ĐẠI HỌC QUỐC GIA, THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH

MẬT MÃ HÓA VÀ AN TOÀN THÔNG TIN (CO3083)

BÁO CÁO BÀI TẬP LỚN

Chủ đề

GVHD: Trịnh Cao Thắng
SV thực hiện: LỮ HOÀNG DUY 2310474
HÀ MINH PHÚ 2312646
LÝ VĨNH THÁI 2213104

TP Hồ Chí Minh, Tháng 12 Năm 2025

Mục lục

Phân công công việc

Sinh viên	Nhiệm vụ	Trạng thái	Tiến độ
LỮ HOÀNG DUY	Tấn công Many-Time Pad	Hoàn thành	100%
	Khai thác điểm yếu app	Hoàn thành	100%
	Viết báo cáo	Hoàn thành	100%
HÀ MINH PHÚ	Chuẩn bị lý thuyết	Hoàn thành	100%
	Phân tích app	Hoàn thành	100%
	Viết và tổng hợp báo cáo	Hoàn thành	100%
LÝ VĨNH THÁI	Chuẩn bị lý thuyết	Hoàn thành	100%
	Viết báo cáo	Hoàn thành	100%

Giới thiệu

Trong mật mã học hiện đại, **tính ngẫu nhiên** đóng vai trò then chốt. Nhiều giao thức mật mã yêu cầu các chuỗi bit ngẫu nhiên để tạo khóa, tạo nonce, hoặc thực hiện các phép mã hóa. Tuy nhiên, việc tạo ra các chuỗi ngẫu nhiên thực sự (true randomness) là rất khó khăn và tốn kém về mặt tài nguyên.
Bộ sinh số giả ngẫu nhiên (Pseudo-Random Generator - PRG) giải quyết vấn đề này bằng cách "kéo dài" một chuỗi ngẫu nhiên ngắn (gọi là *seed*) thành một chuỗi dài hơn nhiều, sao cho chuỗi output này không thể phân biệt được với một chuỗi ngẫu nhiên thực sự bởi bất kỳ thuật toán hiệu quả nào.
Trong bài này, chúng ta sẽ:

- Định nghĩa chính xác các khái niệm về PRG và tính bảo mật của PRG
- Định nghĩa tính không thể dự đoán (unpredictability) của PRG
- Chứng minh rằng tính bảo mật kéo theo tính không thể dự đoán

Phần I

Nguyên cứu và phát triển

Chương 1

Lý thuyết và chứng minh

1.1 Phần (a): Các định nghĩa cơ bản

1.1.1 Bộ sinh số giả ngẫu nhiên (Pseudo-Random Generator - PRG)

1.1.1.1 Định nghĩa hình thức

Định nghĩa 1 (Pseudo-Random Generator - PRG)**.** Cho ℓ là một **tham số bảo mật** (security parameter). Một **bộ sinh số giả ngẫu nhiên (PRG)** là một hàm tất định (deterministic) có thể tính được trong thời gian đa thức:

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

thỏa mãn điều kiện $n = n(\ell) > s = s(\ell)$, trong đó:

- $\{0,1\}^s$ là **không gian seed** (seed space) - tập hợp tất cả các chuỗi bit có độ dài s
- $\{0,1\}^n$ là **không gian output** (output space) - tập hợp tất cả các chuỗi bit có độ dài n
- $s = s(\ell)$ là **độ dài seed**, phụ thuộc vào tham số bảo mật
- $n = n(\ell)$ là **độ dài output**, phụ thuộc vào tham số bảo mật

1.1.1.2 Giải thích chi tiết

Tại sao cần $n > s$?

Điều kiện $n > s$ là bản chất của PRG. Nếu $n \leq s$, thì hàm G không "sinh" thêm bất kỳ bit nào mới, và do đó không có ý nghĩa thực tiễn. Tỷ lệ:

$$\text{Tỷ lệ mở rộng (Expansion Ratio)} = \frac{n}{s} > 1$$

cho biết PRG "kéo dài" seed bao nhiêu lần. Ví dụ, nếu $s = 128$ và $n = 1024$, thì tỷ lệ mở rộng là 8, nghĩa là từ 128 bit seed, ta thu được 1024 bit output.

Tính tất định (Deterministic):

PRG là một hàm tất định, nghĩa là với cùng một seed k, PRG luôn cho ra cùng một output $G(k)$. Điều này rất quan trọng vì:

- Cho phép tái tạo lại chuỗi giả ngẫu nhiên khi cần thiết
- Đảm bảo tính nhất quán trong các giao thức mật mã

Tính hiệu quả (Efficient):

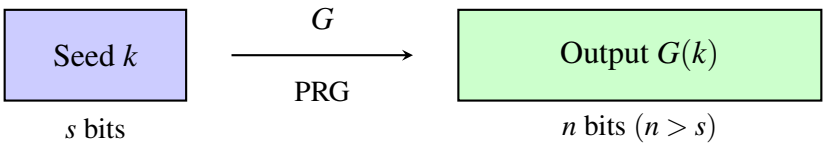
PRG phải có thể tính được trong thời gian đa thức theo tham số bảo mật ℓ. Điều này đảm bảo PRG có thể được sử dụng trong thực tế.

Ví dụ 1 (Minh họa PRG)**.** Giả sử ta có PRG $G : \{0,1\}^{128} \to \{0,1\}^{256}$:

- Input (seed): $k =$ `0x3a7b9c...` (128 bit)
- Output: $G(k) =$ `0x8f2e1d...` (256 bit)

Mặc dù chỉ có 2^{128} seed khả dĩ, output trông như thể được chọn ngẫu nhiên từ 2^{256} chuỗi có thể.

1.1.1.3 Biểu diễn trực quan



1.1.2 Trò chơi bảo mật PRG (PRG Security Game)

1.1.2.1 Ý tưởng trực quan

Trước khi đưa ra định nghĩa hình thức, hãy hiểu ý tưởng cốt lõi:

Ý tưởng: Một PRG được coi là "tốt" nếu output của nó *trông giống như* ngẫu nhiên thực sự. Cụ thể hơn, không có thuật toán hiệu quả nào có thể phân biệt được:

- Output của PRG: $G(k)$ với k được chọn ngẫu nhiên
- Chuỗi ngẫu nhiên thực sự: r được chọn ngẫu nhiên từ $\{0,1\}^n$

Để hình thức hóa khái niệm "không thể phân biệt", chúng ta sử dụng mô hình **trò chơi** (game) giữa hai bên:

- **Challenger** ($\mathscr{C}$): Người thách thức, đại diện cho hệ thống
- **Adversary** ($\mathscr{A}$): Kẻ tấn công, cố gắng phá vỡ tính bảo mật

1.1.2.2 Định nghĩa hình thức

Định nghĩa 2 (PRG Security Game)**.** Cho PRG $G : \{0,1\}^s \to \{0,1\}^n$ và một adversary $\mathscr{A}$. **Trò chơi bảo mật PRG**, ký hiệu $\text{Game}^{PRG}_{G,\mathscr{A}}$, được thực hiện như sau:

Bước 1: Challenger chọn bit thách thức:

Challenger $\mathscr{C}$ chọn ngẫu nhiên một bit $b \xleftarrow{\$} \{0,1\}$ với xác suất đều:

$$\Pr[b = 0] = \Pr[b = 1] = \frac{1}{2}$$

Bước 2: Challenger tạo thách thức:

- **Nếu** $b = 0$ (trường hợp PRG):
  (a) $\mathscr{C}$ chọn seed ngẫu nhiên: $k \xleftarrow{\$} \{0,1\}^s$
  (b) $\mathscr{C}$ tính output của PRG: $y \leftarrow G(k)$
- **Nếu** $b = 1$ (trường hợp ngẫu nhiên thực sự):
  (a) $\mathscr{C}$ chọn chuỗi ngẫu nhiên: $y \xleftarrow{\$} \{0,1\}^n$

Bước 3: Challenger gửi thách thức:

$\mathscr{C}$ gửi y cho adversary $\mathscr{A}$

Bước 4: Adversary phân tích và đoán:

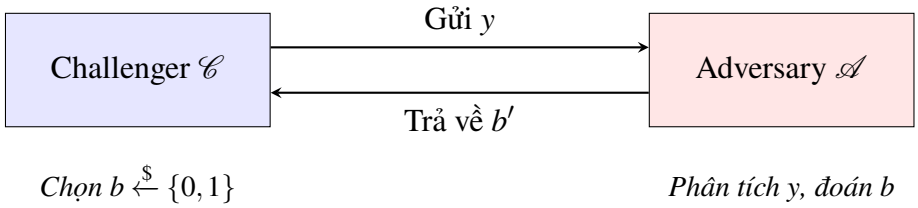
$\mathscr{A}$ phân tích y và xuất ra một bit dự đoán $b' \in \{0,1\}$:

