



# CRIPTOGRAFÍA PARA INGENIER@S

# Class4crypt

© jorgeramio 2022

Aula virtual de  
criptografía  
aplicada

Diapositivas  
utilizadas en las  
clases grabadas  
de Class4crypt

Módulo 10 Criptografía asimétrica

Dr. Jorge Ramió Aguirre © 2022



Attribution-NonCommercial-  
NoDerivatives 4.0 International  
(CC BY-NC-ND 4.0)



<https://www.youtube.com/user/jorgeramio>

Dr. Jorge Ramió Aguirre

*El ingenio es intrínseco al ser humano,  
solo hay que darle una oportunidad  
para que se manifieste.*

<https://www.cryptored.es/cvJorge/index.htm>

# Class4crypt

- Módulo 1. Principios básicos de la seguridad
- Módulo 2. Matemáticas discretas en la criptografía
- Módulo 3. Complejidad algorítmica en la criptografía
- Módulo 4. Teoría de la información en la criptografía
- Módulo 5. Fundamentos de la criptografía
- Módulo 6. Algoritmos de criptografía clásica
- Módulo 7. Funciones hash
- Módulo 8. Criptografía simétrica en bloque
- Módulo 9. Criptografía simétrica en flujo
- Módulo 10. Criptografía asimétrica

# Class4crypt

## Módulo 10. Criptografía asimétrica

10.1 Criptografía asimétrica y la analogía de los candados

10.2 Intercambio de clave de Diffie y Hellman

10.3 Ataque man in the middle a DH

10.4 Algoritmo RSA

10.5 Generación de claves RSA y estándar PKCS#1

10.6a Cifrado y descifrado con RSA (parte 1)

10.6b Cifrado y descifrado con RSA (parte 2)

10.7 Números no cifrables en RSA

10.8 Claves parejas y números piratas en RSA

10.9a Ataques teóricos y prácticos a RSA parte 1 (continúa en la siguiente página)

# Class4crypt

## Módulo 10. Criptografía asimétrica (continuación)

10.9b Ataques teóricos y prácticos a RSA parte 2

10.10 Algoritmo de cifra de Elgamal

10.11 Algoritmos de firma digital RSA y de Elgamal

10.12 Firma digital DSA y ataques a firmas de Elgamal y DSA

10.13 Criptografía con curvas elípticas (guía para estudio personal)

Lista de reproducción del módulo 10 en el canal Class4crypt

<https://www.youtube.com/playlist?list=PLq6etZPDh0kvlQtQ4w6NkisPE4tEobKQd>

# Class4crypt c4c10.1

## Módulo 10. Criptografía asimétrica

### Lección 10.1. Criptografía asimétrica y la analogía de los candados

10.1.1. Buscando enviar un secreto en un medio de transmisión inseguro

10.1.2. Primer escenario con un único candado del receptor

10.1.3. Segundo escenario con dos candados, de emisor y de receptor

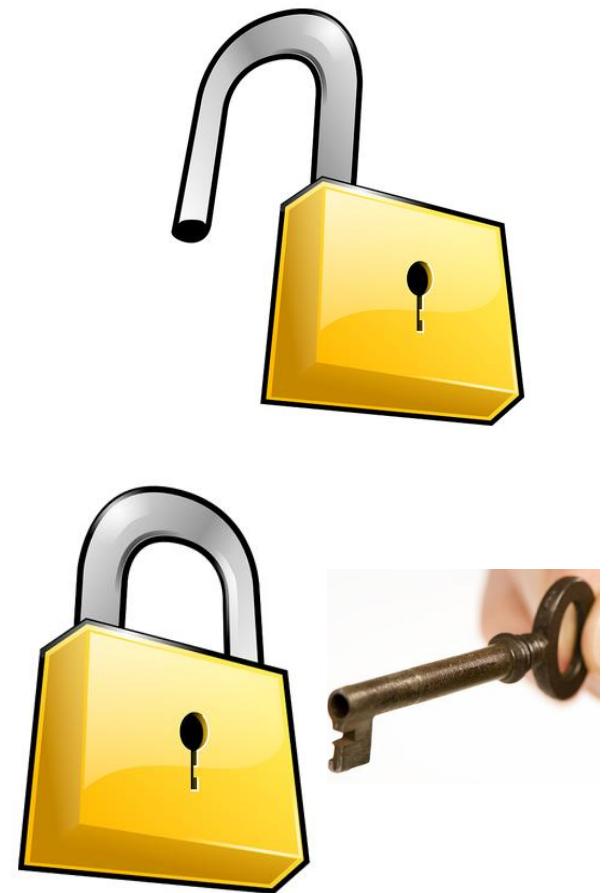
10.1.4. En criptografía el orden es importante

10.1.5. Soluciones al problema de enviar un secreto en un medio de transmisión inseguro

Class4crypt c4c10.1 Criptografía asimétrica y la analogía de los candados  
<https://www.youtube.com/watch?v=j7sEBM9XXUQ>

# Analogía de la cifra con los candados

- Todos sabemos que un candado tiene dos estados, o está abierto o está cerrado
- La operación de cerrar un candado la puede hacer cualquier persona, basta con bajar la clavija hasta escuchar un clic
- En cambio, la operación de abrir ese candado requiere que se tenga una llave. Y no cualquier llave, sino la llave específica de ese candado
- Podríamos decir que la operación de cerrar un candado es una operación pública (la de cifrar), en tanto que la operación de abrir ese candado será una operación privada (la de descifrar)



# Necesidad de intercambiar un secreto

- Alicia y Bernardo, que viven en sendos castillos del siglo XV, muy a su pesar físicamente separados, desean enviarse mensajes secretos, pero saben que el medio de transmisión es, por definición, inseguro
- Van a plantear dos métodos para alcanzar este objetivo, contando con una caja muy segura (a prueba de balas, fuego, etc.) y sendos candados



**Alicia**



**Bernardo**



# Primer escenario: un solo candado



- Alicia le entrega a Bernardo una caja y un candado, ambos abiertos, para que los use Bernardo cuando desee enviarle a ella un mensaje secreto
- Bernardo genera ese mensaje secreto, lo mete en la caja, cierra la caja con el candado de Alicia, y se lo entrega a un mensajero para que se lo lleve a Alicia
- Cuando Alicia recibe la caja, lo único que debe hacer es usar la llave de ese candado que sólo ella tiene, por ser su candado, y abrirla para leer el mensaje secreto de Bernardo
- Parece que todo funciona bien, pero las cosas no serán lo que parecen cuando trasladamos este escenario a los sistemas de cifra



# Los candados y la cifra del César versión 1



Alfabeto de cifra de Alicia  
 $c_i = m_i + 5 \text{ mod } 27$ ,  
conocido por Bernardo

|                   |   |   |   |  |   |   |   |   |   |  |   |   |   |   |   |   |
|-------------------|---|---|---|--|---|---|---|---|---|--|---|---|---|---|---|---|
| Mensaje secreto   | n | o | s |  | v | a | m | o | s |  | m | a | ñ | a | n | a |
| Cifrado Bernardo  | R | T | X |  | A | F | Q | T | X |  | Q | F | S | F | R | F |
| Descifrado Alicia | n | o | s |  | v | a | m | o | s |  | m | a | ñ | a | n | a |

- Todo marcha bien **siempre y cuando** Bernardo conozca el alfabeto de cifra de Alicia. Pero, ¿cómo se lo ha enviado? Necesitamos superar este inconveniente

# Segundo escenario: dos candados



- Alicia introduce su mensaje secreto en una caja muy resistente, la cierra con su candado y se la entrega a un mensajero que se la lleva a Bernardo
- Cuando la caja llega a Bernardo, este añade su propio candado y el mensajero va de vuelta donde Alicia con la caja, ahora cerrada con los dos candados
- Alicia abre su candado, y el mensajero vuelve con la caja hacia Bernardo
- Bernardo abre su candado y ahora puede leer el mensaje secreto de Alicia
- El secreto siempre ha estado bien protegido durante su transmisión
- Si suponemos que cerrar el candado corresponde a cifrar y abrir el candado corresponde a descifrar, ¿podríamos asociar esto con la criptografía? ➔

# Los candados y la cifra del César versión 2

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | ñ | o | p | q | r | s | t | u | v | w | x | y | z |
| F | G | H | I | J | K | L | M | N | Ñ | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E |

Alfabeto de cifra de Alicia:  $c_i = m_i + 5 \text{ mod } 27$



|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | ñ | o | p | q | r | s | t | u | v | w | x | y | z |
| C | D | E | F | G | H | I | J | K | L | M | N | Ñ | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B |

Alfabeto de cifra de Bernardo:  $c_i = m_i + 2 \text{ mod } 27$



| Mensaje secreto |   | n | o | s |  | v | a | m | o | s |  | m | a | ñ | a | n | a |
|-----------------|---|---|---|---|--|---|---|---|---|---|--|---|---|---|---|---|---|
|                 | Paso 1: Alicia cierra su candado (cifra)    | R | T | X |  | A | F | Q | T | X |  | Q | F | S | F | R | F |
|                 | Paso 2: Bernardo cierra su candado (cifra)  | T | V | Z |  | C | H | S | V | Z |  | S | H | U | H | T | H |
|                 | Paso 3: Alicia abre su candado (descifra)   | o | q | u |  | x | c | ñ | q | u |  | ñ | c | p | c | o | c |
|                 | Paso 4: Bernardo abre su candado (descifra) | n | o | s |  | v | a | m | o | s |  | m | a | ñ | a | n | a |

El secreto dentro de la caja ha estado siempre protegido en sus viajes

# Los candados y la cifra por decimación

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | ñ | o | p | q | r | s | t | u | v | w | x | y | z |
| A | C | E | G | I | K | M | Ñ | P | R | T | V | X | Z | B | D | F | H | J | L | N | O | Q | S | U | W | Y |

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | ñ | o | p | q | r | s | t | u | v | w | x | y | z |
| A | F | K | O | T | Y | D | I | N | R | W | B | G | L | P | U | Z | E | J | Ñ | S | X | C | H | M | Q | V |

Alfabeto de cifra de Alicia:  $c_i = 2*m_i \text{ mod } 27$



Alfabeto de cifra de Bernardo:  $c_i = 5*m_i \text{ mod } 27$

| Mensaje secreto |   | n | o | s |  | v | a | m | o | s |  | m | a | ñ | a | n | a |
|-----------------|---|---|---|---|--|---|---|---|---|---|--|---|---|---|---|---|---|
|                 | Paso 1: Alicia cierra su candado (cifra)    | Z | D | L |  | Q | A | X | D | L |  | X | A | B | A | Z | A |
|                 | Paso 2: Bernardo cierra su candado (cifra)  | V | O | B |  | E | A | M | O | B |  | M | A | F | A | V | A |
|                 | Paso 3: Alicia abre su candado (descifra)   | l | u | ñ |  | c | a | g | u | ñ |  | g | a | p | a | l | a |
|                 | Paso 4: Bernardo abre su candado (descifra) | n | o | s |  | v | a | m | o | s |  | m | a | ñ | a | n | a |

Nuevamente el secreto dentro de la caja ha estado siempre protegido... pero ¿funcionará siempre?

# Los candados y la cifra afín

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j | k | l | m | n | ñ | o | q | r | s | t | u | v | w | x | y | z |   |
| R | W | B | G | L | P | U | Z | E | J | Ñ | S | X | C | H | M | Q | V | A | F | K | O | T | Y | D | I | N |

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a | b | c | d | e | f | g | h | i | j | K | l | m | n | ñ | o | q | r | s | t | u | v | w | x | y | z |   |
| G | S | F | R | E | Q | D | P | C | O | B | Ñ | A | N | Z | M | Y | L | X | K | W | J | V | I | U | H | T |

Alfabeto de cifra de Alicia:  $c_i = 5m_i + 18 \text{ mod } 27$



Alfabeto de cifra de Bernardo:  $c_i = 13m_i + 6 \text{ mod } 27$

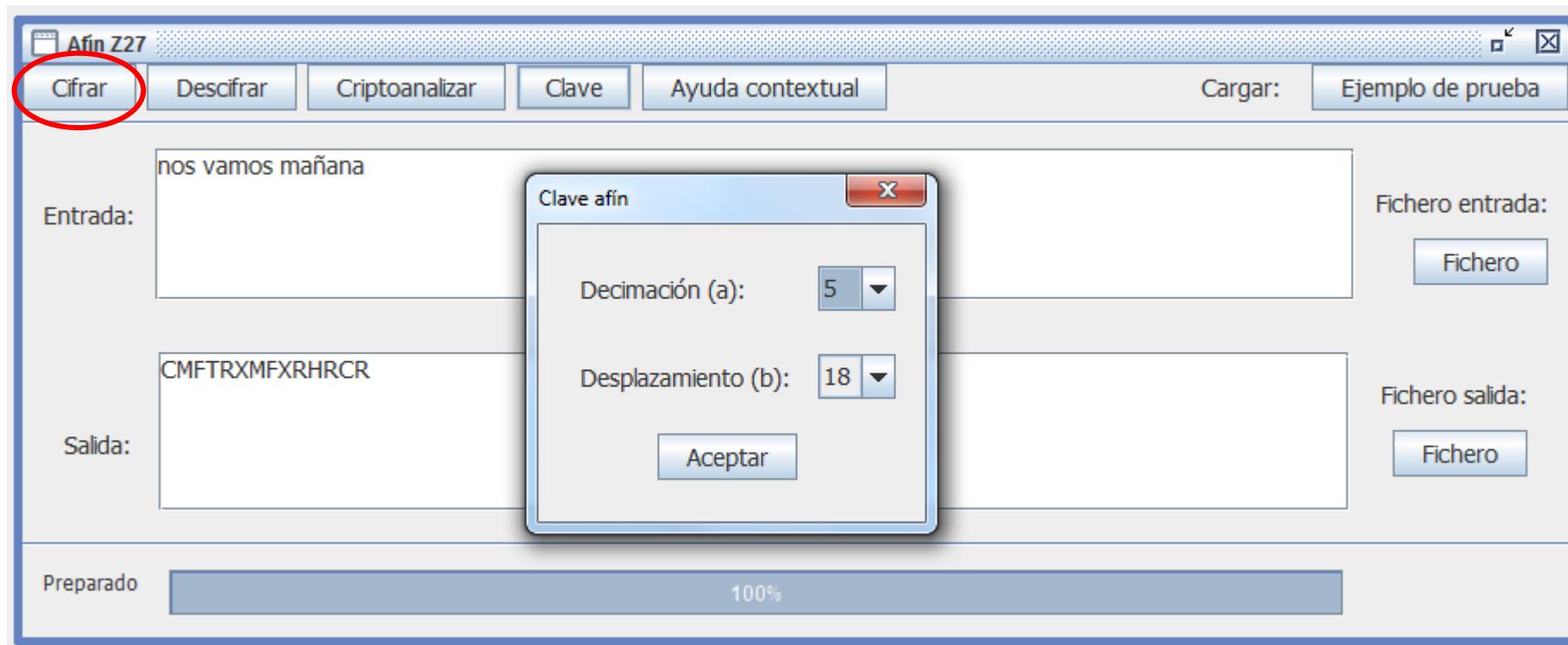


| Mensaje secreto                             | n | o | s |  | v | a | m | o | s |  | m | a | ñ | a | n | a |
|---|---|---|---|--|---|---|---|---|---|--|---|---|---|---|---|---|
| Paso 1: Alicia cierra su candado (cifra)    | C | M | F |  | T | R | X | M | F |  | X | R | H | R | C | R |
| Paso 2: Bernardo cierra su candado (cifra)  | F | A | Q |  | W | X | U | A | Q |  | U | X | P | X | F | X |
| Paso 3: Alicia abre su candado (descifra)   | s | r | p |  | b | m | g | r | p |  | g | m | f | m | s | m |
| Paso 4: Bernardo abre su candado (descifra) | b | d | h |  | k | o | a | d | h |  | a | o | c | o | b | o |

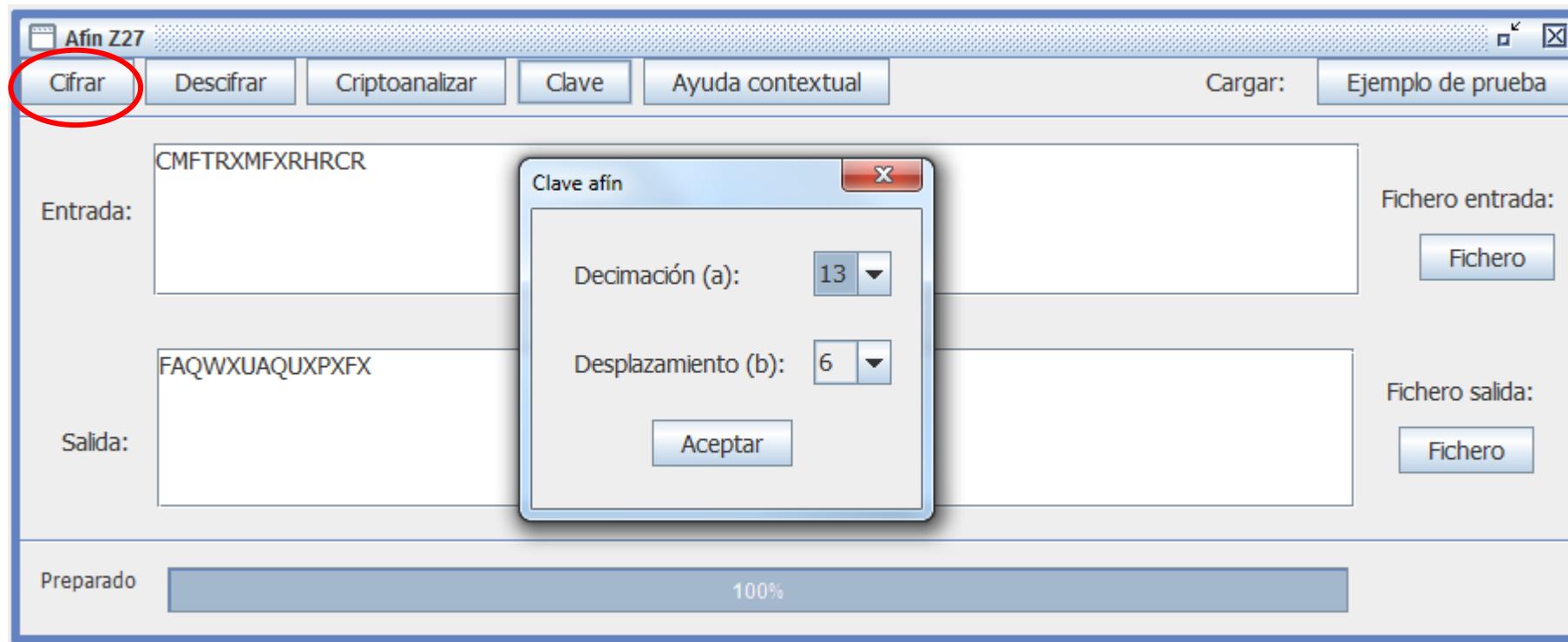
Ahora no se recupera el secreto...

# Cifrado de Alicia (comprobación)

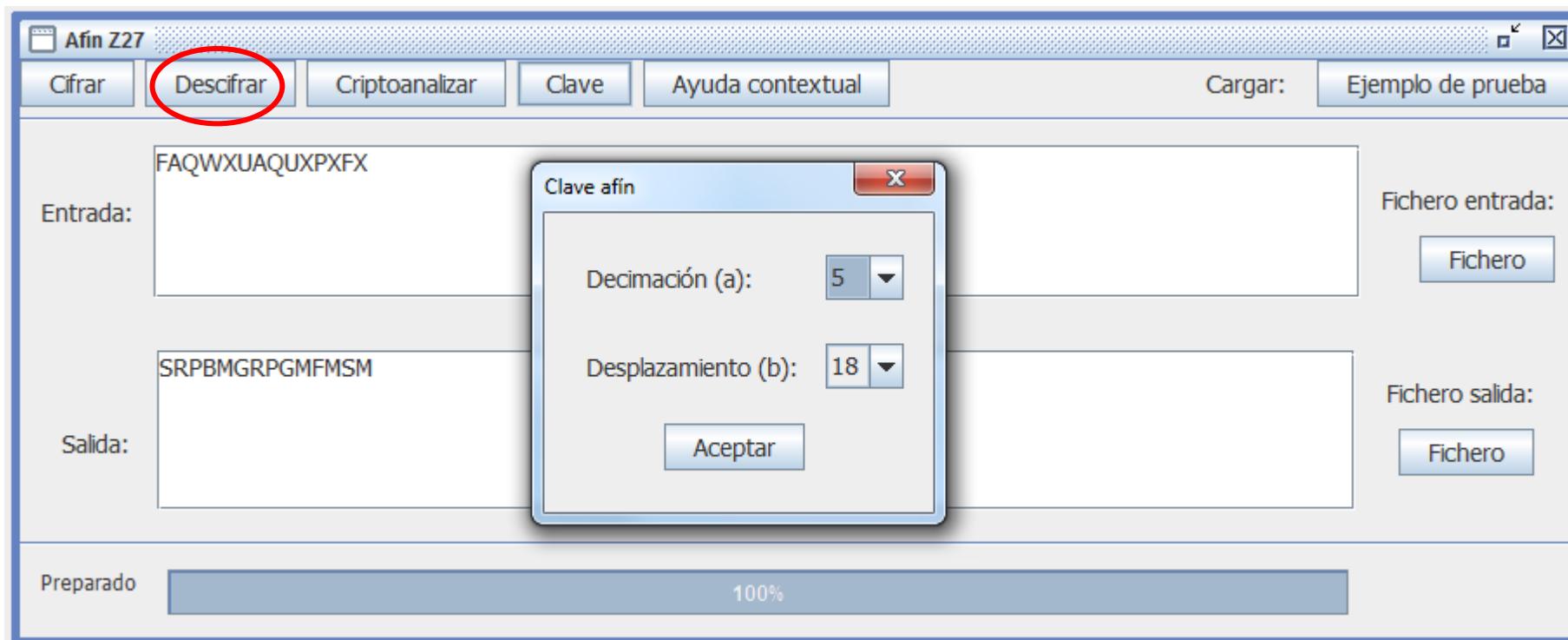
- Software Criptoclásicos v2.1: [https://www.criptored.es/software/sw\\_m001c.htm](https://www.criptored.es/software/sw_m001c.htm)



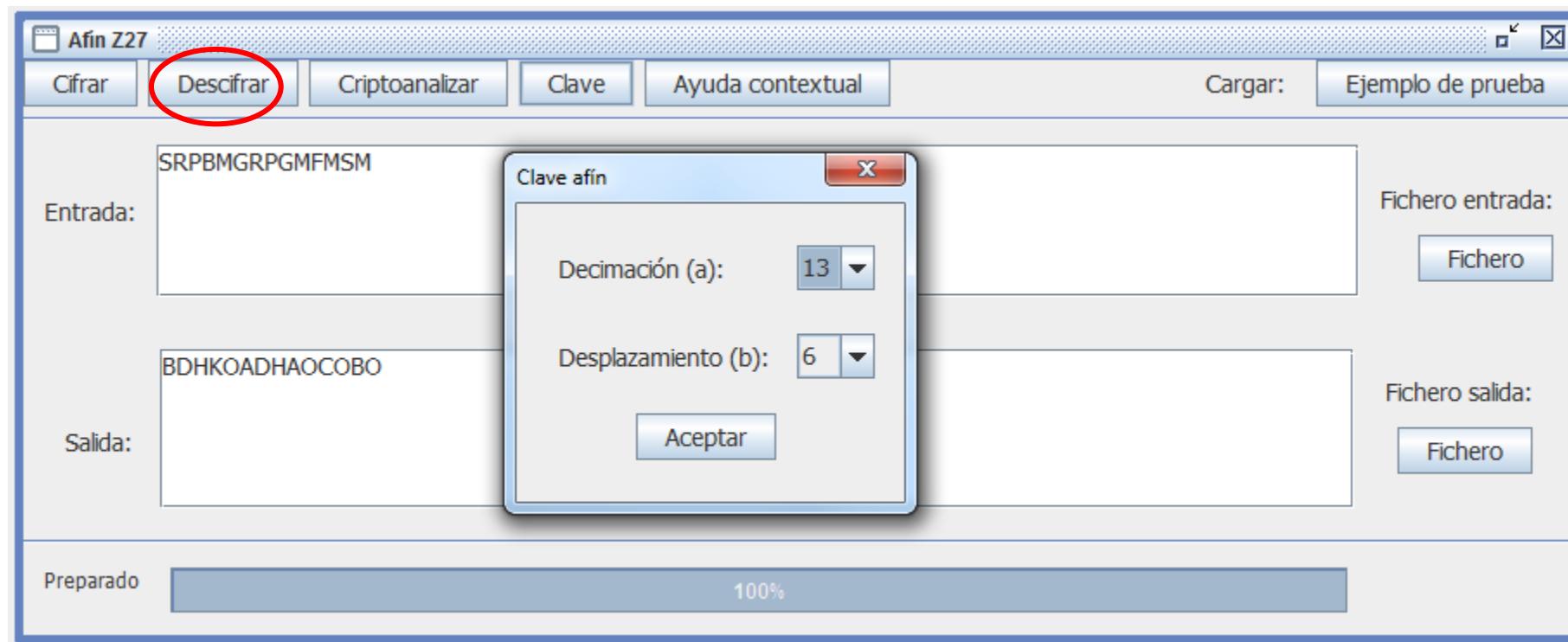
# Cifrado de Bernardo (comprobación)



# Descifrado de Alicia (comprobación)



# Descifrado de Bernardo (comprobación)



# La importancia de los inversos

- En la cifra por sustitución simple monoalfabética con algoritmos del tipo César (desplazamiento puro) y de decimación (multiplicación pura), se recupera siempre el secreto, debido a que los inversos aditivo y multiplicativo cumplen su función
- En el primer caso se desplaza + 5 (Alicia) y después + 2 (Bernardo), un total de 7 espacios. Al descifrar se hace - 5 (Alicia) y después - 2 (Bernardo), obteniendo texto en claro por concepto del inverso aditivo
- En el segundo caso Alicia multiplica por 5, y luego Bernardo multiplica por 2. Al descifrar, Alicia ha usado  $\text{inv}(5, 27) = 11$ , eliminando su cifra, y Bernardo luego ha usado  $\text{inv}(2, 27) = 14$ , eliminando por su parte su cifra, inversos multiplicativos
- En resumen, se ha multiplicado el texto en claro por  $10 = 5 \cdot 2$  en la cifra, y en el descifrado se ha multiplicado por los inversos  $11 \cdot 14 \bmod 27 = 154 \bmod 27 = 19$  que es precisamente el  $\text{inv}(10, 27)$
- En la cifra afín y, muy importante, en la cifra asimétrica, esto ya no se cumple...

# Algo falla, hay que perfeccionarlo

- De lo anterior podría deducirse que el enfoque de los candados para enviar un secreto no funcionaría en la criptografía real, pero no es así
- Habrá que usar funciones de una sola vía (one way functions)
- Como su nombre sugiere, una función de una sola vía es fácil de hacer pero muy difícil de deshacer. En otras palabras, las funciones de doble vía son reversibles, pero las funciones de una sola vía son irreversibles
- Entre estas funciones, se encuentra el problema del logaritmo discreto PLD, que es lo que usarán Diffie y Hellman en su protocolo de intercambio de clave para permitir el envío de un secreto entre dos usuarios separados físicamente
- El intercambio de clave de Diffie y Hellman se verá en la siguiente videoclase

# En la criptografía el orden es importante

- En criptografía hay que realizar las operaciones de forma inversa y en orden
- Es decir, si Alicia ha cerrado la caja con el candado que Bernardo le ha entregado abierto (operación de cifrado) para proteger ese mensaje secreto, la única opción es que a continuación de ello Bernardo use la llave de su candado para abrirlo (operación de descifrado), y recuperar así el secreto
- La solución será dividir una clave muy grande en dos partes: una parte muy pequeña que se dará a conocer a todo el mundo y que se llamará clave pública (la caja y el candado abierto), y la otra parte muy grande que guardará en secreto su dueño y que se llamará clave privada (la llave del candado)
- Quien no tenga una trampa o secreto que sólo conoce el dueño de la clave, no podrá deducir esa clave privada a partir del conocimiento de la clave pública

# Material multimedia en intypedia



Lección 3: Sistemas de cifra con clave pública (intypedia)

<https://www.youtube.com/watch?v=On1clzor4x4>

# Conclusiones de la lección 10.1

- Si se comparte el candado abierto del destinatario, ello permite hacer una cifra a cualquiera, pero descifrar sólo podrá hacerlo quien posea la llave de ese candado. Sabiendo que el canal es inseguro, no tenemos cómo enviarle al destino nuestra clave de cifra, que será necesaria para cifrar el secreto
- En un escenario de cifrado con dos candados, a veces se recupera el secreto, como en la cifra sustitución por desplazamiento puro ( $c_i = m_i + b \text{ mod } n$ ) y con decimación pura ( $c_i = a*m_i \text{ mod } n$ ), pero para otras formas de cifra básica como la cifra afín ( $c_i = a*m_i + b \text{ mod } n$ ), ya no se recupera el secreto
- La analogía de los candados con la cifra, con operaciones de cifra públicas y operaciones de cifra privadas, requiere seguir en destino un orden inverso a las operaciones realizadas en origen para su correcto funcionamiento
- Habrá que usar funciones de un solo sentido como el PLD (Diffie y Hellman)

# Lectura recomendada

- Documento Lección 3 intypedia, Sistemas de cifra con clave pública, Gonzalo Álvarez Marañón, 2010
  - <https://www.criptored.es/intypedia/docs/es/video3/GuionIntypedia003.pdf>
- One-way function (Wikipedia)
  - [https://en.wikipedia.org/wiki/One-way\\_function](https://en.wikipedia.org/wiki/One-way_function)

# Class4crypt c4c10.2

## Módulo 10. Criptografía asimétrica

### Lección 10.2. Intercambio de clave de Diffie y Hellman

10.2.1. Recordando el Problema del Logaritmo Discreto y generadores en un cuerpo

10.2.2. Protocolo de intercambio de clave de Diffie y Hellman

10.2.3. Protocolo de intercambio de clave de Diffie y Hellman no simultáneo

10.2.4. Seguridad del protocolo de intercambio de clave de Diffie y Hellman

Class4crypt c4c10.2 Intercambio de clave de Diffie y Hellman  
<https://www.youtube.com/watch?v=Tr9aR4mc6kw>

# Recordando qué es el PLD

- El problema del logaritmo discreto PLD decía lo siguiente:
  - Sean  $p$  un primo y  $\alpha$  un generador de ese primo, ambos números públicos
    - En una próxima transparencia recordaremos qué es un generador en un primo  $p$
  - Exponenciación modular
    - Si  $x$  es un valor privado (secreto), entonces  $y = \alpha^x \text{ mod } p$  será un resultado que podría ser público. Será, además, de fácil y rápido cálculo, independientemente de los valores de  $\alpha$ ,  $p$  y  $x$
  - Logaritmo discreto
    - No obstante, conociendo  $(\alpha, p, y)$ , encontrar ahora el valor privado  $x$  significa resolver la ecuación  $x = \log_{\alpha} y \text{ mod } p$ , muy costosa en tiempo de cálculo, inabordable cuando  $p$  es un primo muy grande

# Gráfica del PLD



Figura 30. PLD: curva del tiempo de cómputo versus tamaño de n desde 10 a 110 bits.

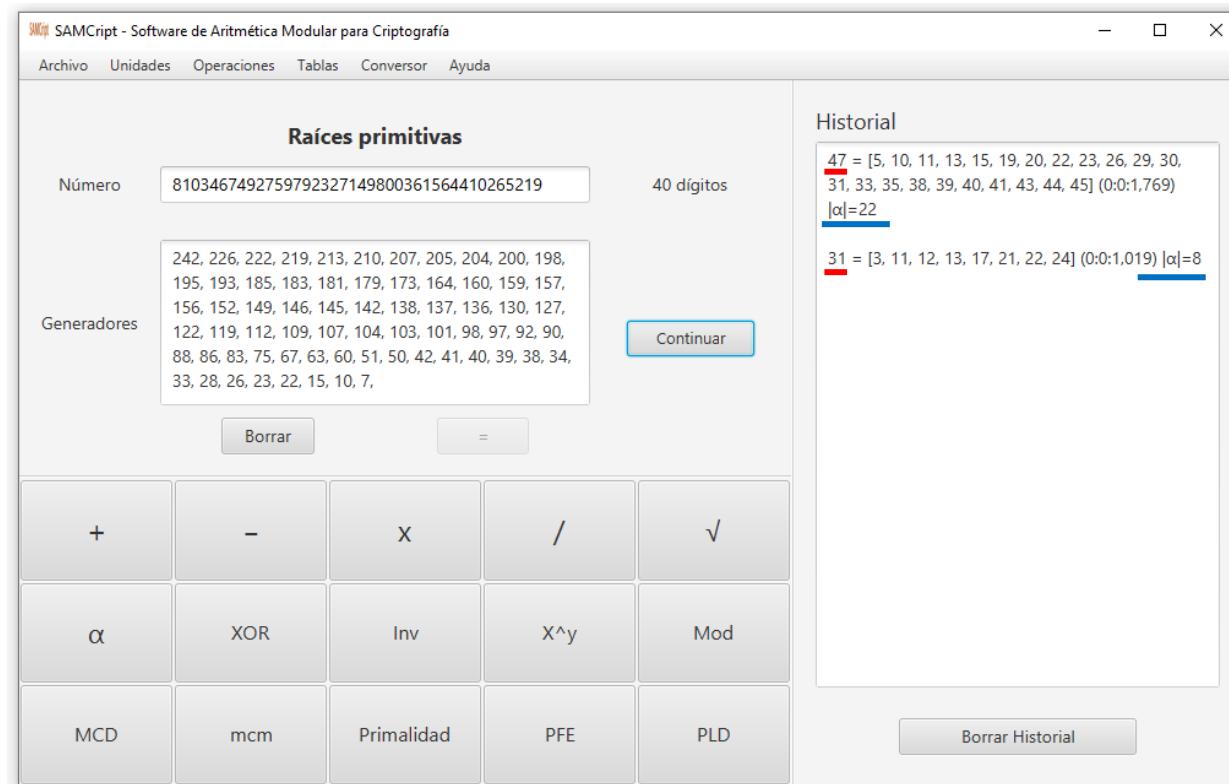
[https://www.criptored.es/descarga/CLCript\\_entrega\\_09\\_Matematicas\\_Discretas\\_Criptografia\\_SAMCript.pdf](https://www.criptored.es/descarga/CLCript_entrega_09_Matematicas_Discretas_Criptografia_SAMCript.pdf)

# Recordando qué es un generador $\alpha$ en $p$

- Un generador  $\alpha$ , o raíz primitiva, es un resto del primo  $p$  que será capaz de generar todo el cuerpo  $p$ , de forma que las operaciones  $\alpha^1 \bmod p$ ,  $\alpha^2 \bmod p$ ,  $\alpha^3 \bmod p$ , ...,  $\alpha^{p-1} \bmod p$ , entregan todos los restos de  $p$ , desde 1 hasta  $p-1$
- No se puede saber a priori los generadores de un primo  $p$ , pero es muy fácil encontrarlos (mediante cálculos de prueba y error) ya que la distribución de estos números entre los restos de  $p$  es bastante uniforme y existe más de un 30% de restos que son generadores de  $p$
- Por ejemplo en el primo 29 son generadores los restos 2, 3, 8, 10, **11**, 14, 15, 18, 19, 21, 26 y 27. Así, **11**<sup>1</sup> mod 29, **11**<sup>2</sup> mod 29, **11**<sup>3</sup> mod 29 ... **11**<sup>28</sup> mod 29 nos entregarán números distintos, “generando” todo el cuerpo  $p$
- Y entonces para  $\alpha^x \bmod p = y$ , habrá un único **x** que entregue ese valor  $y$

# Buscando generadores en p con SAMCript

- $p = 31, p = 47, p = 8103467492759792327149800361564410265219$



# Whitfield Diffie y Martin Hellman

- En noviembre de 1976, los investigadores Whitfield Diffie y Martin Hellman de la Universidad de Stanford, proponen un algoritmo para un intercambio de clave computacionalmente seguro entre dos usuarios separados físicamente
- Computacionalmente seguro quiere decir que para resolver el problema asociado a la seguridad de dicho intercambio de clave, se necesitará una cantidad de tiempo, de cómputo, costes y consumo de energía tan exagerado, que no es abordable actualmente



Ralph Merkle, Martin Hellman, Whitfield Diffie



# Algoritmo de Diffie y Hellman entre A y B

- El algoritmo de DH se utiliza para el intercambio seguro de una clave entre dos interlocutores A (Alicia) y B (Bernardo).
- Ambos seleccionan un número primo  $p$  y un generador  $\alpha$  de  $p$ , valores que son públicos, realizando los siguientes pasos:

## Protocolo



Bernardo

**A** genera un número aleatorio **a** y envía a B  $\alpha^a \text{ mod } p = x_A$

**B** genera un número aleatorio **b** y envía a A  $\alpha^b \text{ mod } p = x_B$

**B** usa su clave secreta **b** y calcula  $x_A^b \text{ mod } p = \alpha^{ab} \text{ mod } p$

**A** usa su clave secreta **a** y calcula  $x_B^a \text{ mod } p = \alpha^{ba} \text{ mod } p$

El secreto compartido es  $K = \alpha^{ab} \text{ mod } p = \alpha^{ba} \text{ mod } p$



Alicia

# Algoritmo DH entre varios usuarios: A, B, C

- **A, B y C** seleccionan un cuerpo  $p$  y un generador  $\alpha$
- **A** genera un número aleatorio  $a$  y envía  $(\alpha^a \bmod p)$  a **B**
- **B** genera un número aleatorio  $b$  y envía  $(\alpha^b \bmod p)$  a **C**
- **C** genera un número aleatorio  $c$  y envía  $(\alpha^c \bmod p)$  a **A**
- **A** recibe  $(\alpha^c \bmod p)$ , calcula  $[(\alpha^c)^a \bmod p]$  y se lo envía a **B**
- **B** recibe  $(\alpha^a \bmod p)$ , calcula  $[(\alpha^a)^b \bmod p]$  y se lo envía a **C**
- **C** recibe  $(\alpha^b \bmod p)$ , calcula  $[(\alpha^b)^c \bmod p]$  y se lo envía a **A**
- **A** recibe  $(\alpha^{bc} \bmod p)$  y calcula  $[(\alpha^{bc})^a \bmod p] = \alpha^{bca} \bmod p$
- **B** recibe  $(\alpha^{ca} \bmod p)$  y calcula  $[(\alpha^{ca})^b \bmod p] = \alpha^{cab} \bmod p$
- **C** recibe  $(\alpha^{ab} \bmod p)$  y calcula  $[(\alpha^{ab})^c \bmod p] = \alpha^{abc} \bmod p$
- El secreto compartido por **A, B** y **C** es el valor  $\alpha^{abc} \bmod p$

# Ejemplo de intercambio de clave de DH

- Alicia (A) y Bernardo (B) van a intercambiar una clave de sesión dentro del cuerpo  $p = 5.981$ , siendo el generador  $\alpha = 3$
- Alicia elegirá el valor secreto  $a = 1.655$
- Bernardo elegirá el valor secreto  $b = 3.904$ 
  1. A calcula  $\alpha^a \text{ mod } p = 3^{1.655} \text{ mod } 5.981 = 2.177$  y se lo envía a B
  2. B calcula  $\alpha^b \text{ mod } p = 3^{3.904} \text{ mod } 5.981 = 2.272$  y se lo envía a A
  3. B recibe 2.177 y calcula  $2.177^{3.904} \text{ mod } 5.981 = 216$
  4. A recibe 2.272 y calcula  $2.272^{1.655} \text{ mod } 5.981 = 216$
  5. La clave compartida es  $\alpha^{ab} \text{ mod } p = 3^{1.655*3.904} \text{ mod } 5.981 = 3^{6.461.120} \text{ mod } 5.981 = 216$
- El inconveniente de este protocolo es que el secreto compartido se conoce a posteriori, una vez terminado el protocolo y ambos deben realizarlo simultáneamente (adecuado Internet)
- Como veremos a continuación, algunas variaciones del algoritmo de DH permitirán usar este protocolo para enviar un número secreto previamente conocido, es decir enviar al receptor una clave de sesión K generada y conocida antes por el emisor

# Comprobación con SAMCript

- Alicia (A) y Bernardo (B) eligen  $p = 5.981$ , siendo el generador  $\alpha = 3$
- Alicia elegirá el valor secreto  $a = 1.655$ , Bernardo elegirá el valor secreto  $b = 3.904$
- La clave compartida será  $K = 216$



# Intercambio de clave DH no simultáneo (1)

- Fase previa
- Claves de Alicia
  - Usa un primo  $p_A$  con un generador  $\alpha_A$
  - Ha elegido su clave privada  $a$
  - Ha calculado su clave pública  $C_{púbA} = \alpha_A^a \text{ mod } p_A$ 
    - Alicia hace público:  $p_A, \alpha_A, C_{púbA}$
- Claves de Bernardo:
  - Usa un primo  $p_B$  con un generador  $\alpha_B$
  - Ha elegido su clave privada  $b$
  - Ha calculado su clave pública  $C_{púbB} = \alpha_B^b \text{ mod } p_B$ 
    - Bernardo hace público:  $p_B, \alpha_B, C_{púbB}$

# Intercambio de clave DH no simultáneo (2)

Operaciones que debe hacer Alicia para enviar a Bernardo una clave K

1. Calcula una clave  $K_{AB} = (C_{PúbB})^a \text{ mod } p_B = (\alpha_B^b \text{ mod } p_B)^a \text{ mod } p_B$
2.  $K_{AB} = \alpha_B^{ba} \text{ mod } p_B$
3. De ese valor  $K_{AB}$  en el cuerpo  $p_B$  (de al menos 1.024 bits), se eligen un conjunto de bits para una clave de cifra simétrica, por ejemplo los primeros 256 bits significativos (primer bit en 1, set) para el AES-256 como clave K (si faltasen bits, muy poco probable, se llenaría)
4. Alicia cifra el mensaje con el algoritmo simétrico AES y la clave K
5. Alicia envía a Bernardo como criptograma ese mensaje cifrado
6. Además Alicia envía a Bernardo el siguiente valor  $K_1 = \alpha_B^a \text{ mod } p_B$

# Intercambio de clave DH no simultáneo (3)

Operaciones que debe hacer Bernardo para recuperar esa clave K

1. El valor recibido  $K_1 = \alpha_B^a \text{ mod } p_B$  lo eleva a su clave privada b
2.  $K_1^b \text{ mod } p_B = (\alpha_B^a \text{ mod } p_B)^b \text{ mod } p_B = \alpha_B^{ab} \text{ mod } p_B = \alpha_B^{ba} \text{ mod } p_B = K_{AB}$
3. Bernardo selecciona los mismos 256 bits que en origen y obtiene K
4. Con esa clave K y usando el algoritmo AES, Bernardo descifra el mensaje cifrado por Alicia y obtiene el texto en claro

