

Proyecto Individual

The Imitation Game

Diseño e Implementación de un ASIP de descryptación mediante RSA

Fecha de asignación: 1 marzo 2023
Grupos: 1 persona

Fecha de entrega: 24 marzo 2023
Profesor: Luis Chavarría Zamora

Mediante el desarrollo de este proyecto, el estudiante aplicará los conceptos de arquitectura de computadores en el diseño e implementación de una aplicación capaz de realizar la descryptación de una imagen usando RSA empleando un ISA específico.

1. Descripción General

El RSA (Rivest-Shamir-Adleman) es un sistema criptográfico de clave pública desarrollado en 1979. Es asimétrico, lo que significa que cuenta con una llave pública (Public Key) y una llave privada (Private Key). El funcionamiento se muestra a continuación:

1. Un cliente (e.g., buscador) envía su llave pública a un servidor y solicita información. El cliente también tiene una llave privada que solo tiene él (esta se debe guardar celosamente).
2. El servidor encripta la información usando la llave pública y se la envía al cliente.
3. El cliente recibe la información y la descrypta usando la llave privada.

El funcionamiento a nivel de esquema de las llaves públicas y privadas se muestra en la Figura 1. Como se observa, cualquier ente que tenga acceso a la llave pública puede encriptar un mensaje, pero no cualquiera puede descryptarlo pues necesita la llave pública. En este [vídeo](#) se encuentra una explicación interactiva del algoritmo RSA.

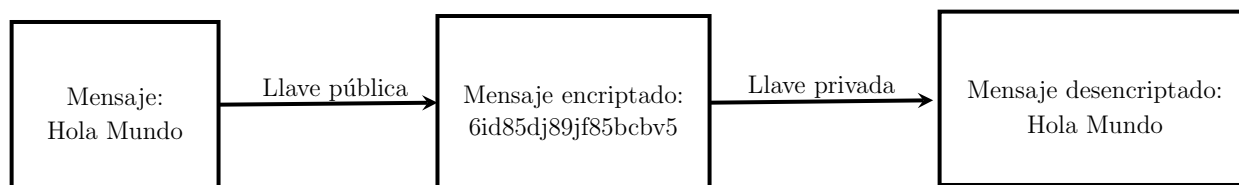


Figura 1: Esquema de encriptación y descryptación

Para encriptar un mensaje se tiene que realizar la operación

$$c = m^e \mod n, \quad (1)$$

donde m es el mensaje, c es el mensaje encriptado y $(e$ y $n)$ son los parámetros de la llave pública.
Para descryptar el mensaje se tiene que realizar la operación

$$m = c^d \pmod n, \quad (2)$$

donde c es el mensaje encriptado, m es el mensaje descryptado o recuperado y $(n^1$ y $d)$ son los parámetros de la llave privada.

Para la generación de llaves se usa el siguiente algoritmo:

1. Seleccione dos números primos aleatorios $(p$ y $q)$. Por ejemplo 7 y 2.
2. Calcule $n = pq$. Este es el módulo de la llave pública y privada. Por ejemplo, en este caso $n = 7 \times 2 = 14$.
3. Calcule la función de Euler $\phi = (p-1)(q-1)$. Por ejemplo, en este caso $\phi = (7-1)(2-1) = 6$.
4. Se selecciona un entero positivo e menor que ϕ (no puede ser igual), este debe ser coprimo a ϕ . Este es el exponente de la llave pública. Un número es coprimo de otro si el máximo común divisor es igual a 1. Dos números coprimos no tienen que ser primos, por ejemplo 6 y 19 son coprimos (19 es primo, 6 no es primo). Si e es muy pequeño podría suponer una vulnerabilidad importante para el sistema. Por ejemplo, en este caso $e = 5 < \phi = 6$.
5. Se selecciona un entero positivo d que satisfaga: $de \pmod \phi = 1$. Por ejemplo, en este caso $de = 11 \times 5 = 55 \rightarrow 55 \pmod 6 = 1$. Entonces $d = 11$ cumple el requerimiento.

De esta manera se genera el siguiente par de llaves:

- Llave pública o de encriptación: $(5,14)$.
- Llave privada o de descryptación: $(11,14)$.

Se pueden encriptar todos los números mayores a 1 ². Por ejemplo se encriptará el número 3 con las llaves públicas y privadas expresadas anteriormente

$$c = m^e \pmod n = 3^5 \pmod 14 = 243 \pmod 14 = 5.$$

Para descryptar se muestra la operación

$$m = c^d \pmod n = 5^{11} \pmod 14 = 48828125 \pmod 14 = 3,$$

¹note que este parámetro n es el mismo que el de la llave pública.