- $b' = 0$: $\mathscr{A}$ đoán rằng y là output của PRG
- $b' = 1$: $\mathscr{A}$ đoán rằng y là ngẫu nhiên thực sự

Bước 5: Xác định kết quả:

- $\mathscr{A}$ **thắng** nếu $b' = b$
- $\mathscr{A}$ **thua** nếu $b' \neq b$

1.1.2.3 Sơ đồ trò chơi



Nếu $b = 0$*:* $y \leftarrow G(k)$
Nếu $b = 1$*:* $y \xleftarrow{\$} \{0,1\}^n$

$\mathscr{A}$ thắng $\Leftrightarrow b' = b$

1.1.2.4 Định nghĩa lợi thế (Advantage)

Định nghĩa 3 (PRG Advantage)**. Lợi thế** của adversary $\mathscr{A}$ trong trò chơi bảo mật PRG được định nghĩa là:

$$\mathbf{Adv}_G^{PRG}(\mathscr{A}) = \left|\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]\right|$$

Giải thích chi tiết về công thức lợi thế:
Hãy phân tích từng thành phần:

- $\Pr[b' = 1 \mid b = 0]$: Xác suất $\mathscr{A}$ xuất ra 1 khi nhận được output của PRG
  - Đây là xác suất $\mathscr{A}$ *nhầm lẫn*, nghĩ rằng output PRG là ngẫu nhiên thực sự
  - Hay còn gọi là **False Positive Rate**
- $\Pr[b' = 1 \mid b = 1]$: Xác suất $\mathscr{A}$ xuất ra 1 khi nhận được chuỗi ngẫu nhiên thực sự
  - Đây là xác suất $\mathscr{A}$ *đoán đúng* khi input là ngẫu nhiên thực sự
  - Hay còn gọi là **True Positive Rate**

Công thức tương đương:
Ta có thể viết lại lợi thế dưới dạng:

$$\mathbf{Adv}_G^{PRG}(\mathscr{A}) = |\Pr[\mathscr{A}(G(k)) = 1] - \Pr[\mathscr{A}(r) = 1]|$$

trong đó:

- $k \xleftarrow{\$} \{0,1\}^s$ là seed ngẫu nhiên
- $r \xleftarrow{\$} \{0,1\}^n$ là chuỗi ngẫu nhiên thực sự

Ý nghĩa của lợi thế:

- **Adv** $= 0$: $\mathscr{A}$ không thể phân biệt được PRG output với ngẫu nhiên thực sự (hoàn toàn an toàn)
- **Adv** $= 1$: $\mathscr{A}$ luôn phân biệt được chính xác (hoàn toàn không an toàn)
- **Adv** $= \varepsilon$ nhỏ: $\mathscr{A}$ chỉ có lợi thế rất nhỏ so với đoán ngẫu nhiên

Nhận xét 1 (Về việc đoán ngẫu nhiên)**.** Một adversary "ngớ ngẩn" có thể chỉ đơn giản đoán ngẫu nhiên $b' \xleftarrow{\$} \{0,1\}$ mà không cần phân tích y. Trong trường hợp này:

$$\Pr[b' = 1 \mid b = 0] = \Pr[b' = 1 \mid b = 1] = \frac{1}{2}$$

Do đó, lợi thế là $|1/2 - 1/2| = 0$. Điều này cho thấy định nghĩa lợi thế đã loại bỏ "may mắn" từ việc đoán ngẫu nhiên.

1.1.3 PRG an toàn (Secure PRG)

1.1.3.1 Hàm không đáng kể (Negligible Function)

Trước khi định nghĩa PRG an toàn, ta cần hiểu khái niệm "không đáng kể";

Định nghĩa 4 (Negligible Function)**.** Một hàm $\mathsf{negl} : \mathbb{N} \to \mathbb{R}_{\geq 0}$ được gọi là **không đáng kể** (negligible) nếu với *mọi* đa thức dương $p(\cdot)$, tồn tại một số nguyên N sao cho với mọi $\ell > N$:

$$\mathsf{negl}(\ell) < \frac{1}{p(\ell)}$$

Giải thích trực quan:
Hàm không đáng kể giảm nhanh hơn bất kỳ hàm $1/p(\ell)$ nào với p là đa thức. Ví dụ:

- $\mathsf{negl}(\ell) = 2^{-\ell}$ là không đáng kể (vì $2^{-\ell} < 1/\ell^c$ với mọi c khi ℓ đủ lớn)
- $\mathsf{negl}(\ell) = 2^{-\sqrt{\ell}}$ là không đáng kể
- $\mathsf{negl}(\ell) = 1/\ell^{100}$ **không** phải là không đáng kể (vì nó là đa thức)

1.1.3.2 Định nghĩa PRG an toàn

Định nghĩa 5 (Secure PRG)**.** Một PRG $G : \{0,1\}^s \to \{0,1\}^n$ được gọi là **an toàn** (secure) nếu với *mọi* adversary $\mathscr{A}$ chạy trong thời gian đa thức (PPT - Probabilistic Polynomial Time), lợi thế của $\mathscr{A}$ là không đáng kể:

$$\mathbf{Adv}_G^{PRG}(\mathscr{A}) \leq \mathsf{negl}(\ell)$$

trong đó ℓ là tham số bảo mật.

Giải thích chi tiết:

1. **"Mọi adversary PPT":** Định nghĩa này đòi hỏi tính bảo mật chống lại *tất cả* các thuật toán chạy trong thời gian đa thức, không chỉ một số thuật toán cụ thể.
2. **"Thời gian đa thức":** Adversary bị giới hạn tài nguyên tính toán. Cụ thể, thời gian chạy của $\mathscr{A}$ là $O(\ell^c)$ cho một hằng số c nào đó.
3. **"Lợi thế không đáng kể":** Ngay cả adversary mạnh nhất (trong giới hạn thời gian đa thức) cũng không thể phân biệt PRG output với ngẫu nhiên thực sự với xác suất đáng kể.

Ý nghĩa thực tiễn: Nếu PRG G an toàn với tham số bảo mật $\ell = 128$, thì bất kỳ kẻ tấn công nào cũng cần thực hiện khoảng 2^{128} phép tính để có thể phân biệt output của G với ngẫu nhiên thực sự. Với công nghệ hiện tại, điều này là không khả thi.

1.1.3.3 Phát biểu tương đương

Định nghĩa PRG an toàn có thể được phát biểu tương đương như sau:

Định lý 1 (Phát biểu tương đương của PRG an toàn)**.** PRG G an toàn khi và chỉ khi hai phân phối sau đây là **không thể phân biệt về mặt tính toán** (computationally indistinguishable):

$$\{G(k) : k \xleftarrow{\$} \{0,1\}^s\} \approx_c \{r : r \xleftarrow{\$} \{0,1\}^n\}$$

Ký hiệu $\approx_c$ nghĩa là không có thuật toán hiệu quả nào có thể phân biệt được hai phân phối này.

1.1.4 Trò chơi PRG không thể dự đoán (Unpredictable PRG Game)

1.1.4.1 Ý tưởng trực quan

Một cách khác để đánh giá chất lượng của PRG là xem xét khả năng **dự đoán** output:

Ý tưởng: Một PRG "tốt" nên có tính chất: cho trước một phần output, không ai có thể dự đoán được phần tiếp theo với xác suất tốt hơn đáng kể so với việc đoán ngẫu nhiên (50%).

Ví dụ thực tế: Nếu bạn biết 100 số đầu tiên của một dãy giả ngẫu nhiên, bạn có thể đoán được số thứ 101 không? Nếu PRG tốt, câu trả lời là "không thể tốt hơn tung đồng xu".

1.1.4.2 Định nghĩa hình thức

Định nghĩa 6 (Unpredictable PRG Game)**.** Cho PRG $G : \{0,1\}^s \to \{0,1\}^n$ và một adversary $\mathscr{A}$. Với mỗi vị trí $i \in \{1,2,\ldots,n\}$, **Trò chơi PRG Không thể Dự đoán tại vị trí** i, ký hiệu $\text{Game}^{pred}_{G,\mathscr{A},i}$, được thực hiện như sau:

Bước 1: Challenger tạo output PRG:

(a) Challenger $\mathscr{C}$ chọn seed ngẫu nhiên: $k \xleftarrow{\$} \{0,1\}^s$

(b) $\mathscr{C}$ tính output đầy đủ: $y = G(k) = y_1 y_2 \cdots y_n$

(c) Ở đây, y_j là bit thứ j của output ($j = 1,2,\ldots,n$)

Bước 2: Challenger gửi tiền tố:
$\mathscr{C}$ gửi **tiền tố** (prefix) cho adversary:

$$\text{prefix} = (y_1, y_2, \ldots, y_{i-1})$$

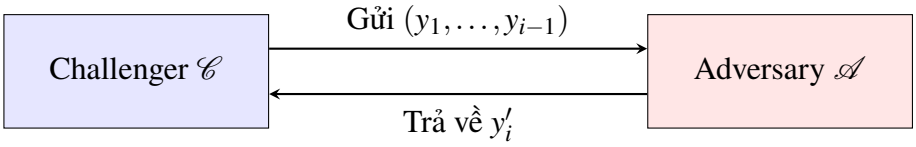
Lưu ý: Adversary nhận được $i-1$ bit đầu tiên, nhưng *không* biết bit thứ i.