# Ejemplo de intercambio DH no simultáneo

## Alicia

$$p_A = 311, \alpha_A = 19, \mathbf{a = 188}, C_{\text{púbA}} = 245$$

Usarán los primeros seis bits significativos de la clave encontrada (relleno si necesario)

### Alicia calcula una clave de sesión K

$$K_{AB} = (C_{\text{púbB}})^a \bmod p_B = 27^{188} \bmod 409 = 180$$

Como  $180 = 10110100$

Elige los primeros 6 bits =  $101101 = 45 = K$

Cifra el mensaje con  $K = 45$  y se lo envía a B

$$\text{Y envía } K_1 = \alpha_B^a \bmod p_B = 21^{188} \bmod 409 = 20$$

## Bernardo

$$p_B = 409, \alpha_B = 21, \mathbf{b = 258}, C_{\text{púbB}} = 27$$

Usarán los primeros seis bits significativos de la clave encontrada (relleno si necesario)

### Bernardo recupera la clave de sesión K

$$\text{Bernardo recibe } K_1 = \alpha_B^a \bmod p_B = 20$$

$$\text{Calcula } K_1^b \bmod p_B = 20^{258} \bmod 409 = 180$$

Como  $180 = 10110100$

Elige los primeros 6 bits =  $101101 = 45 = K$

Descifra el criptograma con la clave  $K = 45$

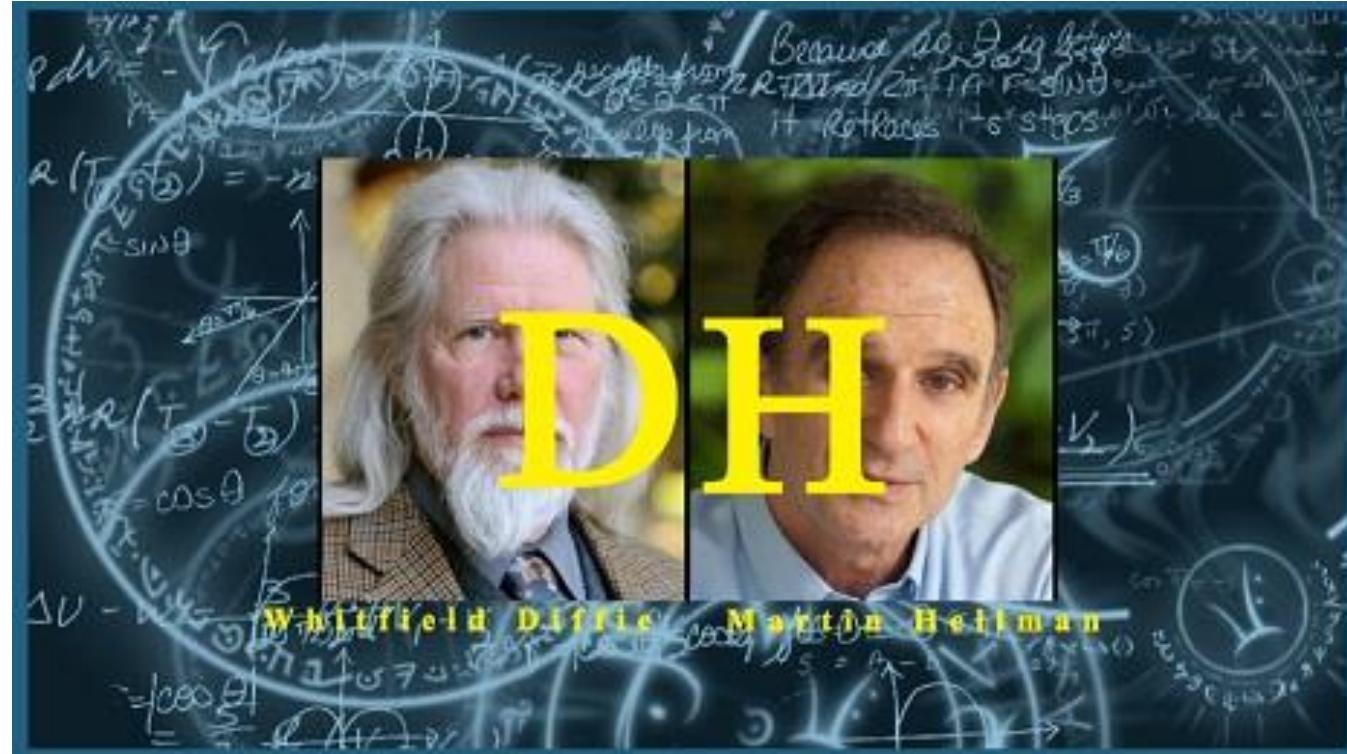
# Seguridad del protocolo de DH (1)

- La seguridad reside en que si el primo  $p$  es un número muy grande (e.g. dos mil bits) y los valores secretos  $a$  y  $b$  de cada interlocutor no son demasiado pequeños, para que  $(\alpha^a)$  y  $(\alpha^b)$  sean mayores que el tamaño de  $p$ , entonces el atacante sólo podría conocer:
  1. El valor del primo  $p$  que es público
  2. El valor del generador  $\alpha$  que es público
  3. El valor  $x_A = (\alpha^a \text{ mod } p)$  que A envía a B, y que el atacante deberá capturar
  4. El valor  $x_B = (\alpha^b \text{ mod } p)$  que B envía a A, y que el atacante deberá capturar

# Seguridad del protocolo de DH (2)

- El atacante se enfrentará al PLD para números grandes
  1. Calcular  $\log_{\alpha} X_A \text{ mod } p$ , para obtener el secreto a (muy difícil)
  2. Calcular  $\log_{\alpha} X_B \text{ mod } p$ , para obtener el secreto b (muy difícil)
  3. Y sólo después podría encontrar la clave  $\alpha^{ab} \text{ mod } p$  (muy fácil)
- Por ejemplo, p puede ser un primo de 2.048 bits y los valores secretos a y b el resultado de una función hash SHA256 de sendas frases de paso de Alicia y Bernardo
- Si  $\alpha$  no es un generador, el protocolo también funciona pero no habrá un único valor de clave secreta x, sino varios. Para primos p tan grandes, esto se puede obviar (OpenSSL usa  $\alpha = 2$  y  $\alpha = 5$ )

# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=TWhax2wQOrU>

Errata en el minuto 0:38. Diffie y Hellman eran investigadores de la Universidad de Stanford. Por error aparece Harvard (disculpas)

# Conclusiones de la lección 10.2

- Recordamos qué es el PLD y qué son los generadores o raíces primitivas
- Establecemos que la seguridad del intercambio de clave de Diffie y Hellman reside en la dificultad de resolver el PLD para un primo  $p$  grande
- El protocolo de DH puede generalizarse para más de dos usuarios, aunque lo habitual es intercambiar una clave entre dos usuarios o equipos
- Existe una variante del protocolo de DH que permite intercambiar una clave conocida de antemano por el emisor, sin necesidad de simultaneidad
- Al usar primos grandes en el protocolo de intercambio de clave de Diffie y Hellman, logramos que éste sea computacionalmente seguro. Es decir, su resolución implica una cantidad de tiempo, capacidad de cómputo, costes y consumo de energía no permitidos con el estado actual de la informática

# Lectura recomendada (1/2)

- Guion píldora formativa Thoth nº 38, ¿Qué es el intercambio de clave de Diffie y Hellman?, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora038.pdf>
- New Directions in Cryptography (november 1976)
  - <https://ee.stanford.edu/~hellman/publications/24.pdf>
- Discrete logarithm
  - [https://en.wikipedia.org/wiki/Discrete\\_logarithm](https://en.wikipedia.org/wiki/Discrete_logarithm)
- On Generators of Diffie-Hellman-Groups
  - [https://florianjw.de/en/insecure\\_generators.html](https://florianjw.de/en/insecure_generators.html)

# Lectura recomendada (2/2)

- What is the difference between Diffie Hellman generator 2 and 5?
  - <https://security.stackexchange.com/questions/54359/what-is-the-difference-between-diffie-hellman-generator-2-and-5>
- DH\_generate\_parameters
  - [https://www.openssl.org/docs/man1.0.2/man3/DH\\_generate\\_parameters.html](https://www.openssl.org/docs/man1.0.2/man3/DH_generate_parameters.html)

# Class4crypt c4c10.3

## Módulo 10. Criptografía asimétrica

### Lección 10.3. Ataque man in the middle al intercambio de clave de DH

10.3.1. Recordando el protocolo de intercambio de clave de Diffie y Hellman

10.3.2. Ataque man in the middle con una tercera parte usando una sola clave secreta

10.3.3. Ataque man in the middle con una tercera parte usando dos claves secretas

10.3.4. Ejercicio práctico

Class4crypt c4c10.3 Ataque man in the middle al intercambio de clave de DH  
<https://www.youtube.com/watch?v=MK546at83sw>

# Recordando protocolo de Diffie y Hellman

## Intercambio de clave de Diffie y Hellman



Bernardo

**A** genera un número aleatorio **a** y envía a B  $\alpha^a \text{ mod } p = x_A$

**B** genera un número aleatorio **b** y envía a A  $\alpha^b \text{ mod } p = x_B$

**B** usa su clave secreta **b** y calcula  $x_A^b \text{ mod } p = \alpha^{ab} \text{ mod } p$

**A** usa su clave secreta **a** y calcula  $x_B^a \text{ mod } p = \alpha^{ba} \text{ mod } p$

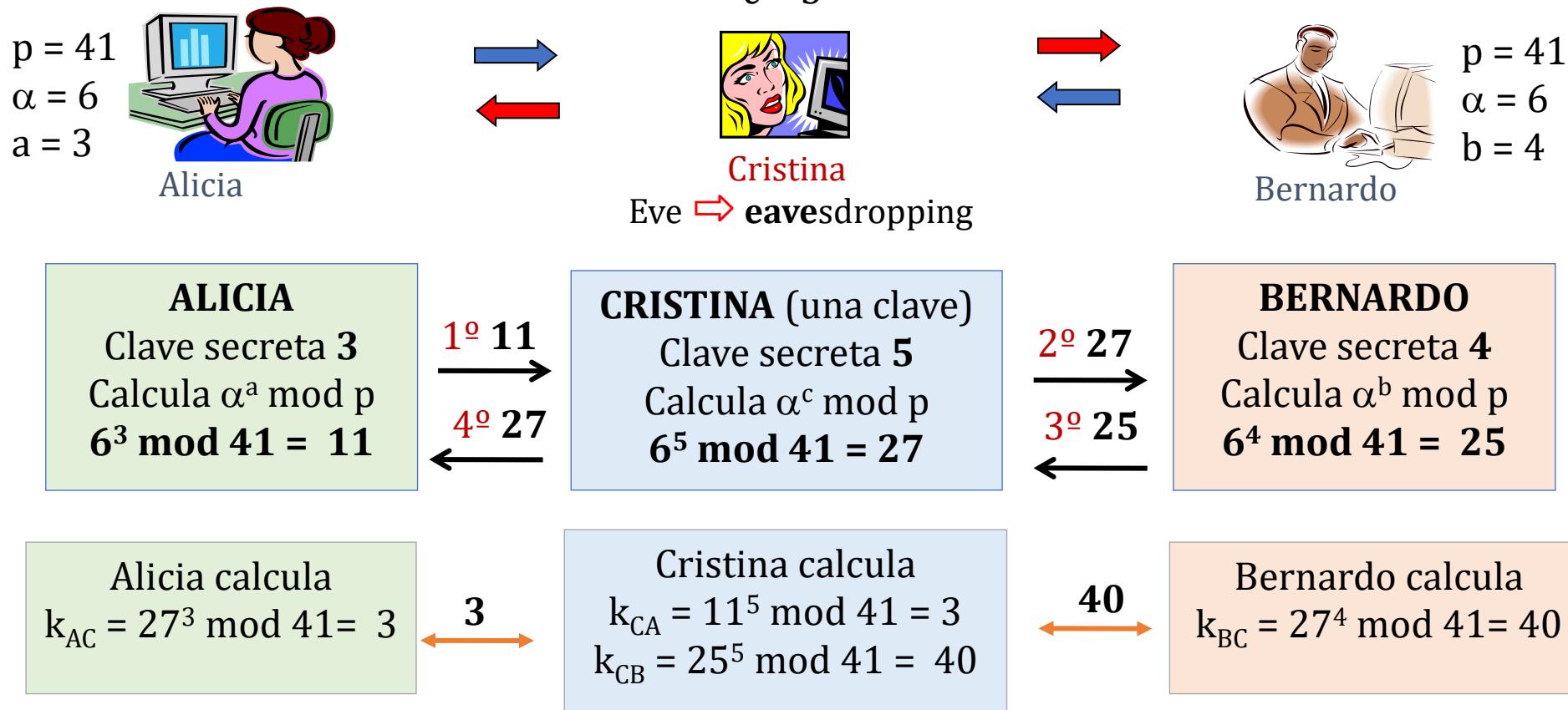
El secreto compartido es  $K = \alpha^{ab} \text{ mod } p = \alpha^{ba} \text{ mod } p$



Alicia

- ¿Pero qué ocurriría si alguien se pone en medio de esta conversación?
  - Y logra capturar el número que Bernardo envía a Alicia...
  - Y logra capturar el número que Alicia envía a Bernardo...
  - Y es capaz de enviar un número diferente tanto a Alicia como a Bernardo...

# Man in the middle a DH con una clave (1)



# Man in the middle a DH con una clave (2)

- Resumiendo:
  - Con los datos públicos  $p$  y  $\alpha$ , y las claves privadas **a** de Alicia, **b** de Bernardo y **c** de Cristina
  - Alicia calcula  $k_{AC} = (\alpha^c \bmod p)^a \bmod p = \alpha^{ca} \bmod p$
  - Cristina calcula  $k_{CA} = (\alpha^a \bmod p)^c \bmod p = \alpha^{ac} \bmod p$
  - Bernardo calcula  $k_{BC} = (\alpha^c \bmod p)^b \bmod p = \alpha^{cb} \bmod p$
  - Cristina calcula  $k_{CB} = (\alpha^b \bmod p)^c \bmod p = \alpha^{bc} \bmod p$
  - Cristina y Alicia se comunican mediante la clave DH  $\alpha^{ac} \bmod p$
  - Cristina y Bernardo se comunican mediante la clave DH  $\alpha^{bc} \bmod p$

# Man in the middle a DH con dos claves (1)

$p = 41$   
 $\alpha = 6$   
 $a = 3$



Alicia

$$c_A = 6 \quad c_B = 5$$



Cristina



Bernardo

$$p = 41 \\ \alpha = 6 \\ b = 4$$

**ALICIA**  
 Clave secreta  $a = 3$   
 Calcula  $\alpha^a \text{ mod } p$   
 $6^3 \text{ mod } 41 = 11$

$$\xrightarrow{1^{\text{o}}} 11$$

$$\xleftarrow{4^{\text{o}}} 39$$

**CRISTINA (dos claves)**

- Clave secreta  $c_B = 5$   
 Calcula  $\alpha^{c_B} \text{ mod } p$   
 $6^5 \text{ mod } 41 = 27 \text{ (-> B)}$
- Clave secreta  $c_A = 6$   
 Calcula  $\alpha^{c_A} \text{ mod } p$   
 $(A <-) 6^6 \text{ mod } 41 = 39$

**BERNARDO**  
 Clave secreta  $b = 4$   
 Calcula  $\alpha^b \text{ mod } p$   
 $6^4 \text{ mod } 41 = 25$

Alicia calcula  
 $k_{AC} = 39^3 \text{ mod } 41 = 33$

$$\xleftarrow{33}$$

Cristina calcula  
 $k_{CA} = 11^6 \text{ mod } 41 = 33$   
 $k_{CB} = 25^5 \text{ mod } 41 = 40$

$$\xleftarrow{40}$$

Bernardo calcula  
 $k_{BC} = 27^4 \text{ mod } 41 = 40$

# Man in the middle a DH con dos claves (2)

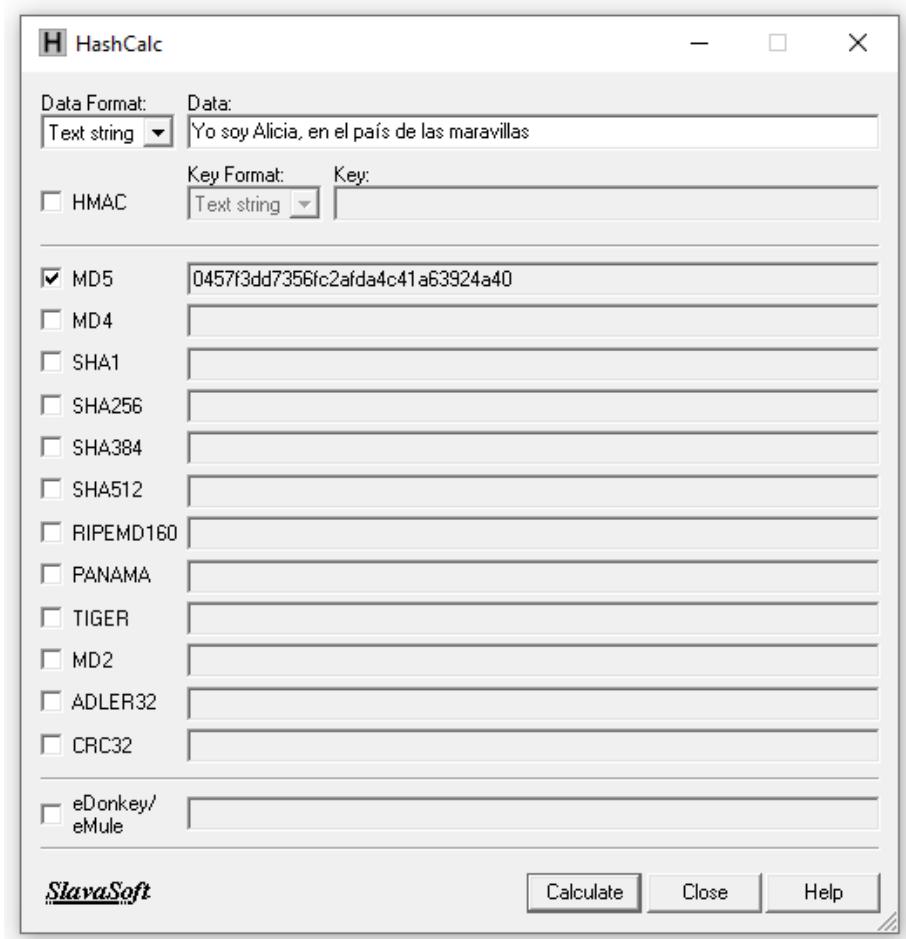
- Resumiendo:
  - Con los datos públicos  $p$  y  $\alpha$ , y las claves privadas  $a$  de Alicia,  $b$  de Bernardo,  $c_A$  de Cristina con Alicia y  $c_B$  de Cristina con Bernardo
  - Alicia calcula  $k_{AC} = (\alpha^{cA} \bmod p)^a \bmod p = \alpha^{cA*a} \bmod p$
  - Cristina calcula  $k_{CA} = (\alpha^a \bmod p)^{cA} \bmod p = \alpha^{a*cA} \bmod p$
  - Bernardo calcula  $k_{BC} = (\alpha^{cB} \bmod p)^b \bmod p = \alpha^{cB*b} \bmod p$
  - Cristina calcula  $k_{CB} = (\alpha^b \bmod p)^{cB} \bmod p = \alpha^{b*cB} \bmod p$
  - Cristina y Alicia se comunican mediante la clave DH  $\alpha^{a*cA} \bmod p$
  - Cristina y Bernardo se comunican mediante la clave DH  $\alpha^{b*cB} \bmod p$

# Ejercicio práctico

- Alicia y Bernardo usan este primo de 160 bits y este generador  $\alpha$ 
  - $p = 0x FAD913871CE8D63FC4B11640459629E1F293F9DF$ ,  $\alpha = 0x 2$
- Alicia usa como valor a el MD5 de: **Yo soy Alicia, en el país de las maravillas**
- Bernardo usa como valor b el MD5 de: **Mi nombre es nardo... Bernardo**
- Cristina usa como valor  $c_A$  el MD5 de: **Soy Cristina, aquí la mala con A**
- Cristina usa como valor  $c_B$  el MD5 de: **Soy Cristina, aquí la mala con B**
- Comprueba que estas son las claves compartidas entre A, B y C tras el ataque:
  - $k_{AC} = k_{CA} = 0x E920D263B470ED244A1F08D6C54D74515C2BF97B$
  - $k_{BC} = k_{CB} = 0x C7AF2CFE5C9A2FFEF4DEC562C2B1E85C202ABE0$
  - SAMCript: [https://www.criptored.es/software/sw\\_m001t.htm](https://www.criptored.es/software/sw_m001t.htm)
  - HashCalc: <https://www.slavasoft.com/hashcalc/>

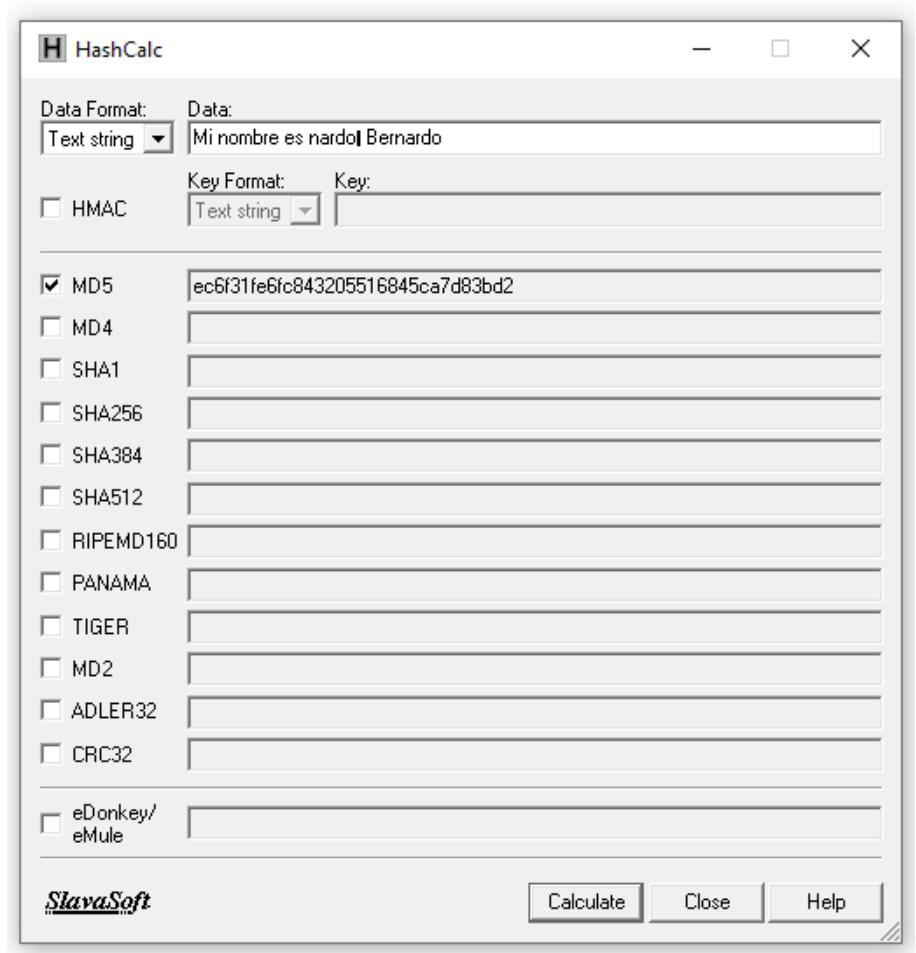
# Hash MD5 de Alicia (clave a)

MD5 (Yo soy Alicia, en el país de las maravillas)  
a = 0457F3DD7356FC2AFDA4C41A63924A40



# Hash MD5 de Bernardo (clave b)

MD5 (Mi nombre es nardo... Bernardo)  
b = EC6F31FE6FC843205516845CA7D83BD2



# Hashes MD5 de Cristina (claves C<sub>A</sub> y C<sub>B</sub>)

MD5 (Soy Cristina, aquí la mala con A)

$$c_A = AB03B36A813205046B703CD0C1190C9B$$

The screenshot shows the HashCalc application interface. In the 'Data' field, the text 'Soy Cristina, aquí la mala con A' is entered. The 'Data Format' dropdown is set to 'Text string'. The 'Key' field is empty. The 'Key Format' dropdown is set to 'Text string'. The 'HMAC' checkbox is unchecked. The 'MD5' checkbox is checked, and its corresponding hash value 'ab03b36a813205046b703cd0c1190c9b' is displayed in the adjacent text field. Other hashing algorithms like MD4, SHA1, SHA256, etc., have their checkboxes unchecked.

MD5 (Soy Cristina, aquí la mala con B)

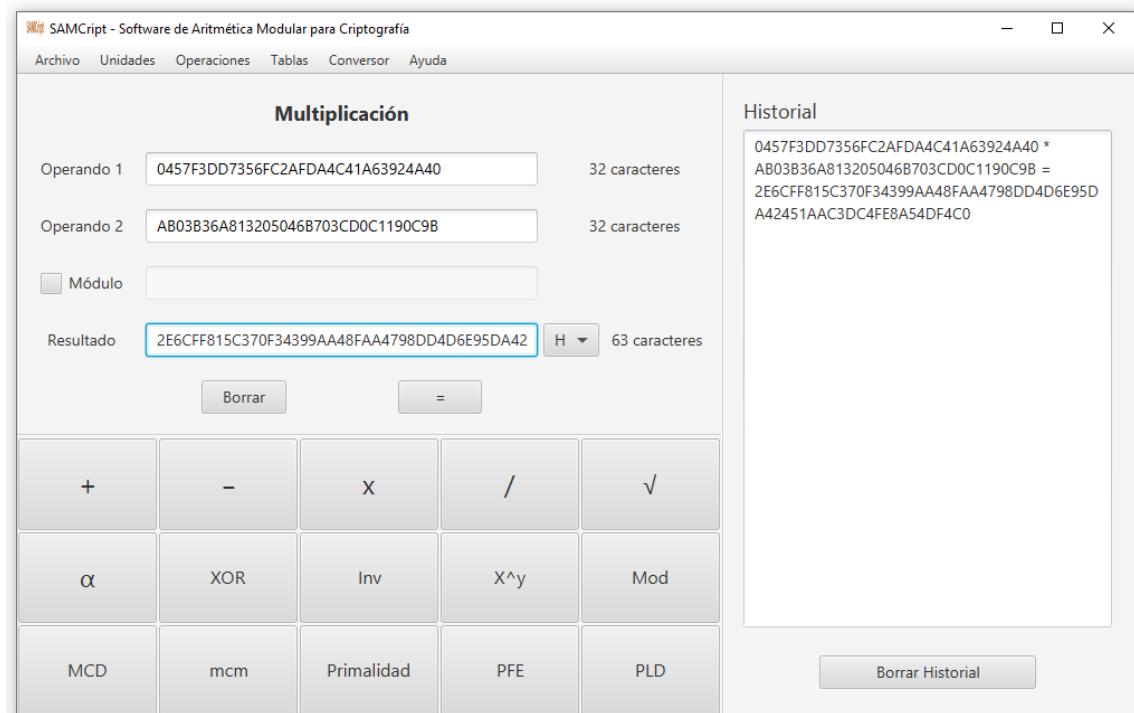
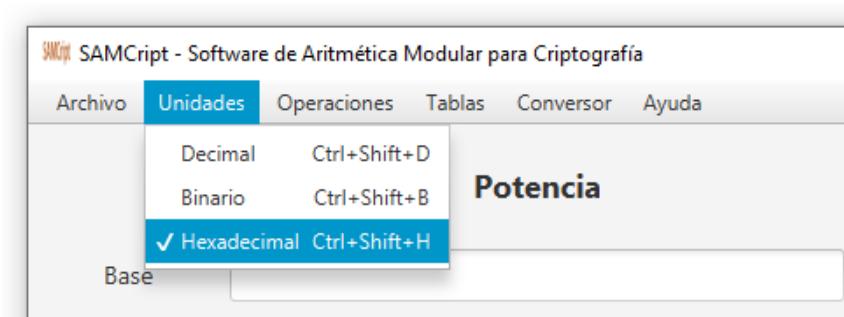
$$c_B = 445260D4BAB0CE8ACB926DE32959B708$$

The screenshot shows the HashCalc application interface. In the 'Data' field, the text 'Soy Cristina, aquí la mala con B' is entered. The 'Data Format' dropdown is set to 'Text string'. The 'Key' field is empty. The 'Key Format' dropdown is set to 'Text string'. The 'HMAC' checkbox is unchecked. The 'MD5' checkbox is checked, and its corresponding hash value '445260d4bab0ce8acb926de32959b708' is displayed in the adjacent text field. Other hashing algorithms like MD4, SHA1, SHA256, etc., have their checkboxes unchecked.

Aunque los textos sólo difieren en que A = 01000001 y B = 01000010 en código ASCII, sólo dos bits, los hashes son completamente diferentes. Es una propiedad muy interesante que veremos en la clase dedicada a las funciones hash

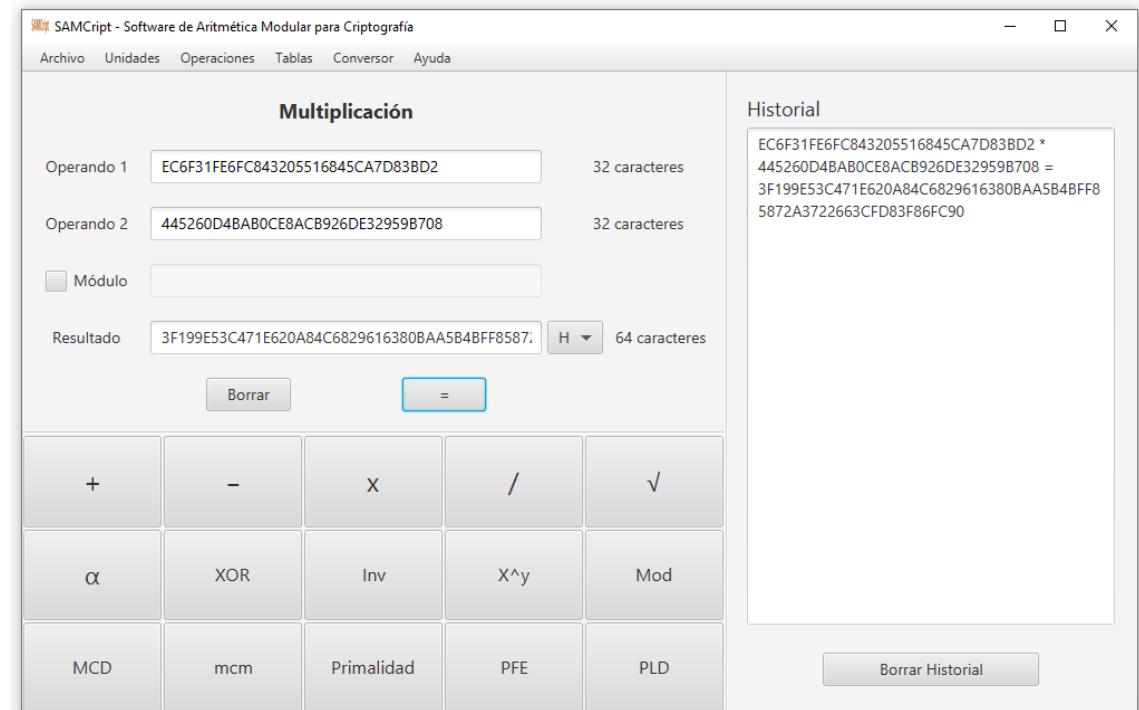
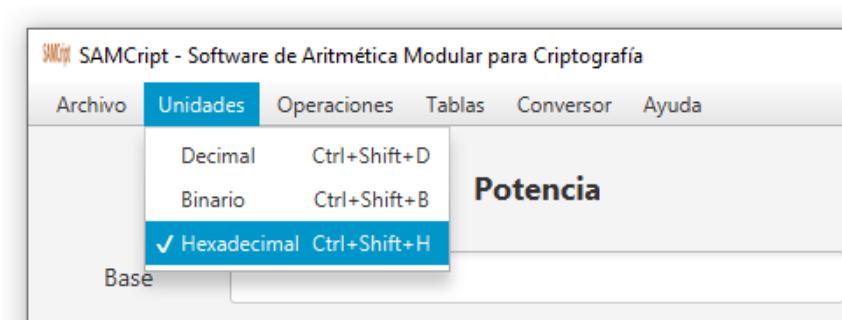
# Producto $a^*c_A$

0x 0457F3DD7356FC2AFDA4C41A63924A40 \* AB03B36A813205046B703CD0C1190C9B  
= 0x 2E6cff815C370F34399AA48FAA4798DD4D6E95DA42451AAC3DC4FE8A54DF4C0



# Producto $b^*c_B$

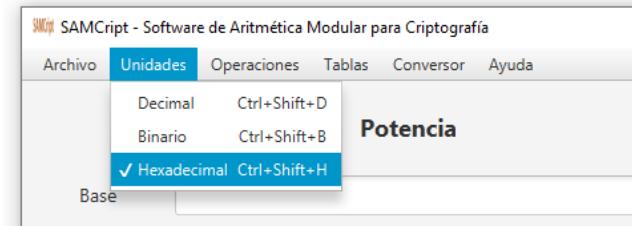
0x EC6F31FE6FC843205516845CA7D83BD2 \* 445260D4BAB0CE8ACB926DE32959B708  
= 0x 3F199E53C471E620A84C6829616380BAA5B4BFF85872A3722663CFD83F86FC90



# Claves compartidas con Cristina

- $k_{AC} = \alpha^{a*cA} \bmod p = k_{CA} = \alpha^{cA*a} \bmod p$ 
  - $k_{AC} = 0x 2^{2E6CFF815C370F34399AA48FAA4798DD4D6E95DA42451AAC3DC4FE8A54DF4C0} \bmod FAD913871CE8D63FC4B11640459629E1F293F9DF$
  - $k_{AC} = k_{CA} = 0x E920D263B470ED244A1F08D6C54D74515C2BF97B$
- $k_{BC} = \alpha^{b*cB} \bmod p = k_{CB} = \alpha^{cB*b} \bmod p$ 
  - $k_{BC} = 0x 2^{3F199E53C471E620A84C6829616380BAA5B4BFF85872A3722663CFD83F86FC90} \bmod FAD913871CE8D63FC4B11640459629E1F293F9DF$
  - $k_{BC} = k_{CB} = 0x C7AF2CFE5C9A2FFEFD4DEC562C2B1E85C202ABE0$

# Claves compartidas $k_{AC} = k_{CA}$ y $k_{BC} = k_{CB}$



The screenshot shows the SAMCrypt software window with the 'Operaciones' tab active, specifically the 'Potencia' section. The calculator displays the equation  $k_{BC} = k_{CB}$ . The historical log on the right side of the interface shows two entries:

- $2 ^{(3F199E53C471E620A84C6829616380BAA5B4BFF8587)} \bmod fad913871ce8d63fc4b11640459629e1f293f9df = C7AF2CFE5C9A2FFEF4DEC562C2B1E85C202ABE0$
- $2 ^{(2E6CFF815C370F34399AA48FAA4798DD4D6E95D4A2451AAC3DC4FE8A54DF4C0)} \bmod fad913871ce8d63fc4b11640459629e1f293f9df = E920D263B470ED244A1F08D6C54D74515C2BF97$

A button labeled 'Borrar Historial' is visible at the bottom right of the history panel.

# Conclusiones de la lección 10.3

- DH no es inmune al ataque conocido como man in the middle MITM
- El atacante C en el medio, puede usar un único valor secreto c para engañar a los usuarios A y B, o bien un valor secreto diferente para cada usuario:  $c_A$  y  $c_B$
- Los valores de las claves privadas de los intervenientes puede ser el resultado de alguna función hash (MD5 con 128 bits, SHA1 con 160 bits , SHA2 con 224, 256, 384 y 512 bits, SHA3 con 224, 256, 384 y 512 bits), lo que haría inviable su ataque por fuerza bruta, aunque esto no será cierto si se trata de ataques por paradoja del cumpleaños para hashes antiguos
- La vulnerabilidad del sistema está en los protocolos de red y no en la posibilidad de romper el sistema de cifrado en sí
- Frase acertada “*la criptografía no se ataca, se esquiva*” (Dr. Alfonso Muñoz)

# Lectura recomendada

- Preventing Man-In-The-Middle Attack in Diffie-Hellman Key Exchange Protocol; Aqeel Sahi, David Lai (2015)
  - [https://www.researchgate.net/publication/280722113 Preventing Man-In-The-Middle Attack in Diffie-Hellman Key Exchange Protocol](https://www.researchgate.net/publication/280722113)
- Man-in-the-Middle Public Wi-Fi Hacking Demo; Keatron Evans, Infosec instructor (2019)
  - <https://www.youtube.com/watch?v=wBVAZiaCda4>
- Diffie-Hellman (Man-in-the-middle); A security site
  - [https://asecuritysite.com/encryption/diffie\\_crack](https://asecuritysite.com/encryption/diffie_crack)

# Class4crypt c4c10.4

## Módulo 10. Criptografía asimétrica

### Lección 10.4. Algoritmo RSA

10.4.1. La historia del algoritmo RSA: el MIT y el GCHQ

10.4.2. Pasos para la generación de una clave RSA

10.4.3. Introducción al cifrado y descifrado con RSA

10.4.4. Fortaleza del algoritmo RSA

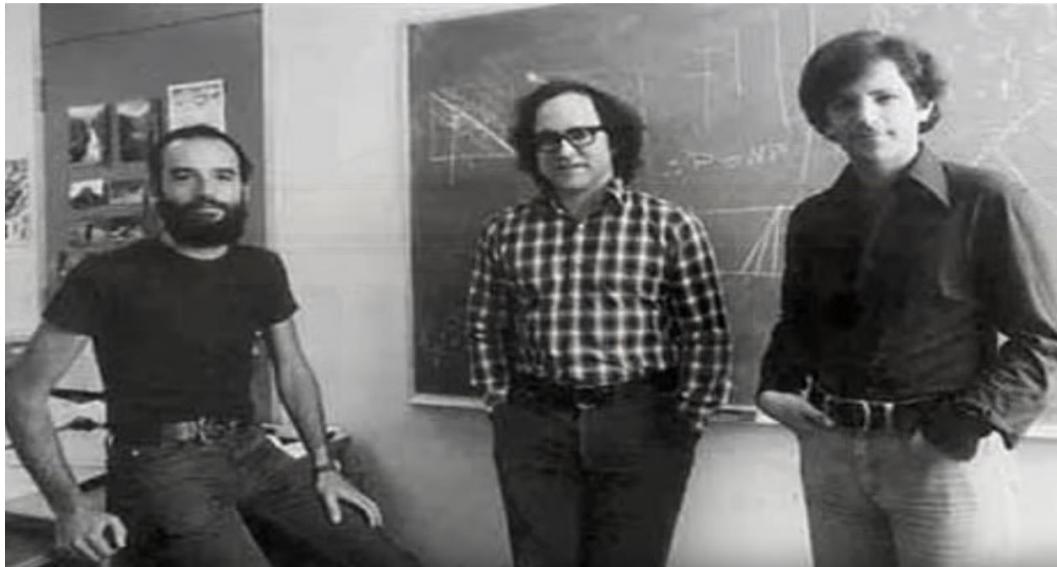
10.4.5. Importancia de los valores elegidos para los primos p y q y para la clave pública e

Class4crypt c4c10.4 Algoritmo RSA

<https://www.youtube.com/watch?v=w5dWeZwfK8w>

# Breve historia del algoritmo RSA

- RSA fue creado por Ron Rivest, Adi Shamir y Leonard Adleman, investigadores del MIT, en febrero de 1978, 15 meses después del intercambio de clave de Diffie y Hellman
- Es uno de los algoritmos de clave pública más populares, aunque va perdiendo protagonismo hoy en día en favor de los sistemas basados en criptografía con curvas elípticas



# La otra historia del algoritmo RSA



- No obstante, Clifford Cocks, un matemático británico que trabajaba para la agencia de inteligencia británica *Government Communications Headquarters* GCHQ, había descrito un sistema equivalente en un documento interno en 1973 (cinco años antes) que RSA
- Debido al elevado coste de las computadoras necesarias para implementarlo en la época, su idea no trascendió en su entorno
- Y, además, su descubrimiento no fue revelado hasta 1997 ya que se trataba de material confidencial (entorno militar), por lo que se supone que Rivest, Shamir y Adleman desarrollaron el algoritmo RSA de forma independiente

# Pasos en la generación de claves RSA (1)

## 1. Cada usuario elige dos números primos p y q distintos

Actualmente encontrar esos dos números primos **p** y **q** de 1.024 bits cada uno (o algo mayores), es una tarea muy sencilla y no debería llevarnos más de un segundo hacerlo

## 2. El grupo de cifra n será la multiplicación de esos números: $n = p \cdot q$

Si p y q tienen 1.024 bits =  $b_{1.023}b_{1.022}b_{1.021} \dots b_2b_1b_0$ , con el bit en posición 1.023 en 1 (set), entonces el módulo n será un número compuesto de  $2 \cdot 1.024$  bits = 2.048 bits. Obviamente **n** será impar, dado que p y q deben ser impares por ser primos, y que el primo 2 no se usa

## 3. Se hace público el valor de n y se guardan en secreto p y q

Se publica n y los primos p y q que sólo los conoce el dueño de la clave se guardan como secreto. En ese secreto va a residir el principio básico de la seguridad en RSA

# Pasos en la generación de claves RSA (2)

## 4. Cada usuario calcula $\phi(n) = (p-1)(q-1)$

Conocido como Función o Indicador de Euler, el valor  $\phi(n)$  será ligeramente menor que n, dado que p y q son números muy grandes de 1.024 bits cada uno, aproximadamente unos 310 dígitos decimales. El Indicador de Euler  $\phi(n)$  será siempre un número par, dado que es el resultado de la multiplicación de dos números pares,  $(p-1)$  y  $(q-1)$ , también de 2.048 bits.

## 5. Cada usuario elige su clave pública e

El valor de la clave pública **e** (de *encrypt*) debe cumplir con:

$$1 < e < \phi(n) \quad \text{mcd } [e, \phi(n)] = 1$$

El valor de la clave pública e debe ser primo relativo con  $\phi(n)$ , para que exista su inverso multiplicativo, que será la clave privada **d** (de *decrypt*)

# Pasos en la generación de claves RSA (3)

## 6. Elección del valor de la clave pública

Observa que la clave pública  $e$  debe ser obligatoriamente un número impar para que cumpla con la condición de existencia del inverso, esto es  $\text{mcd}[1, \phi(n)]$ , porque  $\phi(n)$  será siempre un número par. Es decir,  $e = 3, 5, 7, 9, 11, \dots \phi(n)-1$

Dado que a partir la clave pública  $e$  deduciremos la clave privada  $d$  en el grupo  $\phi(n)$ , por lógica de las matemáticas discretas al ser  $\phi(n)$  un número muy grande (de 2.048 bits), si la clave pública es un número pequeño, la clave privada será un número grande, y viceversa.