²dado (1) y (2): $1^x = 1 \forall x \in R$

se verifica como se obtiene $m = 3$ tanto en el mensaje original como en la descrición. También resulta bastante evidente que como la cantidad de números a encriptar es relativo al módulo.

Es evidente como la dificultad del algoritmo RSA radica en el manejo de números extensos esto se hace evidente viendo las operaciones involucradas en (1) y (2). No es un secreto que entre más extensos sean los números más difícil es encontrar la combinación llave pública y privada. Por esta razón se usan llaves públicas y privadas de 1024 bits ($\approx 1,8 \times 10^{308}$ combinaciones, necesitando 309 dígitos en base decimal), 2048 bits ($\approx 3,2 \times 10^{616}$ combinaciones, necesitando 617 dígitos en base decimal) o incluso más. En este [enlace](#) hay un generador de llaves RSA.

Se vuelve bastante claro como no se pueden procesar las operaciones exponente y modular de forma tradicional, sino el sistema requeriría mucha memoria o sería muy lento. Por esta razón se usa el proceso de exponenciación modular. En el [vídeo 1](#) y [vídeo 2](#) se explica este proceso matemático.

2. Especificación

Se le solicita desarrollar en **completamente en ensamblador** (asm) un programa que descricte información en RSA. Este programa recibe **solamente** la llave privada (puede ser introducida en consola o en un archivo llamado **llaves.txt**). El programa debe realizar la siguiente operación con base en la llave realizará la siguiente operación:

- **Imagen:** El usuario le indica al programa la dirección de la imagen a descricptar. Realiza el proceso de descricptación y le muestra al usuario en una cuadrícula 1×2 la imagen de entrada (encriptada) y la procesada (descricptada).

Este proceso **NO** se realizará completamente en ASM, se realizará bajo los siguientes requisitos generales de funcionalidad:

1. La imágenes deben ser visualizadas en algún software de alto nivel (e.g., Python, Matlab, Octave, etc), **solo** la visualización. El **procesamiento** de la imagen es **completamente en ensamblador (asm)**.
2. Una imagen encriptada de 640×480 con representación de 2 bytes cada píxel³ con una llave pública. El formato de la imagen es el siguiente:
 - Imagen en escala de grises con píxeles con valores entre $[0, 255]$.
 - La representación de cada píxel está dada por 2 bytes sucesivos.
 - La imagen encriptada se observa en el visualizador de imágenes como una de 640×960 .

³la imagen original es de 8 bits, pero como el n de la llave pública y privada determina el rango de números a encriptar, se procuró un $256 < n < 65536$ o n representable con a lo sumo 16 bits, esto también para guardar el alineamiento de la memoria

- **Importante: ¿es una equivocación del profesor que diga que la imagen es de 640×480 y luego dice 640×960 ?** No, recuerde que se encriptó una imagen de 640×480 , como se indicó anteriormente el resultado de la encriptación siempre genera un dato más grande, por esta razón un píxel original de un byte se convierte en dos bytes (doblando el tamaño de la imagen). El píxel 0 (MSB) y 1 (LSB) serán el píxel encriptado 0, luego el 2 (MSB) y el 3 (LSB) serán el píxel encriptado 1.
- 3. Se debe implementar una optimización de código, sino durará mucho tiempo desencriptando 307200 datos ⁴. **Pista:** El usuario sabe de antemano que se encriptó un valor de 8 bits se sabe que habrá a lo sumo 255 combinaciones ⁵, aumentando el rendimiento **1204 veces**.
- 4. El simulador del ISA (e.g., ARM, x86, RISC-V, otros) y otras herramientas de desarrollo podrán ser elegida abiertamente por cada estudiante. Todo el procesamiento de la imagen debe ser en este simulador seleccionado, recuerde que el lenguaje de alto nivel es solo para visualización, **si usa lenguaje de alto nivel en procesamiento obtendrá nota de cero**.
- 5. La imagen que se le da a la/el estudiante es en formato 'crudo', es decir como si fuese un txt con los datos en formato decimal de un byte cada uno.
- 6. El programa automáticamente debe generar un archivos de salida (imagen desencriptada).
- 7. Por facilidad todo este programa debe ser integrado en un solo *framework*. Que tome la imagen en formato 'crudo', procese la imagen en ASM y la visualice al usuario en un arreglo 1×2 .

El profesor le provee un archivo para que realice pruebas con una imagen encriptada en [enlace](#). Está conformado por los siguientes archivos:

- Un archivo de texto plano con la imagen encriptada en formato 'crudo' en una sola dimensión. El formato de la defensa será así.
- Los parámetros de llave pública y privada. Esto es solo una sugerencia del archivo **llaves.txt**. En el proyecto solo tendrán la llave privada pues es solo desencriptación. Los valores de llave privada en la defensa no serán superiores a 65536 ⁶.