Bước 3: Adversary dự đoán:
$\mathscr{A}$ phân tích tiền tố và xuất ra một bit dự đoán $y'_i \in \{0,1\}$

Bước 4: Xác định kết quả:

- $\mathscr{A}$ **thắng** nếu $y'_i = y_i$ (dự đoán đúng)
- $\mathscr{A}$ **thua** nếu $y'_i \neq y_i$ (dự đoán sai)

1.1.4.3 Sơ đồ trò chơi



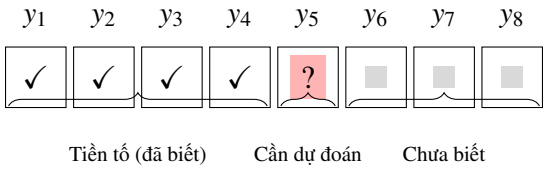
1. *Chọn* $k \xleftarrow{\$} \{0,1\}^s$
2. *Tính* $y = G(k) = y_1 \cdots y_n$

Dự đoán bit thứ i

$\mathscr{A}$ thắng $\Leftrightarrow y'_i = y_i$

1.1.4.4 Minh họa với ví dụ cụ thể

Ví dụ 2. Giả sử PRG G có output 8 bit và ta xét trò chơi tại vị trí $i = 5$:



Adversary nhận được (y_1, y_2, y_3, y_4) và cần dự đoán y_5.

1.1.4.5 Định nghĩa lợi thế trong trò chơi dự đoán

Định nghĩa 7 (Prediction Advantage)**. Lợi thế dự đoán** của adversary $\mathscr{A}$ tại vị trí i được định nghĩa là:

$$\mathbf{Adv}^{pred}_{G,i}(\mathscr{A}) = \left| \Pr[y'_i = y_i] - \frac{1}{2} \right|$$

Lợi thế dự đoán tổng thể là:

$$\mathbf{Adv}^{pred}_{G}(\mathscr{A}) = \max_{i \in \{1,\ldots,n\}} \mathbf{Adv}^{pred}_{G,i}(\mathscr{A})$$

Giải thích:

- $\Pr[y'_i = y_i]$ là xác suất adversary dự đoán đúng bit thứ i
- Một adversary "ngớ ngẩn" đoán ngẫu nhiên có $\Pr[y'_i = y_i] = 1/2$
- Lợi thế đo lường mức độ "tốt hơn" so với đoán ngẫu nhiên
- $\mathbf{Adv} = 0$: Không tốt hơn đoán ngẫu nhiên
- $\mathbf{Adv} = 1/2$: Luôn dự đoán đúng (tối đa có thể)

1.1.5 PRG không thể dự đoán (Unpredictable PRG)

Định nghĩa 8 (Unpredictable PRG)**.** Một PRG $G : \{0,1\}^s \to \{0,1\}^n$ được gọi là **không thể dự đoán** (unpredictable) nếu với *mọi* adversary PPT $\mathscr{A}$ và với *mọi* vị trí $i \in \{1,2,\ldots,n\}$, lợi thế dự đoán là không đáng kể:

$$\mathbf{Adv}^{pred}_{G,i}(\mathscr{A}) \leq \mathsf{negl}(\ell)$$

Phát biểu tương đương:
PRG G không thể dự đoán khi và chỉ khi với mọi adversary PPT $\mathscr{A}$:

$$\Pr[y'_i = y_i] \leq \frac{1}{2} + \mathsf{negl}(\ell)$$

Nói cách khác, xác suất dự đoán đúng chỉ nhỉnh hơn 50% một lượng không đáng kể.

1.1.5.1 So sánh hai khái niệm bảo mật

PRG An toàn (Secure)	**PRG Không thể dự đoán (Unpredictable)**
Dựa trên khả năng *phân biệt*	Dựa trên khả năng *dự đoán*
Adversary nhận *toàn bộ* output	Adversary nhận *một phần* output
Adversary cần phân biệt PRG vs random	Adversary cần dự đoán bit tiếp theo
Định nghĩa "mạnh hơn"	Định nghĩa "yếu hơn"

1.2 Phần (b): Chứng minh Secure PRG $\Rightarrow$ Unpredictable PRG

1.2.1 Phát biểu định lý

Định lý 2 (Secure PRG implies Unpredictable PRG)**.** Nếu $G : \{0,1\}^s \to \{0,1\}^n$ là một PRG an toàn, thì G là không thể dự đoán.

Một cách tương đương (phản đề): Nếu G có thể dự đoán được, thì G không an toàn.

1.2.2 Chiến lược chứng minh

Chúng ta sử dụng kỹ thuật **phép quy dẫn** (reduction), một kỹ thuật chuẩn trong mật mã học:

Ý tưởng quy dẫn: Giả sử tồn tại adversary $\mathscr{A}$ có thể "dự đoán" PRG với lợi thế đáng kể. Ta sẽ xây dựng adversary $\mathscr{B}$ sử dụng $\mathscr{A}$ như một "subroutine" để "phân biệt" PRG với ngẫu nhiên thực sự. Nếu $\mathscr{B}$ thành công, điều này mâu thuẫn với giả thiết PRG an toàn.

Sơ đồ logic:

$$\exists \mathscr{A} \text{ dự đoán được } \xrightarrow{\text{xây dựng } \mathscr{B}} \exists \mathscr{B} \text{ phân biệt được } \xrightarrow{\text{mâu thuẫn}} G \text{ không an toàn}$$

Phản đề:

$$G \text{ an toàn} \Rightarrow G \text{ không thể dự đoán}$$

1.2.3 Giả thiết

Giả sử tồn tại:

- Adversary PPT $\mathscr{A}$
- Vị trí $i \in \{1, 2, \ldots, n\}$
- Lợi thế dự đoán không đáng kể $\varepsilon = \varepsilon(\ell)$

sao cho:

$$\Pr[\mathscr{A}(y_1, \ldots, y_{i-1}) = y_i] = \frac{1}{2} + \varepsilon$$

trong đó $k \xleftarrow{\$} \{0,1\}^s$ và $y = G(k) = y_1 \cdots y_n$.
Mục tiêu: Xây dựng adversary $\mathscr{B}$ cho PRG Security Game với:

$$\mathbf{Adv}_G^{PRG}(\mathscr{B}) \geq \varepsilon$$

1.2.4 Xây dựng Adversary $\mathscr{B}$

1.2.4.1 Mô tả chi tiết thuật toán

Algorithm 1 Adversary $\mathscr{B}$ cho PRG Security Game

Require: Input $y = y_1 y_2 \cdots y_n$ từ Challenger (có thể là $G(k)$ hoặc ngẫu nhiên)
Ensure: Output bit $b' \in \{0,1\}$

// Bước 1: Trích xuất tiền tố
Trích xuất $i-1$ bit đầu tiên: prefix $\leftarrow (y_1, y_2, \ldots, y_{i-1})$

// Bước 2: Sử dụng $\mathscr{A}$ để dự đoán
Gọi predictor: $y_i' \leftarrow \mathscr{A}(\text{prefix})$

// Bước 3: Đưa ra quyết định dựa trên kết quả dự đoán
if $y_i' = y_i$ **then** ▷ $\mathscr{A}$ dự đoán đúng
    return $b' = 0$ ▷ Đoán: y là output của PRG
else ▷ $\mathscr{A}$ dự đoán sai
    return $b' = 1$ ▷ Đoán: y là ngẫu nhiên thực sự
end if

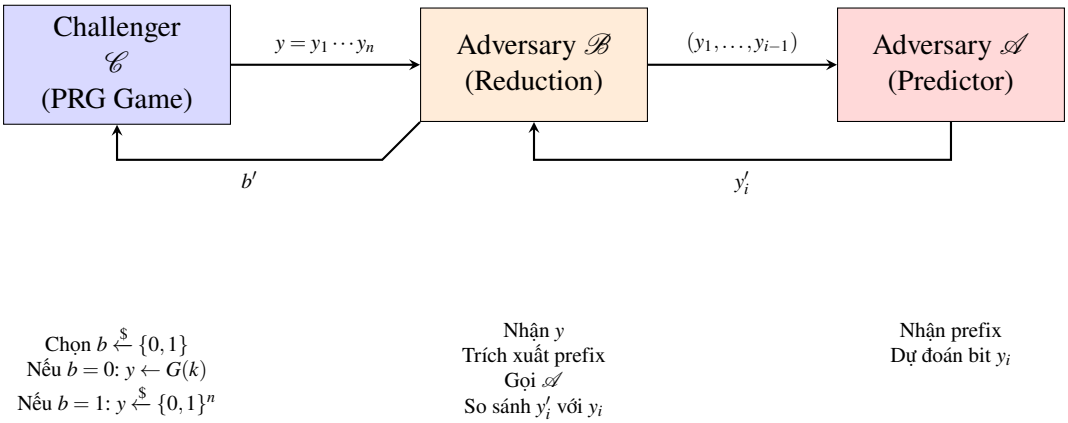
1.2.4.2 Giải thích logic của $\mathscr{B}$