¿Qué valor usamos para  $e$ ? Primero que nada, la clave pública  $e$  deberá ser un valor bajo para que, por contrapartida, la clave privada  $d$  sea un valor muy alto, cercano a  $\phi(n)$ , y no sea adivinable por fuerza bruta. Pero, ¿puede ser cualquier número impar pequeño?

# Pasos en la generación de claves RSA (4)

## 7. El uso del valor 65.537 como clave pública e estándar

Si la clave pública es un valor muy pequeño ( $e = 3$ ) y la usan varios usuarios, cada uno con sus valores de  $n$  diferentes, podrían producirse ataques que desvelen el secreto cifrado

Para evitar este tipo de problemas, mundialmente se usa el valor  $F_4$  (número 4 de Fermat) como clave pública estándar e en RSA

$$F_4 = 2^{2^4} + 1 = 2^{16} + 1 = 65.537 \text{ (un número primo de 17 bits)}$$

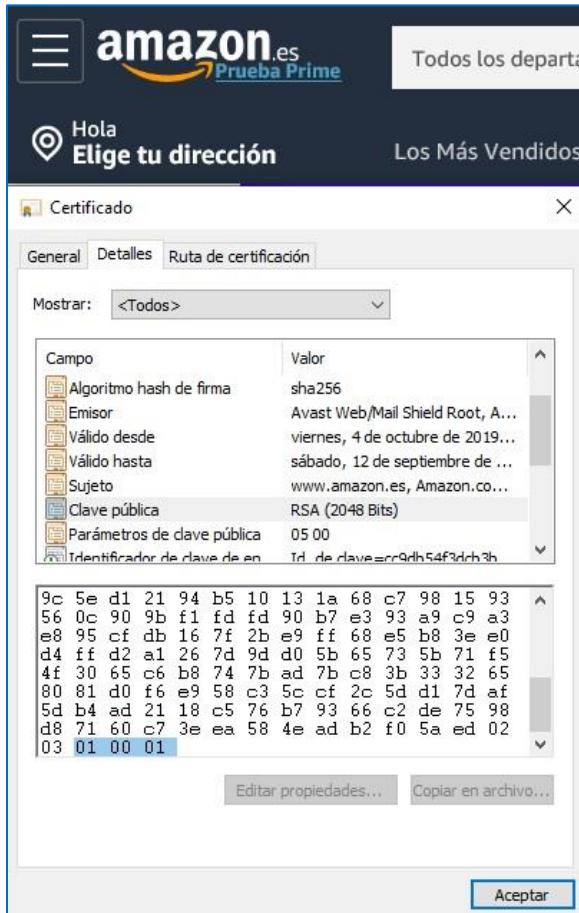
65.537 en hexadecimal representado con 3 bytes es 0x 010001

65.537 en binario es 1000000000000001 (interesante para potencia modular)

Si  $n$  tiene 2.048 bits, como  $e$  tiene solamente 17 bits, entonces la clave privada  $d$  será un número comprendido entre 2.048 bits [tamaño de  $\phi(n)$ ] y  $2.048 - 17$  bits = 2.031 bits

# Pasos en la generación de claves RSA (5)

Internet Explorer



Firefox

The screenshot shows a certificate details page in Firefox. The title bar says "Certificado". Below it, the URL "www.amazon.es" is shown, along with the chain of trust: "DigiCert Global CA G2" (selected) and "DigiCert Global Root G2".

The main content area displays the following certificate information:

**Nombre del asunto**: \_\_\_\_\_  
**País**: US

**Organización**: DigiCert Inc

**Nombre común**: DigiCert Global CA G2

**Nombre del emisor**: \_\_\_\_\_  
**País**: US

**Organización**: DigiCert Inc

**Unidad organizativa**: www.digicert.com

**Nombre común**: DigiCert Global Root G2

**Validez**: \_\_\_\_\_  
**No antes**: 1/8/2013 14:00:00 (hora estándar de Europa central)  
**No después**: 1/8/2028 14:00:00 (hora estándar de Europa central)

**Información de clave pública**: \_\_\_\_\_

**Algoritmo**: RSA

**Tamaño de la clave**: 2048

**Exponente**: 65537

**Módulo**: D3:48:7C:BE:F3:05:86:5D:5B:D5:2F:85:4E:4B:E0:86:AD:15:AC:61:CF:5B:AF:3E:6A:0A:47:FB:9A:76:91:60:0B:8A:6B:CD:CF:...

# Pasos en la generación de claves RSA (6)

## 8. Cálculo de la clave privada d

Con el Algoritmo Extendido de Euclides, cada usuario calcula su clave privada  $d$ , como el inverso de la pública  $e$  en  $\phi(n)$ , es decir  $d = \text{inv}[e, \phi(n)]$  y se cumple que  $e^*d \bmod \phi(n) = 1$

Observa que, al igual que sucedía con la clave pública  $e$ , la clave privada  $d$  será siempre un número impar, en este caso un valor muy cercano a 2.048 bits, el tamaño de la clave RSA

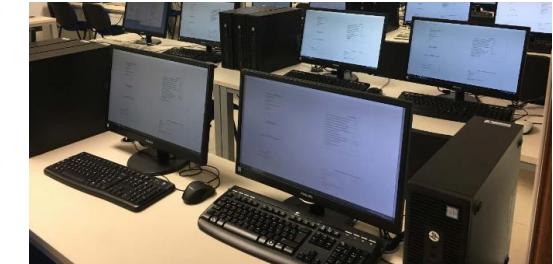
## 9. Publicación de la clave pública

Se hacen públicos los valores de  $n$  y  $e$ , es decir, la clave pública  $(n, e)$

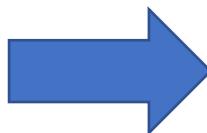
## 10. Guardado de la clave privada

Se mantiene en secreto la clave privada  $d$  y también los valores de los primos  $p$  y  $q$ , para usar el Teorema Chino del Resto en el descifrado y optimizar así el tiempo de cómputo

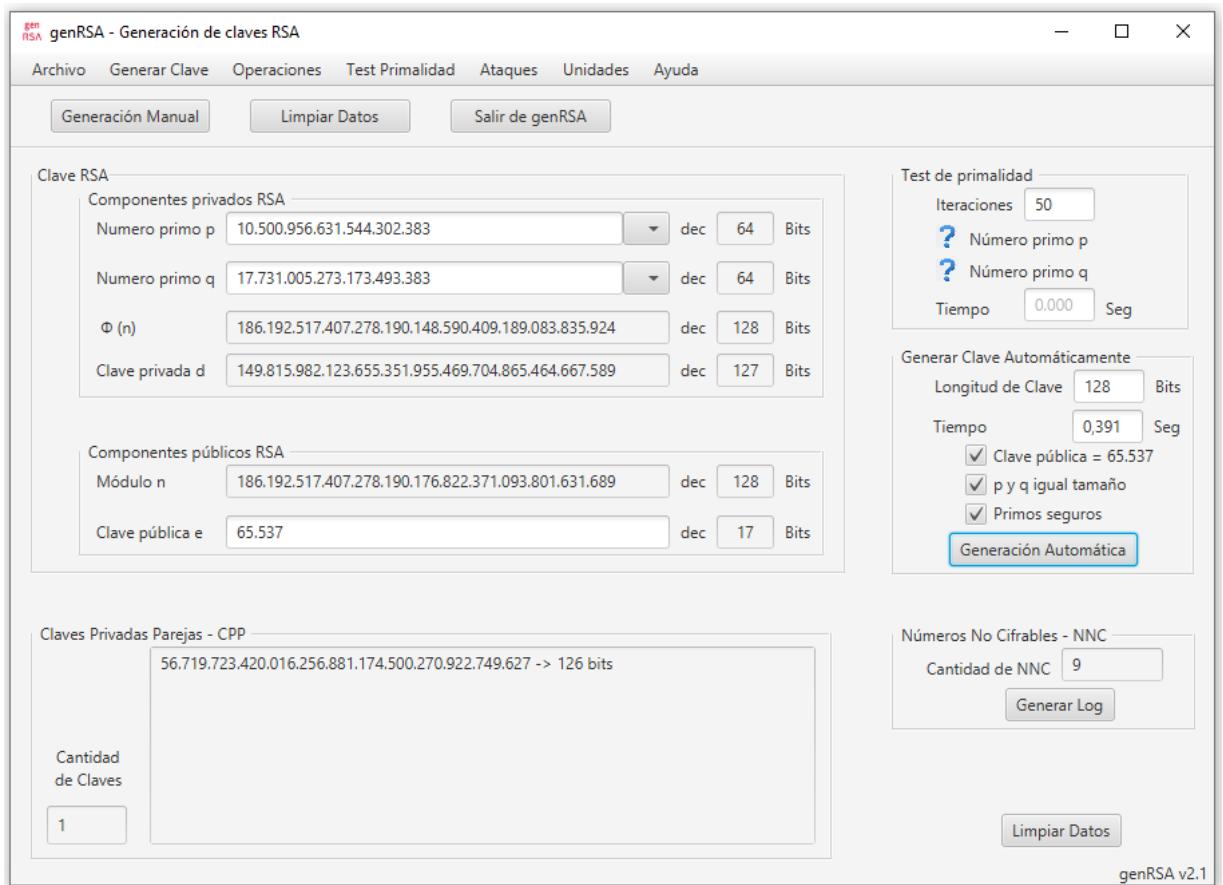
# Laboratorio de prácticas



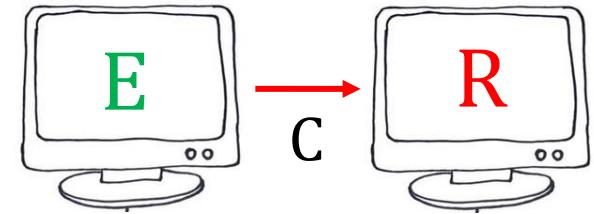
- Ejercicios prácticos de generación de claves RSA con el software genRSA v2.1 realizados en esta videoclase
- Manos a la obra...



Descarga de genRSA v2.1 (ejecutable Java):  
[https://www.criptored.es/software/sw\\_m001d.htm](https://www.criptored.es/software/sw_m001d.htm)

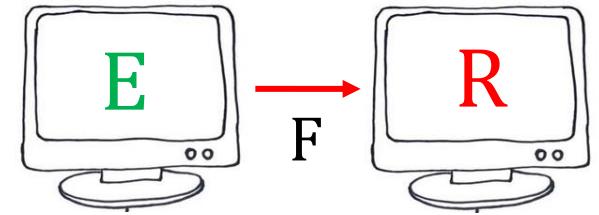


# Cifrado y descifrado con RSA (1)



- Cifrado con la clave pública del receptor (destino)
  - Sea el número secreto  $N$  y  $[e_R, n_R]$  las claves públicas del Receptor
  - El Emisor cifra  $N^{e_R} \text{ mod } n_R = C$  (criptograma que envía al receptor)
- Descifrado con la clave privada del receptor
  - El receptor descifrará el criptograma  $C$  con su clave privada  $d_R$
  - $C^{d_R} \text{ mod } n_R = N$  (texto en claro secreto que se recupera)
- Lo anterior es solo una pequeña introducción al cifrado con RSA; en una próxima lección se profundizará en estas dos operaciones

# Cifrado y descifrado con RSA (2)



- Firma digital con la clave privada de emisor (origen)
  - Sea el número a firmar N y  $d_E$  la clave privada del Emisor y módulo  $n_E$
  - $N^{d_E} \text{ mod } n_E = F$  (criptograma enviado como firma digital a Receptores)
- Comprobación de firma digital con la clave pública de emisor
  - Los receptores o destinatarios descifrarán el criptograma F con la clave pública  $[e_E, n_E]$  del emisor, que todos conocen
  - $F^{e_E} \text{ mod } n_E = N$  (se comprueba la firma al obtener N en el descifrado)
- También se profundizará en este apartado en una próxima lección

# La seguridad del algoritmo RSA

- La seguridad de RSA se basa en la dificultad computacional que supone resolver el Problema de la Factorización Entera PFE, que para valores cercanos a los mil bits se vuelve intratable con la capacidad de cómputo actual
- Por lo tanto, la fortaleza del algoritmo reside en la dificultad de factorizar  $n$  para obtener los primos  $p$  y  $q$ . El atacante, una vez haya encontrado  $p$  y  $q$ , podrá calcular  $\phi(n) = (p-1)(q-1)$  y le será muy fácil calcular la clave privada  $d$  a partir de la clave pública  $e$ , usando el Algoritmo Extendido de Euclides AEE para resolver  $d = \text{inv}[e, \phi(n)]$
- El cuerpo de cifra  $n$  típico actualmente sigue siendo de 2.048 bits, con primos  $p$  y  $q$  de 1.024 bits. Aunque en teoría sería interesante que  $p$  y  $q$  estuviesen separados en unos cuantos bits, para primos tan grandes no hace falta preocuparse de este requisito
- Hace un par de años la NSA sugería que se comenzaran a usar claves RSA de 3.072 bits

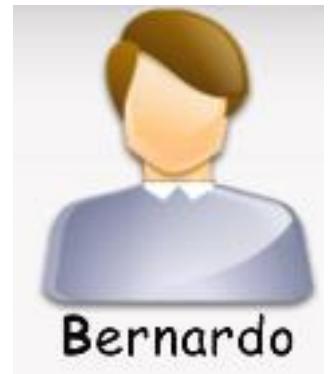
# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=CMe0COxZxb0>

# El origen de Alice y Bob

- En criptografía asimétrica, en lengua anglosajona es común indicar al usuario **A** como *Alice* y al usuario **B** como *Bob*; en español usamos Alicia y Bernardo
- Esto se lo debemos a Rivest, Shamir y Adleman, puesto que así lo hicieron ellos por primera vez en el artículo que daban a conocer su invento RSA
- “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems” de 1978
- <https://people.csail.mit.edu/rivest/Rsapaper.pdf>



# ¿Cómo guardamos la clave privada d?

- Habrá que cifrarla de forma local (cifrado convencional) con un algoritmo simétrico, por ejemplo con AES 256, para guardarla como un archivo en un dispositivo de almacenamiento (disco duro, pendrive, etc.)
- La clave de 256 bits del AES podría ser la función hash SHA256 de nuestra password, preferiblemente una passphrase “muy personal”, es decir que no sea adivinable, con 5 o más palabras, e.g: **La p@ss que NO conose nadie ad79313b39d841d88bd48c9c6a1c60ab7f7e34754c49925dfb173b99a0e000d5**
- Cada vez que se desee usar la clave privada d, desciframos el archivo con AES usando como clave el hash SHA256 de nuestra frase de paso secreta

# No cumplimiento de especificaciones (1)

- ¿Qué sucede si los valores de p o q (o ambos) no son primos?
  - Si por error los valores de p y q no son primos, algo que puede ocurrir al generar claves RSA con OpenSSL de menos de 512 bits (ver referencia), se podría generar una clave RSA, en principio correcta, pero que no permitirá descifrar con la clave privada lo que se haya cifrado con la clave pública. Es decir, aunque se cumpla con los principios de generación de una clave RSA, claves pública y privada correctas, no servirá como algoritmo de cifra

**Referencia.** ¿Son correctas las claves RSA que generamos con Win64 OpenSSL 1.1.0g?

Intentando cifrar con RSA cuando p y q no son primos”, Jorge Ramió, enero de 2018

[https://www.criptored.es/descarga/Win64OpenSSL1.1.0g\\_No\\_Genera\\_Claves\\_Correctas\\_Hasta\\_128\\_Bits.pdf](https://www.criptored.es/descarga/Win64OpenSSL1.1.0g_No_Genera_Claves_Correctas_Hasta_128_Bits.pdf)

# No cumplimiento de especificaciones (2)

- ¿Qué sucede si elegimos una clave pública e demasiado pequeña?
  - Si 3 usuarios usan como clave pública  $e = 3$  teniendo cada uno diferentes módulos de cifra  $n$ , y los 3 reciben el mismo secreto cifrado, un atacante puede descubrir ese secreto sin necesidad de conocer los primos  $p$  y  $q$  ni las claves privadas de esos usuarios
  - Otra anomalía es que si el número a cifrar fuese muy pequeño, por ejemplo el resultado decimal de la palabra ESPIA =  $\text{ASCII}_2 = 01000101\ 01010011\ 01010000\ 01001001\ 01000001 = 297.750.513.985 = N$ , al elevarlo a la clave pública  $e = 3$  queda  $C = 297.750.513.985^3 = 26.397.181.561.727.876.626.570.836.776.946.625$ , un número de solamente 35 dígitos (115 bits), que al reducirlo módulo 2.048 bits nos dará el mismo número  $N$ . Así, se podría descifrar simplemente obteniendo la raíz cúbica de ese criptograma  $C$ . Para evitar esto, en RSA se usará un relleno denominado OAEP Optimal Asymmetric Encryption Padding

# En videoclases futuras de RSA veremos

- Estándares PKCS#1 y versiones
- El uso de la función de Carmichael  $\lambda(n) = \text{mcm}[(p-1), (q-1)]$  en vez de la función de Euler  $\phi(n)$  para el cálculo de la clave privada d
- Generación de claves con OpenSSL y comprobación de los valores que se usan para el descifrado con el teorema chino de los restos
- Particularidades y curiosidades de las claves y de la cifra RSA
- Ataques por clave pública pequeña, por factorización de n, por cifrado cíclico, por la paradoja del cumpleaños y por canal lateral

# Conclusiones de la lección 10.4

- Rivest, Shamir y Adleman inventan RSA en 1978, 15 meses después del protocolo de intercambio de clave de Diffie y Hellman. En 1973 Clifford Cocks hizo algo similar
- A diferencia de DH, RSA permite el intercambio de clave y también la firma digital
- Los primos p y q del grupo de cifra  $n = p * q$  deben ser hoy como mínimo de 1.024 bits
- Se usa un grupo trampa  $\phi(n)$  o  $\lambda(n)$  que sólo puede calcular quien conoce p y q
- La clave pública e debe ser un número pequeño, que cumpla la condición del inverso multiplicativo en  $\phi(n)$  o en  $\lambda(n)$ , pero no demasiado pequeño para evitar ataques
- Se usa de forma estándar como clave pública e el número 4 de Fermat  $F_4 = 65.537$
- La clave privada se calcula mediante el algoritmo extendido de Euclides
- Como la clave pública es pequeña, la clave privada tendrá una magnitud cercana a n
- La seguridad de RSA se basa en la dificultad computacional de encontrar p y q

# Lectura recomendada (1/2)

- MOOC Crypt4you, El algoritmo RSA, Jorge Ramió, 2012
  - Lección 0. Introducción al curso
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion0/leccion00.html>
  - Lección 1. Los principios del algoritmo RSA
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion1/leccion01.html>
  - Lección 2. Valores de diseño de las claves
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion2/leccion02.html>
- Curso de Criptografía Aplicada, Jorge Ramió, 2018
  - Tema 5 Algoritmos de cifra asimétrica, apartado 5.4 El algoritmo RSA, página 105 y siguientes
  - <https://www.criptored.es/descarga/CursoCriptografiaAplicada2018.pdf>

# Lectura recomendada (2/2)

- RSA Algorithm, DI Management Services, Australia, manual online
  - [https://www.di-mgt.com.au/rsa\\_alg.html](https://www.di-mgt.com.au/rsa_alg.html)
- Guion píldora formativa Thoth nº 39, “Cómo funciona el algoritmo RSA”, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora039.pdf>

# Class4crypt c4c10.5

## Módulo 10. Cifra asimétrica

### Lección 10.5. Generación de claves RSA y estándar PKCS#1

10.5.1. Recordando el algoritmo RSA

10.5.2. Generación de una clave RSA manualmente sin ayuda de software

10.5.3. Generación de claves RSA con el software genRSA v2.1

10.5.4. Estándar en RSA con indicador de Euler y función de Carmichael

10.5.5. Generación de claves RSA con OpenSSL

Class4crypt c4c10.5 Generación de claves RSA y estándar PKCS#1  
<https://www.youtube.com/watch?v=AS66q6ykLCs>

# Recordando el algoritmo RSA

- Se eligen dos números primos p y q distintos [Mínimo 1.024 bits]
- Se multiplican estos primos obteniendo  $n = p * q$  [Grupo de cifra]
- Se calcula  $\phi(n) = (p-1)(q-1)$  [Grupo trampa]
- Se busca una clave pública e tal que  $\text{mcd}(e, \phi(n)) = 1$  [Estándar F4]
- Se calcula la clave privada  $d = \text{inv}(e, \phi(n))$  [Tamaño  $\approx \phi(n)$ ]
- Se hacen públicos n y e [Clave pública del sujeto]
- Se guarda en secreto d [Clave privada del sujeto]
- Se recomienda guardar en secreto p y q [Teorema Chino del Resto]
- Cifrado:  $C = N^{clave} \bmod n$    Descifrado:  $N = C^{invclave} \bmod n$  [Inversos]

# Generación manual de una clave RSA

Generar manualmente una clave RSA con los dos primos más grandes de tres dígitos y una clave pública e la menor posible. Usar <https://primes.utm.edu/lists/small/10000.txt>

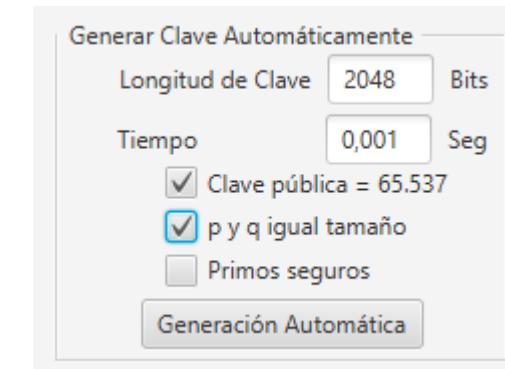
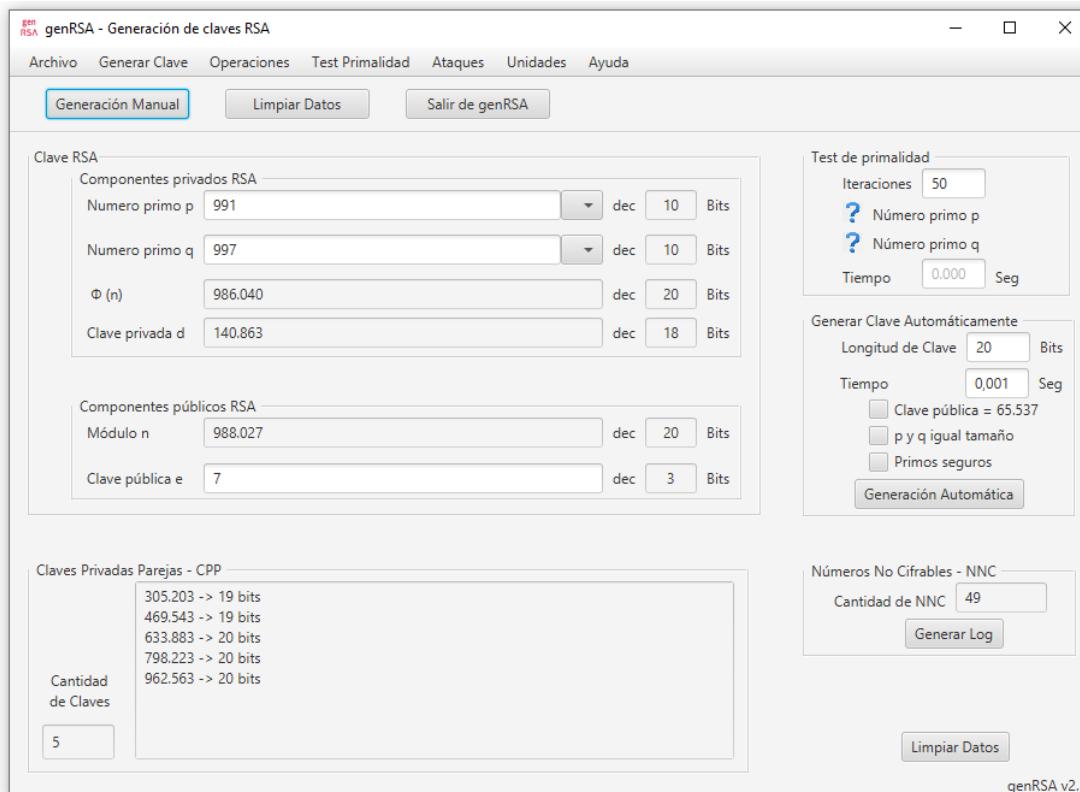
- Primos:  $p = 991$  y  $q = 997$
- Entonces  $n = 991 * 997 = 988.027$
- $\phi(n) = (p-1)(q-1) = 990 * 996 = 986.040$
- Búsqueda de la clave pública e:
  - ¿3? MCD (3, 986.040) = 3 (no válida)
  - ¿5? MCD (5, 986.040) = 5 (no válida)
  - ¿7? MCD (7, 986.040) = 1 (válida)
- Clave pública  $e = 7$ ,  $n = 988.027$
- Clave privada  $d = \text{inv}(7, 986.040) = 140.863$

| i | $y_i$   | $g_i$   | $u_i$ | $v_i$    |
|---|---------|---------|-------|----------|
| 0 | -       | 986.040 | 1     | 0        |
| 1 | -       | 7       | 0     | 1        |
| 2 | 140.862 | 6       | 1     | -140.862 |
| 3 | 1       | 1       | -1    | 140.863  |
| 4 | 6       | 0       | 7     | -986.040 |



# Generación de claves RSA con genRSA v2.1

- Generación manual y automática de claves con números decimales y hexadecimales
- Un paseo práctico por el software genRSA v2.1 generando claves (no cifrado ni ataques)



[https://www.criptored.es/software/sw\\_m001d.htm](https://www.criptored.es/software/sw_m001d.htm)

# Estándar RSA, el caso de $\phi(n)$ y $\lambda(n)$

|         | Versión | Nombre   |
|---------|---------|--|
| PKCS#1  | 2.1     | Estándar criptográfico RSA   |
| PKCS#2  | -       | Obsoleto   |
| PKCS#3  | 1.4     | Estándar de intercambio de claves Diffie-Hellman                                   |
| PKCS#4  | -       | Obsoleto   |
| PKCS#5  | 2.1     | Estándar de cifrado basado en contraseñas  |
| PKCS#6  | 1.5     | Estándar de sintaxis de certificados extendidos                                    |
| PKCS#7  | 1.5     | Estándar sobre la sintaxis del mensaje criptográfico                               |
| PKCS#8  | 1.2     | Estándar sobre la sintaxis de la información de clave privada                      |
| PKCS#9  | 2.0     | Tipos de atributos seleccionados   |
| PKCS#10 | 1.7     | Estándar de solicitud de certificación   |
| PKCS#11 | 2.40    | Interfaz de dispositivo criptográfico ("Cryptographic Token Interface" o cryptoki) |
| PKCS#12 | 1.1     | Estándar de sintaxis de intercambio de información personal                        |
| PKCS#13 | -       | Estándar de criptografía de curva elíptica   |
| PKCS#14 | -       | Generación de número pseudo-aleatorios   |
| PKCS#15 | 1.1     | Estándar de formato de información de dispositivo criptográfico                    |

Public-Key Cryptography Standards

[Docs] [txt|pdf] [draft-moriarty-...] [Tracker] [Diff1] [Diff2] [Errata]

INFORMATIONAL  
Errata Exist

Internet Engineering Task Force (IETF)  
Request for Comments: 8017  
Obsoletes: 3447  
Category: Informational  
ISSN: 2070-1721

K. Moriarty, Ed.  
EMC Corporation  
B. Kaliski  
Verisign  
J. Jonsson  
Subset AB  
A. Rusch  
RSA  
November 2016

PKCS #1: RSA Cryptography Specifications Version 2.2

- Paper Rivest, Shamir, Adleman (*Euler totient function*)
  - Grupo trampa  $\phi(n) = (p-1)(q-1)$
- PKCS#1 v2.2 (*Carmichael function*)
  - Grupo trampa  $\lambda(n) = \text{lcm } (p-1, q-1)$
- ¿Tiene alguna ventaja? 

# Carmichael $\lambda(n)$ versus Euler $\phi(n)$

- $\lambda(n)$  siempre será más pequeño que  $\phi(n)$
- De hecho,  $\lambda(n)$  es un divisor de  $\phi(n)$
- La clave privada  $d_\lambda$  saldrá muchas veces como la primera clave privada pareja, menor que la clave privada  $d$ , usando  $\phi(n)$
- De todas las claves privadas parejas,  $d_\lambda$  será siempre la clave más pequeña. Es decir, aquella que cumple con el primer valor de la ecuación de las claves privadas parejas
- $d_\lambda = \text{inv} [e, \lambda(n)]$  con  $\lambda(n) = \text{mcm} [(p-1), (q-1)]$
- La clave privada  $d_\lambda$  será siempre la menor posible
- Pero, ¿esto tiene alguna ventaja real de cara a la cifra? 

# Uso de $\lambda(n)$ en vez de $\phi(n)$ en inversos

- $\phi(n) = (p-1)*(q-1)$   $d_1 = \text{inv } [e, \phi(n)]$  Ejemplo claves de 32 bits
- $\lambda(n) = \text{mcm } (p-1, q-1)$   $d_2 = \text{inv } [e, \lambda(n)]$  Con  $e = 65.537$
- Y para claves reales de 2.048 bits, ¿qué sucede al usar  $\lambda(n)$  en vez de  $\phi(n)$ ?

| Clave   | p      | q      | $\phi(n)$     | $d_1$         | $\lambda(n)$  | $d_2$         |
|---|--------|--------|---------------|---------------|---------------|---------------|
| Caso en que la clave privada $d_2$ calculada con $\lambda(n)$ es menor que la clave privada $d_1$ calculada con $\phi(n)$ |        |        |               |               |               |               |
| Clave1  | 41.957 | 63.421 | 2.660.849.520 | 1.119.727.313 | 665.212.380   | 454.514.933   |
| Clave2  | 39.019 | 57.373 | 2.238.540.696 | 422.794.097   | 53.298.588    | 49.703.981    |
| Clave3  | 46.679 | 55.619 | 2.596.137.004 | 1.597.129.129 | 1.298.068.502 | 299.060.627   |
| Clave4  | 52.321 | 64.091 | 3.353.188.800 | 3.248.659.073 | 335.318.880   | 230.789.153   |
| Clave5  | 52.889 | 63.419 | 3.354.051.184 | 1.935.397.617 | 1.677.025.592 | 258.372.025   |
| Clave6  | 52.919 | 57.587 | 3.047.335.948 | 1.889.909.053 | 1.523.667.974 | 366.241.079   |
| Caso en que la clave privada $d_2$ calculada con $\lambda(n)$ es igual que la clave privada $d_1$ calculada con $\phi(n)$ |        |        |               |               |               |               |
| Clave7  | 49.711 | 63.587 | 3.160.860.060 | 980.615.633   | 1.580.430.030 | 980.615.633   |
| Clave8  | 62.939 | 53.783 | 3.384.931.516 | 56.297.593    | 1.692.465.758 | 56.297.593    |
| Clave9  | 55.619 | 51.599 | 2.869.777.564 | 1.286.335.745 | 1.434.888.782 | 1.286.335.745 |
| Clave10   | 36.467 | 62.303 | 2.271.904.732 | 267.136.089   | 1.135.952.366 | 267.136.089   |
| Clave11   | 58.067 | 63.587 | 3.692.184.676 | 466.586.409   | 1.846.092.338 | 466.586.409   |
| Clave12   | 40.841 | 64.151 | 2.619.886.000 | 148.789.473   | 261.988.600   | 148.789.473   |



genRSA v2.1

SAMCrypt

# Generación de claves RSA con OpenSSL

```
C:\OpenSSL-Win64\bin>openssl genrsa -out MiClaveRSA 2048
Generating RSA private key, 2048 bit long modulus
.....+ ++
e is 65537 (0x0100001)

C:\OpenSSL-Win64\bin>openssl rsa -in MiClaveRSA -text -o
ut MiClaveRSATexto
writing RSA key

C:\OpenSSL-Win64\bin>type MiClaveRSATexto
Private-Key: (2048 bit)
modulus:
 00:b6:a5:6c:3c:bd:a7:a4:38:b2:7e:30:51:df:81:
 74:ee:bd:ed:2b:67:50:45:25:97:ae:0d:b7:31:72:
c9:d3:bd:f6:82:46:17:e0:de:08:bc:ed:6b:df:
d5:a7:ef:b6:27:12:4f:fa:f7:9f:b8:09:2e:ea:75:
d6:02:8:a4:d4:9e:d2:75:id:c9:95:75:c9:cb:6e:
f6:ea:be:1c:ab:89:98:bb:ef:04:64:7e:58:e8:5:
a7:29:1c:2f:0d:8d:f5:59:06:86:81:74:d8:b4:4b:
c9:3f:aa:5c:a7:34:ed:68:fc:43:eb:30:d8:e5:83:
06:e3:8c:cd:fd:78:e0:78:3e:cc:84:b6:91:dd:
d4:11:8:a4:b7:fd:77:al:1b:7a:b7:5e:2c:67:2b:
3d:23:0d:b0:ca:0e:dc:4f:43:a4:15:25:43:d3:1d:
d5:b4:12:09:a0:30:f5:59:06:86:81:74:d8:b4:4b:
2c:4a:3d:53:74:94:67:f0:2a:fa:7:f1:98:7e:30:
cd:9:a:ad:89:13:c5:e2:a8:11:cb:83:9:c2:b8:79:
57:f3:d6:c1:21:93:33:c2:78:30:98:fb:99:ce:
b2:08:a7:ad:08:e0:1f:63:5c:e6:9:f:3f:65:dc:52:
54:7c:e8:1b:98:8:e:1f:9b:78:48:49:0:e:30:22:93:
43:7b
publicExponent: 65537 (0x010001)
privateExponent:
 00:8:e6:e6:c4:a5:74:d7:9c:34:ab:5e:8:a:f0:60:
 8d:68:ad:d8:fe:78:9:a:e9:3:fc:6a:05:f9:2b:0b:
30:b6:26:25:34:bc:3:a:c0:cc:73:e0:22:51:70:71:
74:ed:2c:34:4c:65:9b:86:34:70:16:c1:b8:9:f:61:
0a:cc:7:a:45:74:c5:be:cc:3:b3:9e:19:5e:4b:40:8b:
d5:4c:dc:e9:f9:72:8:f7:fc:ee:33:9c:c4:a2:
c0:a2:fd:96:bb:8:a:16:48:4d:3e:b7:d3:00:2a:30:
91:88:ea:76:0:b:ac:fl:83:8:e:bc:65:2:b:96:c8:1b:
b8:2:c:19:6:e:5e:89:d:66:a4:e5:e2:3f:5:e:5a:
ab:24:58:44:a2:e7:20:22:18:b8:55:2d:3d:fe:9c:
32:90:50:b6:b2:0:f:9:a:08:9:b:b8:62:0:f:bb:70:8f:
12:0:b:ae:d8:86:f1:58:69:10:43:78:22:22:1:e:e5:
7d:73:d4:6:f:46:35:79:b8:ed:92:3:b:0:e:31:c5:00:
1b:73:15:24:12:f7:0:f:59:54:6:a:8:c:3:a:8:b:30:43:
22:70:20:42:b2:06:db:a4:d9:5e:08:8:f:1:a:al:
50:3:e:80:a9:da:f3:f7:07:5:b:98:d3:5:f:1:f:e6:b9:
7:e:f0:6:a:15:36:21:65:c8:66:49:f4:d9:cb:3:a:d0:
5d:61
```

```
prime1:
 00:dd:58:bc:a9:08:1d:ed:03:87:93:af:2a:9a:47:
d1:5e:b2:0:a:f0:5e:89:72:64:98:d3:f1:59:40:1e:
93:db:97:3d:b5:0b:5b:f8:02:c:a1:b1:26:aa:67:e4:
68:d6:68:1e:9d:02:a5:f9:f9:8:f:c3:fa:05:0f:db:
42:2d:37:ad:ac:c4:82:3f:01:c0:27:20:0:c:a2:7f:
11:c9:71:bd:69:fc:c5:ca:80:c1:80:8b:70:47:3a:
30:5c:06:63:2:a:4:b:83:fd:78:6d:43:49:62:a4:4b:
1a:40:cd:d9:23:6f:ad:67:9d:9b:ef:53:a2:a9:9b:
c7:bf:ca:b9:e3:66:3:b:f7:e7
```

```
prime2:
 00:d3:3d:a0:4b:20:6a:71:22:7b:ae:c4:96:0:f:49:
de:3:c8:5:e:24:78:2:c:eb:69:b4:b6:7d:55:64:78:
ca:75:67:ff:18:65:02:e5:52:df:56:fc:eb:79:67:
8f:67:75:3:f:26:f7:99:ca:1:f:8:a:9d:6:c1:3:a:f9:
0d:e2:82:f3:50:fa:a2:d5:09:6:f3:f9:8:a:11:a:f:
eb:c2:3d:c8:34:db:2:e:db:23:4:e:d5:2:c:28:aa:e0:
6:c4:fc:59:5b:44:d0:55:25:c:bd:86:94:01:7:c:
7f:98:a9:3:a:83:le:7:e:46:3:a:e5:7:f:40:f0:d5:59:
28:0b:43:7:f7:74:8:d:b6:55:4d
```

```
exponent1:
 00:8:a:d5:27:b0:15:c9:3:f:6:a:21:55:7:e:5:f:08:59:
d7:76:bf:d1:a0:a6:8:b:2:b:56:f1:8:c:ae:2:a:be:ce:
ef:0:a:a2:b8:2:f:a:0:2:a:55:65:19:d1:d6:14:b8:09:
49:c0:81:9:a:01:fd:31:41:b3:48:7b:31:8:c:0:f:3:f:
85:28:7:b:6:d:1:99:87:e:0:21:26:f:6:e:3:61:49:62:
ce:33:3d:c6:02:7:e:6:b:48:f:2:ad:6:c:05:09:ad:83:
4:f:5:e:c:93:d6:69:7:b:26:36:ac:c1:b1:7:b:99:83:
fd:b9:7:b:e:3:a:0:18:6:f:9:45:a:0:12:e:2:b:0:7:a:15:
9d:9:a:2:c:7:b:9:f:a:7:f:6:21:2d
```

```
exponent2:
 05:f:2:65:31:d:8:3:e:c6:e2:96:a2:cb:ec:01:90:2d:
ce:7:e:41:90:3:b:45:e:0:76:c:0:5:c:a:6:d:8:a:1:94:78:
f6:4:7:2:f:da:4:a:f5:b2:e:9:0:b:5:a:92:93:3:a:7:f:bc:
dc:c7:c3:38:60:9:e:16:bf:9:e:92:61:fe:6:a:9:f:9a:
51:77:28:4:c:6:e:51:e:9:37:73:cc:65:89:0:a:88:6:
55:b:0:91:93:53:5:c:55:55:00:22:26:35:1:a:e:2:68:
8:a:0:f:e:2:a:0:1:c:26:d:5:4:e:d:0:f:6:90:f:4:57:ba:5:e:
89:38:b5:5:e:ba:9:c:0:97:74:1:d:03:e:1:54:ce:5:c:cd:
30:cb:fc:a:9:96:57:bc:bd
```

```
coefficient:
 0:a:c5:0:e:21:2:e:1:e:21:84:83:c:0:35:89:e:0:b4:f:5:
d4:a:2:5:c:1:d:d9:25:28:bf:27:73:67:ce:b8:4:e:16:
0d:61:21:21:f7:7b:34:c:0:c:8:39:48:c:2:ec:40:9:b:a:
ef:aa:aa:09:5d:2:d:be:d6:d5:62:58:e:5:16:36:48:
1d:57:01:98:46:61:1d:94:86:88:d:2:cf:8:b:8:f:4:c:
5:c:94:c:5:43:e:2:e:7:a:c:f:e4:e:7:c:1:f:fc:50:26:
a:0:41:42:7:d:00:c:5:g:52:24:be:f:8:0:d:85:c:7:80:
cf:a:0:16:0:0:31:d:3:23:41:c:4:f:4:0:2:dc:37:8:a:c:2:
24:2d:f:6:1:c:19:0:7:0:3:69
```



Comandos

```
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAtqVsPLDnpDiyfjBR34F07r3tK2dQRSWXrg23MXLJ071vgkYX
6N7QCLzta9/Vp++2JxJP+vEfAu6nXWAoqk1J7SdR3J1XXJy2726r4cq4mYu+8M
ZH5Y6MnKRwv0I31WQaGgXTYtEvJP6pcpzTtaPxD6zDY5YMG44zc/ed4Hg+zIS4
kd3UEYpIt/13oRt6t14sZys9Iw2wyg7cT0OkFSVD0x3VtBIJoDCbbriy2mDzu1cs
Sj1TdJRn8Cr45/GYfjDNmq2JE8XiqBHLg5zSuH1X89bBIRGTm8J4MJj7mc6yCKet
CQgfY1zmnz913FJUFogbmI4fm3hISQ4wIpNDewIDAQABoIBAQCO5ubEpXTxNdsr
XorwYI1ordj+eJrpPvxqBfkrCz2JiU0vDrAzHPgIlFwcxtLDRMzzuGNHAWwbif
YQrMekv0xb7MoZ4ZKtAi9Vm3On5nHKPt/zum5zEosCi/Za7ihZITT630wAqMjGI
6nyLrPGDjrxlK5bIG7gsGSWe5idzqTf14j9eWqsKWEsi5yAiGLhVLT3+nDKQuLay
D5oIm7hiD7twjx1Lrt8VnpEEEN4iIle5X1z1G+mNxM47Z17DjHFABtzFSQS9w9Z
VGqmOosWQyJwIEKyBtuknV4Ij9EaoVa+gChw5jTxx/muX7wahU2IWX1Zkn0
2cs60F1hAoGBAN1YVkk1He0Dh5OvKppH0V6yCvBeiXJkmNpxWUAek9uXPbULW/gs
obEmqmfpkaNZoHp0Cpfn5j8P6BQ/bq0i03razEgj8BwCcgDKJ/EclxvWn8xcqAwYCL
cEc6MFwGyypLg/14bUNJYk5LGkDN2SNvrWedm+9Toqmbx7/KueNmO/fnAoGBANM9
oEsganEie67Elg9J3jzIXiR4LoptLz9VWR4ynVn/xhlAuVS31b8631nj2d1Pyb3
mcOfip1swTr5DeKC81D6otUjb/P5ihGv68I9yDtBLtsjTtUsKKrgbEr8WVtE0FUl
XL2GlAF8f5ipo4MefkY65X9I8NV2KAtDf3SNt1VNAoGBAIrVJ7AVyT9qIVV+Xwhz
13a/0AcmyitW8YyuKr70wqiuC+gKlv1GR3WFNgJScCBmgH9MUGzSHsxjI8/hSh
bdGZh+AhJvbjYUlijzM9xgj+a0:jyjRwWpD2DT+Xik9zpey2rMGxe5mD/b1746AY
Yf1f0BLisHoVnZose5+n9iEtAoGABfJLMdg+xuKwosvsAZAtzn5BkDtF4HbAXKbY
oZR49qcv2kr1sukLWpKSoN+83MfDOGceFr+ekmH+ap+aUXcoTG5R6TdzGWJvCohu
VbCRk1NcVVUAIiY1GuJoig/ioBwm1U7Q9pD0V7peiTi1XrqcCXQdA+FUz1zNMV8
qZZXvL0CgYAKxQ4hLh4hhIPANYngtF/Uolwd2SUovydzZ864ThYNYSHXezTAyDlI
wuxAm6nvqqoJXdK+1tViWOUWNkgdVwGYRmEd1IaI0s+l+j0xclMVD4ud6z+TnwU/8
UCagQUJ9AMVtUiS++A2Fx4DPqBYAMdMjQcT0Atw3isIkLfYcGQcDaQ==
-----END RSA PRIVATE KEY-----
```

