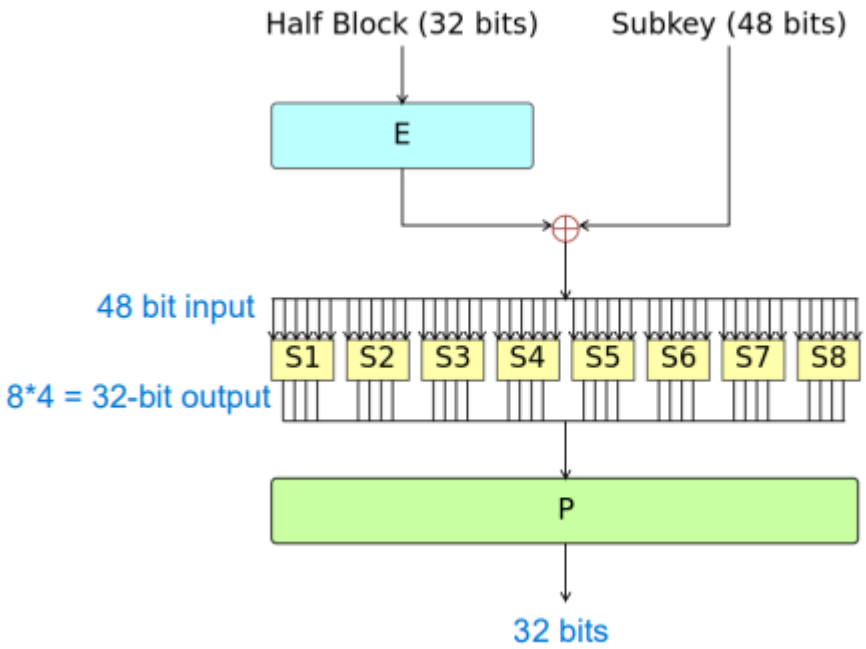
Specifically, DES uses $|K| = 56$ bits and $|x|, |y| = 64$ bits. It uses a 16-round *Feistel* structure for its encryption, which looks like: *draw*



The characteristics of a Feistel cipher is that it computes some function each round, involving a *round* key and some degree of both permutation and substitution as advised by Shannon, when we want to achieve security. A so-called *Key Schedule* is responsible for generating each round key, which in the DES case generates 16, 48-bit keys from the 56-bit key.

We simply refer to this function as the f-function:

$$f(R, K) = P(S(K \oplus E(R)))$$



It takes a 32-bit block as input, *e*xpands this to 48 bits and XOR's the expansion with the round key of the corresponding round. It then *s*ubstitutes the 48-bits, using 8 *substitution boxes* and concatenates their output, yielding 32 bits. It is important to note that each substitution box is a *non-linear* function, which ensures that the input block of DES cannot be retrieved using linear algebra. Finally, the 32-bits are *p*ermuted.

One of the main issues with DES today, is that the key is too short. It *is* feasible to search through all $|\mathcal{K}| = 2^{56}$ possible keys. This is why AES was created, using key-sizes 128, 192 and 256.

The computations of the oracles during CPA proof

REAL

$$\begin{aligned} CBC(m) \Rightarrow \text{choose random} \quad & y_0 \\ \Rightarrow E(y_0 \oplus x_1) &= y_1 \\ \Rightarrow E(y_1 \oplus x_2) &= y_2 \\ & \dots \\ \Rightarrow y_0, y_1, \dots, y_t & \end{aligned}$$

IDEAL

$$R(m) = c = y_0, y_1, \dots, y_t$$

HYBRID

$$\begin{aligned} CBC(m) \Rightarrow \text{choose random} \quad & y_0 \\ \Rightarrow R(y_0 \oplus x_1) &= y_1 \\ \Rightarrow R(y_1 \oplus x_2) &= y_2 \\ & \dots \\ \Rightarrow y_0, y_1, \dots, y_t & \end{aligned}$$

Public-key cryptography from Factoring (Chapter 7 & 8)

Disposition (Kirk):

RSA:

Definition The RSA algorithm:

1. On input (even) security parameter value k, choose random $k/2$-bit primes p, q, and set $n = pq$.
2. Select a number $e \in Z^*_{(p-1)(q-1)}$ and set $d = e - 1 \bmod (p-1)(q-1)$.
3. Output public key $pk = (n, e)$ and secret key $sk = (n, d)$. For RSA, we always have $\mathcal{P} = \mathcal{C} = Z_n$.

Encryption and decryption works as follows:

$$E_{(n,e)}(x) = x^e \bmod n$$

$$D_{(n,d)}(y) = y^d \bmod n$$

△

Public-key cryptography based on discrete log and LWE (Chapter 9 & 10, definition of CPS security in chapter 8)

Disposition (Kirk):

Symmetric authentication and hash functions (Chapter 11)

Signature schemes (Chapter 12)

Appendix

CPA

$$\begin{aligned}\epsilon &= \epsilon' + \left(\frac{\mu}{n}\right)^2 \cdot \frac{1}{2^n} \\ &= \epsilon' + \frac{\mu^2}{n \cdot 2^n}\end{aligned}$$

Solving for $1 = |\epsilon - \epsilon'|$:

$$\begin{aligned}1 &= |\epsilon - \epsilon'| \\ 1 &= \frac{\mu^2}{n \cdot 2^n} \\ n \cdot 2^n &= \mu^2 \\ \sqrt{n \cdot 2^n} &= \mu \\ \sqrt{n} \cdot 2^{n/2} &= \mu\end{aligned}$$

So if we encrypt much less than $2^{n/2}$ we are safe. We disard $\sqrt{n}$ since it is insignificant compared to $2^{n/2}$.

□

CPA

$$\begin{aligned}P[M_j] &= P[M_j|M_{j-1}]P[M_{j-1}] + P[M_j|\neg M_{j-1}]P[\neg M_{j-1}] \quad \text{(Law of Total Probability)} \\ &= \frac{P[M_j, M_{j-1}]}{P[M_{j-1}]}P[M_{j-1}] + P[M_j|\neg M_{j-1}]P[\neg M_{j-1}] \quad \text{(Bayes rule)} \\ &= P[M_j, M_{j-1}] + P[M_j|\neg M_{j-1}]P[\neg M_{j-1}] \\ &= P[M_j]P[M_{j-1}] + P[M_j|\neg M_{j-1}]P[\neg M_{j-1}] \\ &\leq P[M_{j-1}] + P[M_j|\neg M_{j-1}]\end{aligned}$$