Tại sao $\mathscr{B}$ hoạt động? Ý tưởng cốt lõi:

1. **Khi $y = G(k)$ (output của PRG):**
    - Các bit của y có mối tương quan (correlation) với nhau vì chúng đều được sinh từ cùng một seed k
    - Adversary $\mathscr{A}$ (theo giả thiết) có thể khai thác mối tương quan này để dự đoán y_i từ $(y_1, \ldots, y_{i-1})$
    - Do đó, $\mathscr{A}$ dự đoán đúng với xác suất $\frac{1}{2} + \varepsilon$
2. **Khi y là ngẫu nhiên thực sự:**
    - Các bit $y_1, y_2, \ldots, y_n$ là độc lập với nhau
    - Bit y_i hoàn toàn độc lập với tiền tố $(y_1, \ldots, y_{i-1})$
    - Dù $\mathscr{A}$ mạnh đến đâu, $\mathscr{A}$ cũng không thể dự đoán y_i tốt hơn đoán ngẫu nhiên
    - Do đó, $\mathscr{A}$ dự đoán đúng với xác suất chính xác $\frac{1}{2}$

Kết luận: Sự khác biệt về xác suất dự đoán đúng giữa hai trường hợp cho phép $\mathscr{B}$ phân biệt PRG output với ngẫu nhiên thực sự.

1.2.4.3 Sơ đồ quy dẫn chi tiết



1.2.5 Phân tích lợi thế

1.2.5.1 Trường hợp 1: $b = 0$ (Challenger gửi $y = G(k)$)

Khi bit thách thức $b = 0$, challenger gửi $y = G(k)$ với $k \xleftarrow{\$} \{0,1\}^s$.

Theo giả thiết về $\mathscr{A}$:

$$\Pr[\mathscr{A}(y_1, \ldots, y_{i-1}) = y_i \mid y = G(k)] = \frac{1}{2} + \varepsilon$$

Do đó, xác suất $\mathscr{B}$ xuất ra 0 (đoán đúng):

$$\Pr[\mathscr{B}(y) = 0 \mid b = 0] = \Pr[y'_i = y_i \mid y = G(k)] \tag{1.1}$$

$$= \frac{1}{2} + \varepsilon \tag{1.2}$$

1.2.5.2]

Trường hợp 2: $b = 1$ (Challenger gửi $y \xleftarrow{\$} \{0,1\}^n$)

Khi bit thách thức $b = 1$, challenger gửi $y \xleftarrow{\$} \{0,1\}^n$, một chuỗi ngẫu nhiên thực sự.

Quan sát quan trọng: Khi y là ngẫu nhiên thực sự, mỗi bit y_j được chọn độc lập và đều:

$$\Pr[y_j = 0] = \Pr[y_j = 1] = \frac{1}{2} \quad \forall j$$

Đặc biệt, bit y_i **độc lập hoàn toàn** với tiền tố $(y_1, \ldots, y_{i-1})$:

$$\Pr[y_i = b \mid y_1, \ldots, y_{i-1}] = \Pr[y_i = b] = \frac{1}{2} \quad \forall b \in \{0,1\}$$

Do đó, dù $\mathscr{A}$ sử dụng bất kỳ chiến lược nào dựa trên tiền tố, xác suất dự đoán đúng vẫn là:

$$\Pr[\mathscr{A}(y_1,\ldots,y_{i-1}) = y_i \mid y \xleftarrow{\$} \{0,1\}^n] = \frac{1}{2}$$

Do đó, xác suất $\mathscr{B}$ xuất ra 0:

$$\Pr[\mathscr{B}(y) = 0 \mid b = 1] = \Pr[y'_i = y_i \mid y \xleftarrow{\$} \{0,1\}^n] \tag{1.3}$$

$$= \frac{1}{2} \tag{1.4}$$

1.2.5.3 Tính lợi thế của $\mathscr{B}$

Áp dụng định nghĩa lợi thế PRG:

$$\mathbf{Adv}_G^{PRG}(\mathscr{B}) = |\Pr[\mathscr{B}(y) = 0 \mid b = 0] - \Pr[\mathscr{B}(y) = 0 \mid b = 1]| \tag{1.5}$$

$$= \left|\left(\frac{1}{2} + \varepsilon\right) - \frac{1}{2}\right| \tag{1.6}$$

$$= |\varepsilon| \tag{1.7}$$

$$= \varepsilon \tag{1.8}$$

Kết quả quan trọng:

$$\boxed{\mathbf{Adv}_G^{PRG}(\mathscr{B}) = \mathbf{Adv}_{G,i}^{pred}(\mathscr{A}) = \varepsilon}$$

Lợi thế của $\mathscr{B}$ trong PRG Security Game **bằng chính xác** lợi thế của $\mathscr{A}$ trong Unpredictable PRG Game.

1.2.6 Phân tích thời gian chạy

Để hoàn thiện chứng minh, ta cần kiểm tra $\mathscr{B}$ chạy trong thời gian đa thức:

- **Bước 1 (Trích xuất tiền tố):** $O(n)$ - thời gian tuyến tính
- **Bước 2 (Gọi $\mathscr{A}$):** $T_{\mathscr{A}}(\ell)$ - thời gian đa thức (vì $\mathscr{A}$ là PPT)
- **Bước 3 (So sánh và quyết định):** $O(1)$ - thời gian hằng số

Tổng thời gian chạy của $\mathscr{B}$:

$$T_{\mathscr{B}}(\ell) = O(n) + T_{\mathscr{A}}(\ell) + O(1) = O(T_{\mathscr{A}}(\ell) + n)$$

Vì $n = n(\ell)$ là đa thức theo ℓ và $T_{\mathscr{A}}(\ell)$ là đa thức theo ℓ, ta có $T_{\mathscr{B}}(\ell)$ cũng là đa thức theo ℓ. Do đó, $\mathscr{B}$ là adversary PPT.

1.2.7 Kết luận chứng minh

1.2.7.1 Tóm tắt các bước

1. **Giả sử** tồn tại adversary PPT $\mathscr{A}$ có thể dự đoán bit thứ i của PRG output với lợi thế ε không đáng kể.
2. **Xây dựng** adversary PPT $\mathscr{B}$ sử dụng $\mathscr{A}$ như subroutine:
   - $\mathscr{B}$ nhận input y từ PRG Security Game
   - $\mathscr{B}$ trích xuất tiền tố và gọi $\mathscr{A}$
   - $\mathscr{B}$ so sánh dự đoán của $\mathscr{A}$ với bit thực tế để quyết định
3. **Chứng minh** $\mathbf{Adv}_G^{PRG}(\mathscr{B}) = \varepsilon$:
   - Khi $y = G(k)$: $\mathscr{A}$ dự đoán đúng với xác suất $\frac{1}{2} + \varepsilon$
   - Khi y ngẫu nhiên: $\mathscr{A}$ dự đoán đúng với xác suất $\frac{1}{2}$
   - Sự khác biệt chính xác là ε
4. **Mâu thuẫn:** Nếu ε không đáng kể (tức là $\mathscr{A}$ tồn tại), thì $\mathscr{B}$ có lợi thế ε không đáng kể, mâu thuẫn với giả thiết G an toàn.

1.2.7.2 Phát biểu kết luận

Định lý 3 (Kết luận)**.** Cho $G : \{0,1\}^s \to \{0,1\}^n$ là một PRG. Ta có:

$$G \text{ an toàn} \Rightarrow G \text{ không thể dự đoán}$$

Tương đương (phản đề):

$$G \text{ có thể dự đoán} \Rightarrow G \text{ không an toàn}$$

Chứng minh. Đã chứng minh ở trên bằng phép quy dẫn: với mọi adversary $\mathscr{A}$ cho Unpredictable PRG Game, ta xây dựng được adversary $\mathscr{B}$ cho PRG Security Game sao cho:

$$\mathbf{Adv}_G^{PRG}(\mathscr{B}) = \mathbf{Adv}_{G,i}^{pred}(\mathscr{A})$$

Do đó, nếu G an toàn (mọi adversary có lợi thế PRG không đáng kể), thì G không thể dự đoán (mọi adversary có lợi thế prediction không đáng kể). ■ □

1.2.8 Nhận xét bổ sung

Nhận xét 2 (Chiều ngược lại)**.** Câu hỏi tự nhiên: Liệu chiều ngược lại có đúng không?

$$G \text{ không thể dự đoán} \overset{?}{\Rightarrow} G \text{ an toàn}$$

Câu trả lời là **CÓ**, nhưng chứng minh phức tạp hơn và sử dụng kỹ thuật hybrid argument. Điều này có nghĩa là hai khái niệm "an toàn" và "không thể dự đoán" là **tương đương** đối với PRG.