# Comandos OpenSSL generación clave RSA

- El programa se instala por defecto en C:\OpenSSL-Win64>
- C:\OpenSSL-Win64\bin>`openssl genrsa -out MiClaveRSA 2048`
- C:\OpenSSL-Win64\bin>`openssl rsa -in MiClaveRSA -text -out MiClaveRSATexto`
- Si no deseas incluir en el archivo la clave privada, agrega la opción –noout
- Edita MiClaveRSATexto con WordPad:
  - Elimina todos los “:”
  - Elimina todos los “cuatro espacios en blanco” o sangría
  - Deja los valores de prime1 (p) y prime2 (q) en una sola fila cada uno
- Pega esos valores de p y de q en genRSA v2.1 (Unidades en hexadecimal)
- Introduce como clave pública e = 10001 (valor de F4 en hexadecimal)
- Genera esa clave con “Generación Manual”
- Comprueba que los valores del módulo n y de la clave privada d son los mismos

# Captura de pantalla de genRSA v2.1

```
Private-Key (2048 bit)

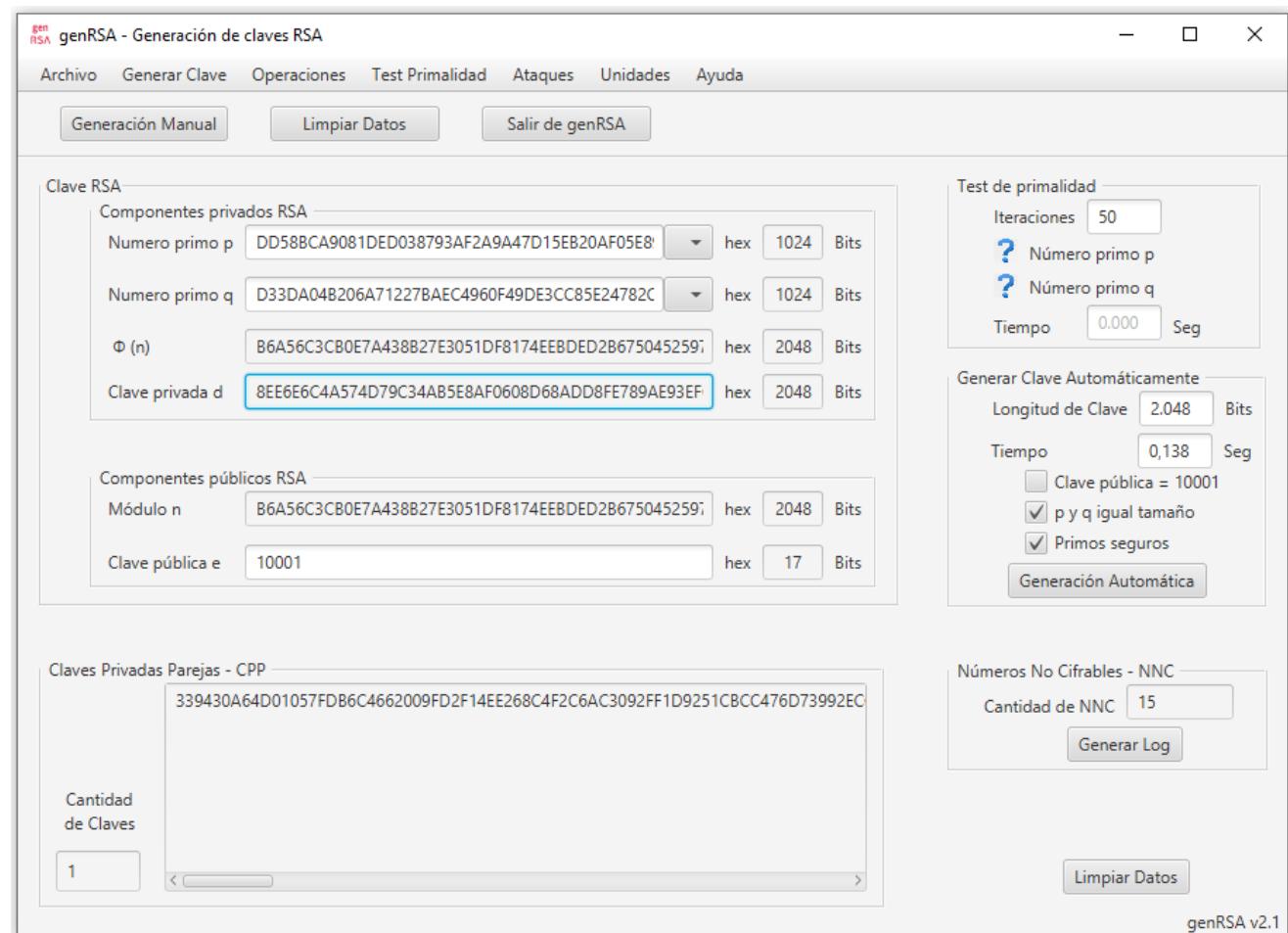
modulus
00b6a56c3cb0e7a438b27e3051df8174eebded2b6750452597ae0db73172c9d3b
d6f824617e8ded008bc0ed6bdfd5a7efb627124ffaf79fb8092eea75d6028aa4d4
9ed2751dc99575c9cb6ef6eabe1cab8998bbef0c647e58e8c5a7291c2fd08df55
90686174d8b44bc93faa5ca734ed68fc43eb30d8e58306e38cdcfde778e0783e
cc84b891ddd4118a48b7fd77a11b7ab75e2c672b3d230db0ca0edc4f43a415254
3d31dd5b41209a0309b6eb8b2da60f3bb572c4a3d53749467f02af8e7f1987e30
cd9aad8913c5e2a811cb839cd2b87957f3d6c121119333c2783098fb99ceb208a
7ad08e81f635ce69f3f65dc52547ce81b988e1f9b7848490e302293437b

publicExponent 65537 (0x10001)

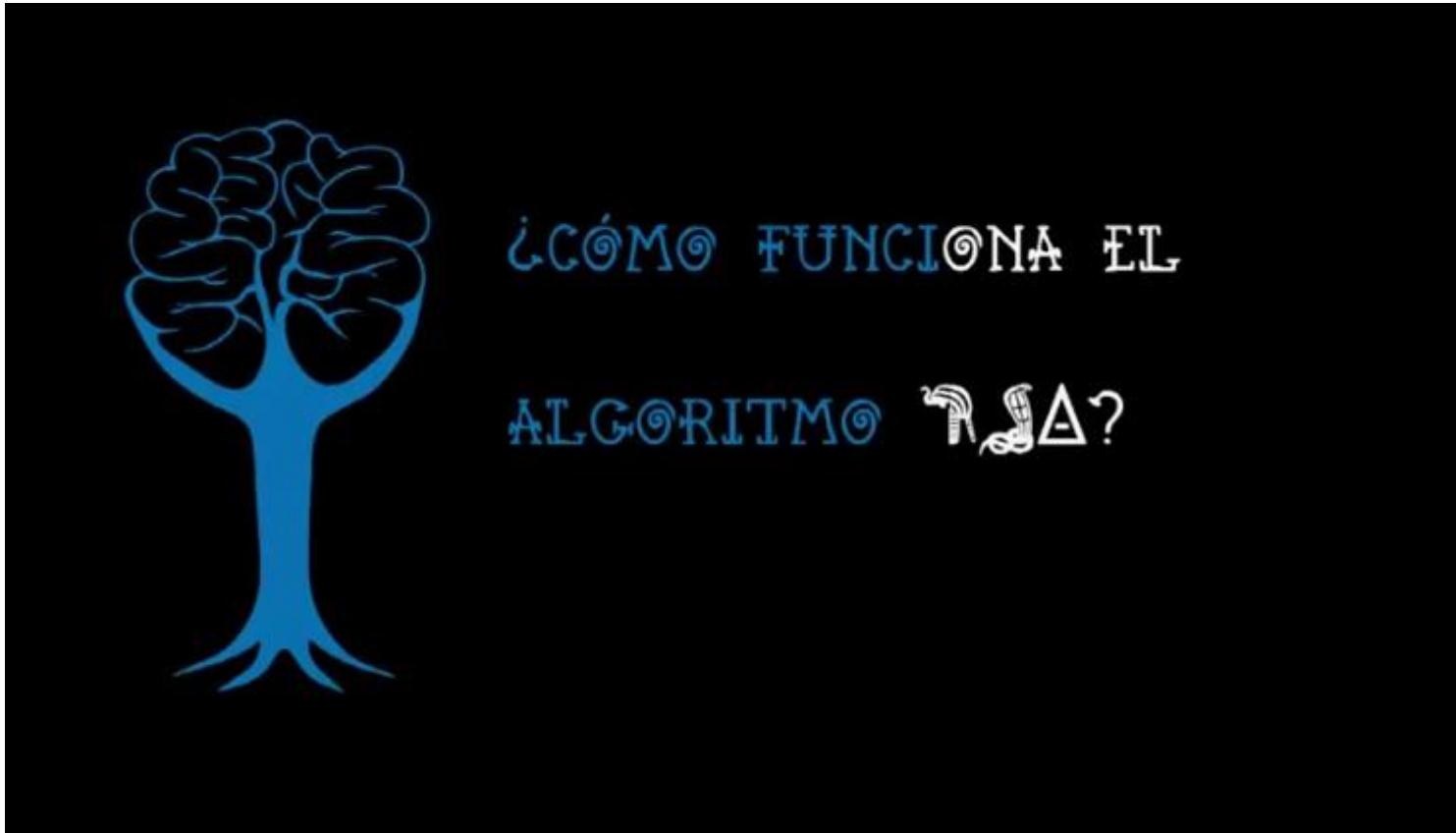
privateExponent
008ee6e6c4a574d79c34ab5e8af0608d68add8fe789ae93efc6a05f92b0b30b62
62534bc3ac0cc73e02251707174ed2c344c659b86347016c1b89f610acc7a4574
c5becc3b3e195e4b408bd54cdce9f99c728fb7fce339cc4a2c0a2fd96bb8a164
84d3eb7d3002a309188ea760bacf1838ebc652b96c81bb82c196e5e89d89d66a4
e5e23f5e5aab245844a2e7202218b8552d3dfe9c329050b6b20f9a089bb8620fb
b708f120baed886f158691043782221ee57d73d46fa63579b8ed923b0e31c500
1b73152412f70f59546a8c3a8b304322702042b206dba49d5e088fd11aa1503e8
0a9daf3f7075b98d35f1fe6b97ef06a15362165c86649f4d9cb3ad05d61

prime1
00dd58bca9081ded038793af2a9a47d15eb20af05e89726498d3f159401e93db9
73db50b5bf82ca1b126aa67e468d6681e9d02a5f9f98fc3fa050fdb422d37adac
c4823f01c027200ca27f11c971bd69fcc5ca80c1808b70473a305c06632a4b83f
d786d4349624e4b1a40cdd9236fad679d9bef53a2a99bc7bfcab9e3663bf7e7

prime2
00d33da04b206a71227baec4960f49de3cc85e24782ceb69b4b67d556478ca756
7ff186502e552df56fceb79678f67753f26f799ca1f8a9d6cc13af90de282f350
faa2d5096ff3f98a11afebc23dc834db2edb234ed52c28aae06c4afc595b44d05
5255cbd8694017c7f98a9a3831e7e463ae57f48f0d559280b437f748db6554d
```



# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=CMe0COxZxb0>

# Conclusiones de la lección 10.5

- En la generación de claves RSA, podemos usar dos grupos trampa para el cálculo de la clave privada  $d$ , conociendo la clave pública  $e$ 
  - El indicador de Euler  $\phi(n) = (p-1)(q-1)$   $d_\phi = \text{inv } [e, \phi(n)]$
  - La función de Carmichael  $\lambda(n) = \text{lcm } (p-1, q-1)$   $d_\lambda = \text{inv } [e, \lambda(n)]$
- En ambos casos, la clave generada tendrá al menos una clave privada pareja
- Cuando se elige usar Euler, la clave privada  $d$  podrá ser cualquiera de esas claves privadas parejas. Cuando se elige usar Carmichael, se asegura que la clave privada  $d$  es la más pequeña posible de todas ellas
- El estándar mundial de RSA es el PKCS#1, versión 2.2
- Generar una clave RSA actual de 2.048, bits implica un cómputo de menos de un segundo. Si usamos primos seguros este tiempo puede tardar mucho más

# Lectura extra recomendada (1/2)

- A Method for Obtaining Digital Signatures and Public-Key Cryptosystems, R.L. Rivest, A. Shamir, L. Adleman
  - <https://people.csail.mit.edu/rivest/Rsapaper.pdf>
- PKCS Wikipedia
  - <https://en.wikipedia.org/wiki/PKCS>
- PKCS#1: RSA Cryptography Specifications Version 2.2
  - <https://tools.ietf.org/html/rfc8017>
- Main purpose for Carmichael's Function in RSA, Crypto Stack Exchange
  - <https://crypto.stackexchange.com/questions/54280/main-purpose-for-carmichaels-function-in-rsa>

# Lectura extra recomendada (2/2)

- El algoritmo RSA, Lección 7 Generación de claves con OpenSSL, MOOC Crypt4you, Jorge Ramió, 2012
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion7/leccion07.html>
- OpenSSL command cheatsheet
  - <https://www.freecodecamp.org/news/openssl-command-cheatsheet-b441be1e8c4a/#0507>
- Generate OpenSSL RSA Key Pair from the Command Line, Frank Rietta, 2019
  - <https://rietta.com/blog/openssl-generating-rsa-key-from-command/>
- Guion píldora formativa Thoth nº 39, “Cómo funciona el algoritmo RSA”, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora039.pdf>

# Class4crypt c4c10.6a

## Módulo 10. Cifra asimétrica

### Lección 10.6a. Cifrado y descifrado con RSA (parte 1)

10.6a.1. Recordando el algoritmo RSA

10.6a.2. Aquí ciframos números

10.6a.3. ¿Por qué podemos usar como exponente la clave inversa para descifrar?

10.6a.4. Operaciones simples de cifra y firma de números usando RSA

Class4crypt c4c10.6a Cifrado y descifrado con RSA (parte 1)  
<https://www.youtube.com/watch?v=SzIqUXeqAG0>

# Recordando el algoritmo RSA

- Se eligen dos números primos p y q distintos Mínimo 1.024 bits
- Se multiplican estos primos obteniendo  $n = p * q$  Grupo de cifra
- Se calcula  $\phi(n) = (p-1)(q-1)$  o  $\lambda(n) = \text{mcm}(p-1, q-1)$  Grupo trampa
- Se busca una clave pública e tal que  $\text{mcd}[e, \phi(n)] = 1$  Estándar F4
- Se calcula la clave privada  $d = \text{inv}(e, \phi(n))$  o  $\text{inv}[e, \lambda(n)]$  Tamaño  $\approx \phi(n)$
- Se hacen públicos n y e Clave pública del sujeto
- Se guarda en secreto d Clave privada del sujeto
- Se recomienda guardar en secreto p y q Teorema Chino del Resto
- Cifrado:  $C = N^{clave} \bmod n$  Descifrado:  $N = C^{invclave} \bmod n$  Claves inversas

# La criptografía asimétrica cifra números

- La operación característica de la cifra asimétrica es mediante un cifrado exponencial modular
- La operación a realizar será  $C = N^x \text{ mod } n$ , en donde n es el módulo de cifra, hoy en día de 2.048 bits, x será una clave pública del receptor de 17 bits para el intercambio de un número secreto N, o bien una clave privada del emisor de unos 2.048 bits para la firma digital de un hash N, y C el criptograma resultante de la operación de cifra o firma
- N será siempre un resto del módulo n, por lo general de unas centenas de bits. Si se desea cifrar texto, se hará sólo como ejercicio de laboratorio y con objetivos docentes
- Esto es así porque este tipo de cifra es muy lenta (comparada con la simétrica) y sería muy costoso en tiempo cifrar mensajes o documentos con decenas o cientos de Megabytes
- Por lo tanto, cuando se cifre con la clave pública del receptor para hacer un intercambio de clave, se tratará de un número N por ejemplo de 256 bits (una clave de sesión para una cifra simétrica), y cuando se cifre con la clave privada del emisor para lograr una firma digital, se tratará de un número N de 256 bits, por ejemplo el hash SHA-2 de un documento M

# ¿Por qué se descifra con la clave inversa? 1

Como  $C = N^e \text{ mod } n$  y  $N = C^d \text{ mod } n$ , entonces  $(N^e)^d \text{ mod } n = N = N^{e*d} \text{ mod } n$  

Sabemos que  $e*d \text{ mod } \phi(n) = 1$

Es decir:  $e*d = k*\phi(n) + 1$  

Por el Teorema de Euler:

$N^{k\phi(n)} \text{ mod } n = 1$ , para todo  $N$  coprimo con  $n$ . Entonces:

$$\begin{aligned} N^{e*d} \text{ mod } n &= N^{k\phi(n)+1} \text{ mod } n \\ &= N^{k\phi(n)} * N \text{ mod } n = 1 * N = N \end{aligned}$$

Y recuperamos el secreto

Si no fuesen coprimos, esto no nos serviría para descifrar...



Pero...

Por el Teorema Chino del Resto se tiene que:

$N^{e*d} \text{ mod } n = N$ , si y solo si

$N^{e*d} \text{ mod } p = N$

$N^{e*d} \text{ mod } q = N$



Luego, el sistema de cifra será válido para cualquier valor de  $N$

# Comprobación teorema de Euler y TCR

Sean:  $p = 7, q = 13, n = 91, \phi(n) = 6 * 12 = 72$

$$\phi(n) + 1 = 73$$

$$N^{k\phi(n)} \bmod n = 1$$



$$N^{e*d} \bmod n = N^{k\phi(n)+1} \bmod n = N$$

Porque  $N^{e*d} \bmod p = N$  y  $N^{e*d} \bmod q = N$

Potencia

|  |     |
|--|-----|
| Base                                       | 27  |
| Exponente                                  | 72  |
| <input checked="" type="checkbox"/> Módulo | 91  |
| Resultado                                  | 1 ← |

Potencia

|  |      |
|--|------|
| Base                                       | 27   |
| Exponente                                  | 73   |
| <input checked="" type="checkbox"/> Módulo | 91   |
| Resultado                                  | 27 ← |

# ¿Por qué se descifra con la clave inversa? 2

Si  $n = p \cdot q = 23 \cdot 41 = 943$  entonces  $\phi(n) = (23-1)(41-1) = 880$

Sea el número  $N = 756 = 2^2 \cdot 3^3 \cdot 7$  (un resto de  $n$ )

Se elige  $e = 3$ , entonces  $d = \text{inv}[e, \phi(n)] = \text{inv}(3, 880) = 587$

Se cumple que  $e \cdot d \bmod \phi(n) = 3 \cdot 587 \bmod 880 = 1761 \bmod 880 = 1 \quad [[2\phi(n) + 1]]$

$C = N^e \bmod n = 756^3 \bmod 943 = (2^2 \cdot 3^3 \cdot 7)^3 \bmod 943$

$C = [(2^2)^3 \bmod 943 * (3^3)^3 \bmod 943 * (7)^3 \bmod 943] \bmod 943 \dots$  entonces

$N = C^d \bmod n = \{[(2^2)^3 \bmod 943 * (3^3)^3 \bmod 943 * (7)^3 \bmod 943] \bmod 943\}^{587} \bmod 943$

$N = [(2^2)^{3*587} \bmod 943 * (3^3)^{3*587} \bmod 943 * (7)^{3*587} \bmod 943] \bmod 943$

$N = [(2^2)^{2\phi(n)+1} * (3^3)^{2\phi(n)+1} * (7)^{2\phi(n)+1}] \bmod 943$

Por los teoremas de Euler y Chino del Resto:  $N = 2^2 \cdot 3^3 \cdot 7 \bmod 943 = 756$

# Operaciones de cifrado y descifrado RSA

genRSA v2.1 

- Cifrado y descifrado con confidencialidad

- $C = N^{e_R} \text{ mod } n_R$        $N = C^{d_R} \text{ mod } n_R$      $(e_R, n_R)$  clave pública;  $d_R$  clave privada (Receptor)
- Con la clave privada de recepción se descifra  $C$  y se produce un intercambio de clave. Como tanto  $C$  como la clave privada  $d$  serán números muy grandes, de magnitud similar al módulo  $n$ , se recomienda realizar este descifrado usando el Teorema Chino del Resto
- Si deseamos cifrar textos con RSA, se deben formar bloques... se verá más adelante

- Cifrado y descifrado con integridad (y autenticidad)

- $C = N^{d_E} \text{ mod } n_E$        $N = C^{e_E} \text{ mod } n_E$      $(e_E, n_E)$  clave pública;  $d_E$  clave privada (Emisor)
- Con la clave pública del emisor se descifra  $C$  y se comprueba una firma digital
- Cifrado y descifrado convencional o local (confidencialidad) No se usa por baja velocidad
  - $C = N^{e_E} \text{ mod } n_E$        $N = C^{d_E} \text{ mod } n_E$      $(e_E, n_E)$  clave pública;  $d_E$  clave privada (Emisor)

# genRSA v2.1: generación de clave RSA

Clave RSA

| Componentes privados RSA |            |     |         |
|--------------------------|------------|-----|---------|
| Numero primo p           | 2.789      | dec | 12 Bits |
| Numero primo q           | 3.779      | dec | 12 Bits |
| $\Phi(n)$                | 10.533.064 | dec | 24 Bits |
| Clave privada d          | 7.022.043  | dec | 23 Bits |

| Componentes públicos RSA |            |     |         |
|--------------------------|------------|-----|---------|
| Módulo n                 | 10.539.631 | dec | 24 Bits |
| Clave pública e          | 3          | dec | 2 Bits  |

# genRSA v2.1: cifrado RSA y descifrado

**Cifrado - Clave Pública Destinatario**

Datos para el Cifrado

|                  |            |
|------------------|------------|
| Módulo           | 10.539.631 |
| Clave Pública    | 3          |
| Datos Originales | 2.001      |

Tienen texto los Datos Originales

Resultados del Cifrado

|                |           |
|----------------|-----------|
| Datos Cifrados | 1.886.441 |
|----------------|-----------|

**Cifrar Datos** **Información** **Limpiar Datos**

**Descifrado - Clave Privada Destinatario**

Datos para el Descifrado

|                |            |
|----------------|------------|
| Módulo         | 10.539.631 |
| Clave Privada  | 7.022.043  |
| Datos Cifrados | 1.886.441  |

Tienen texto los Datos Originales

Resultados del Descifrado

|                   |       |
|-------------------|-------|
| Datos Descifrados | 2.001 |
|-------------------|-------|

**Descifrar Datos** **Información** **Limpiar Datos**

# genRSA v2.1: firma RSA y comprobación

**Firma - Clave Privada Emisor**

Datos para la Firma

|  |   |
|--|---|
| Clave Privada  | <input type="text" value="7.022.043"/>  |
| Módulo   | <input type="text" value="10.539.631"/> |
| Datos Originales   |   |
| <input type="checkbox"/> Tienen texto los Datos Originales |   |

Resultados de la Firma

|                |  |
|----------------|--|
| Datos Firmados | <input type="text" value="4.294.038"/> |
|----------------|--|

**Validar firma - Clave Pública Emisor**

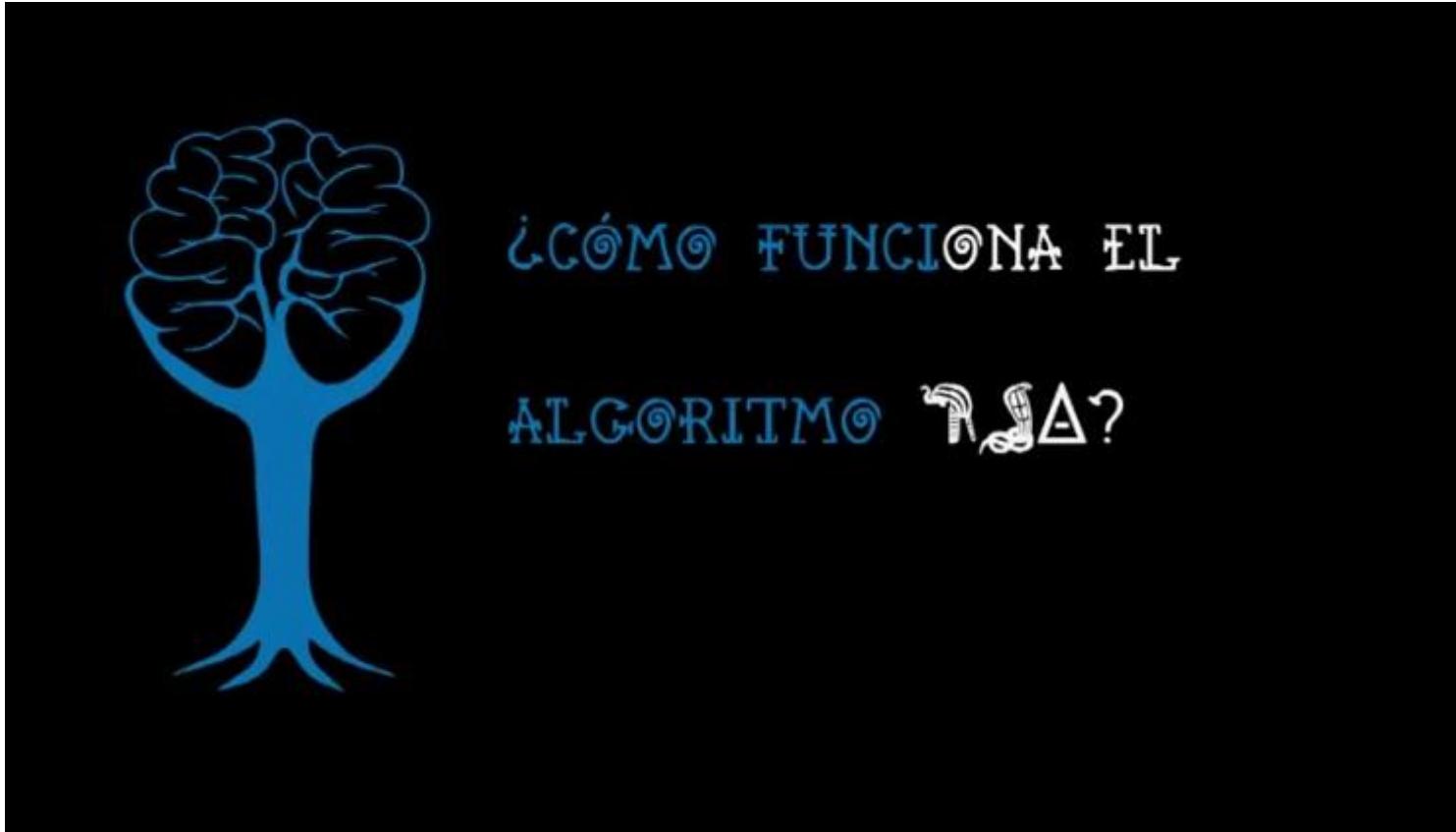
Datos para validar la Firma

|  |   |
|--|---|
| Clave Pública  | <input type="text" value="3"/>          |
| Módulo   | <input type="text" value="10.539.631"/> |
| Datos Firmados   |   |
| <input type="checkbox"/> Tienen texto los Datos Originales |   |

Resultados de la validación

|                 |                                    |
|-----------------|------------------------------------|
| Datos Validados | <input type="text" value="2.001"/> |
|-----------------|------------------------------------|

# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=CMe0COxZxb0>

# Conclusiones de la lección 10.6a

- En criptografía asimétrica real sólo se cifran números, no mensajes
- Usando el Teorema de Euler y el Teorema Chino de los Restos, podemos comprobar por qué en RSA se usa como operación de descifrado la misma que la de cifrado, pero cambiando la clave pública e por la clave privada d
- Las operaciones de cifra con RSA pueden ser:
  - Intercambio de una clave K  $C = K^{e_R} \text{ mod } n_R$  ( $R = \text{Receptor}$ )
  - Firma digital sobre un hash  $h(M)$   $C = h(M)^{d_E} \text{ mod } n_E$  ( $E = \text{Emisor}$ )
  - Cifrado convencional o local de  $M$   $C = M^{e_E} \text{ mod } n_E$  ( $E = \text{Emisor}$ , sin sentido)
- ¿Con RSA podremos cifrar (confidencialidad) y firmar (integridad) de forma simultánea? Sí, lo veremos en la clase Class4crypt10.6b
- ¿Con RSA podremos cifrar textos? Sí, lo veremos en la clase Class4crypt10.6b

# Lectura recomendada (1/2)

- El algoritmo RSA, Lección 6 RSA y el teorema chino del resto, MOOC Crypt4you, Jorge Ramió, 2012
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion6/leccion06.html>
- El algoritmo RSA, Lección 3 Cifrado de números y mensajes, MOOC Crypt4you, Jorge Ramió, 2012
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion3/leccion03.html>
- Guion píldora formativa Thoth nº 39, “Cómo funciona el algoritmo RSA”, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora039.pdf>

# Lectura recomendada (2/2)

- RSA Algorithm, David Ireland, DI Management, 2018
  - [https://www.di-mgt.com.au/rsa\\_alg.html#encryption](https://www.di-mgt.com.au/rsa_alg.html#encryption)
- Simple RSA key generation, A security site, Bill Buchanan
  - <https://asecuritysite.com/encryption/rsa>

# Class4crypt c4c10.6b

## Módulo 10. Cifra asimétrica

### Lección 10.6b. Cifrado y descifrado con RSA (parte 2)

10.6b.1. Recordando la analogía de los candados: el orden en el cifrado y descifrado es muy importante

10.6b.2. Opciones de cifra y firma simultáneas con RSA

10.6b.3. Cifrando texto o bloques de texto con RSA

Class4crypt c4c10.6b Cifrado y descifrado con RSA (parte 2)  
[https://www.youtube.com/watch?v=Abe7YAO\\_qKM](https://www.youtube.com/watch?v=Abe7YAO_qKM)

# ¿Sigue el problema del orden en la cifra?

- En Class4cryptc4c10.1 “Criptografía asimétrica y la analogía de los candados”, teníamos un problema: no se recuperaba el mensaje. Veamos qué sucedería ahora si Alicia y Bernardo utilizan RSA con sus claves pública y privada en este símil de los candados.
- Alicia tiene la siguiente clave RSA:  $n = 11 \cdot 13 = 143$ ,  $e = 23$ ,  $d = 47$
- Bernardo tiene la siguiente clave RSA:  $n = 7 \cdot 19 = 133$ ,  $e = 11$ ,  $d = 59$
- Alicia desea enviar el secreto 51 a Bernardo usando el protocolo de la caja y los candados
  - **Alicia** usa su clave pública (cierra su candado) y envía:  $51^{23} \text{ mod } 143 = 90$
  - **Bernardo** usa su clave pública (cierra su candado) y envía:  $90^{11} \text{ mod } 133 = 13$
  - **Alicia** usa su clave privada (abre su candado) y obtiene:  $13^{47} \text{ mod } 143 = 117$
  - **Bernardo** usa su clave privada (abre su candado) y obtiene:  $117^{59} \text{ mod } 133 = 129$
- El secreto ha viajado seguro ... ¡pero no se ha recuperado en recepción! ¿Por qué?

# Deshacer lo cifrado en el mismo orden

- Si se realiza una cifra con la clave pública de destino, deberá volver a cifrarse (descifrado) con la clave privada de destino, y viceversa. Luego las operaciones de doble cifrado como la del ejemplo de los candados deben hacerse en un orden y en el mismo grupo de cifra.
- Supongamos que Alicia desea enviar el secreto 75 a Bernardo:
- Usará en ese caso las claves públicas de Bernardo:  $75^{11} \text{ mod } 133 = 94$
- Y Bernardo descifrará con su clave privada:  $94^{59} \text{ mod } 133 = 75$
- Alicia también podría desear enviar a Bernardo el valor 75 ahora firmado:
- Así, Alicia cifra con su clave privada y su módulo:  $75^{47} \text{ mod } 143 = 4$
- Y Bernardo descifrará con las claves públicas de Alicia:  $4^{23} \text{ mod } 143 = 75$
- Se pueden hacer las dos cosas a la vez, pero en orden. Es decir, si en emisión primero se cifra y después se firma, en recepción primero se comprueba la firma y luego se recupera el secreto. O bien al revés. Desde el punto de vista teórico ¿qué sería mejor o más eficiente?

# Caso 1: primero se firma y después se cifra

- En emisión 1º se firma y 2º se cifra  $C = (M^{d_A} \text{ mod}_{n_A})^{e_B} \text{ mod}_{n_B}$
- En recepción 1º se descifra y 2º se comprueba la firma  $M = (C^{d_B} \text{ mod}_{n_B})^{e_A} \text{ mod}_{n_A}$
- Sea el mensaje secreto 11

Claves de Alicia:  $p_A = 11, q_A = 47, n_A = 517, e_A = 17, d_A = 433$

Claves de Bernardo:  $p_B = 23, q_B = 83, n_B = 1.909, e_B = 17, d_B = 849$

- Alicia. Firma en  $n_A$  y cifra en  $n_B$ :
  - Firma:  $11^{433} \text{ mod } 517 = 374$  Cifra:  $374^{17} \text{ mod } 1.909 = 1.898$
- Bernardo. Descifra en  $n_B$  y comprueba firma en  $n_A$ :
  - $1.898^{849} \text{ mod } 1.909 = 374$   $374^{17} \text{ mod } 517 = 11$
- Se recupera el secreto 11 (todo bien)

# Caso 2: primero se cifra y después se firma

- En emisión 1º se cifra y 2º se firma  $C = (M^{e_B} \text{ mod}_{n_B})^{d_A} \text{ mod}_{n_A}$
- En recepción 1º se comprueba la firma y 2º se descifra  $M = (C^{e_A} \text{ mod}_{n_A})^{d_B} \text{ mod}_{n_B}$
- Sea el mensaje secreto 11

Claves de Alicia:  $p_A = 11, q_A = 47, n_A = 517, e_A = 17, d_A = 433$

Claves de Bernardo:  $p_B = 23, q_B = 83, n_B = 1.909, e_B = 17, d_B = 849$

- Alicia. Cifra en  $n_B$  y firma en  $n_A$ :
  - Cifra:  $11^{17} \text{ mod } 1.909 = 934$  Firma:  $934^{433} \text{ mod } 517 = 87$  (*¡cuidado!*)
- Bernardo. Comprueba firma en  $n_A$  y descifra en  $n_B$ :
  - $87^{17} \text{ mod } 517 = 417$   $417^{849} \text{ mod } 1.909 = 1.182$
- No se ha recuperado el secreto 11. ¿Por qué?

# Sólo pueden cifrarse restos del módulo

- En la segunda operación en emisión Alicia ha firmado el valor **934**, un número mayor que su grupo de cifra **517** ( $934^{433} \bmod 517 = 87$ ) y eso no se debe hacer. En el fondo, Alicia ha firmado el valor  $934 \bmod 517 = 417$ , pues 934 era una clase de equivalencia en 517, que lógicamente es otro valor
- Si se desea continuar el protocolo, como Alicia ha firmado un elemento fuera de su grupo de cifra, deberá indicar a Bernardo cuántas veces k su grupo  $n_A$  ha entrado en el número que firmó, en este caso 1 veces ( $934/517$ ). Bernardo al resultado que llegue al comprobar esa firma, deberá agregarle  $1*n_A = 517$
- Bernardo calcula  $87^{17} \bmod 517 = 417$  (la clase de equivalencia que Alicia firmó) y le añade (en este caso) 517, obteniendo ( $417 + 517 = 934$ )
- Finalmente Bernardo recupera la cifra calculando  $934^{849} \bmod 1.909 = 11$

# Cifra y posterior firma con $n_B \gg n_A$

- Sea el mensaje 51

Claves Alicia:  $p_A = 43, q_A = 59, n_A = 2.537, e_A = 5, d_A = 1.949$

Claves Bernardo:  $p_B = 167, q_B = 233, n_B = 38.911, e_B = 3, d_B = 25.675$

- Cifra:  $51^3 \bmod 38.911 = 15.918$
- Firma:  $15.918^{1.949} \bmod 2.537 = 2.189$
- Pero Alicia en realidad ha firmado  $15.918 \bmod 2.537 = 696$ . Entonces Alicia debe indicarle a Bernardo que ha firmado un número 6 veces mayor que su grupo, puesto que  $15.918/2.537 = 6,2743397713835238470634607804493$ .
- Bernardo comprueba la firma  $2.189^5 \bmod 2.537 = 696$  (lógicamente). A ese valor deberá sumarle  $6*2.537 = 15.222$  y obtiene el valor real **15.918** ... etc.