El proceso de diseño debe incluir propuestas y comparación de viabilidad de las mismas, estas deben ser no fácilmente descartables.

⁴ $640 \times 480 = 307200$

⁵ $2^8 = 256 - 1 = 255$, no se usa el cero

⁶ $2^{16} = 65536$

3. Metodología de trabajo

El proyecto debe seguir los siguientes aspectos de desarrollo, sino, la parte funcional no será calificada y obtendrá nota de cero:

1. Utilice una cuenta de repositorio gratuita.
2. Cree un repositorio con el siguiente nombre: `<user_id>.computer.architecture_1_2022`. El `user_id` estará compuesto por la primera letra del nombre y el apellido. Por ejemplo, para el estudiante Luis Chavarría, el nombre del repositorio será:
`lchavarria.computer.architecture_1_2022`.
3. Si el repositorio es privado, proporcione acceso a `luchazam` (bitbucket) o `luchazam` (GitHub).
4. El repositorio de Git contendrá dos ramas principales: `master` y `development`.
5. Inicialmente, la rama de `development` se crea a partir del `master`.
6. Al trabajar en un proyecto, el estudiante debe crear una nueva rama de trabajo desde `develop` y cuando la función esté lista, la rama debe fusionarse para `develop`. Cualquier corrección o modificación adicional después de `merge` debería requerir que se repita el proceso (es decir, crear la rama desde `develop` y fusionar los cambios más tarde). Una vez que el código de desarrollo esté listo, se fusionará con `master` y se debe crear una `tag`. El proceso se describe en la siguiente Figura 2.

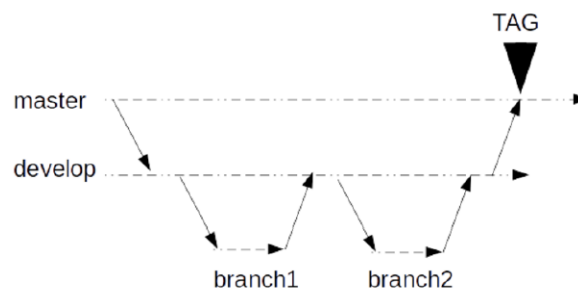


Figura 2: *Git workflow*

Adicionalmente se coloca este [enlace recomendado](#).

7. Después de haber realizado algunos proyectos la rama `master` debe verse así:

- `master/`
 - `proyecto_1`

- proyecto_2
- ...
- ...

Donde cada directorio de `proyecto_x` contiene todos los entregables para cada proyecto.

No es permitido realizar todo el trabajo en un solo commit, es decir, que realice el trabajo de forma local y solo suba el último entregable en el repositorio. Si no, obtendrá nota de cero. Debe mostrar avance incremental (se revisarán estadísticas).

4. Evaluación y entregables

La evaluación del proyecto se da bajos los siguientes rubros:

- Presentación proyecto 100 % funcional (75 %): Una defensa de 20 minutos donde el profesor evaluará el rendimiento del algoritmo. El/la estudiante llevará a la defensa una imagen libre que cumpla los requerimientos. **Debe notificar al profesor del repositorio a más tardar el lunes después de realizar la entrega del enunciado a las 11:59 pm.**
- Documentación de diseño (25 %): Este documento se encuentra directamente ligado con el atributo AP. La documentación del diseño deberá contener las siguientes secciones:
 1. Listado de requerimientos del sistema: Cada estudiante deberá determinar los requerimientos de ingeniería del problema planteado, considerando partes involucradas, estado del arte, estándares, normas, entre otros.
 2. Elaboración de opciones de solución al problema: Para el problema planteado deberán documentarse al menos dos opciones de solución. Cada solución deberá ser acompañada de algún tipo de diagrama. **Estas opciones de solución no deben ser fácilmente descartables y deben llevar un análisis objetivo con base en criterios técnicos o teóricos.**
 3. Comparación de opciones de solución: Se deberán comparar explícitamente las opciones de solución, de acuerdo con los requerimientos y otros aspectos aplicables de salud, seguridad, ambientales, económicos, culturales, sociales y de estándares. **La comparación debe ser objetiva basado en las características técnicas y teóricas del ISA.**
 4. Selección de la propuesta final: Se deberá evaluar de forma objetiva, válida y precisa las soluciones planteadas al problema y escoger una solución final.

5. Archivo tipo README donde especifiquen las herramientas que usaron y las instrucciones de uso. **Es un documento README.MD aparte.**
- Los documentos serán sometidos a control de plagios para eliminar cualquier intento de plagio con trabajos de semestres anteriores, actual o copias textuales, tendrán nota de cero los datos detectados. Se prohíbe el uso de referencias hacia sitios no confiables.
 - No coloque código fuente en los documentos, quita espacio y aporta poco. Mejor explique el código, páselo a pseudocódigo o use un diagrama.