Nhận xét 3 (Ý nghĩa thực tiễn)**.** Định lý này cho phép chúng ta:

- Chứng minh tính bảo mật của PRG bằng cách chứng minh tính không thể dự đoán (đôi khi dễ hơn)
- Hiểu rằng bất kỳ điểm yếu nào trong khả năng dự đoán đều dẫn đến điểm yếu trong tính bảo mật tổng thể

1.3 Phần (c): Chứng minh Unpredictable PRG $\Rightarrow$ Secure PRG

1.3.1 Phát biểu định lý

Định lý 4 (Unpredictable PRG implies Secure PRG)**.** Nếu G là một PRG không thể dự đoán (unpredictable), thì G là một PRG an toàn (secure).

Phản đề (tương đương): Nếu G không an toàn (tồn tại adversary phân biệt được G), thì G có thể dự đoán được (tồn tại adversary dự đoán được bit tiếp theo).

1.3.2 Chiến lược chứng minh: Lập luận lai ghép (Hybrid Argument)

Để chứng minh chiều này, chúng ta sử dụng kỹ thuật **Hybrid Argument**. Ý tưởng là biến đổi dần dần từ phân phối ngẫu nhiên thực sự sang phân phối PRG thông qua một chuỗi các phân phối trung gian.

Giả sử tồn tại một adversary $\mathscr{A}$ (Distinguisher) chiến thắng trò chơi PRG Security Game với lợi thế ε không đáng kể. Ta sẽ xây dựng một adversary $\mathscr{B}$ (Predictor) chiến thắng trò chơi Unpredictable PRG Game với lợi thế liên quan đến ε/n.

1.3.2.1 Định nghĩa các phân phối lai ghép (Hybrids)

Cho $G : \{0,1\}^s \to \{0,1\}^n$. Ta định nghĩa một chuỗi $n+1$ phân phối xác suất $\{H_k\}_{k=0}^n$ trên $\{0,1\}^n$ như sau:

- H_k: Là phân phối bao gồm k bit đầu tiên lấy từ PRG $G(s)$, và $n-k$ bit còn lại được chọn ngẫu nhiên thực sự.

Cụ thể, chuỗi mẫu x thuộc phân phối H_k có dạng:

$$x = (\underbrace{g_1, g_2, \ldots, g_k}_{\text{từ PRG}}, \underbrace{r_{k+1}, \ldots, r_n}_{\text{ngẫu nhiên}})$$

trong đó $(g_1 \ldots g_n) = G(s)$ với $s \xleftarrow{\$} \{0,1\}^s$, và $r_j \xleftarrow{\$} \{0,1\}$.

Hai trường hợp cực đoan:

- H_n: Toàn bộ bit đều từ PRG $\Rightarrow H_n \equiv \{G(s)\}$.
- H_0: Toàn bộ bit đều ngẫu nhiên $\Rightarrow H_0 \equiv \{U_n\}$ (Uniform).

1.3.3 Phân tích lợi thế của Adversary $\mathscr{A}$

Theo giả thiết, $\mathscr{A}$ phân biệt được $G(s)$ và U_n với lợi thế ε:

$$\mathbf{Adv}_G^{PRG}(\mathscr{A}) = |\Pr[\mathscr{A}(H_n) = 1] - \Pr[\mathscr{A}(H_0) = 1]| = \varepsilon$$

Sử dụng bất đẳng thức tam giác cho tổng chuỗi (Telescoping sum):

$$\begin{aligned}\varepsilon &= \left| \sum_{k=1}^{n} (\Pr[\mathscr{A}(H_k) = 1] - \Pr[\mathscr{A}(H_{k-1}) = 1]) \right| \\ &\leq \sum_{k=1}^{n} |\Pr[\mathscr{A}(H_k) = 1] - \Pr[\mathscr{A}(H_{k-1}) = 1]|\end{aligned}$$

Theo nguyên lý Dirichlet (Pigeonhole Principle), vì tổng của n số hạng lớn hơn hoặc bằng ε, nên phải tồn tại ít nhất một chỉ số $i \in \{1, \ldots, n\}$ sao cho:

$$|\Pr[\mathscr{A}(H_i) = 1] - \Pr[\mathscr{A}(H_{i-1}) = 1]| \geq \frac{\varepsilon}{n}$$

Điều này có nghĩa là $\mathscr{A}$ có khả năng phân biệt giữa hai phân phối lân cận H_i và H_{i-1} với lợi thế ít nhất là ε/n.

1.3.4 Xây dựng Adversary $\mathscr{B}$

Hãy xem xét sự khác biệt giữa H_i và H_{i-1}:

- $H_i = (g_1, \ldots, g_{i-1}, \mathbf{g_i}, r_{i+1}, \ldots, r_n)$

- $H_{i-1} = (g_1, \ldots, g_{i-1}, \mathbf{r_i}, r_{i+1}, \ldots, r_n)$

Chúng chỉ khác nhau ở vị trí thứ i: trong H_i là bit PRG thật g_i, còn trong H_{i-1} là bit ngẫu nhiên r_i.

Ta xây dựng $\mathscr{B}$ để dự đoán bit thứ i của PRG bằng cách sử dụng khả năng phân biệt của $\mathscr{A}$.

Algorithm 2 Adversary $\mathscr{B}$ cho Unpredictable PRG Game (tại vị trí i)

Require: Tiền tố $y_{prefix} = (y_1, \ldots, y_{i-1})$ từ Challenger
Ensure: Dự đoán bit y'_i

// Bước 1: Tạo phần đuôi ngẫu nhiên
Chọn chuỗi ngẫu nhiên $\rho = (r_{i+1}, \ldots, r_n) \xleftarrow{\$} \{0,1\}^{n-i}$

// Bước 2: Đoán thử bit thứ i
Chọn ngẫu nhiên một bit $b \xleftarrow{\$} \{0,1\}$ (đây là ứng cử viên cho y_i)

// Bước 3: Ghép chuỗi và gọi $\mathscr{A}$
Tạo chuỗi input: $X = (y_1, \ldots, y_{i-1}, b, r_{i+1}, \ldots, r_n)$
Gọi Distinguisher: $out \leftarrow \mathscr{A}(X)$

// Bước 4: Quyết định dựa trên phản hồi của $\mathscr{A}$
if $out = 1$ **then**
    return b ▷ $\mathscr{A}$ nghĩ X là PRG $\Rightarrow$ b có vẻ đúng
else
    return $1 - b$ ▷ $\mathscr{A}$ nghĩ X là Random $\Rightarrow$ b có vẻ sai
end if

1.3.5 Chứng minh tính đúng đắn

Ta cần tính xác suất $\mathscr{B}$ dự đoán đúng y_i (ký hiệu là sự kiện Win).

$$\Pr[\text{Win}] = \Pr[\mathscr{B}(\text{prefix}) = y_i]$$

Xét input X mà $\mathscr{B}$ đưa cho $\mathscr{A}$:

- Nếu $b = y_i$ (xác suất $1/2$): X có dạng $(y_1 \ldots y_{i-1}, \mathbf{y_i}, \text{random})$. Đây chính là phân phối H_i.
- Nếu $b \neq y_i$ (xác suất $1/2$): Vì b chọn ngẫu nhiên và khác y_i, nên tại vị trí i là một giá trị ngẫu nhiên. X có dạng phân phối H_{i-1}.