# Cifra de textos con RSA

- Aunque no sea lo habitual, también se podrá cifrar mensajes con RSA, codificando previamente los caracteres del mensaje con números, como un ejercicio de laboratorio sólo con fines docentes
- Si la codificación numérica (deberá usarse siempre el código ASCII) del texto en claro fuese mayor que el tamaño del módulo, deberemos formar bloques cuyo tamaño fuese menor al módulo de cifra, para cifrar siempre restos
- En el caso de elegir tamaños óptimos del bloque a cifrar, se tiene que:
  - Si ciframos en dígitos, el bloque deberá tener un dígito menos que  $n$  
  - Si ciframos en bytes, el bloque deberá tener un byte menos que  $n$  
  - Si ciframos en bits, el bloque deberá tener un bit menos que  $n$

# Armando bloques de dígitos

- Si  $p = 89$  y  $q = 127$ , el grupo de cifra será  $n = 11.303$  y el grupo trampa será  $\phi(n) = 11.088$ : Si elegimos  $e = 25$ , se obtiene  $d = 10.201$
- Sea el mensaje ASCII  $\text{léame} = 108 \ 233 \ \underline{097} \ 115 \ 101$  (bloques de 3 dígitos)
- Como  $n = 11.303$  son 5 dígitos, cifraremos bloques de 4 dígitos
- Bloques de texto en claro:  $1082 \ 3309 \ 7115 \ \underline{0101}$  (cuatro bloques de 4 dígitos)
- Cada bloque (restos de  $n = 11.303$ ) se cifrará de forma independiente, por ejemplo  $\text{Bloque}^e \bmod n = \text{Bloque}^{25} \bmod 11.303$

# Cifrando y descifrando bloques de dígitos

- Texto en claro: 1.082; 3.309; 7.115; 101
- Clave RSA: e = 25, d = 10.201; n = 11.303

## Descifrado

$$4.625^{10.201} \bmod 11.303 = 1.082$$

$$8.991^{10.201} \bmod 11.303 = 3.309$$

$$11.161^{10.201} \bmod 11.303 = 7.115$$

$$1.970^{10.201} \bmod 11.303 = 101$$



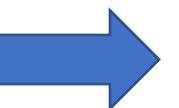
## Cifrado

$$1.082^{25} \bmod 11.303 = 4.652$$

$$3.309^{25} \bmod 11.303 = 8.991$$

$$7.115^{25} \bmod 11.303 = 11.161$$

$$101^{25} \bmod 11.303 = 1.970$$

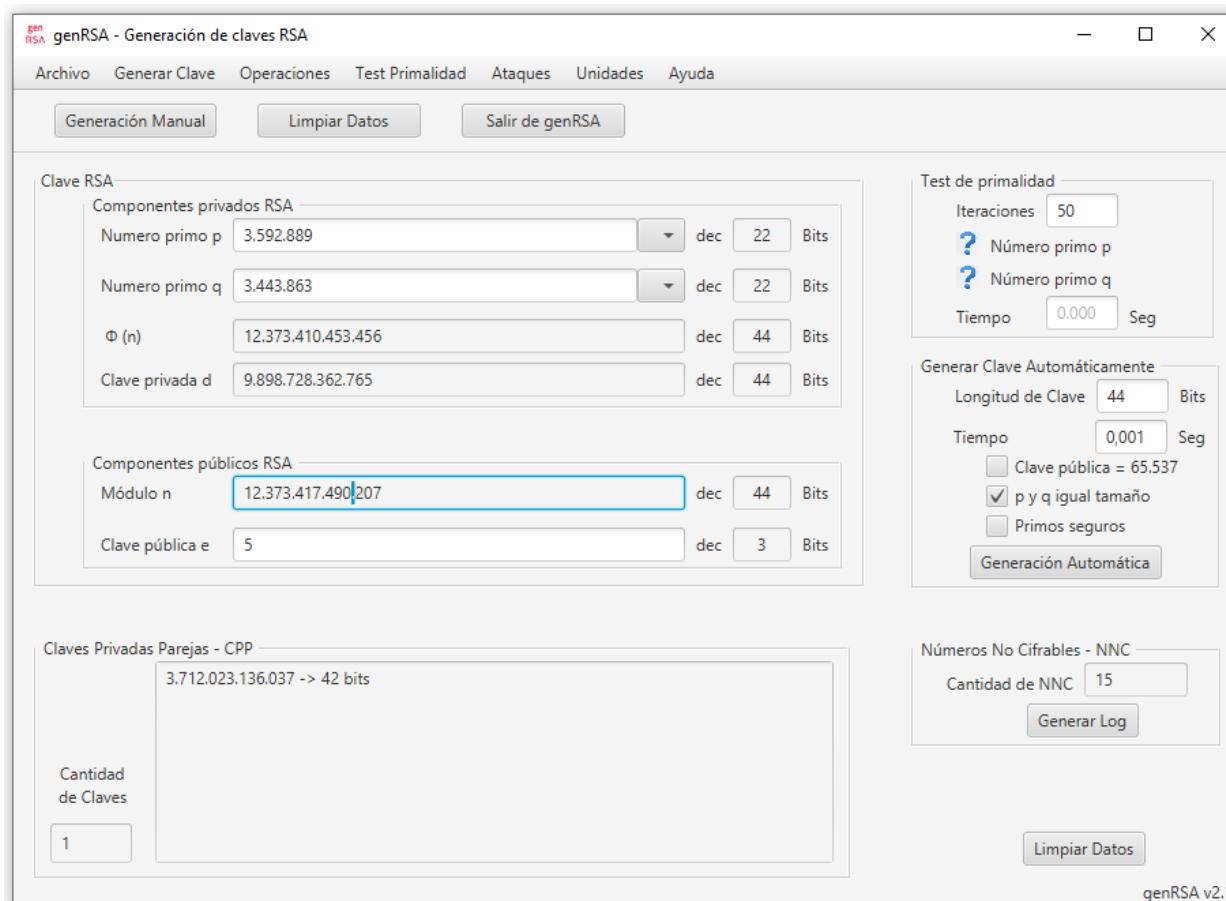


Agrupación en ASCII decimal con 3 dígitos: 108, 233, 097, 115, 101  
Texto = léame

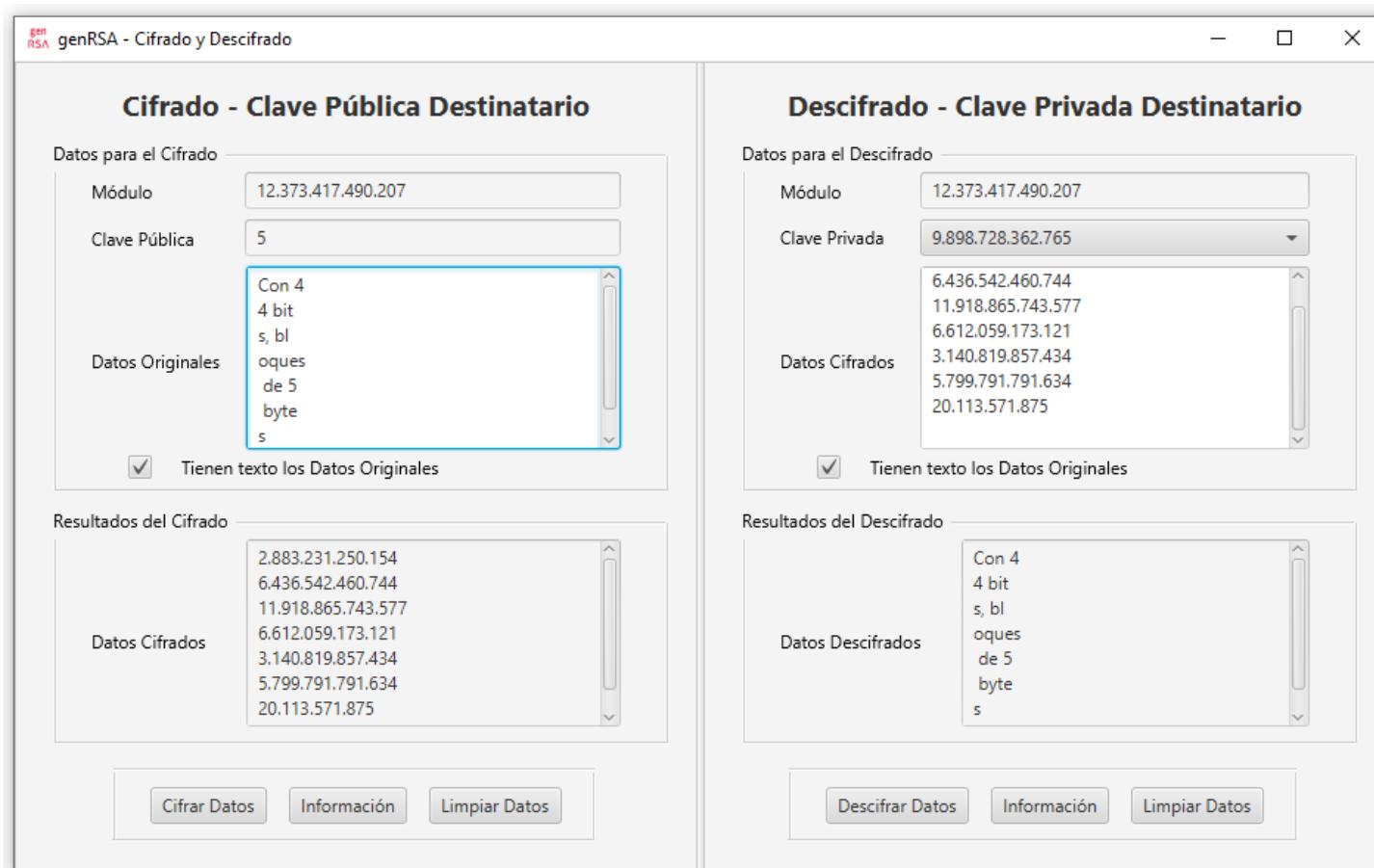
# Armando bloques de bytes

- Sea la clave RSA con  $p = 3.592.889$ ,  $q = 3.443.863$ ,  $n = 12.373.417.490.207$ ,  
 $\phi(n) = 12.373.410.453.456$ ,  $e = 5$ ,  $d = 9.898.728.362.765$       (**n de 44 bits**)
- Deberemos cifrar bloques de 5 bytes porque  $5*8 = 40$  bits y  $6*8 = 48$  bits
- Aunque los bits de la clave fuese un múltiplo de 8, por ejemplo  $48$  bits = 6 bytes, los bloques tendrán un byte menos para asegurar que ciframos restos
- Cifraremos el siguiente texto ASCII
  - **Con 44 bits, bloques de 5 bytes**

# Clave de 44 bits con genRSA v2.1



# Cifra bloques de 5 bytes con genRSA v2.1



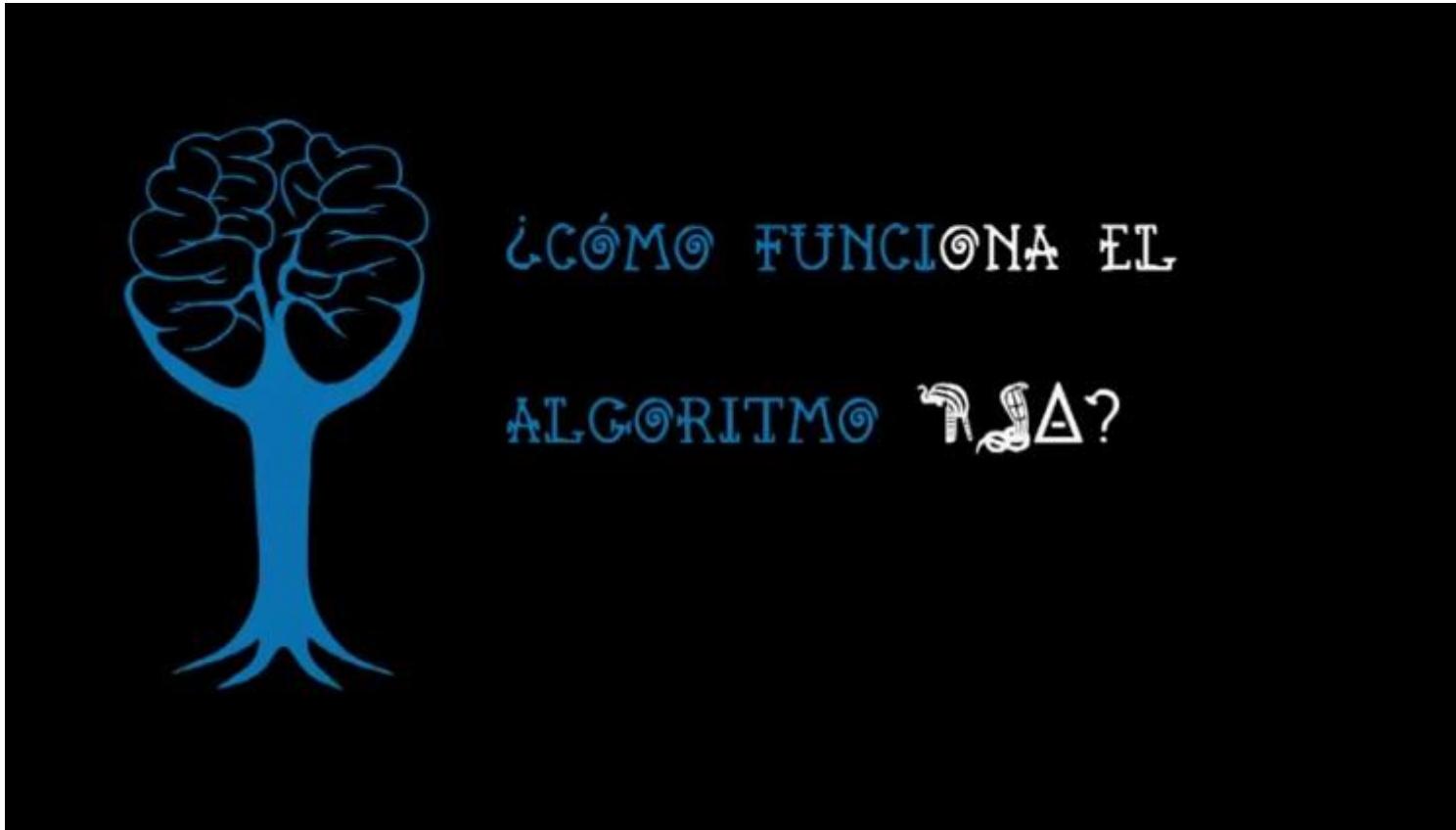
# Comprobación 1er bloque con SAMCript

- 1er bloque: **Con 4** (ASCII): 01000011 01101111 01101110 00100000 00110100
- $010000110110111101101110001000000110100 = 289.632.297.012$
- **Cifrado:**
  - $289.632.297.012^5 \bmod 12.373.417.490.207$
  - $= 2.883.231.250.154$
- **Descifrado:**
  - $2.883.231.250.154^{9.898.728.362.765} \bmod 12.373.417.490.207 = 289.632.297.012$
  - $289.632.297.012 = \underline{0}1000011 01101111 01101110 00100000 00110100 = \text{Con 4}$



| Potencia                                   |                    |
|--|--------------------|
| Base                                       | 289.632.297.012    |
| Exponente                                  | 5                  |
| <input checked="" type="checkbox"/> Módulo | 12.373.417.490.207 |
| Resultado                                  | 2.883.231.250.154  |

# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=CMe0COxZxb0>

# Conclusiones de la lección 10.6b

- Las operaciones de descifrado deben realizarse en el orden inverso al que se realizaron en el cifrado. Si se firma un número (integridad) y después este resultado se cifra (confidencialidad), en recepción primero se descifra (confidencialidad) y después se comprueba la firma (integridad). Y al revés, si primero cifra y después firma, se procede de igual manera, en sentido inverso
- En cifrados encadenados con diferentes grupos de cifra, hay que tener mucho cuidado con cifrar siempre restos de dichos grupos o módulos
- Si se desea cifrar textos como ejercicio de laboratorio, habrá que formar bloques de manera que cada bloque sea menor que el módulo de cifra. No tiene sentido cifrar archivos con RSA salvo en algún tipo de ransomware
- Entonces, se crearán bloques por dígitos numéricos, bytes o bits, siempre un dígito, un byte o un bit menos que el tamaño del módulo, para cifrar restos

# Lectura recomendada (1/2)

- Lección 3. Cifrado de números y mensajes, MOOC El algoritmo RSA, Jorge Ramió, 2012
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion3/leccion03.html>
- CLCrypt 00: Códigos y tablas de uso frecuente en criptografía, Cuadernos de Laboratorio de Criptografía, Jorge Ramió, 2019
  - [https://www.criptored.es/descarga/Codigos\\_y\\_tablas\\_de\\_uso\\_frecuente\\_en\\_crip\\_tografia.pdf](https://www.criptored.es/descarga/Codigos_y_tablas_de_uso_frecuente_en_crip_tografia.pdf)
- Guion píldora formativa Thoth nº 39, “Cómo funciona el algoritmo RSA”, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora039.pdf>

# Lectura recomendada (2/2)

- RSA with varying prime number sizes, A Security site (*igual a diapositiva 17*)
  - [https://asecuritysite.com/encryption/rsa\\_12\\_2](https://asecuritysite.com/encryption/rsa_12_2)
- Online RSA Encryption, Decryption And Key Generator Tool (*formato Base64*)
  - <https://www.devglan.com/online-tools/rsa-encryption-decryption>

# Class4crypt c4c10.7

## Módulo 10. Criptografía asimétrica

### Lección 10.7. Números no cifrables en RSA

10.7.1. Números no cifrables

10.7.2. Cantidad de números no cifrables

10.7.3. Buscando los números no cifrables

10.7.4. Supuesta debilidad de RSA debido a los números no cifrables

Class4crypt c4c10.7 Números no cifrables en RSA  
<https://www.youtube.com/watch?v=39fXlxt67zc>

# Números no cifrables NNC en RSA

- N es un número no cifrable NNC con cualquier clave en RSA, si se cumple que  $N^e \text{ mod } n = N^d \text{ mod } n = N$ , es decir, el criptograma de la cifra con la clave pública o de la firma digital con la clave privada será el mismo valor en claro
- En RSA habrá, como mínimo, 9 números no cifrables
  - El número 0, puesto que  $0^x \text{ mod } n = 0$  (condición obvia)
  - El número 1, puesto que  $1^x \text{ mod } n = 1$  (condición obvia)
  - El número  $n-1$ , puesto que  $(n-1)^x \text{ mod } n = n-1$  (demostración matemática)
  - Y mínimo otros 6 números que habrá que calcularlos usando fuerza bruta
- En el peor de los casos, todos los números ( $0 \leq N < n$ ) podrían ser no cifrables
- Esto podría ser un problema en RSA, puesto que si se usa este algoritmo para realizar un intercambio de clave K, al cifrar ese secreto K éste podría ir en claro y perdería, por tanto, su característica de confidencialidad

# Ejemplo de números no cifrables en RSA

- Sea la clave RSA de 10 bits
    - $p = 17$
    - $q = 31$
    - $n = 17 \cdot 31 = 527$
    - $\phi(n) = (p-1)(q-1) = 16 \cdot 30 = 480$
    - $e = 11$
    - $d = \text{inv}(11, 480) = 131$
  - Comprobación
    - $35^{11} \bmod 527 = 35$
    - $35^{131} \bmod 527 = 35$
    - $101^{11} \bmod 527 = 101$
    - $101^{131} \bmod 527 = 101$
    - $188^{11} \bmod 527 = 188$
    - $188^{131} \bmod 527 = 188$
- 
- Serán números no cifrables NNC de esta clave estos 33 números
    - 0, 1, 16, 33, 35, 85, 101, 120, 153, 154, 170, 171, 186, 187, 188, 221, 256, 271, 306, 339, 340, 341, 356, 357, 373, 374, 407, 426, 442, 492, 494, 511, 526
    - De 527 números, 33 van en claro

# Cantidad de números no cifrables

- La cantidad  $\sigma_n$  de NNC dentro de una clave RSA viene dada por
  - $\sigma_n = [1 + \text{mcd}(e-1, p-1)] [1 + \text{mcd}(e-1, q-1)]$
- Como  $\sigma_n$  solo depende de los valores de  $p$ ,  $q$  y  $e$ , será muy sencillo encontrar dicha cantidad
- En el ejemplo anterior, con  $p = 17$ ,  $q = 31$  y  $e = 11$ 
  - $\sigma_n = \sigma_{527} = [1 + \text{mcd}(11-1, 17-1)] [1 + \text{mcd}(11-1, 31-1)]$
  - $\sigma_{527} = [1 + \text{mcd}(10, 16)] [1 + \text{mcd}(10, 30)]$
  - $\sigma_{527} = [1 + 2] [1 + 10] = 3 * 11 = 33$
- La pregunta es, ¿además del 0, el 1 y el  $(n-1)$ , cómo puedo conocer cada uno de esos restantes valores de  $N$  que no se cifran en  $n$ ?

# Buscando los números no cifrables

- Ecuación para encontrar cada uno de los números no cifrables
  - $N = [q\{\text{inv}(q, p)\}N_p + p\{\text{inv}(p, q)\}N_q] \bmod n$ 
    - Donde  $N_p$  son las soluciones de  $N^e \bmod p = N$ , para  $0 \leq N \leq p-1$
    - Donde  $N_q$  son las soluciones de  $N^e \bmod q = N$ , para  $0 \leq N \leq q-1$
- Estas dos últimas ecuaciones, de fuerza bruta en  $p$  y en  $q$ , es lo que dificulta un ataque con valores reales
- Por ejemplo, si  $p$  y  $q$  son primos de 1.024 bits en una clave RSA de 2.048 bits, hacer las operaciones de consulta  $\{N^e \bmod p = N\}$  y  $\{N^e \bmod q = N\}$ , va a requerir  $2^{1.024}$  cálculos, imposible en la actualidad



# Encontrando los NNC del ejemplo anterior

- $N = [q\{\text{inv}(q, p)\}N_p + p\{\text{inv}(p, q)\}N_q] \bmod n$  ( $p = 17, q = 31, e = 11$ )
- $N_p$ : ¿ $N^{11} \bmod 17 = N$ ? para  $N = 0, 1, 2, \dots, 14, 15, 16$ 
  - $N_p$ :  $N = \{0, 1, 16\}$
- $N_q$ : ¿ $N^{11} \bmod 31 = N$ ? para  $N = 0, 1, 2, \dots, 28, 29, 30$ 
  - $N_q$ :  $N = \{0, 1, 2, 4, 8, 15, 16, 23, 27, 29, 30\}$
- Calculamos  $\text{inv}(q, p) = \text{inv}(31, 17) = \text{inv}(14, 17) = 11$
- Calculamos  $\text{inv}(p, q) = \text{inv}(17, 31) = 11$
- Reemplazamos valores en  $N$
- $N = [31*11\{0,1,16\} + 17*11\{0,1,2,4,8,15,16,23,27,29,30\}] \bmod 527$
- $N = [341\{0,1,16\} + 187\{0,1,2,4,8,15,16,23,27,29,30\}] \bmod 527$

Resolvemos,  
ordenamos y  
obtenemos los  
33 NNC del  
ejemplo  
anterior

# Distribución de los números no cifrables

- Para la clave  $n = 527 = 17 \cdot 31$  (10 bits) y  $e = 11$  tenemos
  - $33 \text{ NNC} = 0, 1, 16, 33, 35, 85, 101, 120, 153, 154, 170, 171, 186, 187, 188, 221, 256, 271, 306, 339, 340, 341, 356, 357, 373, 374, 407, 426, 442, 492, 494, 511, 526$
  - Excepto el 0, la suma de los extremos siempre será igual a  $n$ 
    - $1+526 = 16+511 = 33+494 = 35+492 = 85+442 = 101+426 = 120+407 = 153+374 = 154+373 = 170+357 = 171+356 = 186+341 = 187+340 = 188+339 = 221+306 = 256+271 = 527$
  - Aunque esto sea algo curioso, no es una debilidad en una clave RSA real, porque para números muy grandes el valor no cifrable siguiente al número 1 conocido será imposible adivinarlo, así como la distribución entre dichos números

# Distribución de los NNC en claves grandes

- (14 bits) Para la clave  $n = 11.009 = 101 \cdot 109$  y  $e = 7$ 
  - 21 NNC = 0, 1, 808, 809, 2.726, 3.333, 3.334, 3.534, 4.141, 4.142, 4.950, 6.059, 6.867, 6.868, 7.475, 7.675, 7.676, 8.283, 10.200, 10.201, 11.008
- (20 bits) Para la clave  $n = 819.703 = 839 \cdot 977$  y  $e = 3$ 
  - 9 NNC = 0, 1, 148.503, 148.504, 297.007, 522.696, 671.199, 671.200, 819.702
- (26 bits) Para la clave  $n = 50.813.951 = 6.733 \cdot 7.547$  y  $e = 5$ 
  - 15 NNC = 0, 1, 4.452.730, 9.343.187, 13.795.916, 13.795.917, 18.248.647, 23.222.118, 27.591.833, 32.565.304, 37.018.034, 37.018.035, 41.470.764, 46.361.221, 50.813.950
- Excepto 0 y 1, los demás NNC serán muy grandes

# Minimizando los números no cifrables

- Para que la cantidad de números no cifrables  $\sigma_n$  sea 9, la mínima posible, debemos elegir la clave pública e de forma que
  - $\text{mcd}(e-1, p-1) = 2$
  - $\text{mcd}(e-1, q-1) = 2$
  - $\sigma_n = [1 + \text{mcd}(e-1, p-1)] [1 + \text{mcd}(e-1, q-1)] = (1 + 2)(1 + 2) = 9$
  - Esto se logra en la mayoría de los casos usando primos seguros, es decir en el caso del primo p si  $p = 2^*r + 1$  siendo r un primo
- Si  $p = 11 = 2^*5 + 1$  y  $q = 47 = 2^*23 + 1$ ,  $n = 517$ ,  $\phi(n) = 10^*46 = 460$  y elegimos  $e = 7$  ya que  $\text{mcd}(7, 460) = 1$ , entonces  $\sigma_n = 9$  porque
  - $\text{mcd}(e-1, p-1) = \text{mcd}(7-1, 11-1) = \text{mcd}(6, 10) = 2$
  - $\text{mcd}(e-1, q-1) = \text{mcd}(7-1, 47-1) = \text{mcd}(6, 46) = 2$

# Cantidad máxima de números no cifrables

- Los números no cifrables pueden llegar a ser  $n = p^*q$  si
  - $\text{mcd}(e-1, p-1) = p-1$  y  $\text{mcd}(e-1, q-1) = q-1$
  - $\sigma_n = [1 + \text{mcd}(e-1, p-1)] [1 + \text{mcd}(e-1, q-1)] = (1 + (p-1))(1 + (q-1)) = p^*q$
- Esto se logra por lo general cuando se elige una clave pública  $e$  que cumpla con esta condición
  - $e = \phi(n)/k + 1$ , con  $k > 1$  y obviamente siendo  $e$  un valor válido
- En el ejemplo anterior con  $\phi(n) = 460$ , si elegimos  $e = \phi(n)/2 + 1 = 231$ , un valor de clave pública válido, entonces
  - $\text{mcd}(e-1, p-1) = \text{mcd}(230, 10) = 10 = p-1$
  - $\text{mcd}(e-1, q-1) = \text{mcd}(230, 46) = 46 = q-1$
  - $\sigma_n = 517 = n$

# NNC con clave pública $e = \phi(n)/k + 1$

- Nunca podrá usarse  $e = \phi(n)/2 + 1$  (si es una clave válida) ya que la clave de descifrado será igual a 1 y por tanto no será cifrable ningún número de n
- Sea  $p = 101$ ,  $q = 761$ ,  $n = 76.861$ . Luego  $\phi(n) = 100*760 = 76.000$
- Algunos valores de e válidos como clave relacionados con  $\phi(n)$ 
  - $e = \phi(n)/2 + 1 = 38.001 \Rightarrow 76.861$  NNC (100 %)
  - $e = \phi(n)/4 + 1 = 19.001 \Rightarrow 76.861$  NNC (100 %)
  - $e = \phi(n)/5 + 1 = 15.201 \Rightarrow 76.861$  NNC (100 %)
  - $e = \phi(n)/8 + 1 = 9.501 \Rightarrow 38.481$  NNC (50 % aproximadamente)
  - $e = \phi(n)/10 + 1 = 7.601 \Rightarrow 76.861$  NNC (100 %)
  - $e = \phi(n)/16 + 1 = 4.751 \Rightarrow 9.741$  NNC (12,5 % aproximadamente)
  - $e = \phi(n)/19 + 1 = 4.001 \Rightarrow 4.141$  NNC (5 % aproximadamente)
  - $e = \phi(n)/20 + 1 = 3.801 \Rightarrow 76.861$  NNC (100 %)
  - $e = \phi(n)/50 + 1 = 1.521 \Rightarrow 15.981$  NNC (20 % aproximadamente)
  - $e = \phi(n)/100 + 1 = 761 \Rightarrow 15.981$  NNC (20 % aproximadamente)
  - $e = \phi(n)/1.000 + 1 = 77 \Rightarrow 385$  NNC (0,5 % aproximadamente)
- En claves reales, e es muchísimo más pequeño que  $\phi(n)$  y por tanto esto no sucede

# Intercambio de clave y firma RSA con NNC

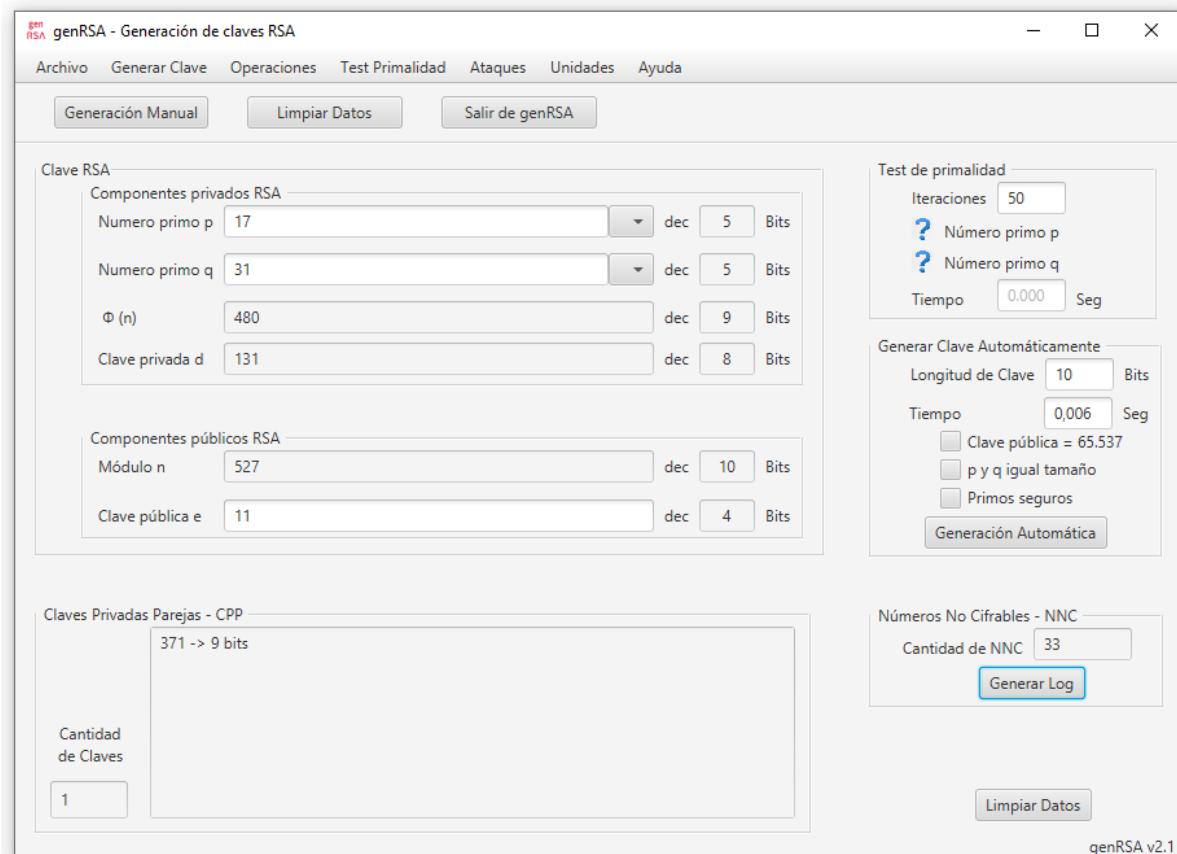
- Como la cantidad de números no cifrables NNC puede llegar a ser valor muy grande, se podría plantear un problema cuando
  - a) Intercambiamos una clave de sesión K siendo ese valor un NNC, ya que al realizar la operación  $C = K^e \text{ mod } n$ , C será el mismo número K
  - b) Firmamos el hash de un mensaje  $h(M)$  siendo ese valor un NNC, ya que al realizar la operación  $F = h(M)^d \text{ mod } n$ , F será el mismo hash  $h(M)$
- Para un módulo de 2.048 bits, en ambos casos, la cifra o la firma, irán en claro y por lo tanto ese criptograma tendría -por ejemplo- solo los 256 bits de la clave K o del hash  $h(M)$  como bits significativos y habría varias centenas de bits en cero al comienzo, por lo que sería fácilmente detectable

# Los NNC no serán una preocupación real

- A la derecha se observa el informe del software genRSA v2.1 para una clave de 60 bits con 25 NNC
- A medida que vamos aumentando el tamaño del módulo, los NNC comienzan a ser cada vez más grandes y están más cerca del módulo n, y su distribución deja de tener esa apariencia uniforme dentro el grupo y que podía observarse en claves con módulos de cifra pequeños
- Por lo tanto, para claves de 2.048 bits, los NNC estarán muy cerca del módulo n y la probabilidad de que alguno de esos números tenga 256 bits es nula

0  
1  
16.563.694.364.824.882  
97.634.547.149.596.920  
114.198.241.514.421.801  
114.198.241.514.421.802  
158.188.081.673.967.981  
174.751.776.038.792.862  
211.832.788.664.018.722  
228.396.483.028.843.603  
238.357.281.704.916.494  
254.920.976.069.741.375  
272.386.323.188.389.782  
352.555.523.219.338.295  
370.020.870.337.986.702  
386.584.564.702.811.583  
396.545.363.378.884.474  
413.109.057.743.709.355  
450.190.070.368.935.215  
466.753.764.733.760.096  
510.743.604.893.306.275  
510.743.604.893.306.276  
527.307.299.258.131.157  
608.378.152.042.903.195  
624.941.846.407.728.076

# Prácticas con el software genRSA v2.1



**Unidades: Decimal**  
Número primo P generado:  
17  
Número primo Q generado:  
31  
Módulo N generado:  
527  
Clave Pública e generada:  
11  
Clave Privada d generada:  
131

## NÚMEROS NO CIFRABLES

La cantidad de Números No Cifrables es: 33

0  
1  
16  
33  
35  
85  
101  
120  
153  
154  
170  
171  
186  
187  
188  
221  
256  
271  
306  
339  
340  
341  
356  
357  
373  
374  
407  
426  
442  
492  
494  
511  
526

[https://www.criptored.es/software/sw\\_m001d.htm](https://www.criptored.es/software/sw_m001d.htm)

# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=mZymE3eMEpI>

# Conclusiones de la Lección 10.7

- Existirán como mínimo 9 números no cifrables NNC, es decir, que al cifrarlos con la clave pública  $e$  o con la clave privada  $d$  en módulo  $n$  irán en claro
- El valor mínimo de 9 NNC se logra usando primos seguros
- Estos números son el 0, el 1, el  $n-1$  y como mínimo otros 6 números
- Será fácil conocer la cantidad de números no cifrables, pero no así sus valores puesto que esto requiere un ataque por fuerza bruta a los primos  $p$  y  $q$
- Si no se tiene cuidado en la elección de la clave pública  $e$ , podría darse el caso de que todos los números del grupo  $n$  de cifra vayan en claro. Esto sucede cuando se elige  $e = [\phi(n)/2] + 1$ , que lógicamente cumpla con  $\text{mcd}(e, \phi(n)) = 1$
- Para claves reales con números grandes esto no sucede ya que todos estos NNC son valores de magnitud similar al módulo  $n$  y no será una debilidad

# Lectura recomendada

- Curso de Criptografía Aplicada, Jorge Ramió, 2018
  - [https://www.criptored.es/guiateoria/gt\\_m001s1.htm](https://www.criptored.es/guiateoria/gt_m001s1.htm)
- Libro Electrónico de Seguridad Informática y Criptografía, versión 4.1, Jorge Ramió, 2006
  - [https://www.criptored.es/guiateoria/gt\\_m001a.htm](https://www.criptored.es/guiateoria/gt_m001a.htm)
- Guion píldora formativa Thoth nº 40, ¿Es vulnerable el algoritmo RSA?, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora040.pdf>
- CLCript 06 Números no cifrables en RSA, Jorge Ramió, 2019
  - [https://www.criptored.es/descarga/CLCript\\_entrega\\_06\\_Numeros\\_No\\_Cifrables\\_RSA.pdf](https://www.criptored.es/descarga/CLCript_entrega_06_Numeros_No_Cifrables_RSA.pdf)

# Class4crypt c4c10.8

## Módulo 10. Criptografía asimétrica

### Lección 10.8. Claves parejas y números piratas en RSA

10.8.1. Qué son y cómo se calculan las claves privadas parejas

10.8.2. Supuesta vulnerabilidad en RSA por las claves privadas parejas

10.8.3. Qué son y cómo se calculan las claves públicas parejas

10.8.4. Supuesta vulnerabilidad en RSA por las claves públicas parejas

10.8.5. Descifrando RSA con números piratas

10.8.6. Supuesta vulnerabilidad en RSA por los números piratas

Class4crypt c4c10.8 Claves parejas y números piratas en RSA  
<https://www.youtube.com/watch?v=i5yYJEi2N58>

# Existencia de claves parejas en RSA

- En RSA las claves pública e y privada d son inversas en el grupo trampa  $\phi(n) = (p-1)(q-1)$ , con  $d = \text{inv}(e, \text{mod } \phi(n))$  un valor único
- Pero la cifra y la firma se hacen en el grupo público  $n = p*q$ , y en  $n$  ya no se cumple que exista una única clave privada  $d$  que descifre lo que se haya cifrado con la clave pública  $e$
- En el grupo  $n$ , para cada clave pública  $e$  existirán otros valores de clave privada  $d'$  (claves privadas parejas CPP) que actúan como si fuesen inversos multiplicativos de  $e$  en  $\phi(n)$
- Y de la misma forma, en  $n$  para cada clave privada  $d$  existirán otros valores de clave pública  $e'$  (claves públicas parejas CPúbP) que actúan como si fuesen inversos multiplicativos de  $d$  en  $\phi(n)$

# ¿Qué son las claves privadas parejas?

- En RSA habrá como mínimo una clave privada pareja que cumplirá la misma función que su homóloga, es decir, permite descifrar con confidencialidad o firmar digitalmente con autenticidad
- Si se usan números primos seguros, entonces la cantidad de claves privadas parejas se minimiza, siendo casi siempre igual a 1
- Una clave RSA que tenga 9 números no cifrables y 1 clave privada pareja, es decir los mínimos, se conocerá como clave óptima
- Hay que tener en cuenta que programas estándar como OpenSSL por lo general no usan primos seguros en la generación de claves, aunque en últimas versiones sí se minimizan los NNC y las CPP

# Ejemplo de claves privadas parejas

- Sean  $p = 13$ ,  $q = 19$ ,  $n = 13 \cdot 19 = 247$  y  $\phi(n) = 12 \cdot 18 = 216$
- Se elige como clave pública  $e = 5$  pues  $\text{mcd}(5, 216) = 1$
- Se calcula la clave privada  $d = \text{inv}(e, \phi(n)) = \text{inv}(5, 216) = 173$
- Si ciframos el valor secreto  $K = 100$ ,  $C = 100^5 \pmod{247} = 237$
- Desciframos  $C$  con la clave privada  $d$ ,  $237^{173} \pmod{247} = 100$
- Pero también desciframos  $C$  con estas 6 claves privadas parejas  $d'$ 
  - $d' = 29, 65, 101, 137, 209$  y  $245$
  - $237^{29} \pmod{247} = 100$        $237^{65} \pmod{247} = 100$
  - $237^{101} \pmod{247} = 100$        $237^{137} \pmod{247} = 100$
  - $237^{209} \pmod{247} = 100$        $237^{245} \pmod{247} = 100$

# Cantidad de claves privadas parejas

- La clave privada de menor valor se conoce como  $d_\gamma$ 
    - $d_\gamma = \text{inv}(e, \gamma)$
    - Con  $\gamma = \text{mcm}[(p-1), (q-1)]$
  - Las claves privadas parejas  $d_i$  serán
    - $d_i = d_\gamma + i^* \gamma$ , donde  $i = 0, 1, \dots, \lambda$  (deberá cumplirse que  $d_\lambda < n$ )
    - En este caso, la cantidad de claves privadas parejas será  $\lambda = \lfloor (n - d_\gamma) / \gamma \rfloor$
  - En el ejemplo:  $p = 13$ ,  $q = 19$ ,  $n = 247$ ,  $\phi(n) = 216$ ,  $e = 5$ ,  $d = 173$ 
    - $\gamma = \text{mcm}(12, 18) = 36$
    - $d_\gamma = \text{inv}(e, \gamma) = \text{inv}(5, 36) = 29$
    - $\lambda = \lfloor (n - d_\gamma) / \gamma \rfloor = \lfloor (247 - 29) / 36 \rfloor = \lfloor 6,05 \rfloor = 6$
    - $d_i = d_\gamma + i^* \gamma = 29 + i * 36 = 29, 65, 101, 137, 173, 209, 245$  (**173** es el valor  $d$ )
-  La clave privada  
será una de éstas

# Casos especiales en cantidad de CPP

- Dependiendo del valor elegido para la clave pública  $e$ , es posible que la cantidad de claves privadas parejas sea  $\lambda - 1$  en vez de  $\lambda$
- Siguiendo el ejemplo anterior ( $p = 13$ ,  $q = 19$ ,  $n = 247$ ,  $\phi(n) = 216$ ) pero ahora con clave pública  $e = 7$ , y por tanto  $d = 31$ , tenemos
  - $\gamma = \text{mcm}(12, 18) = 36$
  - $d_\gamma = \text{inv}(e, \gamma) = \text{inv}(7, 36) = 31$
  - $\lambda = \lfloor (n - d_\gamma)/\gamma \rfloor = \lfloor (247 - 31)/36 \rfloor = \lfloor 6,00 \rfloor = 6$  (pero no será cierto)
  - Se supone que en la ecuación  $d_i = d_\gamma + i * \gamma$  usaremos  $\lambda = 0, 1, 2, 3, 4, 5, 6$
  - Pero para  $\lambda = 6$  tenemos  $d_\lambda = 31 + 6 * 36 = 247 = n$  (que no es un resto válido)
  - En este caso se tendrá que contabilizar hasta  $\lambda - 1 = 5$ , es decir:
  - $d_i = d_\gamma + i * \gamma = 31 + i * 36 = 31, 67, 103, 139, 175, 211$  (**31** es el valor  $d$ )

# Minimizando las claves privadas parejas

- Eligiendo p y q primos seguros, la cantidad de claves privadas parejas será siempre muy bajo, aunque no se garantiza que sea 1
  - Si  $p = 11$ ,  $q = 47$  (primos seguros),  $n = 11 \cdot 47 = 517$ ,  $\phi(n) = 10 \cdot 46 = 460$
- Elegimos  $e = 3$  y calculamos  $d = \text{inv}(e, \phi(n)) = \text{inv}(3, 460) = 307$ 
  - $\gamma = \text{lcm}[(p-1), (q-1)] = \text{lcm}(10, 46) = 230$
  - $d_\gamma = \text{inv}(e, \gamma) = \text{inv}(3, 230) = 77$
  - $\lambda = \lfloor (n - d_\gamma)/\gamma \rfloor = \lfloor (517 - 77)/230 \rfloor = \lfloor 1,90 \rfloor = 1$
  - $d_i = d_\gamma + i \gamma = 77 + i \cdot 230 = 77, 307$
- Pero si elegimos  $e = 7$ 
  - $d_\gamma = \text{inv}(e, \gamma) = \text{inv}(7, 230) = 33$
  - $\lambda = \lfloor (n - d_\gamma)/\gamma \rfloor = \lfloor (517 - 33)/230 \rfloor = \lfloor 2,10 \rfloor = 2$
  - $d_i = d_\gamma + i \gamma = 33 + i \cdot 230 = 33, 263, 493$

# Las CPP no serán una preocupación real

- Para claves reales todas las claves privadas parejas tendrán valores muy cercanos al grupo  $\phi(n)$ , con una magnitud de 2.048 bits
- Como la clave pública  $e$  ( $F_4$ ) será de 17 bits y  $\gamma = \text{mcm}[(p-1), (q-1)]$  será un valor solo algo menor que  $\phi(n)$  pero de una magnitud muy similar, la clave privada o privada pareja más pequeña  $d_0$  tendrá una magnitud en torno a los 2.048 – 17 bits es decir unos 2.031 bits
- Hacer  $2^{2.031}$  cálculos, o de ese orden, para intentar adivinar algunas de estas claves privadas parejas es un cómputo inalcanzable actualmente
- Por lo tanto, para el caso de claves RSA reales de 2.048 bits, las claves privadas parejas no serán una debilidad o vulnerabilidad del algoritmo

# ¿Qué son las claves públicas parejas?

- En RSA habrá como mínimo dos claves públicas parejas que cumplirán la misma función que su homóloga, es decir, que nos permita realizar un intercambio de clave o que compruebe la validez de una firma digital
- Si se usan números primos seguros, entonces la cantidad de claves públicas parejas se minimiza, siendo su valor igual a 1 o 2
- Por lo general habrá una clave pública pareja más que privada
- Aunque a simple vista pareciera que no debería preocuparnos que haya otros números desconocidos que hagan lo mismo que la clave pública, por ser ésta pública, veremos que puede darse una situación extraña en la comprobación de firmas digitales con RSA

# Ejemplo de claves públicas parejas

- Sean  $p = 13$ ,  $q = 19$ ,  $n = 13 \cdot 19 = 247$  y  $\phi(n) = 12 \cdot 18 = 216$
- Se elige como clave pública  $e = 5$  pues  $\text{mcd}(5, 216) = 1$
- Se calcula la clave privada  $d = \text{inv}(e, \phi(n)) = \text{inv}(5, 216) = 173$
- Si firmamos el valor secreto  $K = 200$ ,  $F = 200^{173} \bmod 247 = 109$
- Comprobamos la firma  $F$  con clave pública  $e$ ,  $109^5 \bmod 247 = 200$
- Y también comprobamos  $F$  con estas 6 claves públicas parejas  $e'$ 
  - $e' = 41, 77, 113, 149, 185$  y  $221$
  - $109^{41} \bmod 247 = 200$        $109^{77} \bmod 247 = 200$
  - $109^{113} \bmod 247 = 200$        $109^{149} \bmod 247 = 200$
  - $109^{185} \bmod 247 = 200$        $109^{221} \bmod 247 = 200$

# Cantidad de claves públicas parejas

- La clave pública de menor valor se conoce como  $e_\gamma$ 
  - $e_\gamma = \text{inv}(d, \gamma)$
  - Con  $\gamma = \text{mcm}[(p-1), (q-1)]$
- Las claves públicas parejas  $e_i$  serán
  - $e_i = e_\gamma + i^* \gamma$ , donde  $i = 0, 1, \dots, \lambda$  (deberá cumplirse que  $e_\lambda < n$ )
  - En este caso, la cantidad de claves públicas parejas será  $\lambda = \lfloor (n - e_\gamma) / \gamma \rfloor$
- En el ejemplo:  $p = 13$ ,  $q = 19$ ,  $n = 247$ ,  $\phi(n) = 216$ ,  $e = 5$ ,  $d = 173$ 
  - $\gamma = \text{mcm}(12, 18) = 36$
  - $e_\gamma = \text{inv}(d, \gamma) = \text{inv}(137, 36) = \text{inv}(29, 36) = 5$
  - $\lambda = \lfloor (n - e_\gamma) / \gamma \rfloor = \lfloor (247 - 5) / 36 \rfloor = \lfloor 6,72 \rfloor = 6$
  - $e_i = e_\gamma + i^* \gamma = 5 + i * 36 = 5, 41, 77, 113, 149, 185, 221$  (**5** es el valor  $e$ )

# Firma digital y claves públicas parejas

- En el ejemplo anterior con clave  $p = 13$ ,  $q = 19$ ,  $n = 247$ ,  $\phi(n) = 216$ ,  $e = 5$  y  $d = 173$ , teníamos un total de 6 claves públicas parejas
- La firma del valor 200 en entregaba  $F = 200^{173} \text{ mod } 247 = 109$
- Que se podía comprobar en recepción, usando la clave pública por todos conocida  $e = 5$ , es decir,  $109^5 \text{ mod } 247 = 200$ . Pero también se podía comprobar la firma usando 41, 77, 113, 149, 185 y 221
- ¿Significa esto que el algoritmo RSA de firma queda en entredicho?
- No, porque para números muy grandes las claves públicas parejas serán números también muy grandes, todos cerca del módulo  $n$ , y por tanto no podrán conocerse ni adivinarse

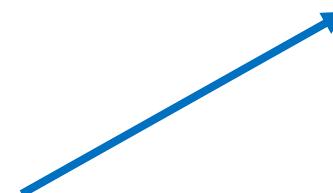
# Números piratas en RSA



- Definición de pirata en la RAE: 2. adj. Illegal, que carece de la debida licencia o que está **falsificado**
- **Falsificado**: quizás sería la forma más adecuada de llamar a estos números que, sin ser la clave privada ni tampoco ninguna de las claves privadas parejas, son capaces de descifrar un criptograma
- Supongamos que hemos cifrado un número secreto  $K$  con la clave pública del receptor  $e_R$ , obteniendo el criptograma  $C = K^{e_R} \text{ mod } n_R$ 
  - Descifraremos  $C$  con la clave privada  $d_R$   $C^{d_R} \text{ mod } n_R = K$
  - Descifraremos  $C$  con una clave privada pareja  $d'_R$   $C^{d'R} \text{ mod } n_R = K$
  - Y descifraremos  $C$  con un número pirata  $N_p$   $C^{Np} \text{ mod } n_R = K$

# Cifrado RSA de todos los restos de n

- Clave RSA con  $p = 7$ ,  $q = 19$ ,  $n = 133$ ,  $\phi(n) = 108$ ,  $e = 5$  y  $d = 65$
- $d' = 11, 29, 47, 83, 101, 119$
- Cifraremos  $N^5 \bmod 133$
- Con  $0 \leq N < 133$
- En amarillo se indican los 9 números no cifrables de esta clave
- En el cifrado se obtienen todos los restos del módulo  $n = 133$ , es decir, la cifra será única, por ejemplo, el caso de  $44^5 \bmod 133 = 81$



| N       | Cifrado: $N^5 \bmod 133$ |     |     |     |     |     |     |     |     |     |
|---------|--------------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|         | 0                        | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
| 0-9     | 0                        | 1   | 32  | 110 | 93  | 66  | 62  | 49  | 50  | 130 |
| 10-19   | 117                      | 121 | 122 | 90  | 105 | 78  | 4   | 82  | 37  | 38  |
| 20-29   | 20                       | 70  | 15  | 74  | 47  | 100 | 87  | 69  | 35  | 22  |
| 30-39   | 102                      | 103 | 128 | 10  | 97  | 42  | 120 | 18  | 19  | 58  |
| 40-49   | 108                      | 34  | 112 | 85  | 81  | 68  | 107 | 73  | 41  | 7   |
| 50-59   | 8                        | 109 | 124 | 2   | 80  | 6   | 56  | 57  | 39  | 89  |
| 60-69   | 72                       | 17  | 104 | 119 | 106 | 88  | 54  | 79  | 45  | 27  |
| 70-79   | 14                       | 29  | 116 | 61  | 44  | 94  | 76  | 77  | 127 | 53  |
| 80-89   | 131                      | 9   | 24  | 125 | 126 | 92  | 60  | 26  | 65  | 52  |
| 90-99   | 48                       | 21  | 99  | 25  | 75  | 114 | 115 | 13  | 91  | 36  |
| 100-109 | 123                      | 5   | 30  | 31  | 111 | 98  | 64  | 46  | 33  | 86  |
| 110-119 | 59                       | 118 | 63  | 113 | 95  | 96  | 51  | 129 | 55  | 28  |
| 120-129 | 43                       | 11  | 12  | 16  | 3   | 83  | 84  | 71  | 67  | 40  |
| 130-139 | 23                       | 101 | 132 |     |     |     |     |     |     |     |

# Descifrando con claves y números piratas

- Pero el descifrado ya NO será único en n
- Sea el criptograma  $C = 81 = 44^5 \text{ mod } 133$
- Recuperamos el secreto 44 descifrando
  - a) Con la clave privada  $d = 65$  (amarillo)
  - b) Con las claves privadas parejas  $d' = 11, 29, 47, 83, 101$  y  $119$  (naranja,  $\Delta = 18$ )
  - c) Y con los números piratas  $N_p = 2, 20, 38, 56, 74, 92, 110, 128$  (verde,  $\Delta = 18$ )
- En este caso, los números piratas son solo valores pares y siguen una distribución
- Se forman anillos de números. Para esta clave: 1, 81, 44, 106, 74, 9, 64, 130, 23

| X       | Descifrado de $C = 81, 81^x \text{ mod } 133$ |     |     |     |     |     |     |     |     |     |
|---------|---|-----|-----|-----|-----|-----|-----|-----|-----|-----|
|         | 0   | 1   | 2   | 3   | 4   | 5   | 6   | 7   | 8   | 9   |
| 0-9     | 1   | 81  | 44  | 106 | 74  | 9   | 64  | 130 | 23  | 1   |
| 10-19   | 81  | 44  | 106 | 74  | 9   | 64  | 130 | 23  | 1   | 81  |
| 20-29   | 44  | 106 | 74  | 9   | 64  | 130 | 23  | 1   | 81  | 44  |
| 30-39   | 106   | 74  | 9   | 64  | 130 | 23  | 1   | 81  | 44  | 106 |
| 40-49   | 74  | 9   | 64  | 130 | 23  | 1   | 81  | 44  | 106 | 74  |
| 50-59   | 9   | 64  | 130 | 23  | 1   | 81  | 44  | 106 | 74  | 9   |
| 60-69   | 64  | 130 | 23  | 1   | 81  | 44  | 106 | 74  | 9   | 64  |
| 70-79   | 130   | 23  | 1   | 81  | 44  | 106 | 74  | 9   | 64  | 130 |
| 80-89   | 23  | 1   | 81  | 44  | 106 | 74  | 9   | 64  | 130 | 23  |
| 90-99   | 1   | 81  | 44  | 106 | 74  | 9   | 64  | 130 | 23  | 1   |
| 100-109 | 81  | 44  | 106 | 74  | 9   | 64  | 130 | 23  | 1   | 81  |
| 110-119 | 44  | 106 | 74  | 9   | 64  | 130 | 23  | 1   | 81  | 44  |
| 120-129 | 106   | 74  | 9   | 64  | 130 | 23  | 1   | 81  | 44  | 106 |
| 130-139 | 74  | 9   | 64  |     |     |     |     |     |     |     |

Clave privada que descifra  $C = 81: x = 65$

Claves privadas parejas que descifran  $C = 81: x = 11, 29, 47, 83, 101, 119$

Números piratas que descifran el  $C = 81: x = 2, 20, 38, 56, 74, 92, 110, 128$

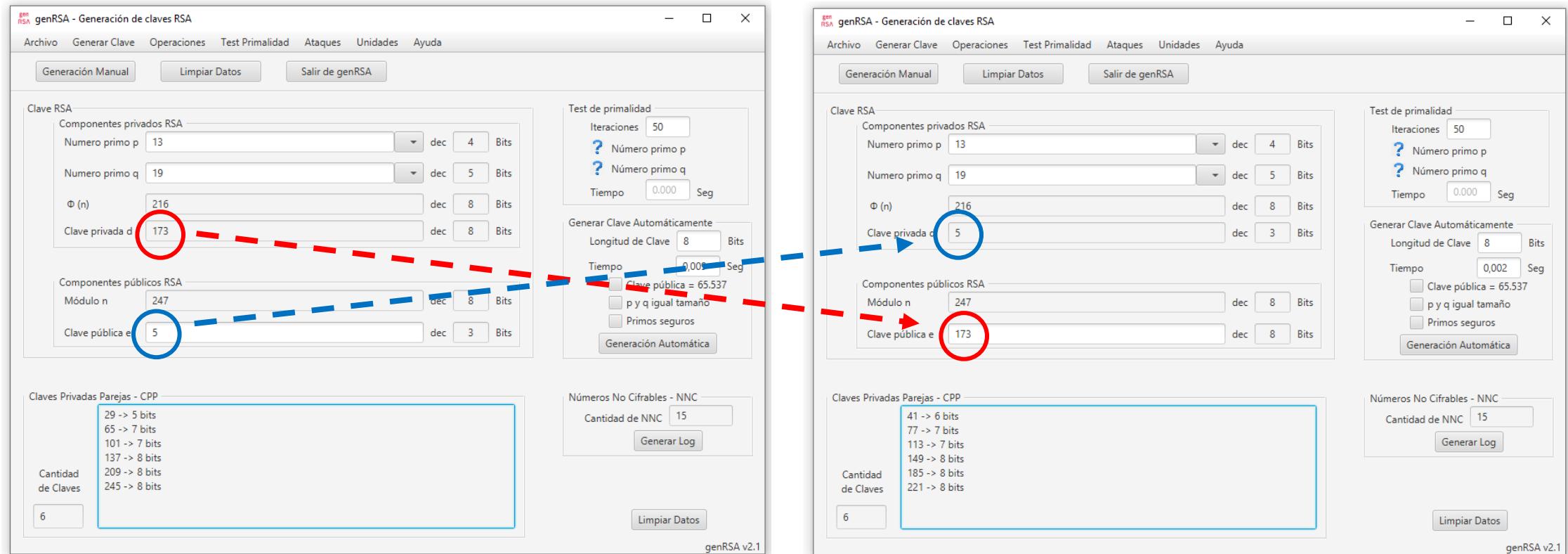
# Otros ejemplos de números piratas

- Para  $p = 23$ ,  $q = 31$ ,  $n = 713$ ,  $e = 7$ ,  $d = 283$ ,  $d' = 613$ , la cifra  $N^7 \bmod 713$  con  $0 \leq N < 713$  entrega todos los restos de  $n = 713$ , la cifra es única
- Si se cifra el valor secreto  $N = 100$ ,  $C = 100^7 \bmod 713 = 679$
- $C = 679$  se descifra con  $d = 283$ , se descifra con  $d' = 613$  y también se descifra con el número pirata  $N_p = 118$ , ya que  $679^{118} \bmod 713 = 100$
- Para  $679^x \bmod 713$ , se forma un anillo de 165 valores, por ejemplo desde  $x = 0$  hasta  $x = 164$ : 1, 679, 443, 624, 174, ... 676, 545, 8, 441, 692
- El criptograma 679 se descifra con el número pirata 118, con  $118+165 = 283$  (la clave privada);  $283+165 = 448$  (otro número pirata);  $448+165 = 613$  (la CPP)... **incluso** con  $613+165 = 778$ ,  $778+165 = 943$ ,  $943+165 = 1.108$ ... números fuera del grupo y que no se han reducido mod 713

# Números piratas dignos de estudio

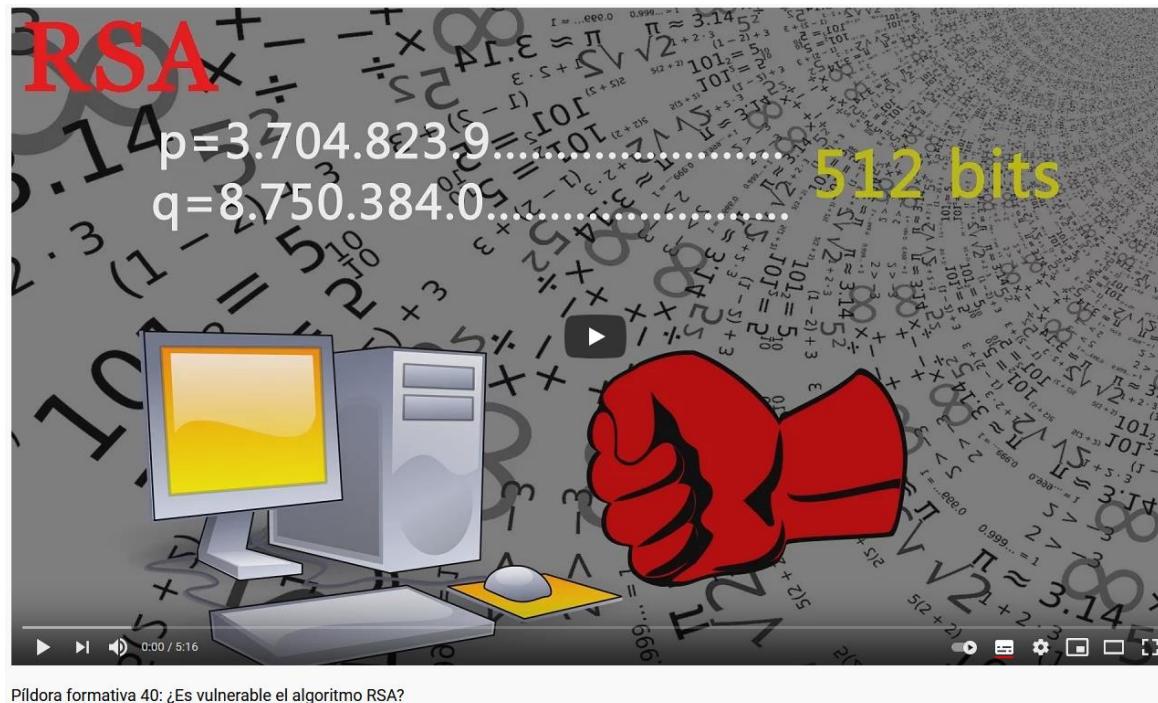
- Según los datos obtenidos, en la aparición de una mayor o menor cantidad de estos números especiales, está claro que intervendrán los valores de los primos p y q, la cantidad de claves parejas CPP y el valor elegido para la clave pública e
- Aunque sea un tema de interés académico, es digno de estudio
- Al igual que sucedía con las claves privadas parejas y con las claves pública parejas, todos los valores de números piratas para claves RSA reales de 2.048 bits serán inmensos, imposibles de adivinar
- Los anillos que se formarán en el descifrado  $C^x \text{ mod } n$  con  $0 \leq x < n$  serán inmensos y por tanto será imposible hacerle un seguimiento

# Prácticas con el software genRSA v2.1



[https://www.criptored.es/software/sw\\_m001d.htm](https://www.criptored.es/software/sw_m001d.htm)

# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=mZymE3eMEpI>

# Conclusiones de la Lección 10.8

- Toda clave RSA tendrá al menos una clave privada pareja CPP d'
- De la misma manera, existirán claves pública parejas e', aunque en muchos casos la cantidad de claves públicas parejas será una unidad mayor que la de claves privadas parejas
- Estas claves parejas estarán separadas por un valor constante, se pueden calcular de forma rápida y sencilla, incluso para números muy grandes, y se minimizan usando primos seguros
- Para claves RSA actuales, las claves privadas parejas d' (y las claves públicas parejas) serán números de una magnitud similar a la del módulo de cifra n, imposibles de adivinar, lo mismo que otros números diferentes a d y d' dentro del grupo n, que también descifran el criptograma y que este profesor llama números piratas. Por lo tanto, no serán una debilidad del algoritmo

# Lectura recomendada

- Curso de Criptografía Aplicada, Jorge Ramió, 2018
  - [https://www.criptored.es/guiateoria/gt\\_m001s1.htm](https://www.criptored.es/guiateoria/gt_m001s1.htm)
- Libro Electrónico de Seguridad Informática y Criptografía, versión 4.1, Jorge Ramió, 2006
  - [https://www.criptored.es/guiateoria/gt\\_m001a.htm](https://www.criptored.es/guiateoria/gt_m001a.htm)
- Guion píldora formativa Thoth nº 40, ¿Es vulnerable el algoritmo RSA?, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora040.pdf>
- CLCript 05 Claves parejas en RSA, Jorge Ramió, 2019
  - [https://www.criptored.es/descarga/CLCript\\_entrega\\_05\\_Claves\\_Parejas\\_RSA.pdf](https://www.criptored.es/descarga/CLCript_entrega_05_Claves_Parejas_RSA.pdf)

# Class4crypt c4c10.9a

Módulo 10. Criptografía asimétrica

Lección 10.9a. Ataques teóricos y prácticos a RSA parte 1

10.9a.1. Ataques significativos a RSA

10.9a.2. Ataque por factorización entera del módulo n

10.9a.3. Ataque por cifrado cíclico

Próxima lección 10.9b Ataques teóricos y prácticos a RSA parte 2

10.9b.1. Ataque por la paradoja del cumpleaños

10.9b.2. Ataque acústico por canal lateral

10.9b.3. Conclusiones de la lección 10.9

Class4crypt c4c10.9a Ataques teóricos y prácticos a RSA parte 1  
<https://www.youtube.com/watch?v=7XI8WzWu4kE>

# Ataques a RSA

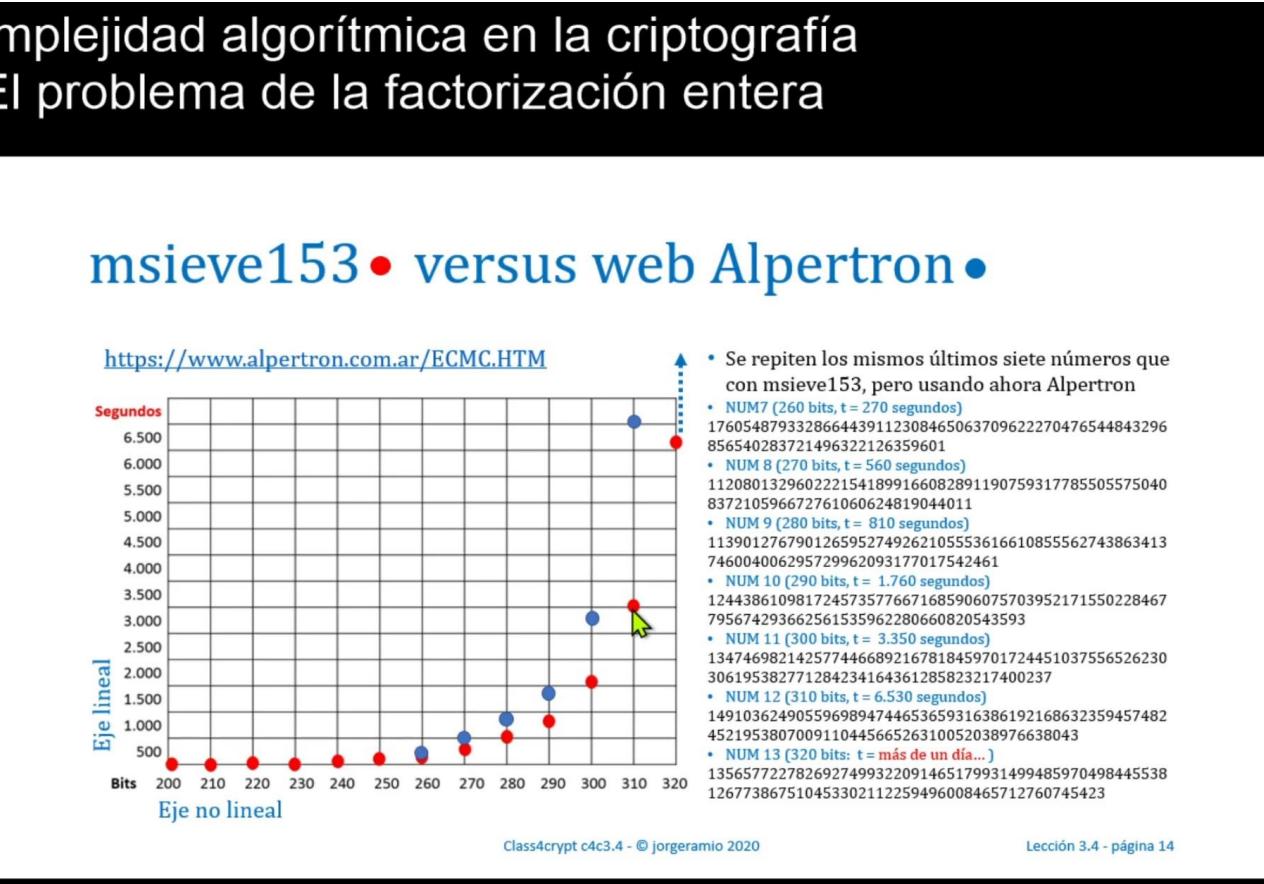
- Ataques teóricos más conocidos
  - Ataque basado en la factorización del módulo n
    - Resolviendo el Problema de la Factorización Entera PFE, se encuentra la clave privada d
  - Ataque por cifrado cíclico
    - Conociendo el criptograma, se realiza un cifrado cíclico usando siempre la clave pública de la víctima y es posible romper el secreto cifrado con confidencialidad pero no la clave privada d
  - Ataque por la paradoja del cumpleaños
    - Conociendo solamente los datos públicos de la víctima, es posible encontrar la clave privada d o una clave privada pareja d' aunque pueden aparecer falsos positivos muy poco frecuentes
  - Existen otros ataques: de exponente público e bajo, de módulo n en común, etc.
- Ataques reales
  - Ataque acústico por canal lateral
    - La ejecución del algoritmo de exponenciación modular provoca sonidos entre los 30 KHz y los 40 KHz que permiten descubrir los bits ceros y unos de la clave privada d

# Ataque por factorización entera

- Objetivo
  - El atacante quiere conocer la clave secreta  $d$  a partir del conocimiento de los valores públicos de la clave, es decir,  $n$  y  $e$
- ¿A qué se enfrenta el atacante?
  - Al Problema de la Factorización Entera PFE
  - La complejidad asociada al problema de la factorización entera para un número  $n$ , viene dada por una ecuación exponencial en donde los bits del número  $n$  a factorizar se encuentran en su exponente, un problema NP
- ¿Por qué esto rompería el algoritmo RSA?
  - Al factorizar  $n$  se obtienen los primos  $p$  y  $q$ , se calcula  $\phi(n) = (p-1)(q-1)$  y se encuentra la clave privada calculando  $d = \text{inv}[e, \phi(n)]$

# Problema de la factorización entera PFE

Módulo 3. Complejidad algorítmica en la criptografía  
Lección 3.4. El problema de la factorización entera



- Temática ya abordada en la Class4crypt c4c3.4 El problema de la factorización entera

[https://www.youtube.com/watch?v=of\\_5ioayJo0](https://www.youtube.com/watch?v=of_5ioayJo0)

# Desafíos de factorización módulos RSA

| RSA number               | Decimal digits | Binary digits | Cash prize offered        | Factored on                        | Factored by   |
|--------------------------|----------------|---------------|---------------------------|------------------------------------|---|
| RSA-100                  | 100            | 330           | US\$1,000 <sup>[8]</sup>  | April 1, 1991 <sup>[9]</sup>       | Arjen K. Lenstra  |
| RSA-110                  | 110            | 364           | US\$4,429 <sup>[8]</sup>  | April 14, 1992 <sup>[9]</sup>      | Arjen K. Lenstra and M.S. Manasse   |
| RSA-120                  | 120            | 397           | US\$5,898 <sup>[8]</sup>  | July 9, 1993 <sup>[10]</sup>       | T. Denny <i>et al.</i>  |
| RSA-129 <sup>[a]</sup>   | 129            | 426           | US\$100                   | April 26, 1994 <sup>[9]</sup>      | Arjen K. Lenstra <i>et al.</i>  |
| RSA-130                  | 130            | 430           | US\$14,527 <sup>[8]</sup> | April 10, 1996                     | Arjen K. Lenstra <i>et al.</i>  |
| RSA-140                  | 140            | 463           | US\$17,226                | February 2, 1999                   | Herman te Riele <i>et al.</i>   |
| RSA-150                  | 150            | 496           |                           | April 16, 2004                     | Kazumaro Aoki <i>et al.</i>   |
| RSA-155                  | 155            | 512           | US\$9,383 <sup>[8]</sup>  | August 22, 1999                    | Herman te Riele <i>et al.</i>   |
| RSA-160                  | 160            | 530           |                           | April 1, 2003                      | Jens Franke <i>et al.</i> , University of Bonn                              |
| RSA-170 <sup>[b]</sup>   | 170            | 563           |                           | December 29, 2009                  | D. Bonenberger and M. Krone <sup>[c]</sup>                                  |
| RSA-576                  | 174            | 576           | US\$10,000                | December 3, 2003                   | Jens Franke <i>et al.</i> , University of Bonn                              |
| RSA-180 <sup>[b]</sup>   | 180            | 596           |                           | May 8, 2010                        | S. A. Danilov and I. A. Popovyan, Moscow State University <sup>[11]</sup>   |
| RSA-190 <sup>[b]</sup>   | 190            | 629           |                           | November 8, 2010                   | A. Timofeev and I. A. Popovyan  |
| RSA-640                  | 193            | 640           | US\$20,000                | November 2, 2005                   | Jens Franke <i>et al.</i> , University of Bonn                              |
| RSA-200 <sup>[b]</sup> ? | 200            | 663           |                           | May 9, 2005                        | Jens Franke <i>et al.</i> , University of Bonn                              |
| RSA-210 <sup>[b]</sup>   | 210            | 696           |                           | September 26, 2013 <sup>[12]</sup> | Ryan Propper  |
| RSA-704 <sup>[b]</sup>   | 212            | 704           | US\$30,000                | July 2, 2012                       | Shi Bai, Emmanuel Thomé and Paul Zimmermann                                 |
| RSA-220 <sup>[b]</sup>   | 220            | 729           |                           | May 13, 2016                       | S. Bai, P. Gaudry, A. Kruppa, E. Thomé and P. Zimmermann                    |
| RSA-230 <sup>[b]</sup>   | 230            | 762           |                           | August 15, 2018                    | Samuel S. Gross, Noblis, Inc. <sup>[d]</sup>                                |
| RSA-232 <sup>[b]</sup>   | 232            | 768           |                           | February 17, 2020 <sup>[13]</sup>  | N. L. Zamarashkin, D. A. Zheltkov and S. A. Matveev                         |
| RSA-768 <sup>[b]</sup>   | 232            | 768           | US\$50,000                | December 12, 2009                  | Thorsten Kleinjung <i>et al.</i> <sup>[14]</sup>                            |
| RSA-240 <sup>[b]</sup>   | 240            | 795           |                           | Dec 2, 2019 <sup>[15]</sup>        | F. Boudot, P. Gaudry, A. Guillevic, N. Heninger, E. Thomé and P. Zimmermann |
| RSA-250 <sup>[b]</sup>   | 250            | 829           |                           | Feb 28, 2020 <sup>[16]</sup>       | F. Boudot, P. Gaudry, A. Guillevic, N. Heninger, E. Thomé and P. Zimmermann |
| RSA-260                  | 260            | 862           |                           |                                    |   |
| RSA-270                  | 270            | 895           |                           |                                    |   |
| RSA-896                  | 270            | 896           | US\$75,000 <sup>[d]</sup> |                                    |   |

Decenas de miles de máquinas en pocos meses

- RSA-250 (829 bits), febrero de 2020
- Fabrice Boudot, Pierrick Gaudry, Aurore Guillevic, Nadia Heninger, Emmanuel Thomé, Paul Zimmermann
- Computadores Francia, USA, Alemania
- “This computation was performed with the Number Field Sieve algorithm, using the open-source CADO-NFS software
- The total computation time was roughly 2700 core-years, using Intel Xeon Gold 6130 CPUs as a reference (2.1GHz):
  - RSA-250 sieving: 2450 physical core-years
  - RSA-250 matrix: 250 physical core-years”

<https://sympa.inria.fr/sympa/arc/cado-nfs/2020-02/msg00001.html>

# Ataque a RSA por cifrado cíclico

- Objetivo
  - Se trata de encontrar, a partir de un criptograma C, el número secreto N que se ha cifrado con la clave pública e de destino o víctima, sin necesidad de conocer la clave privada d de ese receptor
- ¿Cómo se logra esto?
  - Como  $C = N^e \text{ mod } n$ , a partir de este criptograma al que llamaremos  $C_0$ , realizaremos cifrados sucesivos con la clave pública e del receptor hasta volver a obtener el criptograma inicial  $C_0$
- Entonces
  - Si en el cifrado i-ésimo se encuentra nuevamente el criptograma inicial  $C_0$ , lógicamente el cifrado anterior ( $C_{i-1}$ ) será el número secreto N buscado

# Ejemplo de ataque a RSA por cifrado cíclico

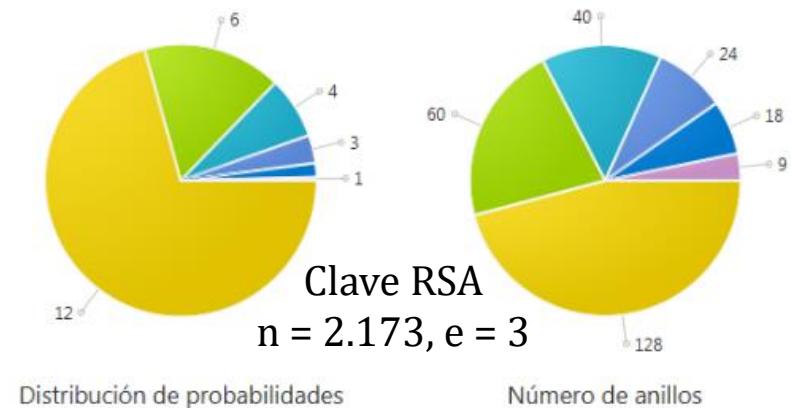
- Sea  $p = 13$ ,  $q = 19$ ,  $n = p \cdot q = 247$  y  $\phi(n) = (p-1)(q-1) = 216$ . Elegimos  $e = 5$  y encontramos como clave privada  $d = 173$ , un valor desconocido
- Sea el número secreto a cifrar  $N = 123$ , entonces  $C = 123^5 \text{ mod } 247 = 54$

|                                     |  |   |
|-------------------------------------|--|---|
| $C_0 = 54$ (valor que se captura)   | $C_4 = 6^5 \text{ mod } 247 = 119$                     | ↓ |
| $C_1 = 54^5 \text{ mod } 247 = 175$ | $C_5 = 119^5 \text{ mod } 247 = 123$                   | ↓ |
| $C_2 = 175^5 \text{ mod } 247 = 93$ | El atacante no se sabe que 123 era el secreto          |   |
| $C_3 = 93^5 \text{ mod } 247 = 6$   | $C_6 = 123^5 \text{ mod } 247 = 54$ (ahora sí lo sabe) |   |

- El ataque ha prosperado rápidamente porque son números muy pequeños
- Si usamos primos seguros, el ataque va a requerir una búsqueda mayor. Sean  $p = 11$  y  $q = 23$  ambos primos seguros,  $n = 253$  y con  $e = 3$ . El secreto 123 que se cifra como  $123^3 \text{ mod } 253 = 52$ , se recupera ahora en la vuelta 9 en vez de 5

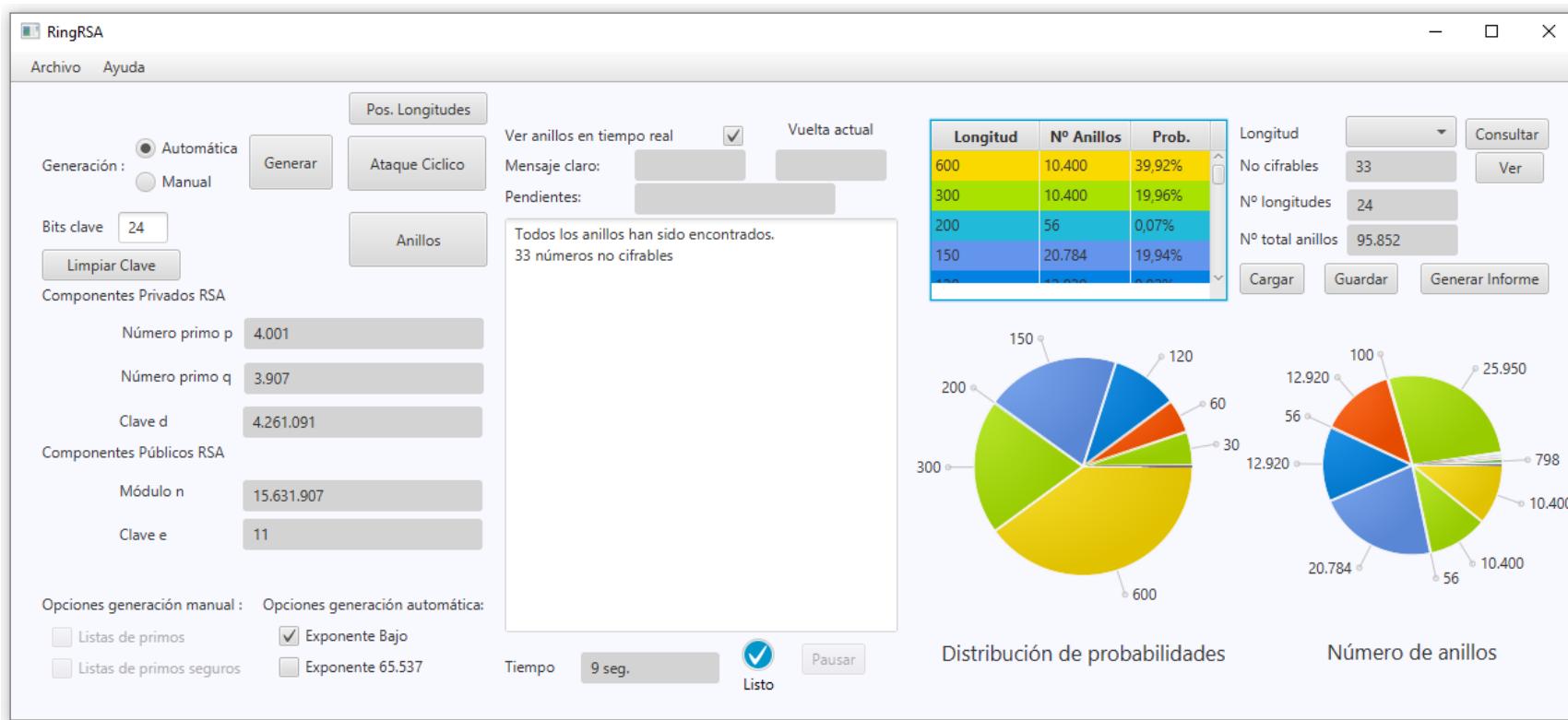
# Formación de varios anillos de números

- Anillos clave RSA  $p = 41$ ,  $q = 53$ ,  $n = 2.173$ ,  $e = 3$
- Al realizar una cifra cíclica  $A^b \text{ mod } n$ , en la que el nuevo resultado  $A'$  (o  $C$ ) se vuelve a cifrar con la misma clave pública  $e$ , se forman anillos de números
- En esos anillos, de longitudes distintas, se van ubicando todos los restos del módulo  $n$  o grupo de cifra, desde 0 hasta  $n-1$
- Habrá anillos de corta longitud (pocos números) y anillos de larga longitud (muchos números)
- Uno de los 128 anillos de longitud 12 es el que se indica con estos doce números  $\{2, 8, 512, 210, 1.847, 336, 1.168, 538, 1.519, 1.853, 840, 866\}$



| Longitud | Nº Anillos | Prob.  |
|----------|------------|--------|
| 12       | 128        | 70,69% |
| 6        | 60         | 16,57% |
| 4        | 40         | 7,36%  |
| 3        | 24         | 3,31%  |
| 2        | 18         | 1,66%  |
| 1        | 9          | 0,41%  |

# Anillos de números con RingRSA: 24 bits

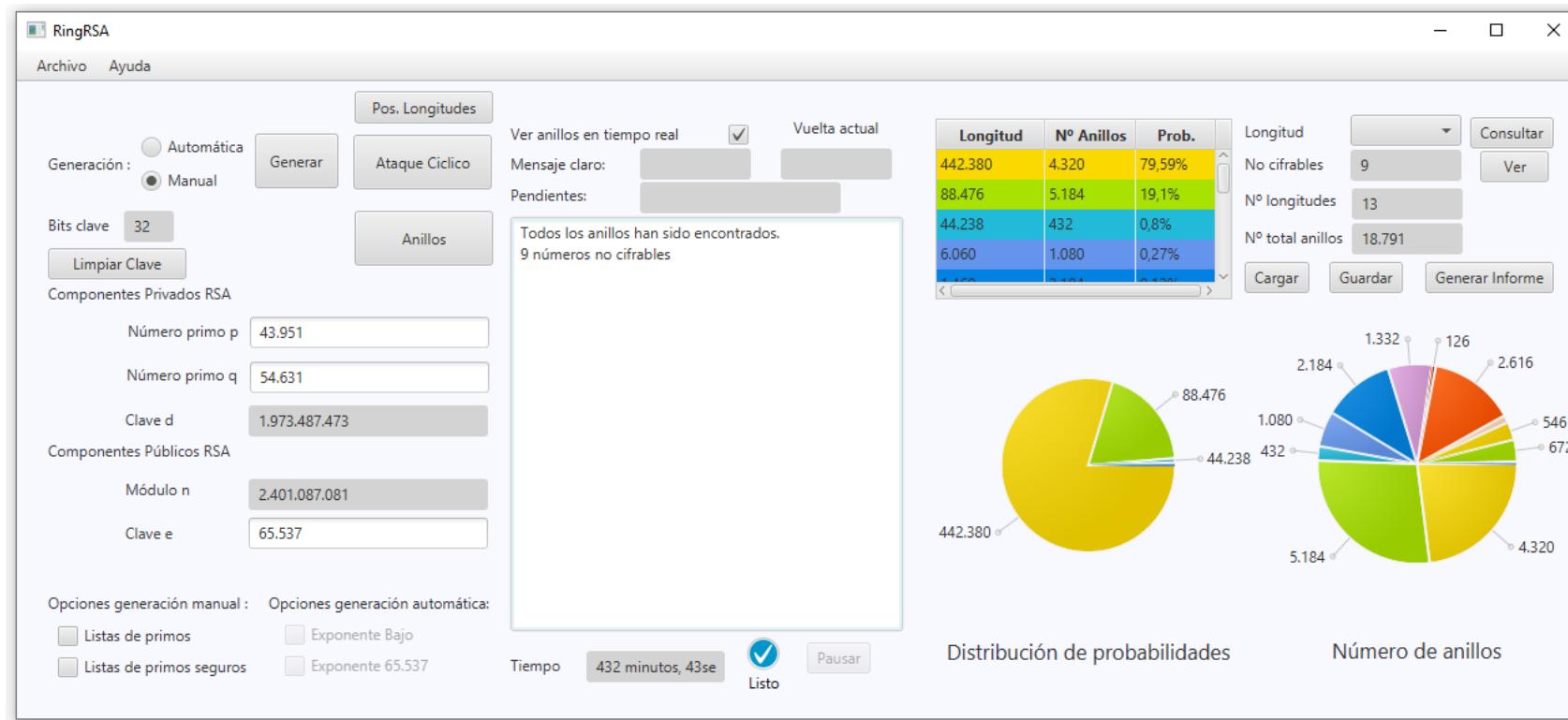


95.852 anillos para clave de 24 bits resuelto en 9 segundos

## DATOS DE LOS ANILLOS

| Longitud | Nº Anillos | Prob.   |
|----------|------------|---------|
| 600      | 10.400     | 39,92%  |
| 300      | 10.400     | 19,96%  |
| 200      | 56         | 0,07%   |
| 150      | 20.784     | 19,94%  |
| 120      | 12.920     | 9,92%   |
| 100      | 56         | 0,04%   |
| 75       | 32         | 0,02%   |
| 60       | 12.920     | 4,96%   |
| 50       | 100        | 0,03%   |
| 40       | 56         | 0,01%   |
| 30       | 25.950     | 4,98%   |
| 25       | 24         | 0,0038% |
| 24       | 400        | 0,06%   |
| 20       | 56         | 0,0072% |
| 15       | 32         | 0,0031% |
| 12       | 400        | 0,03%   |
| 10       | 100        | 0,0064% |
| 8        | 70         | 0,0036% |
| 6        | 798        | 0,03%   |
| 5        | 24         | 0,0008% |
| 4        | 70         | 0,0018% |
| 3        | 44         | 0,0008% |
| 2        | 127        | 0,0016% |
| 1        | 33         | 0,0002% |

# Anillos de números con RingRSA: 32 bits



## DATOS DE LOS ANILLOS

| Longitud | Nº Anillos | Prob.       |
|----------|------------|-------------|
| 442.380  | 4.320      | 79,59%      |
| 88.476   | 5.184      | 19,1%       |
| 44.238   | 432        | 0,8%        |
| 6.060    | 1.080      | 0,27%       |
| 1.460    | 2.184      | 0,13%       |
| 1.212    | 1.332      | 0,07%       |
| 606      | 126        | 0,0032%     |
| 292      | 2.616      | 0,03%       |
| 146      | 228        | 0,0014%     |
| 20       | 546        | 0,0005%     |
| 4        | 672        | 0,0001%     |
| 2        | 62         | 0,00000516% |
| 1        | 9          | 0,00000037% |