Xác suất $\mathscr{B}$ trả về đúng y_i xảy ra trong 2 trường hợp:

1. $\mathscr{B}$ chọn đúng $b = y_i$ VÀ $\mathscr{A}$ trả về 1 (nhận diện là H_i).
2. $\mathscr{B}$ chọn sai $b \neq y_i$ VÀ $\mathscr{A}$ trả về 0 (nhận diện là H_{i-1}, $\mathscr{B}$ đảo bit thành đúng).

$$\Pr[\text{Win}] = \frac{1}{2} \cdot \Pr[\mathscr{A}(H_i) = 1] + \frac{1}{2} \cdot \Pr[\mathscr{A}(H_{i-1}) = 0]$$

Thay $\Pr[\mathscr{A}(H_{i-1}) = 0] = 1 - \Pr[\mathscr{A}(H_{i-1}) = 1]$, ta có:

$$\begin{aligned}\Pr[\text{Win}] &= \frac{1}{2}\Pr[\mathscr{A}(H_i) = 1] + \frac{1}{2}(1 - \Pr[\mathscr{A}(H_{i-1}) = 1]) \\ &= \frac{1}{2} + \frac{1}{2}\underbrace{(\Pr[\mathscr{A}(H_i) = 1] - \Pr[\mathscr{A}(H_{i-1}) = 1])}_{\text{Lợi thế phân biệt tại bước } i}\end{aligned}$$

Lợi thế dự đoán của $\mathscr{B}$:

$$\mathbf{Adv}^{pred}_{G,i}(\mathscr{B}) = \left|\Pr[\text{Win}] - \frac{1}{2}\right| = \frac{1}{2}\left|\Pr[\mathscr{A}(H_i) = 1] - \Pr[\mathscr{A}(H_{i-1}) = 1]\right|$$

Như đã chứng minh ở phần 4.3, tồn tại chỉ số i sao cho hiệu số này $\geq \varepsilon/n$. Do đó:

$$\mathbf{Adv}^{pred}_{G,i}(\mathscr{B}) \geq \frac{\varepsilon}{2n}$$

1.3.6 Kết luận

Nếu G không an toàn, tồn tại $\mathscr{A}$ có lợi thế ε là không đáng kể (non-negligible). Vì n là đa thức (polynomial), thì $\frac{\varepsilon}{2n}$ cũng là không đáng kể.

Điều này có nghĩa là tồn tại một vị trí i và một adversary $\mathscr{B}$ có thể dự đoán bit thứ i với lợi thế đáng kể. Điều này mâu thuẫn với giả thiết G là Unpredictable.

Vậy: **Unpredictable PRG** $\Rightarrow$ **Secure PRG**. ■

Chương 2

Tấn công Many-Time Pad

2.1 Lỗ hổng Many-Time Pad (Tái sử dụng khóa)

Khi một khóa K được dùng để mã hóa cho nhiều bản rõ khác nhau $(P_1, P_2, \ldots)$ trong One-Time Pad. Đây gọi là lỗ hổng Many-Time Pad.

Giả sử ta có hai bản mã C_1 và C_2 được mã hóa bởi cùng một khóa K:

$$C_1 = P_1 \oplus K$$
$$C_2 = P_2 \oplus K$$

Nếu kẻ tấn công thực hiện phép XOR giữa hai bản mã này, khóa K sẽ bị triệt tiêu:

$$C_1 \oplus C_2 = (P_1 \oplus K) \oplus (P_2 \oplus K) = P_1 \oplus P_2 \tag{2.1}$$

Kết quả thu được là XOR của hai bản rõ. Mặc dù kẻ tấn công chưa biết trực tiếp nội dung của P_1 hay P_2, nhưng cấu trúc ngôn ngữ tự nhiên (tính dư thừa) cho phép khôi phục lại nội dung thông qua các phương pháp thống kê hoặc đoán từ (crib dragging).

2.2 Ý tưởng chính: Space Prediction

Trong tiếng Anh, dấu cách là ký tự xuất hiện thường xuyên nhất. Thuật toán hoạt động dựa trên giả định sau:

> Tại một vị trí index i bất kỳ, nếu ta có đủ nhiều bản mã, xác suất cao là ít nhất một trong các bản rõ gốc tại vị trí đó là dấu cách.

2.3 Quy trình thuật toán

Quy trình tấn công được thực hiện qua các bước sau:

1. **Duyệt từng cột:** Thuật toán duyệt qua từng vị trí byte i (từ 0 đến độ dài của bản mã dài nhất).

2. **Giả định và tìm khóa ứng viên:** Với mỗi bản mã C_j, thuật toán giả định rằng ký tự tại vị trí i của bản rõ tương ứng là dấu cách ($P_j[i] = \texttt{0x20}$). Khi đó, byte khóa dự đoán K_{guess} sẽ là:
$$K_{guess} = C_j[i] \oplus \texttt{0x20} \tag{2.2}$$

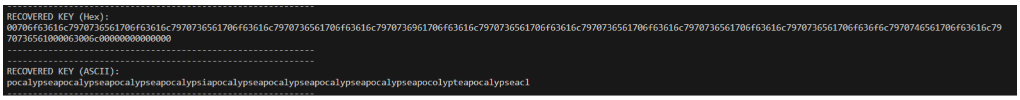
3. **Kiểm chứng:** Để kiểm tra xem K_{guess} có đúng không, thuật toán dùng nó để giải mã vị trí i của tất cả các bản mã còn lại (C_k):
$$P_k[i] = C_k[i] \oplus K_{guess} \tag{2.3}$$
Sau đó kiểm tra xem $P_k[i]$ có phải là ký tự hợp lệ hay không (chữ cái a-z, A-Z, hoặc dấu câu phổ biến).

4. **Đánh giá độ tin cậy:** Nếu tỷ lệ các ký tự sau khi giải mã thử là hợp lệ vượt ngưỡng $(70 - 90\%)$, thì K_{guess} được xem là byte khóa chính xác tại vị trí i.

5. **Tổng hợp:** Sau khi duyệt hết độ dài chuỗi, các byte khóa tìm được sẽ được ghép lại thành chuỗi khóa hoàn chỉnh để giải mã toàn bộ thông điệp.

2.4 Kết quả tấn công:

Từ đoạn bản mã sau khi chạy code Python để đoán key, thu được key như sau:



Ta đoán được key là "apocalypse", dùng để giải mã hết bản mã thì thu được kết quả như sau:

```
[0] The end of the world refers to the ultimate fate of humanity or the planet Earth.
[1] Scientists often study the end of the world in terms of astrophysical and environmental scenarios.
[2] One major hypothesis for the end of Earth is the eventual expansion of the Sun into a red giant.
[3] When the Sun expands, it may engulf Mercury and Venus, and possibly Earth.
[4] In about five billion years, the Sun will exhaust its hydrogen fuel and begin fusing helium.
[5] Climate change is a contemporary concern that could lead to societal collapse long before cosmic events occur.
[6] Nuclear war is considered one of the most immediate man-made threats to global survival.
[7] The detonation of multiple nuclear weapons could trigger a nuclear winter, blocking sunlight for years.
[8] Massive asteroid impacts are another potential cause of global extinction.
[9] A large enough impact could create fires, tsunamis, and block sunlight with dust and debris.
[10] Super volcanic eruptions could release enough ash and gas to disrupt global climate systems.
[11] The Yellowstone Caldera in the United States is one of the most studied supervolcanoes.
[12] Pandemics caused by highly contagious pathogens could drastically reduce human populations.
[13] Artificial intelligence poses both opportunities and existential risks if not properly aligned with human values.
[14] Uncontrolled nanotechnology could theoretically consume resources in a self-replicating 'gray goo' scenario.
[15] Climate collapse may lead to widespread famine, water scarcity, and forced migration.
[16] Ecological imbalance threatens biodiversity and long-term planetary stability.
```

[17] Rising sea levels may submerge major coastal cities around the world.
[18] The collapse of global economies could accompany large-scale environmental disasters.
[19] Nuclear proliferation increases the chance of catastrophic conflict.
[20] A nearby gamma-ray burst could strip away the Earth's ozone layer, exposing the surface to lethal radiation.
[21] The Milky Way is expected to collide with the Andromeda galaxy in about four billion years.
[22] This galactic collision could reshape stellar orbits, though Earth may already be uninhabitable by then.
[23] Theoretical physics suggests the universe might end through heat death, big crunch, or big rip scenarios.
[24] Heat death would occur when entropy reaches its maximum and no usable energy remains.
[25] In the big crunch, gravity would eventually reverse cosmic expansion, collapsing everything inward.
[26] The big rip hypothesis predicts that expansion will accelerate until all matter is torn apart.
[27] Black holes may slowly evaporate through Hawking radiation over unimaginable timescales.
[28] Humanity could face extinction long before cosmic events due to resource depletion or self-destruction.
[29] Environmental degradation is already causing measurable harm to ecosystems worldwide.
[30] Technological advancement without ethical restraint may accelerate existential risks.
[31] Efforts such as planetary defense aim to detect and divert dangerous asteroids.
[32] International cooperation is vital to manage global catastrophic threats.
[33] Philosophers and theologians also explore the end of the world in moral and spiritual terms.
[34] Many religions describe apocalyptic or transformative end-time events.
[35] In some beliefs, the end of the world represents renewal rather than destruction.
[36] The concept of the apocalypse has influenced literature, art, and culture for centuries.
[37] Human civilization has survived multiple near-catastrophic events throughout history.
[38] Preparing for global risks involves both scientific research and policy planning.
[39] Some scientists advocate for establishing self-sustaining colonies beyond Earth.
[40] Space exploration could serve as a safeguard against total planetary extinction.
[41] Theoretical models estimate a finite lifespan for habitable conditions on Earth.
[42] Even without external threats, natural evolution will eventually change all life forms beyond recognition.
[43] Cosmic radiation, magnetic field shifts, and solar storms can pose localized but severe dangers.
[44] Long-term survival of humanity depends on sustainable technology and ecological balance.
[45] Public awareness and education play essential roles in preventing man-made global threats.
[46] The end of the world, in practical terms, may begin gradually rather than suddenly.
[47] Every civilization must confront the question of its own fragility and legacy.
[48] While extinction is statistically probable, resilience and adaptation remain possible.
[49] The end of the world, whether physical or symbolic, reminds humanity of its shared responsibility to endure.