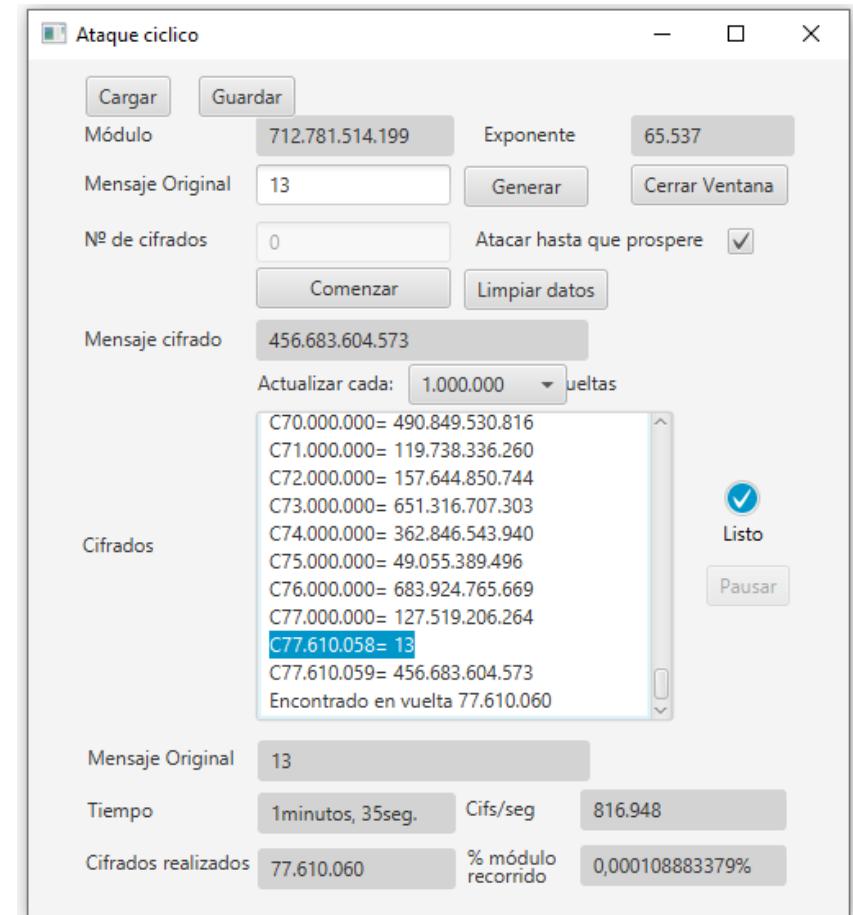
Progreso 100%  
Tiempo 432 minutos, 43seg.  
Nº Longitudes 13  
Nº Total Anillos 18.791

18.791 anillos para clave de 32 bits resuelto en 432 minutos

# Ataque cifrado cíclico a 40 bits: RingRSA

- Sea la clave RSA de 40 bits con
  - $p = 757.507$
  - $q = 940.957$
  - $n = 712.781.514.199$
  - $e = 65.537$
  - $d = 281.742.544.313$
- Y el secreto a cifrar  $N = 13$
- $13^{65.537} \text{ mod } 712.781.514.199 = 456.683.604.573$
- El ataque cíclico al criptograma  $456.683.604.573$  requiere **77.610.060** operaciones en un grupo de cifra  $n$  mayor que 712 mil millones (tan solo el 0,0001 %) y éste se resuelve en 95 segundos de cómputo con una tasa de 820.000 cif/seg

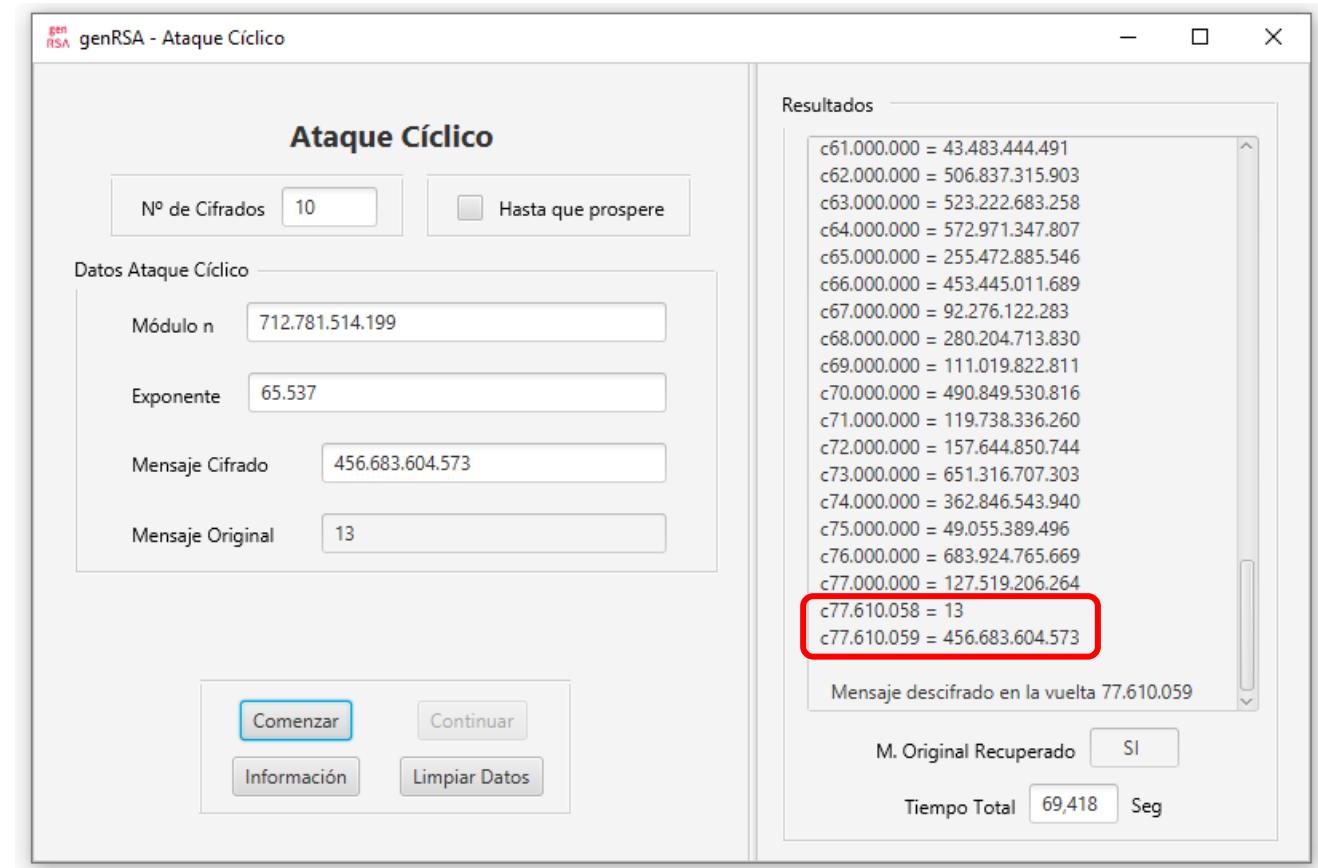
Tanto N como C se encuentran en un anillo de 77.610.060 números



[https://www.criptored.es/software/sw\\_m001q.htm](https://www.criptored.es/software/sw_m001q.htm)

# Ataque cifrado cíclico a 40 bits: genRSA

- Para la misma clave de 40 bits de  $n = 712.781.514.199$ , con clave pública  $e = 65.537$
- Con  $C = 456.683.604.573$ , genRSA v2.1 encuentra el secreto 13 en un tiempo de 69 segundos, con una tasa de cómputo de 1.125.000 cif/seg
- El texto en claro  $N$  y el criptograma  $C$  se encuentran en un anillo de 77.610.060 números diferentes



# Esta clase continúa en la lección 10.9b

- Esta clase continuará en la lección c4c10.9b
- Ataques teóricos y prácticos a RSA parte 2
- De próxima publicación en este canal y en la que se tratarán los siguientes temas:
  - Ataque por la paradoja del cumpleaños
  - Ataque acústico por canal lateral
  - Lectura recomendada de la lección 10.9
  - Conclusiones generales de la lección 10.9

# Class4crypt c4c10.9b

## Módulo 10. Criptografía asimétrica

### Lección 10.9b. Ataques teóricos y prácticos a RSA parte 2

#### 10.9b.1. Resumen de la parte 1

10.9b.1a. Ataque por factorización entera del módulo n

10.9b.1b. Ataque por cifrado cíclico

10.9b.2. Ataque por la paradoja del cumpleaños

10.9b.3. Ataque acústico por canal lateral

10.9b.4. Lectura recomendada y conclusiones de la lección 10.9

Class4crypt c4c10.9b Ataques teóricos y prácticos a RSA parte 2  
<https://www.youtube.com/watch?v=80RIG7xMt8A>

# Ataques a RSA (clase parte a)

- Ataques teóricos conocidos
  - Ataque basado en la factorización del módulo n
  - Ataque por cifrado cíclico
- Temas vistos en la clase anterior

Módulo 10 Criptografía asimétrica  
Lección 10.9 Ataques teóricos y prácticos a RSA parte 1

Class4crypt

Clase c4c10.9a  
12/07/2021



Anillos de números con RingRSA: 24 bits

DATOS DE LOS ANILLOS

| Longitud | Nº Anillos | Prob.   |
|----------|------------|---------|
| 600      | 10 400     | 39,92%  |
| 300      | 10 400     | 19,96%  |
| 200      | 56         | 0,07%   |
| 150      | 20 784     | 19,94%  |
| 120      | 12 920     | 9,92%   |
| 100      | 56         | 0,04%   |
| 75       | 32         | 0,02%   |
| 60       | 12 920     | 4,96%   |
| 50       | 100        | 0,03%   |
| 40       | 56         | 0,01%   |
| 30       | 25 950     | 4,98%   |
| 25       | 24         | 0,0038% |
| 24       | 400        | 0,06%   |
| 20       | 56         | 0,0072% |
| 15       | 32         | 0,0031% |
| 12       | 400        | 0,03%   |
| 10       | 100        | 0,0064% |
| 8        | 70         | 0,0036% |
| 6        | 798        | 0,03%   |
| 5        | 24         | 0,0008% |
| 4        | 70         | 0,0018% |
| 3        | 44         | 0,0008% |
| 2        | 127        | 0,0016% |
| 1        | 33         | 0,0002% |

95.852 anillos para clave de 24 bits resuelto en 9 segundos

RingRSA

Archivo Ayuda

Generación: Automática Manual Generar Ataques Clásicos Analisis

Bits clave: 24

Componentes Privados RSA:

- Número primo p: 4.001
- Número primo q: 4.907
- Clave d: 4.261.091

Componentes Públicos RSA:

- Módulo n: 15.643.097
- Clave e: 11

Opciones generación manual:

- Lista de primos
- Lista de primos seguros

Opciones generación automática:

- Exponente bajo
- Exponente 65.537

Tiempo: 9 seg. Lote: 1

Distribución de probabilidades Número de anillos

Class4crypt c4c10.9a - © jorgeramio 2022

Lección 10.9a - página 17

<https://www.youtube.com/watch?v=7XI8WzWu4kE>

# Ataques a RSA (clase parte b)

- Ataques teóricos más conocidos (continuación)
  - Ataque por la paradoja del cumpleaños
    - Conociendo solamente los datos públicos de la víctima, es posible encontrar la clave privada d o una clave privada pareja d' aunque pueden aparecer falsos positivos muy poco frecuentes
    - Existen otros ataques: de exponente público e bajo, de módulo n en común, etc.
- Ataques prácticos
  - Ataque acústico por canal lateral
    - La ejecución del algoritmo de exponenciación modular provoca sonidos entre los 30 KHz y los 40 KHz que permiten descubrir los bits ceros y unos de la clave privada d
- Lectura recomendada
- Conclusiones generales de la lección 10.9

# Qué era la paradoja del cumpleaños



- La paradoja del cumpleaños dice que para tener confianza, esto es una probabilidad igual o mayor que el 50%, en que dos personas al azar estén de cumpleaños el mismo día, es suficiente con que haya 23 personas en el grupo
- Al haber 365 días al año, este resultado estadístico nos produce cierta perplejidad y por eso se le llama paradoja. No obstante, no es una paradoja
- Escenario práctico:
  - En un aula marcamos en una pizarra con recuadros los 365 días del año (no bisiesto)
  - Un grupo de personas vamos entrando al aula de uno en uno y vamos marcando nuestra fecha de cumpleaños, hasta que haya una colisión, es decir, dos cumpleaños en igual fecha, y apuntamos número el de personas
  - La persona que entra en la posición 23 ya tendrá más de un 50% de probabilidad (es decir confianza, que no certeza) de que su casilla en la pizarra ya esté marcada. Lógicamente, quien primero entra tiene 365 posiciones libres, quien entra en segundo lugar tiene 364 posiciones libres, y así sucesivamente
  - Se repite este proceso varias veces para obtener la media de personas necesarias para una colisión
  - Se trata de una serie y para llegar a esa confianza bastará con que haya de media unas 23 personas, un valor que es aproximadamente igual a  $365^{1/2}$ , es decir, la raíz cuadrada de todas las posibilidades de cumpleaños
- Si el universo de valores binarios posibles fuese  $n$ , desde este punto de vista no habría que realizar  $2^n$  intentos (o  $2^{n-1}$  intentos de media), sino que bastaría con una búsqueda en un espacio igual a  $2^{n/2}$ , muchísimo menor si  $n$  es grande

# Ataque a RSA por paradoja del cumpleaños

Algoritmo propuesto por Merkle y Hellman en 1981

1. El atacante elige dos números aleatorios distintos  $[i, j]$  dentro del módulo  $n$ ; normalmente  $i = 0$  y  $j$  la parte entera de  $n/2$ . Con ello divide el grupo de cifra  $n$  en dos mitades, la izquierda  $i$  y la derecha  $j$
2. Lo interesante es que ahora no necesita capturar ningún valor sino que elige un número  $N$  cualquiera para iniciar el ataque, siendo recomendable usar  $N = 2$  para una mayor rapidez en los cálculos
3. Con  $i = 0$  se calcula  $N^i \bmod n$  y con  $j = n/2$  se calcula  $N^j \bmod n$ , incrementando a continuación  $i$  y  $j$  en una unidad, y repitiendo el proceso sucesivamente hasta que haya una colisión, es decir, que uno de los resultados del espacio  $i$  coincida con uno de los resultados del espacio  $j$ . Como en este caso de cifra se formará otro tipo de anillo de números, bastará que el primer resultado del espacio  $j$  coincida con uno de los resultados del espacio  $i$ , o viceversa, el primer resultado del espacio  $i$  con uno de  $j$
4. Esto último permite parallelizar este ataque, es decir, aplicar divide y vencerás, al igual que sucedía en los ataques de la cifra simétrica en bloque, utilizado por ejemplo en el DES Challenge
5. Cuando se encuentra una coincidencia para una pareja  $[i, j]$ , el atacante podrá obtener la clave privada  $d$ , alguna de las claves privadas parejas o, en muy pocos casos, encontrar un falso positivo

# Ataque por paradoja del cumpleaños



Sea  $p = 13$ ,  $q = 19$  y  
 $d = 59$  (privados)

El atacante solo conoce  $n = 247$  y la clave pública  $e = 11$

El atacante usará el valor  $N = 2$  y elige los valores iniciales  $i = 0$  y  $j = 124$

Hace 17 cálculos en  $i$  (desde  $i=0$  a  $i=16$ ) y 17 cálculos en  $j$  (desde  $j=124$  hasta  $j=140$ ) y aparece la primera colisión, el criptograma  $C = 81$

|        |                                 |
|--------|---------------------------------|
| $i=0$  | $2^0 \text{ mod } 247 = 1$      |
| $i=1$  | $2^1 \text{ mod } 247 = 2$      |
| $i=2$  | $2^2 \text{ mod } 247 = 4$      |
| $i=3$  | $2^3 \text{ mod } 247 = 8$      |
| $i=4$  | $2^4 \text{ mod } 247 = 16$     |
| $i=5$  | $2^5 \text{ mod } 247 = 32$     |
| $i=6$  | $2^6 \text{ mod } 247 = 64$     |
| $i=7$  | $2^7 \text{ mod } 247 = 128$    |
| $i=8$  | $2^8 \text{ mod } 247 = 9$      |
| $i=9$  | $2^9 \text{ mod } 247 = 18$     |
| $i=10$ | $2^{10} \text{ mod } 247 = 36$  |
| $i=11$ | $2^{11} \text{ mod } 247 = 72$  |
| $i=12$ | $2^{12} \text{ mod } 247 = 144$ |
| $i=13$ | $2^{13} \text{ mod } 247 = 41$  |

|        |                                 |
|--------|---------------------------------|
| $i=14$ | $2^{14} \text{ mod } 247 = 82$  |
| $i=15$ | $2^{15} \text{ mod } 247 = 164$ |
| $i=16$ | $2^{16} \text{ mod } 247 = 81$  |
| $i=17$ | $2^{17} \text{ mod } 247 = 162$ |
| $i=18$ | $2^{18} \text{ mod } 247 = 77$  |

Hay una colisión con el mismo resultado  $C = 81$  para  $i = 16$  y para  $j = 124$

A partir de ese punto, las dos secuencias están encadenadas y entregan los mismos resultados: 162, 77, 154, 61, 122, etc.

Se formará un anillo menor que  $n$ , como en descifrado con CPP y números piratas

|         |                                  |
|---------|----------------------------------|
| $j=124$ | $2^{124} \text{ mod } 247 = 81$  |
| $j=125$ | $2^{125} \text{ mod } 247 = 162$ |
| $j=126$ | $2^{126} \text{ mod } 247 = 77$  |
| $j=127$ | $2^{127} \text{ mod } 247 = 154$ |
| $j=128$ | $2^{128} \text{ mod } 247 = 61$  |
| $j=129$ | $2^{129} \text{ mod } 247 = 122$ |
| $j=130$ | $2^{130} \text{ mod } 247 = 244$ |
| $j=131$ | $2^{131} \text{ mod } 247 = 241$ |
| $j=132$ | $2^{132} \text{ mod } 247 = 235$ |
| $j=133$ | $2^{133} \text{ mod } 247 = 223$ |
| $j=134$ | $2^{134} \text{ mod } 247 = 199$ |
| $j=135$ | $2^{135} \text{ mod } 247 = 151$ |
| $j=136$ | $2^{136} \text{ mod } 247 = 55$  |
| $j=137$ | $2^{137} \text{ mod } 247 = 110$ |

|         |                                  |
|---------|----------------------------------|
| $j=138$ | $2^{138} \text{ mod } 247 = 220$ |
| $j=139$ | $2^{139} \text{ mod } 247 = 193$ |
| $j=140$ | $2^{140} \text{ mod } 247 = 139$ |
| $j=141$ | $2^{141} \text{ mod } 247 = 31$  |
| $j=142$ | $2^{142} \text{ mod } 247 = 62$  |

El ataque logra encontrar la clave privada  $d = 59$

La zona  $i$  y la zona  $j$  pueden dividirse en bloques

Divide y vencerás: a los bloques de la zona  $i$  se les entrega como target el primer valor de la zona  $j$  y a los bloques de la zona  $j$  se les entrega como target el primer valor de la zona  $i$

# Resolviendo el ataque por paradoja

- En el ejemplo anterior, la colisión se da para  $i = 16$  y  $j = 124$
- Como el atacante conoce la clave pública  $e = 11$ , calcula:
  - $w = (i - j)/\text{mcd}(e, |i - j|) = (16 - 124)/\text{mcd}(11, |16-124|)$
  - $w = -108/\text{mcd}(11, 108) = -108/1 = -108$
- Existirán entonces valores  $s$  y  $t$  de que forma que se cumpla
  - $w*s + e*t = 1$                        $-108*s + 11*t = 1$
  - Por una parte:                           $w*s \bmod e = 1$
  - $-108*s \bmod 11$                          $s = \text{inv}(-108, 11) = \text{inv}(2, 11) = 6$
  - Por otra parte                             $e*t \bmod w = 1$
  - $11*t \bmod 108$                          $t = \text{inv}(11, 108) = 59$  (que es la clave privada)
-  Vía rápida:  $d = \text{inv}(e, w)$  [privada, privada pareja o falso positivo]

# También aparece un anillo de números



- Al igual que en el descifrado con todos los valores de n, en el ataque por paradoja del cumpleaños también se crea un anillo de números, como se ve en la figura, con esta secuencia de 36 números: 1, 2, 4, 8, ... 139, 31, 62, 124
- Al contrario de lo que sucedía con los anillos del cifrado cíclico, la base A se mantiene constante y lo que cambia es el exponente b en  $A^b \bmod n$

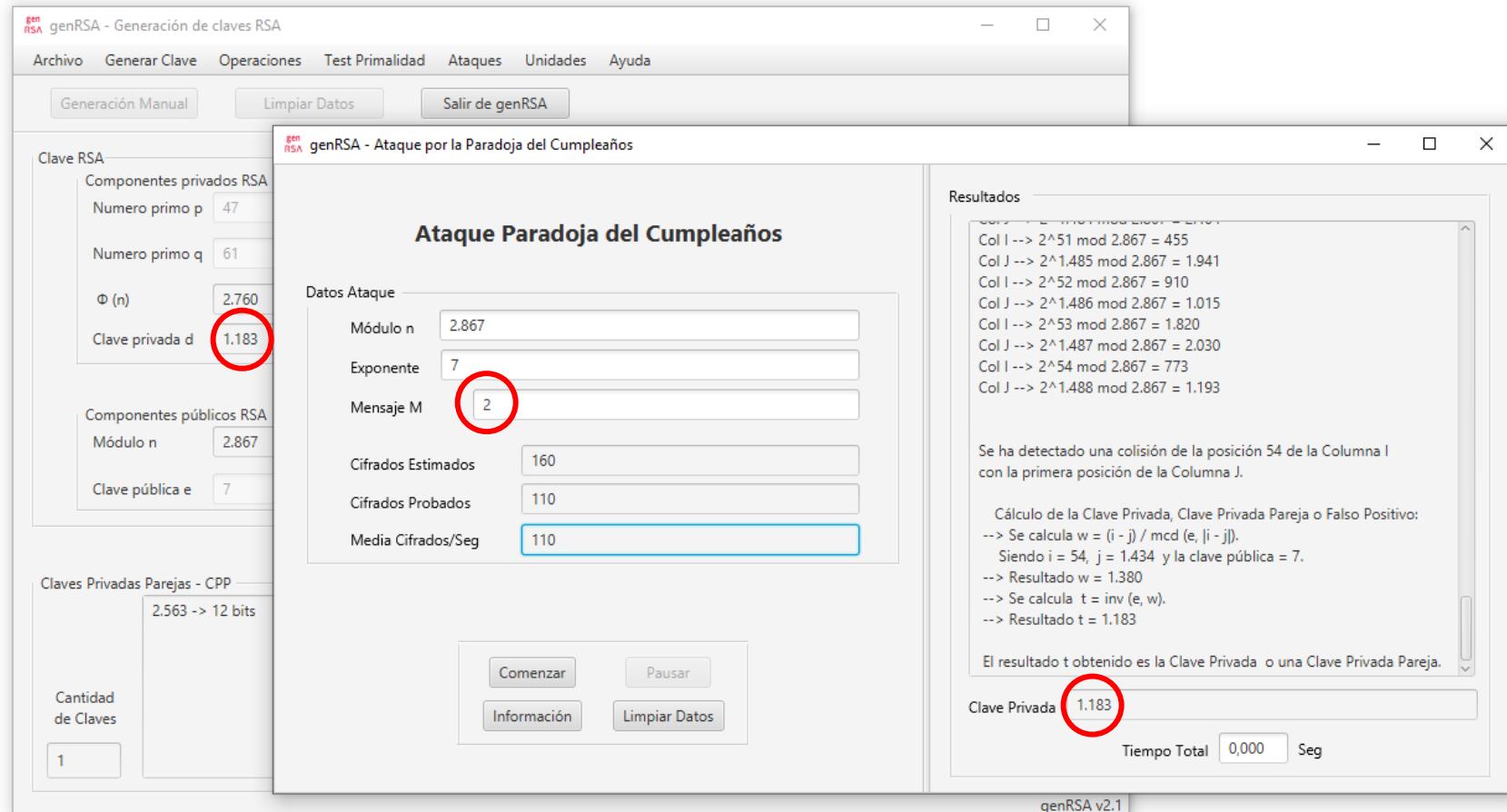
|      |                          |      |                          |      |                          |
|------|--------------------------|------|--------------------------|------|--------------------------|
| i=0  | $2^0 \bmod 247 = 1$      | i=14 | $2^{14} \bmod 247 = 82$  | i=28 | $2^{28} \bmod 247 = 55$  |
| i=1  | $2^1 \bmod 247 = 2$      | i=15 | $2^{15} \bmod 247 = 164$ | i=29 | $2^{29} \bmod 247 = 110$ |
| i=2  | $2^2 \bmod 247 = 4$      | i=16 | $2^{16} \bmod 247 = 81$  | i=30 | $2^{30} \bmod 247 = 220$ |
| i=3  | $2^3 \bmod 247 = 8$      | i=17 | $2^{17} \bmod 247 = 162$ | i=31 | $2^{31} \bmod 247 = 193$ |
| i=4  | $2^4 \bmod 247 = 16$     | i=18 | $2^{18} \bmod 247 = 77$  | i=32 | $2^{32} \bmod 247 = 139$ |
| i=5  | $2^5 \bmod 247 = 32$     | i=19 | $2^{19} \bmod 247 = 154$ | i=33 | $2^{33} \bmod 247 = 31$  |
| i=6  | $2^6 \bmod 247 = 64$     | i=20 | $2^{20} \bmod 247 = 61$  | i=34 | $2^{34} \bmod 247 = 62$  |
| i=7  | $2^7 \bmod 247 = 128$    | i=21 | $2^{21} \bmod 247 = 122$ | i=35 | $2^{35} \bmod 247 = 124$ |
| i=8  | $2^8 \bmod 247 = 9$      | i=22 | $2^{22} \bmod 247 = 244$ | i=36 | $2^{36} \bmod 247 = 1$   |
| i=9  | $2^9 \bmod 247 = 18$     | i=23 | $2^{23} \bmod 247 = 241$ | i=37 | $2^{37} \bmod 247 = 2$   |
| i=10 | $2^{10} \bmod 247 = 36$  | i=24 | $2^{24} \bmod 247 = 235$ | i=38 | $2^{38} \bmod 247 = 4$   |
| i=11 | $2^{11} \bmod 247 = 72$  | i=25 | $2^{25} \bmod 247 = 223$ | i=39 | $2^{39} \bmod 247 = 8$   |
| i=12 | $2^{12} \bmod 247 = 144$ | i=26 | $2^{26} \bmod 247 = 199$ | i=40 | $2^{40} \bmod 247 = 16$  |
| i=13 | $2^{13} \bmod 247 = 41$  | i=27 | $2^{27} \bmod 247 = 151$ | i=41 | $2^{41} \bmod 247 = 32$  |



# Posibilidad de paralelizar este ataque

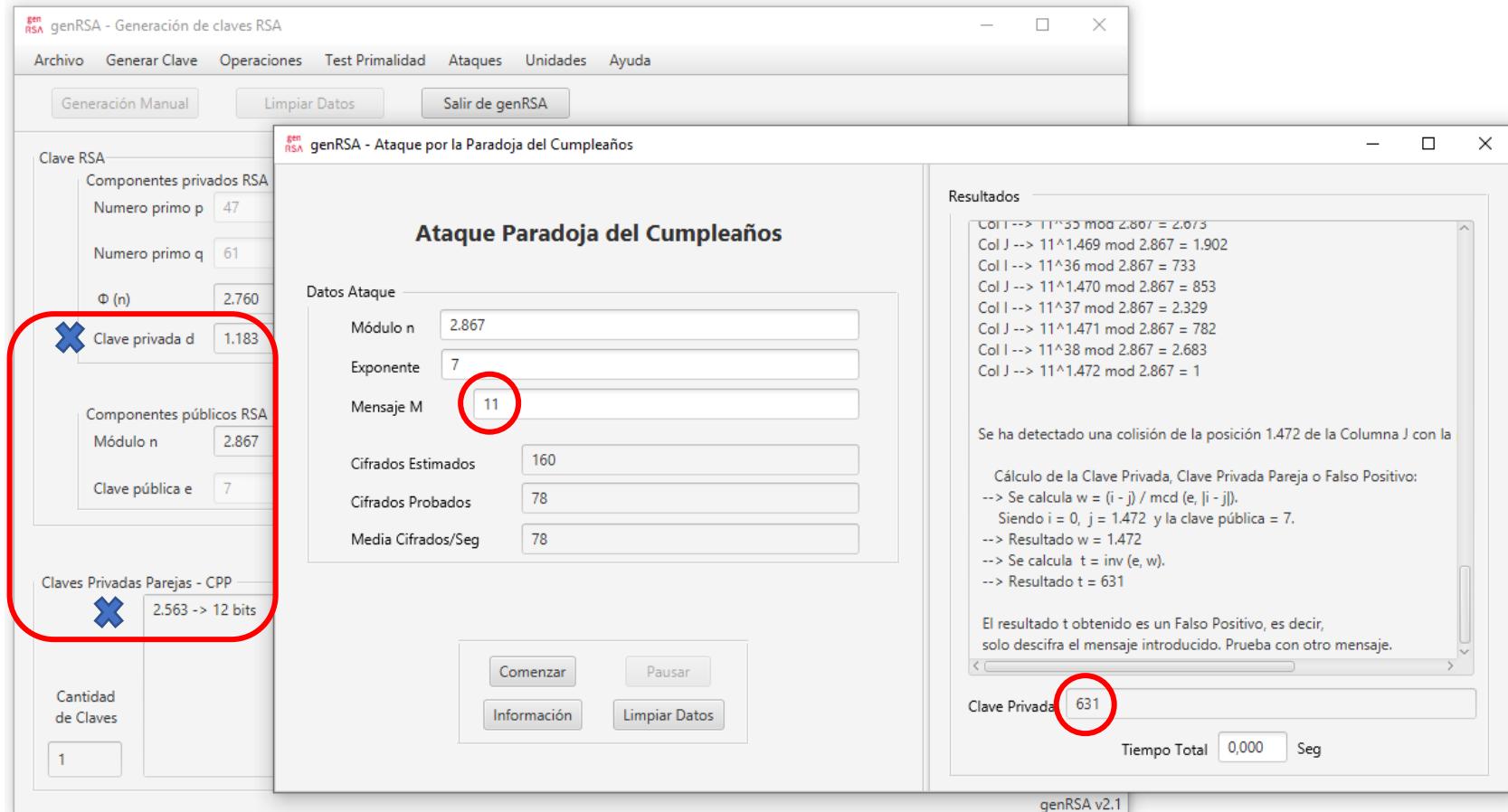
- El ataque basado en la paradoja del cumpleaños no sería factible realizarlo en un solo PC para RSA de 2.048 bits por el alto coste computacional, pues va a requerir  $\sqrt{2^{2.048}} = 2^{1.024}$  operaciones en media
- Pero puede realizarse un ataque distribuido en red con un servidor y muchas máquinas trabajando como clientes
- El servidor tendrá como función distribuir entre las máquinas de los clientes unos bloques de cifra para diferentes intervalos de valores en i y en j
- Se puede asegurar que la colisión siempre existirá
  - Entre un resultado intermedio de i y el primer resultado de j, o bien entre un resultado intermedio de j y el primer resultado de i
  - Por lo tanto, se les entrega a las máquinas clientes i como *target* el primer valor de j, y a las máquinas clientes j como *target* el primer valor de i

# Paradoja cumpleaños con genRSA (CP)



- Para  $n = 2.867$  con  $e = 7$  y  $M = 2, 3, 4, 5, 6, 7, 8, 9$  y  $10$  el ataque por paradoja del cumpleaños nos entrega la clave privada  $d = 1.183$

# Paradoja cumpleaños con genRSA (FP)



- Para  $n = 2.867$ , con  $e = 7$  y ahora con  $M = 11$ , el ataque por la paradoja del cumpleaños nos entrega la clave 631 y se trata de un falso positivo: solo descifra el criptograma si se ha cifrado  $M = 11$

# Comprobación de un falso positivo

The image displays two windows of the SAMCript software, both titled "Potencia".

**Left Window (Initial State):**

- Base: 11 (2 dígitos)
- Exponente: 7 (1 dígito)
- Módulo: 2.867 (4 dígitos) - checked
- Resultado: 172 (3 dígitos)

**Right Window (After Modification):**

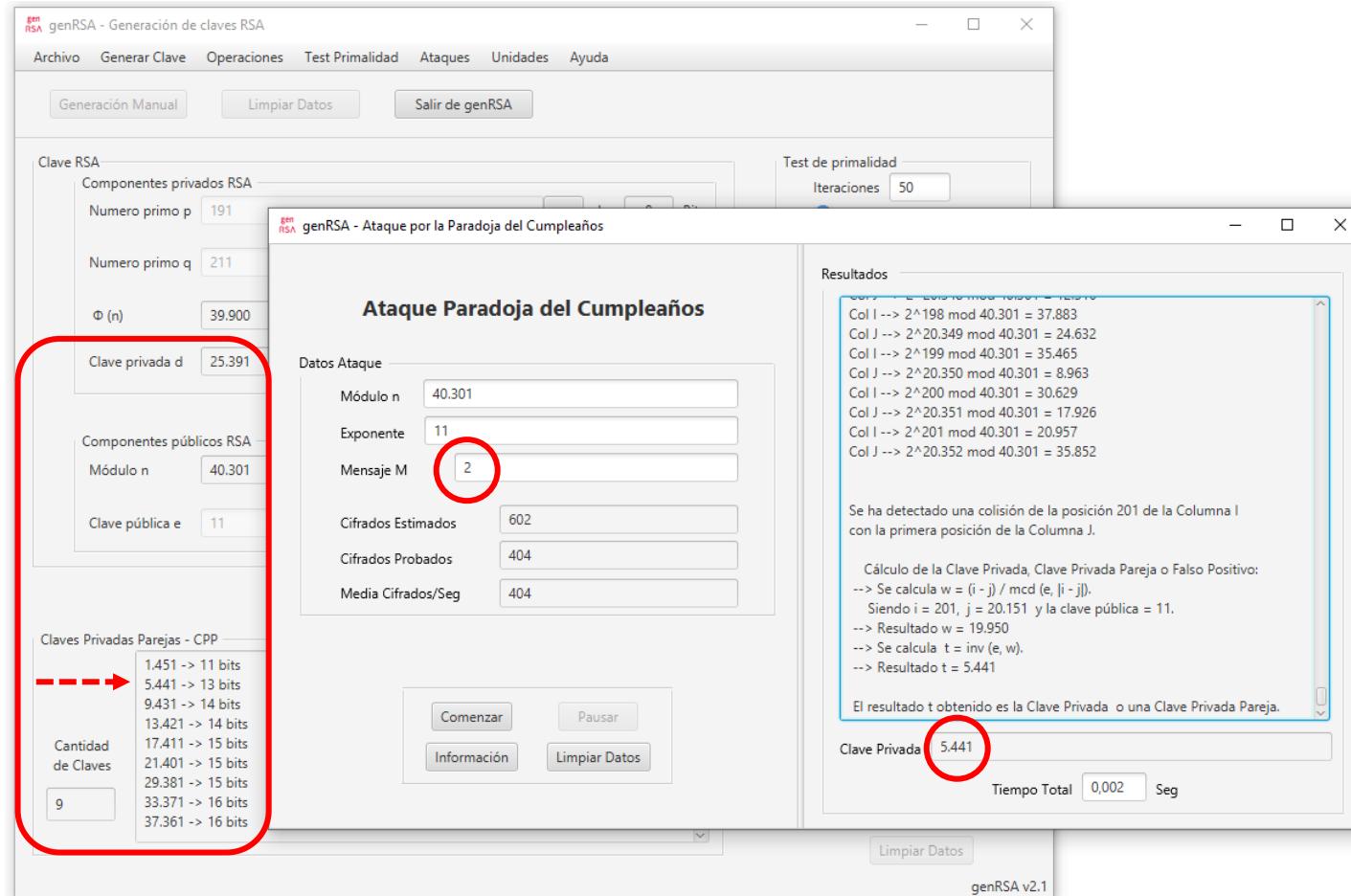
- Base: 172 (3 dígitos)
- Exponente: 631 (3 dígitos) - circled in red
- Módulo: 2.867 (4 dígitos) - checked
- Resultado: 11 (2 dígitos)

Red arrows indicate the flow of data from the initial result back to the base and exponent fields in the second window.

[https://www.criptored.es/software/sw\\_m001t.htm](https://www.criptored.es/software/sw_m001t.htm)

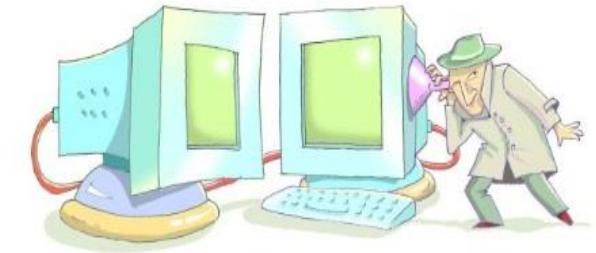
- El criptograma 172, resultado de la cifra de  $M = 11$ , se puede descifrar con 631, que no es la clave privada ni tampoco una clave privada pareja
- En algunos casos, un falso positivo es alguno de aquellos “números piratas”

# Paradoja cumpleaños con genRSA (CPP)



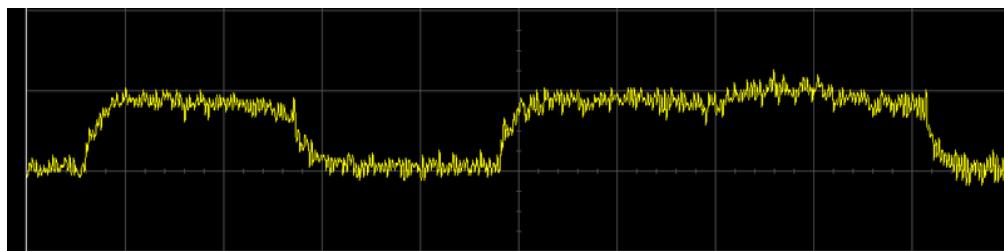
- Para la clave  $n = 40.301$  ( $p = 191$ ,  $q = 211$ ) con clave pública  $e = 11$ , el ataque por paradoja del cumpleaños con  $N = 2$  nos devuelve la clave privada pareja 5.441
- Para  $M = 7$  entrega 7.331,  $M = 14$  entrega 14.675 y  $M = 23$  entrega 1.831, todos falsos positivos

# Ataque acústico por canal lateral



## Ataques por canal lateral

- Consumo de energía
- Radiaciones electromagnéticas
- Emisiones de ruido
- Disipación de calor, etc.



Efecto acústico algoritmo de exponentiación rápida AER en un PC: a la izquierda operación  $x^2 \text{ mod } n$ , a la derecha operación  $x^2 * A \text{ mod } n$

<https://www.cs.tau.ac.il/~tromer/acoustic/>

Daniel Genkin, Adi Shamir y Eran Tromer, Israel, 2014

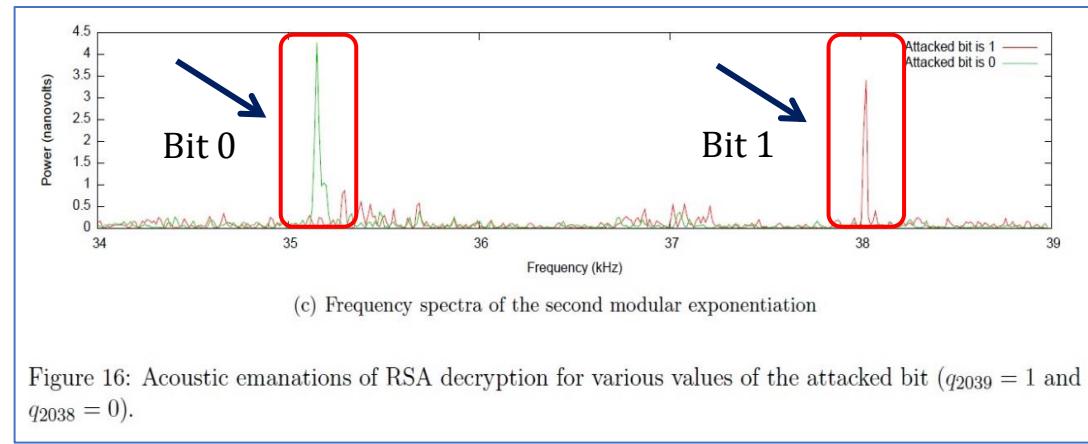


Figure 16: Acoustic emanations of RSA decryption for various values of the attacked bit ( $q_{2039} = 1$  and  $q_{2038} = 0$ ).



# Más información en píldoras Thoth



Píldora formativa 41: ¿Cómo podemos atacar al algoritmo RSA?