Phần II

Khai thác SMC

Chương 3

Bắt chặn và phân tích gói API

Mục tiêu: Phân tích giao thức bảo mật của ứng dụng SMC, tái dựng luồng hoạt động của Client và thực hiện tấn công giả mạo Server (Server Impersonation) đối với người dùng `group-2`.

3.1 Cấu hình môi trường

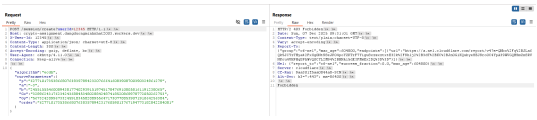
Như đã trình bày, chúng tôi sử dụng kỹ thuật Patch APK (‘apk-mitm‘) trên giả lập LDPlayer (Android 9) để vượt qua cơ chế SSL Pinning và Root Detection. Proxy được cấu hình để chuyển tiếp toàn bộ lưu lượng HTTP/HTTPS qua Burp Suite.

3.2 Kết quả bắt gói tin (Interception Results)

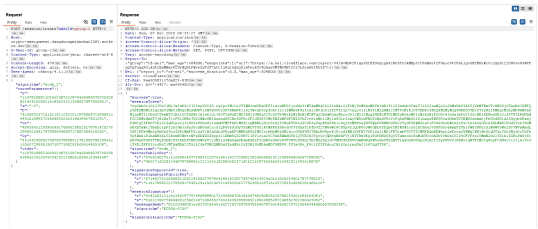
Theo phân công của đồ án, nhóm được cấp định danh người dùng là `group-2`. Để hiểu rõ cơ chế phản hồi của Server, chúng tôi thực hiện bắt gói tin trong hai kịch bản kiểm thử đối chứng:

1. **Kịch bản kiểm chứng lỗi (Negative Test):** Sử dụng một ID ngẫu nhiên không tồn tại (`12345`) để quan sát cơ chế từ chối dịch vụ.
2. **Kịch bản hoạt động chuẩn (Positive Test):** Sử dụng ID đúng (`group-2`) để thu thập toàn bộ quy trình bắt tay (Handshake) và trao đổi khóa.

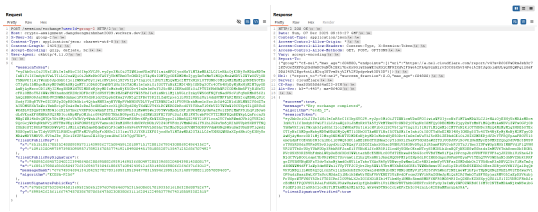
Dưới đây là các bằng chứng thực nghiệm thu được từ Burp Suite:



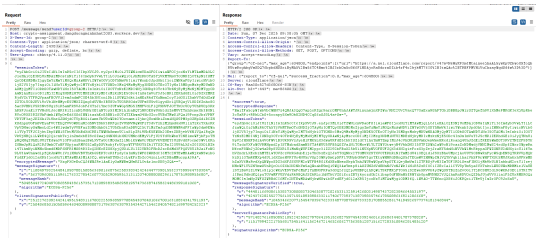
(a) Kịch bản 1: Server trả về 403 Forbidden khi sai ID



(b) Kịch bản 2: Server chấp nhận ID `group-2`



(c) Giai đoạn Exchange: Client gửi Public Key và Token



(d) Gửi tin nhắn được mã hóa và ký số

Hình 3.1: Phân tích các gói tin API trên Burp Suite

Việc so sánh hình 3.1a và 3.1b cho thấy Server thực hiện kiểm tra định danh ngay tại endpoint `/session/create`. Chỉ khi ID hợp lệ, Server mới trả về các tham số mật mã (Curve Parameters, Public Keys) để bắt đầu phiên làm việc.

Chương 4

Tái tạo SMC

GitHub tái tạo SMC

4.1 Sơ đồ luồng giao thức (Protocol Flow)

Dựa trên thứ tự gọi API, quy trình thiết lập kết bảo mật được tái dựng như sau:

1. **Negotiation (Create):** Client đề xuất tham số đường cong (Curve P-256). Server chấp nhận và gửi về khóa công khai của nó (PK_{Server}) cùng chữ ký số để chứng minh nguồn gốc.

2. **Key Agreement (Exchange):** Client tạo cặp khóa Ephemeral, gửi khóa công khai (PK_{Client}) lên Server. Cả hai bên sử dụng thuật toán ECDH để tính ra khóa bí mật chung (Shared Secret).

3. **Secure Messaging:** Dữ liệu được mã hóa đối xứng (khả năng cao là AES) sử dụng khóa phiên vừa tạo.

4.2 Mã giả logic Client (Client Pseudocode)

Algorithm 3 Client Logic: Handshake và Verification

Bước 1: Khởi tạo Session
$params \leftarrow$ {"algorithm" : "ecdh", "curveParameters" : P-256_PARAMS}
$res1 \leftarrow$ POST("/session/create?userId=group-2", $params$)
Bước 2: Xử lý phản hồi từ Server (Lỗ hổng tại đây)
$server\_ecdh\_pk \leftarrow res1$.serverPublicKey
$server\_sign\_pk \leftarrow res1$.serverSignaturePublicKey ▷ Key đến từ Network!
$signature \leftarrow res1$.sessionSignature
$isValid \leftarrow$ ECDSA_Verify($signature, server\_sign\_pk$)
if NOT $isValid$ **then**
    return Error: Signature Invalid
end if
▷ Client tin tưởng server_sign_pk mà không kiểm tra CA!
Bước 3: Trao đổi khóa (Exchange)
$client\_sk, client\_pk \leftarrow$ GenerateECCKeyPair()
$shared\_secret \leftarrow$ ECDH($client\_sk, server\_ecdh\_pk$)
$session\_token \leftarrow res1$.sessionToken
$res2 \leftarrow$ POST("/session/exchange", $\{client\_pk, session\_token, \dots\}$)
Bước 4: Gửi tin nhắn
$ciphertext \leftarrow$ Encrypt($msg, shared\_secret$)
POST("/message/send", $ciphertext$)

4.3 Các thách thức kỹ thuật và Chi tiết bảo mật (Implementation Challenges)

Trong quá trình tái dựng Client (Re-implementation), chúng tôi đã phát hiện nhiều cơ chế bảo mật ngầm định (implicit) không được mô tả trong tài liệu API nhưng bắt buộc phải tuân thủ để giao tiếp thành công với Server.

1. **Sự sai lệch về Tham số Đường cong (Curve Parameter Mismatch):**
    - *Quan sát:* Server yêu cầu thuật toán `ecdh_2` (tương ứng chuẩn P-192) cho User ID `group-2`. Tuy nhiên, nếu gửi khóa P-192, quá trình tính toán Shared Secret thất bại.
    - *Phân tích:* Dựa trên độ dài khóa trong gói tin phản hồi (trường `x, y` của Public Key), chúng tôi xác định Server thực chất đang sử dụng đường cong **SECP256R1 (P-256)**.

- *Giải pháp:* Client phải gửi payload "lai": khai báo thuật toán là `ecdh_2` để vượt qua validation, nhưng gửi kèm tham số và khóa của **P-256**.

2. **Cơ chế Xoay vòng Token (Token Rotation):**
   - *Quan sát:* Sau khi hoàn tất bước trao đổi khóa (Key Exchange) thành công, nếu Client tiếp tục sử dụng `sessionToken` ban đầu để gửi tin nhắn, Server trả về lỗi `"Key exchange not completed"`.
   - *Phân tích:* Server áp dụng cơ chế bảo mật Token Rotation. Ngay sau khi bắt tay hoàn tất, Server cấp phát một token mới trong phản hồi của API `/session/exchange`.
   - *Giải pháp:* Client cần cập nhật token mới này vào bộ nhớ đệm để sử dụng cho các request nhắn tin tiếp theo.
3. **Cấu hình hàm dẫn xuất khóa (KDF Specifics):**
   - *Quan sát:* Việc sử dụng Shared Secret thô (Raw) hoặc chuẩn HKDF hiện đại đều dẫn đến lỗi `"Decryption failed"`.
   - *Phân tích:* Thông qua việc phân tích hành vi phổ biến của các ứng dụng Java/Android cũ (Legacy), chúng tôi xác định Server sử dụng hàm **PBKDF2-HMAC-SHA256** với Salt rỗng (16 bytes 0) và 1000 vòng lặp. Đây là chi tiết không thể thấy được qua việc bắt gói tin mà phải suy luận.
4. **Chuẩn hóa dữ liệu JSON (Canonicalization):**
   - *Vấn đề:* Chữ ký số (Signature) rất nhạy cảm với từng byte dữ liệu.
   - *Giải pháp:* Python mặc định serialize JSON không theo thứ tự cố định. Chúng tôi phải áp dụng `sort_keys=True` và loại bỏ khoảng trắng (`separators=(',', ':')`) để đảm bảo chuỗi hash khớp tuyệt đối với Server.

Kết luận: Việc tái dựng thành công giao thức không chỉ đòi hỏi kiến thức về mật mã (ECDH, AES) mà còn yêu cầu khả năng suy luận về các chi tiết triển khai (Implementation Details) đặc thù của hệ thống đích.

Chương 5

Khai thác lỗ hổng và thử nghiệm

GitHub kịch bản tấn công

5.1 Tổng quan lỗ hổng

Hệ thống xác thực SMC tồn tại lỗ hổng trong cơ chế sinh chữ ký số ECDSA. Cụ thể, giá trị ngẫu nhiên (nonce k) được sử dụng trong quá trình ký không đảm bảo tính ngẫu nhiên mật mã đầy đủ (Cryptographically Secure Randomness).

Các phân tích thực nghiệm cho thấy nonce k bị giới hạn độ dài ở khoảng khoảng 160 bit (20 byte) thay vì 256 bit chuẩn. Sự cố này tạo ra một "thiên lệch"(bias) lớn (96 bit cao nhất luôn bằng 0), cho phép kẻ tấn công khôi phục khóa bí mật bằng phương pháp tấn công Lattice

5.2 Cơ sở Toán học và Quy trình Khai thác

5.2.1 Thu thập dữ liệu

Do độ thiên lệch (bias) của nonce rất lớn (2^{160} so với 2^{256}), cuộc tấn công chỉ yêu cầu một lượng mẫu cực nhỏ.

- **Số lượng mẫu (m):** 5 chữ ký.
- **Dữ liệu thu được:** Các cặp (r_i, s_i) và mã băm thông điệp h_i tương ứng.

5.2.2 Bài toán HNP (Hidden Number Problem)

Quy trình ký ECDSA tạo ra chữ ký (r,s) từ khóa bí mật d, nonce k, và mã băm h theo phương trình:

$$s \equiv k^{-1}(h + d \cdot r) \pmod{n} \tag{5.1}$$

Trong đó n là cấp (order) của đường cong P-256. Để tấn công, ta biến đổi phương trình để cô lập nonce k:

$$k \equiv s^{-1}h + d \cdot (s^{-1}r) \pmod{n} \tag{5.2}$$

Đặt $t_i = s_i^{-1} r_i \pmod{n}$ và $u_i = s_i^{-1} h_i \pmod{n}$, phương trình trở thành:

$$k_i - d \cdot t_i - u_i \equiv 0 \pmod{n} \tag{5.3}$$

Hoặc viết dưới dạng đẳng thức trên tập số nguyên (với a_i là số nguyên nào đó):

$$k_i - d \cdot t_i - u_i = a_i \cdot n \tag{5.4}$$

5.2.3 Ràng buộc kích thước

Lỗ hổng nằm ở chỗ k_i nhỏ hơn nhiều so với n.

$$0 < k_i < B \tag{5.5}$$

Trong trường hợp này, $B \approx 2^{160}$. Do $n \approx 2^{256}$, khoảng cách (gap) giữa B và n là 2^{96}. Đây là cơ sở để áp dụng thuật toán Lattice.

5.2.4 Lattice Construction

Chúng tôi chuyển hệ phương trình tuyến tính nghiệm nguyên trên về bài toán tìm vector ngắn nhất (SVP). Một Lattice $\mathscr{L}$ được dựng bởi ma trận cơ sở M kích thước $(m+2) \times (m+2)$. Với $m = 5$, ma trận có kích thước 7×7:

$$M = \begin{pmatrix} n \cdot B & 0 & \cdots & 0 & 0 & 0 \\ 0 & n \cdot B & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ t_1 \cdot B & t_2 \cdot B & \cdots & t_5 \cdot B & 1 & 0 \\ u_1 \cdot B & u_2 \cdot B & \cdots & u_5 \cdot B & 0 & n \end{pmatrix}$$

Trong đó B là trọng số (scaling factor) để cân bằng độ lớn giữa các thành phần của vector.

5.2.5 Lattice Basis Reduction

Áp dụng thuật toán LLL (Lenstra–Lenstra–Lovász) lên ma trận M. Thuật toán sẽ tìm một cơ sở mới gồm các vector ngắn (gần trực giao). Vector ngắn nhất $v_{shortest}$ trong cơ sở mới sẽ chứa thông tin về khóa bí mật:

$$v_{shortest} \approx (k_1, k_2, \ldots, k_5, d, \text{const})$$

Thành phần thứ $(m+1)$ của vector này chính là giá trị d cần tìm.

5.3 Proof of Concept

5.3.1 Kết quả khôi phục khóa

Script khai thác trên SageMath đã giải mã thành công khóa bí mật chỉ trong thời gian < 1 giây.

Private Key (d) recovered:

```
Hex: 0xec6809b67b6407972b1d662079db06730e0764b7189aeea38b56b30143342698
Decimal: 106929651395809795849503103048857480321800635263136332109986298710169...
```

5.3.2 Phân tích Nonce

Để chứng minh tính xác thực của lỗ hổng, chúng tôi sử dụng khóa d vừa tìm được để tính ngược lại giá trị nonce k của các chữ ký đã thu thập. Kết quả cho thấy toàn bộ nonce đều xấp xỉ ở khoảng 160 bit.

Bảng 5.1: Phân tích độ dài Nonce của 5 chữ ký đầu tiên

ID	Giá trị Nonce (k) [Hex]	Độ dài (Bit)
1	`0xed7ae567cae371cd0530ffa3...`	156
2	`0x378cdc04e59ecc7c669e5b32...`	158
3	`0xa5fc554e8d1a35cb55f89897...`	160
4	`0xd9fc401166717324010a39db...`	160
5	`0x3ab2f618813143399e787b08...`	158

5.3.3 Xác thực lại Khóa bí mật

Để khẳng định tính chính xác tuyệt đối của khóa bí mật d vừa khôi phục, chúng tôi thực hiện quy trình kiểm tra ngược bằng cách tính toán lại khóa công khai từ d và so sánh với khóa công khai thực tế của server.

$$Q_{calculated} = d_{recovered} \cdot G \tag{5.6}$$

Nếu $Q_{calculated}$ trùng khớp hoàn toàn với toạ độ (x, y) của khóa công khai Q_{server} mà server đã gửi, ta có thể kết luận khóa bí mật là chính xác.

- **Recovered Private Key (d):**

```
106929651395809795849503103048857480321800635263136332109986298710169199322776
```

- **Calculated Public Key ($Q = d \cdot G$):**

```
x: 640740963310091336466589733734923645529052856643208905733165893241530051091 25
y: 100058113606145378275936824014648964526228691176618082589064372273733778957161
```

Kết luận: Điểm $Q_{calculated}$ khớp hoàn toàn với khóa công khai của server. Điều này chứng minh kẻ tấn công có toàn quyền mạo danh server để ký bất kỳ dữ liệu nào.

5.4 Khuyến nghị khắc phục

1. **Cải thiện bộ sinh số ngẫu nhiên:** Đảm bảo hàm sinh nonce đọc đủ 32 byte (256 bit) từ nguồn entropy an toàn (CSPRNG).
2. **Chuyển đổi sang RFC 6979:** Khuyến nghị sử dụng cơ chế sinh nonce đơn định (Deterministic ECDSA).

   $$k = \text{HMAC\_SHA256}(d, h)$$

   Cơ chế này loại bỏ hoàn toàn sự phụ thuộc vào bộ sinh số ngẫu nhiên của hệ điều hành, ngăn chặn triệt để các tấn công kênh kề liên quan đến bias.
3. **Thu hồi và cấp mới khóa:** Do khóa bí mật hiện tại đã bị lộ, hệ thống cần thực hiện quy trình thu hồi (Key Revocation) và tạo cặp khóa mới ngay lập tức.

Tài liệu tham khảo

[BS15] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. 2015. Available at: `https://toc.cryptobook.us/`

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography*. CRC Press, 2nd edition, 2014.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.