<https://www.youtube.com/watch?v=0D8uAk5VFcc>

# Conclusiones de la Lección 10.9

- El PFE no se ha resuelto, sigue teniendo complejidad NP y al año 2021 ni siquiera se ha podido factorizar números del tipo  $n = p*q$  de mil bits
- En el ataque por cifrado cíclico, aunque se forman anillos de números de varios tamaños, y algunos podrían ser pequeños, la mayoría serán anillos con muchos números, y para claves de 2.048 bits esta cantidad será inmensa
- En el ataque por la paradoja del cumpleaños, aunque éste se puede parallelizar y lograr un divide y vencerás en una ataque en red, buscando sólo en el espacio de claves  $\sqrt{n}$ , este último número sigue siendo inmenso, del orden de  $2^{1.024}$  valores
- Los ataques por canal lateral son los más peligrosos, porque no se centran en el algoritmo en sí sino en las manifestaciones físicas de su ejecución en un equipo informático. No obstante, si se conoce el efecto, se puede enmascarar como sucedió en el ataque acústico de 2014 de Genkin, Shamir y Tromer

# Lectura recomendada (1/3)

- Curso de Criptografía Aplicada, Jorge Ramió, 2018
  - [https://www.criptored.es/guiateoria/gt\\_m001s1.htm](https://www.criptored.es/guiateoria/gt_m001s1.htm)
- Libro Electrónico de Seguridad Informática y Criptografía, versión 4.1, Jorge Ramió, 2006
  - [https://www.criptored.es/guiateoria/gt\\_m001a.htm](https://www.criptored.es/guiateoria/gt_m001a.htm)
- Guion píldora formativa Thoth nº 41, ¿Cómo podemos atacar al algoritmo RSA?, Jorge Ramió, 2016
  - <https://www.criptored.es/thoth/material/texto/pildora041.pdf>
- CLCript 07: Ataque por paradoja de cumpleaños a RSA, parte 1
  - [https://www.criptored.es/descarga/CLCript\\_entrega\\_07\\_Ataque\\_RSA\\_Paradoja\\_Cumple%C3%B1os\\_1.pdf](https://www.criptored.es/descarga/CLCript_entrega_07_Ataque_RSA_Paradoja_Cumple%C3%B1os_1.pdf)

# Lectura recomendada (2/3)

- CLCript 08: Ataque por paradoja de cumpleaños a RSA, parte 2
  - [https://www.criptored.es/descarga/CLCript\\_entrega\\_08\\_Ataque\\_RSA\\_Paradoja\\_Cumplea%C3%B1os\\_2.pdf](https://www.criptored.es/descarga/CLCript_entrega_08_Ataque_RSA_Paradoja_Cumplea%C3%B1os_2.pdf)
- CLCript 10: Ataque por cifrado cíclico a RSA con genRSA v2.1
  - [https://www.criptored.es/descarga/CLCript\\_entrega\\_10\\_Ataque\\_Cifrado\\_Ciclico\\_con\\_genRSA.pdf](https://www.criptored.es/descarga/CLCript_entrega_10_Ataque_Cifrado_Ciclico_con_genRSA.pdf)
- CLCript 11: Anillos en ataque por cifrado cíclico a RSA con RingRSA
  - [https://www.criptored.es/descarga/CLCript\\_entrega\\_11\\_Anillos\\_Ataque\\_Cifrado\\_Ciclico\\_con\\_RingRSA.pdf](https://www.criptored.es/descarga/CLCript_entrega_11_Anillos_Ataque_Cifrado_Ciclico_con_RingRSA.pdf)
- CLCript 14: Curiosidades y Falsos Positivos en Ataques por Paradoja del Cumpleaños a RSA
  - [https://www.criptored.es/descarga/CLCript\\_entrega\\_14\\_Curiosidades\\_y\\_Falsos\\_Positivos\\_Ataque\\_Paradoja\\_RSA.pdf](https://www.criptored.es/descarga/CLCript_entrega_14_Curiosidades_y_Falsos_Positivos_Ataque_Paradoja_RSA.pdf)

# Lectura recomendada (3/3)

- MOOC El algoritmo RSA, Jorge Ramió, 2012
  - <https://www.criptored.es/crypt4you/temas/RSA/leccion0/leccion00.html>
- Primeros avances en el estudio de anillos en ataques cíclicos al criptosistema RSA, XIII Reunión Española sobre Criptología y Seguridad de la Información, España, Juan Pedro Hecht, Ramió Aguirre, Abel Casado, 2014
  - [https://www.researchgate.net/publication/274380830\\_Primeros\\_avances\\_en\\_el\\_estudio\\_de\\_anillos\\_en\\_ataques\\_ciclicos\\_al\\_criptosistema\\_RSA](https://www.researchgate.net/publication/274380830_Primeros_avances_en_el_estudio_de_anillos_en_ataques_ciclicos_al_criptosistema_RSA)
- RSA Factoring Challenge, Wikipedia
  - [https://en.wikipedia.org/wiki/RSA\\_Factoring\\_Challenge](https://en.wikipedia.org/wiki/RSA_Factoring_Challenge)
- Weaknesses in RSA, RSA Algorithm, DI Management Services, Australia
  - [https://www.di-mgt.com.au/rsa\\_alg.html#weaknesses](https://www.di-mgt.com.au/rsa_alg.html#weaknesses)

# Class4crypt c4c10.10

## Módulo 10. Criptografía asimétrica

### Lección 10.10. Algoritmo de cifra de Elgamal

10.10.1. Breve repaso del PLD e intercambio de clave de Diffie y Hellman

10.10.2. La figura de Taher Elgamal

10.10.3. Generación de claves en el algoritmo de Elgamal

10.10.4. Cifrado de Elgamal

10.10.5. Descifrado de Elgamal

10.10.6. Operaciones de cifra sobre números y textos

Class4crypt c4c10.10 Algoritmo de cifra de Elgamal  
<https://www.youtube.com/watch?v=n7UFUePUW3Q>

# Recordando el PLD e intercambio clave DH

- Problema del logaritmo discreto PLD
  - $\beta = \alpha^x \text{ mod } p$ , con  $x$  como variable, se resuelve en un tiempo polinomial (P)
  - Pero  $x = \log_{\alpha} \beta \text{ mod } p$ , tiene una solución polinomial no determinista (NP)
- Intercambio de clave de Diffie y Hellman
  - En este problema PLD, de muy difícil solución para primos muy grandes, se basaba la seguridad computacional del intercambio de clave de Diffie y Hellman propuesto en noviembre de 1976, que dio inicio a la criptografía de clave pública
  - Si Alicia usaba  $x = a$  y Bernardo usaba  $x = b$ , intercambiaban  $K = \alpha^{ab} \text{ mod } p$
  - En este intercambio de clave, puede decirse que el valor  $\beta$  ( $\beta_A$  para Alicia y  $\beta_B$  para Bernardo) hace las veces de clave pública (por todos conocida) y el valor  $x$  ( $a$  y  $b$ ) hace las veces de clave privada (conocida solo por su dueño)
- Basado en este principio, Taher Elgamal propone su sistema de cifra en 1985

# La figura de Taher Elgamal

- En 1985, el investigador egipcio Taher Elgamal, popularmente conocido como el padre de SSL Secure Socket Layer y actualmente residente en los Estados Unidos, propone sendos algoritmos de cifra y firma digital que llevan su nombre
- Su seguridad reside en el PLD, al igual que en el intercambio de clave de Diffie y Hellman
- Aunque el algoritmo es muy seguro, no logra desbancar a RSA como estándar de cifra
- No obstante, una variación de su algoritmo de firma, conocido como DSA Digital Signature Algorithm, se establece como estándar de firma digital por el NIST en 1994



# Generación de claves en Elgamal

- Cada usuario elige un primo  $p$  muy grande como cuerpo de cifra, un valor que será público
- En el cuerpo  $p$  se elige un valor  $\alpha$  raíz primitiva o generador del cuerpo, que también será público. Es recomendable, no obligatorio, que  $\alpha$  sea una raíz
- Cada usuario elige un número aleatorio secreto  $\lambda$  dentro de  $p$ 
  - Su clave privada  $K_{priv}$  será el valor  $\lambda$
- Cada usuario calcula su clave pública  $K_{pub} = \alpha^\lambda \text{ mod } p$ 
  - Los valores  $K_{pub}$ ,  $p$  y  $\alpha$  formarán los datos públicos del usuario
- Seguridad del sistema
  - Para descubrir la clave privada  $\lambda$ , el atacante deberá enfrentarse al problema del logaritmo discreto PLD, intratable para valores de  $p$  grandes

# Operación de cifrado con Elgamal

- Alicia (A) desea enviar un número secreto  $N$  a Bernardo (B)
- Bernardo ha elegido un primo  $p_B$ , una raíz primitiva  $\alpha_B$  y un número secreto  $b$  dentro de  $p_B$  que será su clave privada  $K_{\text{pri}B} = b$ , con el que calcula su clave pública  $K_{\text{púb}B} = \alpha_B^b \text{ mod } p_B$
- Datos públicos de Bernardo:  $p_B$ ,  $\alpha_B$  y  $K_{\text{púb}B}$
- Con  $\alpha_B$  y  $p_B$ , Alicia usará un número aleatorio cualquiera  $v$  (nu) de sesión y calculará  $N_1$ :
  - $N_1 = \alpha_B^v \text{ mod } p_B$
- Y ahora con la clave pública de Bernardo  $K_{\text{púb}B}$ , Alicia calculará  $N_2$ :
  - $N_2 = N * (K_{\text{púb}B})^v \text{ mod } p_B = N * (\alpha_B^b \text{ mod } p_B)^v \text{ mod } p_B$
- Alicia envía a Bernardo el par  $[N_1, N_2]$ , es decir,  $[\alpha_B^v \text{ mod } p_B, N * (\alpha_B^b)^v \text{ mod } p_B]$
- Observa que solamente Bernardo puede hacer la operación  $(N_1)^b \text{ mod } p_B$  con su clave secreta  $b$  para obtener  $[(\alpha_B^v)]^b \text{ mod } p_B = [(\alpha_B^b)]^v \text{ mod } p_B$  (parte de  $N_2$ ) y con ello puede despejar el valor secreto  $N$ , simplemente usando el concepto de inversos en ese valor  $N_2$  recibido

# Operación de descifrado con Elgamal

- Datos públicos de Bernardo
  - $p_B, \alpha_B$  y  $K_{\text{púbB}} = \alpha_B^b \mod p_B$
- Datos secretos de Bernardo
  - $K_{\text{privB}} = b$
- Bernardo recibe el par  $[N_1, N_2] = [(\alpha_B^v) \mod p_B, N * (\alpha_B^b)^v \mod p_B]$
- Bernardo calcula  $(N_1)^b \mod p_B = (\alpha_B^v)^b \mod p_B = N_3$ 
  - $N_3 = (\alpha_B^v)^b \mod p_B = (\alpha_B^b)^v \mod p_B$  (parte del número  $N_2$  que ha recibido Bernardo)
- Es decir,  $N_3 = (\alpha_B^v)^b \mod p_B$  y  $N_2 = N * (\alpha_B^b)^v \mod p_B$  sólo difieren en el secreto  $N$  ( $N_2 = N * N_3$ )
- Bernardo recuperará  $N$  haciendo el siguiente cálculo
  - $N = N_2 * \text{inv}[N_3, p_B] \mod p_B = N_2 * \text{inv}[(\alpha_B^v)^b \mod p_B, p_B] \mod p_B$
- Observa que  $\text{inv}[N_3, p_B]$  siempre existirá porque el cuerpo de cifra  $p_B$  es un primo y, entonces, todos sus elementos o restos del cuerpo serán primos relativos o coprimos con el cuerpo  $p_B$

# Ejercicio de cifrado y descifrado

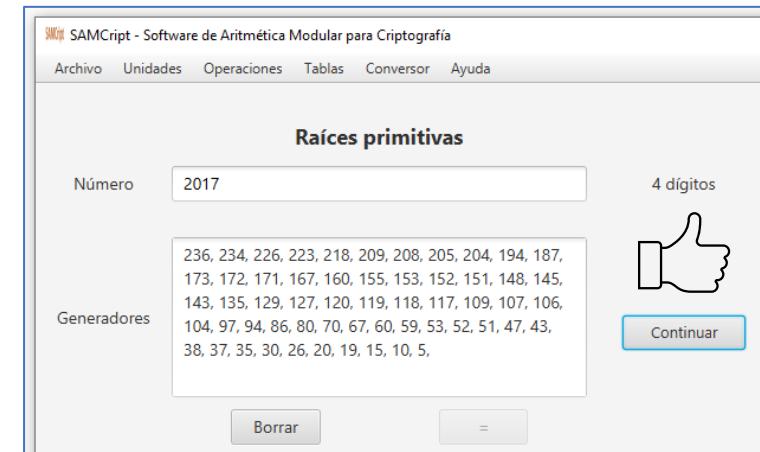
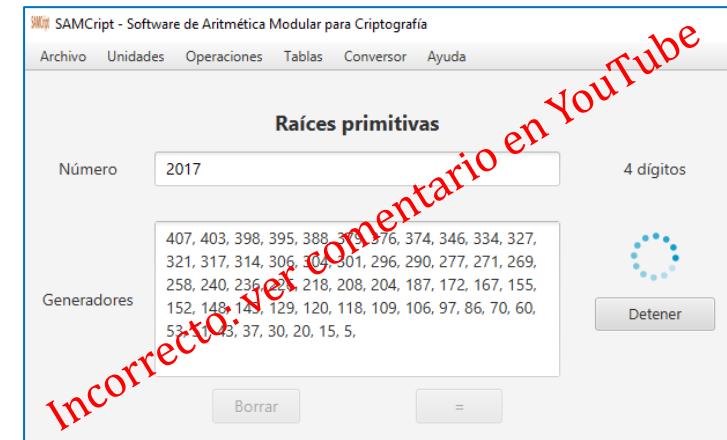
SAMCrypt

Software de Aritmética Modular para  
Criptografía

- Bernardo tiene como valores públicos  $p_B = 2.017$ ,  $\alpha_B = 5$  y  $K_{púbB} = 1.375$
- Alicia desea enviar a Bernardo el número  $N = 1.234$  con confidencialidad
  - Alicia elige  $v = 1.810$  y calcula  $N_1$  y  $N_2$
  - $N_1 = (\alpha_B^v) \text{ mod } p_B = 5^{1.810} \text{ mod } 2.017 = 1.754$
  - $N_2 = N * (\alpha_B^b)^v \text{ mod } p_B = 1.234 * (1.375)^{1.810} \text{ mod } 2.017 = 1.033$
  - Alicia envía a Bernardo el par  $[N_1, N_2] = [1.754, 1.033]$
- Bernardo tiene como clave privada  $b = 491$  ( $K_{púbB} = 5^{491} \text{ mod } 2.017 = 1.375$ )
- Bernardo hace los siguientes cálculos
  - $N_1^b \text{ mod } p_B = 1.754^{491} \text{ mod } 2.017 = 179 = N_3$
  - $N = N_2 * \text{inv}[N_3, p_B] \text{ mod } p_B = 1.033 * \text{inv}(179, 2.017) \text{ mod } 2.017$
  - $N = 1.033 * 462 \text{ mod } 2.017 = 1.234$  (y recupera el secreto o texto en claro)

# Qué sucede si no se usa un generador $\alpha$

- Para el primo  $p = 2.017$  existen 576 (aproximadamente 30%) generadores o raíces primitivas  $\alpha$ , siendo 5 el valor más pequeño
- Si se usa como  $\alpha$  un valor que no sea una raíz primitiva, por ejemplo aquí el 2, el cifrado y descifrado de Elgamal siguen siendo correctos
- Solamente sucederá que para la clave pública  $K_{púb}$  ahora habrá más de un valor de clave privada que sea válido
- Eso significa una menor seguridad



# Cifra de bloques de texto con Elgamal

- Al igual que en RSA, lo normal en Elgamal es cifrar números, el número de la clave en un intercambio de clave o el valor de un hash en una firma digital
- Si queremos cifrar textos, habrá que formar bloques de texto ASCII de un tamaño menor que el módulo de cifra
- Si el módulo es el primo  $p = 266.677 = 1000001000110110101$  de 19 bits y queremos cifrar el mensaje  $M = \text{HOLA AMIGOS}$ , se puede formar 6 bloques de dos letras de 16 bits cada uno
- $M = \text{HO LA } \& \text{A MI GO S}$  (en donde & representa aquí el espacio en blanco)
- Observa que no será necesario incluir relleno en el último bloque
  - $H = 01001000 \quad O = 01001111 \quad L = 01001100 \quad A = 01000001$
  - $= 00100000 \quad A = 01000001 \quad M = 01001101 \quad I = 01001001$
  - $G = 01000111 \quad O = 01001111 \quad S = 01010011$

# Formando bloques de texto con Elgamal

- Los 6 bloques de texto a cifrar serán
  - $HO = 0100100001001111 = 18.511$        $LA = 0100110001000001 = 19.521$
  - $A = 0010000001000001 = 8.257$        $MI = 0100110101001001 = 19.785$
  - $GO = 0100011101001111 = 18.255$        $S = 01010011 = 83$
- Datos del sistema de cifra
  - Bernardo elige  $p_B = 266.677$ ,  $\alpha_B = 5$  y como clave privada  $K_{\text{pri}B} = b = 8.912$ . Su clave pública  $K_{\text{púb}B} = \alpha_B^b \text{ mod } p_B = 5^{8.912} \text{ mod } 266.677 = 63.616$
  - Valores públicos de Bernardo:  $p_B = 266.677$ ,  $\alpha_B = 5$ ,  $K_{\text{púb}B} = 63.616$
  - Valor privado de Bernardo:  $b = 8.912$
  - Alicia desea enviar a Bernardo el texto de 11 bytes  $M = \text{HOLA AMIGOS}$ 
    - Cifraremos solamente los 2 primeros bytes del texto:  $HO = 18.511$

# Cifrando HO con ExpoCrip



- Alicia cifrará  $\text{HO} = 18.511$  en el primo  $p_B = 266.677$ , eligiendo  $v = 5.003$ 
  - $N_1 = (\alpha_B^v) \bmod p_B = 5^{5.003} \bmod 266.677 = 49.796$
  - $N_2 = 18.511 * (\alpha_B^b)^v \bmod p_B = 18.511 * (63.616)^{5.003} \bmod 266.677 = 108.072$
  - Alicia envía la dupla  $[N_1, N_2] = [49.796, 108.072]$
- Bernardo descifrará el criptograma  $[N_1, N_2]$ 
  - $N_1^b \bmod p_B = 49.796^{8.912} \bmod 266.677 = 135.282 = N_3$
  - $N_2 * \text{inv}[N_3, p_B] \bmod p_B = 108.072 * \text{inv}[135.282, 266.677]$
  - $= 108.072 * 248.153 \bmod 266.677 = 18.511 = 100100001001111$
  - Agrupando esta cadena binaria de 15 bits en bloques de 8 bits de derecha a izquierda, y agregando ceros a la izquierda si fuese necesario, se obtiene
  - **01001000 01001111 = HO** (el primer bloque de texto en claro)

# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=N3ANa0i9gAc>

# Conclusiones de la Lección 10.10

- El algoritmo de Elgamal se basa en el algoritmo de intercambio de clave de Diffie y Hellman de 1976
- La seguridad de la cifra reside en el problema del logaritmo discreto PLD
- El algoritmo de Elgamal se presenta en 1985
- Para el cifrado el emisor usará los valores públicos del destinatario
  - Un primo  $p$ , una raíz  $\alpha$  del primo y la clave pública del destinatario  $K_{púb}$
  - Usará además un valor aleatorio o de sesión  $v$ , que generará en cada cifra
- Para el descifrado, el destinatario recuperará ese valor  $v$  usando su clave privada  $y$ , posteriormente, con este valor  $v$  podrá descifrar el criptograma
- Es una cifra muy segura pero se ve penalizada por la necesidad de enviar dos números del tamaño del módulo, hoy en día un primo de al menos 2.048 bits, y haber visto la luz siete años después que el algoritmo RSA

# Lectura recomendada (1/2)

- A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, , Taher Elgamal, IEEE Transactions on Information Theory, vol. it-31, No. 4, july 1985
  - <https://caislab.kaist.ac.kr/lecture/2010/spring/cs548/basic/B02.pdf>
- Guion píldora formativa Thoth nº 42, ¿Cómo funciona el algoritmo de Elgamal?, Jorge Ramió, 2017
  - <https://www.criptored.es/thoth/material/texto/pildora042.pdf>
- Curso de Criptografía Aplicada, Jorge Ramió, 2018
  - [https://www.criptored.es/guiateoria/gt\\_m001s1.htm](https://www.criptored.es/guiateoria/gt_m001s1.htm)
- SAMCrypt: Software de Aritmética Modular para Criptografía, María Nieto Díaz y Jorge Ramió Aguirre, 2018
  - [https://www.criptored.es/software/sw\\_m001t.htm](https://www.criptored.es/software/sw_m001t.htm)

# Lectura recomendada (2/2)

- ExpoCrip: Software para Generación de Claves, Cifra y Firma RSA, ElGamal y DSS, Olga Mariana González Ming y Jorge Ramió Aguirre, 2003
  - [https://www.criptored.es/software/sw\\_m001l.htm](https://www.criptored.es/software/sw_m001l.htm)
- CLCript 00: Códigos y tablas de uso frecuente en criptografía
  - [https://www.criptored.es/descarga/Codigos\\_y\\_tablas\\_de\\_uso\\_frecuente\\_en\\_criptografia.pdf](https://www.criptored.es/descarga/Codigos_y_tablas_de_uso_frecuente_en_criptografia.pdf)

# Class4crypt c4c10.11

## Módulo 10. Criptografía asimétrica

### Lección 10.11. Algoritmos de firma digital RSA y de Elgamal

10.11.1. Firma electrónica y firma digital

10.11.2. Autenticación asimétrica vía hash

10.11.3. Firma digital RSA

10.11.4. Ejemplo de firmado RSA y comprobación de la firma

10.11.5. Firma digital de Elgamal

10.11.6. Ejemplo de firmado de Elgamal y comprobación de la firma

Class4crypt c4c10.11 Algoritmos de firma digital RSA y de Elgamal  
<https://www.youtube.com/watch?v=o8vBlQGhDdQ>

# Firma electrónica y firma digital

## Electrónica

Hace referencia a cuestiones legales, organizativas, técnicas, etc.

- (Art. 3.1) La firma electrónica es el conjunto de datos en forma electrónica, consignados junto a otros o asociados con ellos, que pueden ser utilizados como medio de **identificación del firmante** (ver la bibliografía)
- (Art. 3.2) La firma electrónica avanzada es la firma electrónica que permite identificar al firmante y detectar **cualquier cambio ulterior de los datos firmados**, que está vinculada al firmante de manera única y a los datos a que se refiere y que ha sido creada por medios que el firmante puede mantener bajo su exclusivo control

## Digital

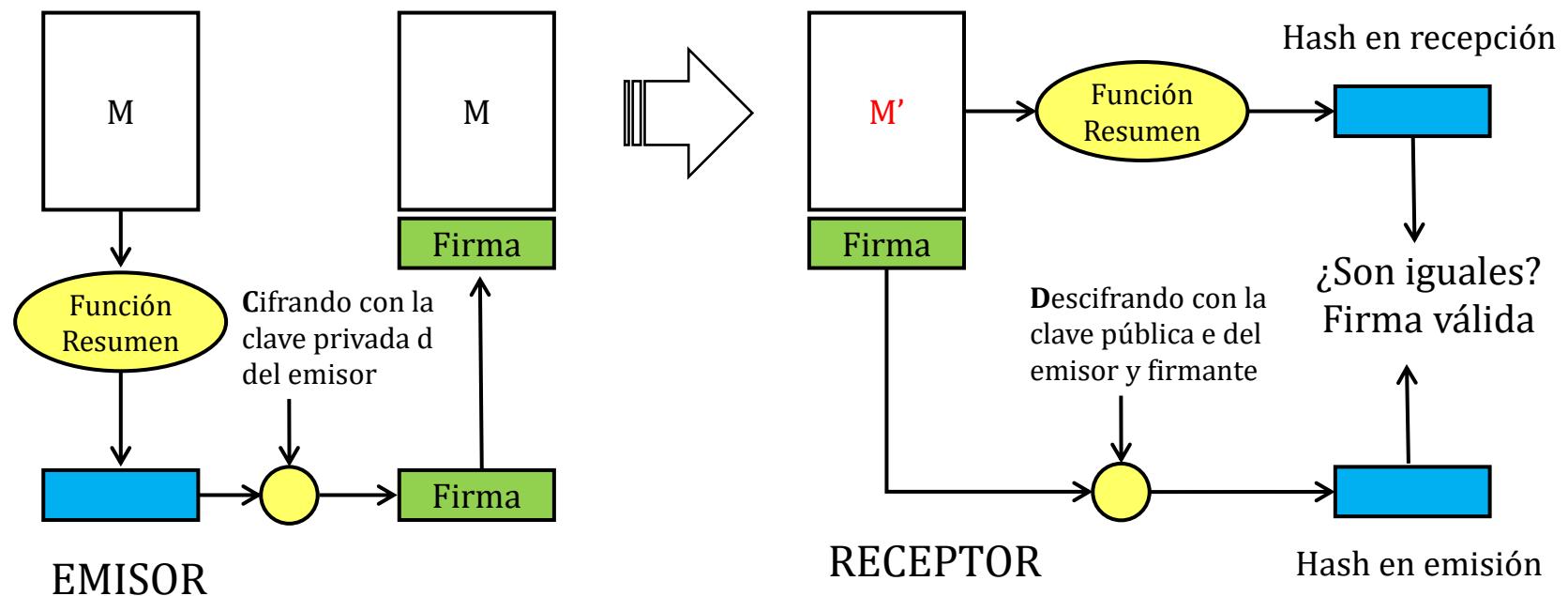
Algoritmos



- La firma digital hace referencia a un conjunto de **métodos criptográficos** y técnicos. Fundamentalmente es un concepto técnico y en esto nos centraremos en la clase

# Autenticación asimétrica vía hash

- Se envía el mensaje  $M$  en claro (si además se requiere confidencialidad, puede cifrarse) junto con el hash  $h(M)$  cifrado con la clave privada  $d$  del emisor
- En recepción se comprueba la firma con la clave pública e del emisor, se aplica la función hash sobre el mensaje  $M'$  recibido y se comparan resultados



# Pasos de una firma digital con RSA



Bernardo

$e_B \rightarrow$  Clave pública  
 $n_B \rightarrow$  Clave pública  
 $d_B \rightarrow$  Clave privada



Alicia

Bernardo va a firmar con RSA un hash del mensaje  $M$  para enviárselo a Alicia

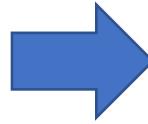
1. Bernardo calcula la firma o rúbrica sobre ese hash:  $r = h(M)^{d_B} \text{ mod } n_B$
2. Bernardo envía a Alicia el mensaje en claro  $M$  (o bien cifrado si éste requiere además de confidencialidad) junto con su rúbrica:  $\{M, r\}$
3. Alicia recibe el par  $\{M', r\}$  y comprueba la rúbrica, usando la clave pública de Bernardo  
 $r^{e_B} \text{ mod } n_B = [h(M)^{d_B}]^{e_B} \text{ mod } n_B = h(M)$ , recuperando así el hash de origen
4. Alicia calcula ahora en destino el resumen  $h(M')$  del mensaje  $M'$  que le ha llegado (que puede no ser el mensaje  $M$ ) y lo compara con el hash  $h(M)$  del origen encontrado en el descifrado
5. Si  $h(M') = h(M)$  entonces se sabe que  $M' = M$  y se acepta la firma  $r$  como válida  
Observa que sólo es 100% verdadero que si  $M' = M$  entonces  $h(M') = h(M)$ , no al revés

# Ejemplo de firma digital con RSA



Bernardo

$$\begin{aligned}e_B &= 35 \\n_B &= 65.669 \\d_B &= 53.771\end{aligned}$$



Puesto que  $2^{16} < n_B = 65.669 < 2^{17}$ , vamos a suponer que se usa una función hash “especial” que entrega un resumen de 16 bits, es decir, desde  $h(M) = 0x\ 0000$  hasta  $h(M) = 0x\ FFFF$



Alicia

- Bernardo va a firmar con RSA un hash del mensaje M (por ejemplo el documento info.docx) para enviárselo a Alicia
- Sea entonces en este escenario por ejemplo  $h(M) = 0x\ F3A9 = 62.377$
- Firma de Bernardo:  $r = h(M)^{d_B} \bmod n_B = 62.377^{53.771} \bmod 65.669 = 24.622$
- Bernardo envía a Alicia el par  $\{M, r\} = \{\text{info.docx}, 24.622\}$
- Alicia comprueba la firma:  $r^{e_B} \bmod n_B = 24.622^{35} \bmod 65.669 = 62.377$
- Alicia calcula el hash en destino del archivo recibido (info.docx o info'.docx) y comprueba si tiene el mismo hash que el de emisión. Si también obtiene 62.377 acepta la firma como válida
- En la práctica los hashes entregan centenas de bits y la cifra RSA es en módulo de 2.048 bits

# Firma digital de Elgamal de Alicia (1/2)



- Datos de Alicia que va a firmar el hash  $h(M)$  de un mensaje  $M$  con Elgamal
  - El cuerpo o módulo de la firma: primo  $p_A$  (público)
  - La raíz primitiva o generador de  $p_A$ :  $\alpha_A$  (público)
  - La clave privada de Alicia  $K_{privA} = a$
  - La clave pública de Alicia  $K_{pubA}$  resultado de  $(\alpha_A^a \bmod p_A)$
- Pasos que debe realizar Alicia para firmar  $h(M)$  mediante la dupla  $\{r, s\}$ 
  - **Paso 1:** genera un número aleatorio  $h$ , primo relativo con  $\phi(p_A) = p_A - 1$ , es decir, deberá cumplirse que  $\text{mcd}\{h, \phi(p_A)\} = 1$
  - **Paso 2:** calcula el inverso de este número en  $p_A$ :  $h^{-1} = \text{inv}[h, \phi(p_A)]$
  - **Paso 3:** calcula su rúbrica  $r = \alpha_A^h \bmod p_A$  (**primera parte de su firma**)
  - A continuación Alicia calcula  $s$  mediante una congruencia con  $h(M)$

# Firma digital de Elgamal de Alicia (2/2)



- Primer valor de la firma de Alicia:  $r = \alpha_A^h \text{ mod } p_A$
- **Paso 4:** Alicia resuelve la siguiente congruencia
  - $h(M) = a*r + h*s \text{ mod } \phi(p_A)$ , es decir:
  - $s = [h(M) - a*r] * \text{inv}[h, \phi(p_A)] \text{ mod } \phi(p_A)$  (**segunda parte de su firma**)
- **Paso 5:** Alicia envía al destinatario M y  $\{r, s\}$ 
  - $\{r, s\} = \{\alpha_A^h \text{ mod } p_A, [h(M) - a*r] * \text{inv}[h, \phi(p_A)] \text{ mod } \phi(p_A)\}$
- Muy importante: observa que, como se usa un valor h aleatorio, la firma de Elgamal sobre dos mensajes iguales entregará resultados distintos, algo muy interesante y que no sucedía con la firma RSA
- No obstante, esta firma sigue siendo más pesada que RSA al tener que enviar dos números r y s de magnitudes de orden  $p_A$  (2.048 bits) en vez de uno solo



# Comprobación de la firma de Elgamal

- Bernardo comprobará la firma digital realizada por Alicia, conociendo los siguientes datos públicos de Alicia:  $p_A$ ,  $\alpha_A$  y  $K_{\text{púbA}}$
- **Paso 1:** Bernardo recibe  $M'$  y  $\{r, s\}$  y calcula  $N_1$  y  $N_2$ :
  - $N_1 = r^s \bmod p_A$
  - $N_2 = K_{\text{púbA}}^r \bmod p_A$
- **Paso 2:** Bernardo calcula  $k = N_1 * N_2 \bmod p_A = [r^s * K_{\text{púbA}}^r] \bmod p_A$ 
  - Como  $r = (\alpha_A^h \bmod p_A)$  y  $K_{\text{púbA}} = (\alpha_A^a \bmod p_A)$ , entonces sustituyendo en  $k$ :
  - $k = [\alpha_A^{h*s} * \alpha_A^{a*r}] \bmod p_A = \alpha_A^{(h*s + a*r)} \bmod p_A$
  - Se trata de una ecuación del tipo:  $\alpha^\beta \bmod p$
  - $h(M) = (a*r + h*s) \bmod \phi(p_A)$  era la congruencia resuelta por Alicia en el módulo  $\phi(p_A)$  y se cumplirá que  $\alpha^\beta = \alpha^\gamma$  **si y solo si**  $\beta = \gamma \bmod \phi(p_A)$
- **Paso 3:** Bernardo obtiene el hash de  $M'$  recibido y comprueba si  $\alpha_A^{h(M')} \bmod p_A = k$
- Si ambos valores son iguales, sabe que  $M' = M$  y acepta la firma  $\{r, s\}$  como válida

# Ejemplo de firma de Elgamal de Alicia



- Sea:  $K_{privA} = a = 20$
- Con  $p_A = 79.903$ ,  $\alpha_A = 10$ ,  $K_{pubA} = 3.631$  ( $K_{pubA} = \alpha_A^a \mod p_A = 10^{20} \mod 79.903 = 3.631$ )
- El documento a firmar tiene un hash de 16 bits  $h(M) = 0x A69B = 42.651$
- Alicia elige el valor  $h = 31$  ya que  $mcd(31, 79.902) = 1$
- Calcula  $h^{-1} = \text{inv}[h, \phi(p_A)] = \text{inv}(31, 79.902) = 5.155$
- Calcula  $r = \alpha_A^h \mod p_A = 10^{31} \mod 79.903 = 11.755$
- Resuelve  $s = [h(M) - a * r] * \text{inv}(h, \phi(p_A)) \mod \phi(p_A)$ 
  - $s = [42.651 - 20 * 11.755] * \text{inv}(31, 79.902) \mod 79.902$
  - $s = [42.651 - 235.100] * 5.155 \mod 79.902$
  - $s = -(192.449) * 5.155 \mod 79.902$
  - $s = -992.074.595 \mod 79.902 = 68.539$
- La firma de Alicia sobre el hash  $h(M) = 42.651$  será el par  $\{r, s\} = \{11.755, 68.539\}$



Bernardo

# Comprobación de la firma de Elgamal

- Datos públicos de Alicia que conoce Bernardo:  $p_A = 79.903$ ,  $\alpha_A = 10$ ,  $K_{\text{púb}A} = 3.631$
- Bernardo recibe  $M'$  y  $\{r, s\} = M', \{11.755, 68.539\}$  y el hash original era  $h(M) = 42.651$
- Calcula  $N_1 = r^s \bmod p_A = 11.755^{68.539} \bmod 79.903 = 66.404$
- Calcula  $N_2 = K_{\text{púb}A}^r \bmod p_A = 3.631^{11.755} \bmod 79.903 = 12.023$
- Calcula  $k = N_1 * N_2 \bmod p_A = (66.404 * 12.023) \bmod 79.903 = 64.419$
- Sobre el mensaje  $M'$  recibido, calcula  $h(M')$  y pregunta ¿ $\alpha_A^{h(M')} \bmod p_A = k$ ?
- Si Bernardo recibe el mismo mensaje  $M$  (es decir  $M' = M$ ) que firmó Alicia, en recepción calcula el hash  $h(M')$  y obtendrá 42.651 que es el mismo valor  $h(M)$  de origen
- Entonces calcula  $10^{42.651} \bmod 79.903 = 64.419 = k$
- Como es el mismo valor, Bernardo sabe que  $M' = M$  y acepta la firma de Alicia como válida

# Más información en píldoras Thoth



<https://www.youtube.com/watch?v=N3ANaOj9gAc>

# Conclusiones de la Lección 10.11

- La autenticación asimétrica vía hash permite comprobar la autenticidad del emisor así como la integridad del mensaje firmado (integridad completa)
- La seguridad de la firma RSA descansa en el problema de la factorización entera y en la firma de Elgamal en el problema del logaritmo discreto
- La firma RSA es similar al cifrado; en este caso con la clave privada del emisor
- Para la firma de Elgamal, el emisor usará un valor aleatorio  $h$  (firma siempre distinta) y deberá resolver una congruencia en  $\phi(p)$ , en donde intervienen el mensaje  $M$  o función hash  $h(M)$  a firmar y la clave privada del firmante
- Firma sobre  $h(M)$ :  $\{r, s\} = (\alpha^h \bmod p), [h(M) - K_{priv} * r] * \text{inv}[h, \phi(p)] \bmod \phi(p)$
- Para comprobar la firma, el receptor calcula  $N_1 = r^s \bmod p$  y  $N_2 = K_{pub}^r \bmod p$
- Calcula  $k = N_1 * N_2 \bmod p$ , y calcula además  $h(M')$  del mensaje  $M'$  recibido
- Si  $\alpha^{h(M')} \bmod p = k = N_1 * N_2$ , entonces la firma se acepta como válida

# Lectura recomendada (1/2)

- A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms, , Taher Elgamal, IEEE Transactions on Information Theory, vol. it-31, No. 4, july 1985
  - <https://caislab.kaist.ac.kr/lecture/2010/spring/cs548/basic/B02.pdf>
- Guion píldora formativa Thoth nº 42, ¿Cómo funciona el algoritmo de Elgamal?, Jorge Ramió, 2017
  - <https://www.criptored.es/thoth/material/texto/pildora042.pdf>
- Base legal de la Firma Electrónica, Portal Administración Electrónica, Gobierno de España
  - <https://firmaelectronica.gob.es/Home/Ciudadanos/Base-Legal.html>

# Lectura recomendada (2/2)

- Curso de Criptografía Aplicada, Jorge Ramió, 2018
  - [https://www.criptored.es/guiateoria/gt\\_m001s1.htm](https://www.criptored.es/guiateoria/gt_m001s1.htm)
- SAMCrip: Software de Aritmética Modular para Criptografía, María Nieto Díaz y Jorge Ramió Aguirre, 2018
  - [https://www.criptored.es/software/sw\\_m001t.htm](https://www.criptored.es/software/sw_m001t.htm)
- ExpoCrip: Software para Generación de Claves, Cifra y Firma RSA, ElGamal y DSS, Olga Mariana González Ming y Jorge Ramió Aguirre, 2003
  - [https://www.criptored.es/software/sw\\_m001l.htm](https://www.criptored.es/software/sw_m001l.htm)

# Class4crypt c4c10.12

## Módulo 10. Criptografía asimétrica

### Lección 10.12. Firma digital DSA y ataques a firmas de Elgamal y DSA

10.12.1. Firma digital DSA y DSS

10.12.2. Ejemplo de firma DSA

10.12.3. Ejercicio práctico de firma DSA con ExpoCrip

10.12.4. Ejemplo de ataque con éxito a una firma de Elgamal

10.12.5. Ejemplo de ataque fallido a una firma de Elgamal

10.12.6. Seguridad y ataques a la firma DSA y ECDSA

Class4crypt c4c10.12 Firma digital DSA y ataques a firmas de Elgamal y DSA  
<https://www.youtube.com/watch?v=QpGRGEx7694>

# Hace ahora justo 2 años... el 15/01/2020

- Comenzábamos el proyecto Class4crypt
- Hoy tenemos publicadas 70 clases y 7 ejercicios o desafíos
- Con más de 3.000 seguidores y más de 71.000 visitas
- Hemos hecho un repaso a la “criptografía aplicada básica”
- Obviamente, quedan aún muchos temas pendientes
- **PERO**, es mejor dejar las clases hasta aquí y comenzar la parte práctica, con preguntas de test y ejercicios típicos de examen, en lo que seguro estarás muy interesado

# Firmas DSA y DSS

- En 1991 el National Institute of Standards and Technology (NIST) propone DSA, Digital Signature Algorithm, una variante de los algoritmos de Elgamal y Schnorr
- En 1994 se establece como estándar el DSA y se conoce como DSS, Digital Signature Standard
- Parámetros públicos originales de la firma DSA:
  - Un número primo  $p$  ( $512 \text{ bits} < p < 1024 \text{ bits}$ )
  - Un número primo  $q$  (160 bits) divisor de  $p-1$  [se usa SHA-1]
  - Un generador  $\alpha$  de orden  $q$  del grupo  $p$ , que se calcula  $\alpha = g^{(p-1)/q} \text{ mod } p$ , con  $\alpha \neq 1$  y siendo  $g$  un valor aleatorio dentro del primo  $p$  y  $q$  uno de los divisores de  $(p-1)$
- Elección de parámetros:
  - Calcular la clave pública del sujeto  $y = \alpha^x \text{ mod } p$ , donde  $x$  es la clave privada
  - Hacer público los parámetros  $p, q, \alpha, y$

# Generación de firma digital DSA por Alicia

- Valores públicos de Alicia: primos  $p_A$ ,  $q_A$  y el generador  $\alpha_A$ 
  - Firmará el valor  $1 \leq h(M) \leq q_A$
- Clave privada de la firma: un número aleatorio  $a$  ( $1 \leq a < q_A$ )
- Clave pública de la firma:  $y_A = \alpha_A^a \text{ mod } p_A$
- Para firmar, Alicia elegirá un valor aleatorio ( $1 \leq k < q_A$ )
- Luego calculará  $r$  y  $s$ :
  - $r = (\alpha_A^k \text{ mod } p_A) \text{ mod } q_A$   
Si  $r = 0$ , deberá elegir otro valor de  $k$  y volver a calcular  $r$
  - $s = [(h(M) + a \cdot r) * \text{inv}(k, q_A)] \text{ mod } q_A$   
Si  $s = 0$ , deberá elegir otro valor de  $k$  y volver a calcular  $r$  y  $s$
- La firma digital de Alicia sobre  $h(M)$  será el par  $(r, s)$
- Observa que ahora los valores  $r$  y  $s$  enviados como firma están reducidos al cuerpo  $q_A$  de 160 bits si usamos SHA-1 o a 256 bits si usamos SHA-2



# Comprobación de firma DSA por Bernardo

- Se recibe el par  $(r, s)$
- El receptor calcula:
  - $w = \text{inv}(s, q_A)$
  - $u = h(M) * w \bmod q_A$   $h(M)$  calculado en destino (o bien  $h(M')$ )
  - $v = r * w \bmod q_A$
- Y comprueba que se cumple la siguiente igualdad:
  - $(\alpha_A^u y_A^v \bmod p_A) \bmod q_A = r$
- Si se cumple lo anterior, el receptor acepta la firma DSA como válida
- Observa que la comprobación de la firma se hace sobre  $r$ , un valor en el que no interviene el valor del mensaje  $M$  o  $h(M)$  y que, además, la firma se hace ahora sobre valores del cuerpo  $q$ , mucho menores que  $p$



Bernardo

# Ejemplo de una firma DSA de Alicia

- Valores públicos de Alicia:  $p_A = 223$ ,  $q_A = 37$ ,  $\alpha_A = 64$
- Clave privada de Alicia:  $a = 25$
- Clave pública de Alicia:  $y_A = \alpha_A^a \text{ mod } p_A = 64^{25} \text{ mod } 223 = 4$
- Comprobación de que  $p_A$ ,  $q_A$  y  $\alpha_A$  son correctos:
  - $p_A = 223$  es primo y  $(p_A - 1) = 222 = 2 * 3 * 37$  (elige el primo mayor 37 como  $q_A$ )
  - Si  $g = 2$ , entonces  $g^{(p-1)/q} \text{ mod } p = 2^{(222)/37} \text{ mod } 223 = 64 \neq 1$ , entonces  $\alpha_A = 64$
- Alicia firmará el valor  $h(M) = 30$  ( $h(M) < q_A$ )
- Para firmar elige aleatoriamente  $k = 12$  que cumple  $\{1 \dots q_A - 1\}$
- Luego calculará:
  - $r = (\alpha_A^k \text{ mod } p_A) \text{ mod } q_A$   
 $r = (64^{12} \text{ mod } 223) \text{ mod } 37 = 56 \text{ mod } 37 = 19$
  - $s = [(h(M) + a * r) * \text{inv}(k, q_A)] \text{ mod } q_A$   
 $s = (30 + 25 * 19) * \text{inv}(12, 37) \text{ mod } 37 = 505 * 34 \text{ mod } 37 = 2$

La firma digital de Alicia sobre  $h(M) = 30$  será el par  $(r, s) = (19, 2)$

# Comprobación de la firma DSA del ejemplo

- Se recibe el par  $(r, s) = (19, 2)$
- Valores públicos de Alicia:  $p_A = 223$ ,  $q_A = 37$ ,  $\alpha_A = 64$ ,  $y_A = 4$
- El receptor calcula:
  - $w = \text{inv}(s, q_A) = \text{inv}(2, 37) = 19$
  - $u = h(M) * w \bmod q_A = 30 * 19 \bmod 37 = 15$
  - $v = r * w \bmod q_A = 19 * 19 \bmod 37 = 28$
- Y comprueba que se cumple la siguiente relación:
  - $i(\alpha_A^u * y_A^v \bmod p_A) \bmod q_A = (64^{15} * 4^{28} \bmod 223) \bmod 37 = r?$   
 $(197 * 15 \bmod 223) \bmod 37 = 56 \bmod 37 = 19 = r$
- Como se llega al mismo valor 19 que tiene el valor  $r$  recibido, el receptor acepta la firma DSA de Alicia como válida

# Ejercicios prácticos

- Al ser la última clase de Class4crypt, la comprobación de los ejercicios presentados con el software SAMCript se deja como tarea para el estudiante



- En esta clase comprobaremos el ejemplo de firma DSA del libro “Handbook of Applied Cryptography” de Alfred Menezes, Paul van Oorschot y Scott Vanstone (1997) usando el software ExpoCript
- Chapter 11 - Digital Signatures, 11.57 Example (DSA signature generation with artificially small parameters), página 453 (de libre descarga en Internet)

# Ejemplo 11.57 de firma DSA

## 11.57 Example (*DSA signature generation with artificially small parameters*)

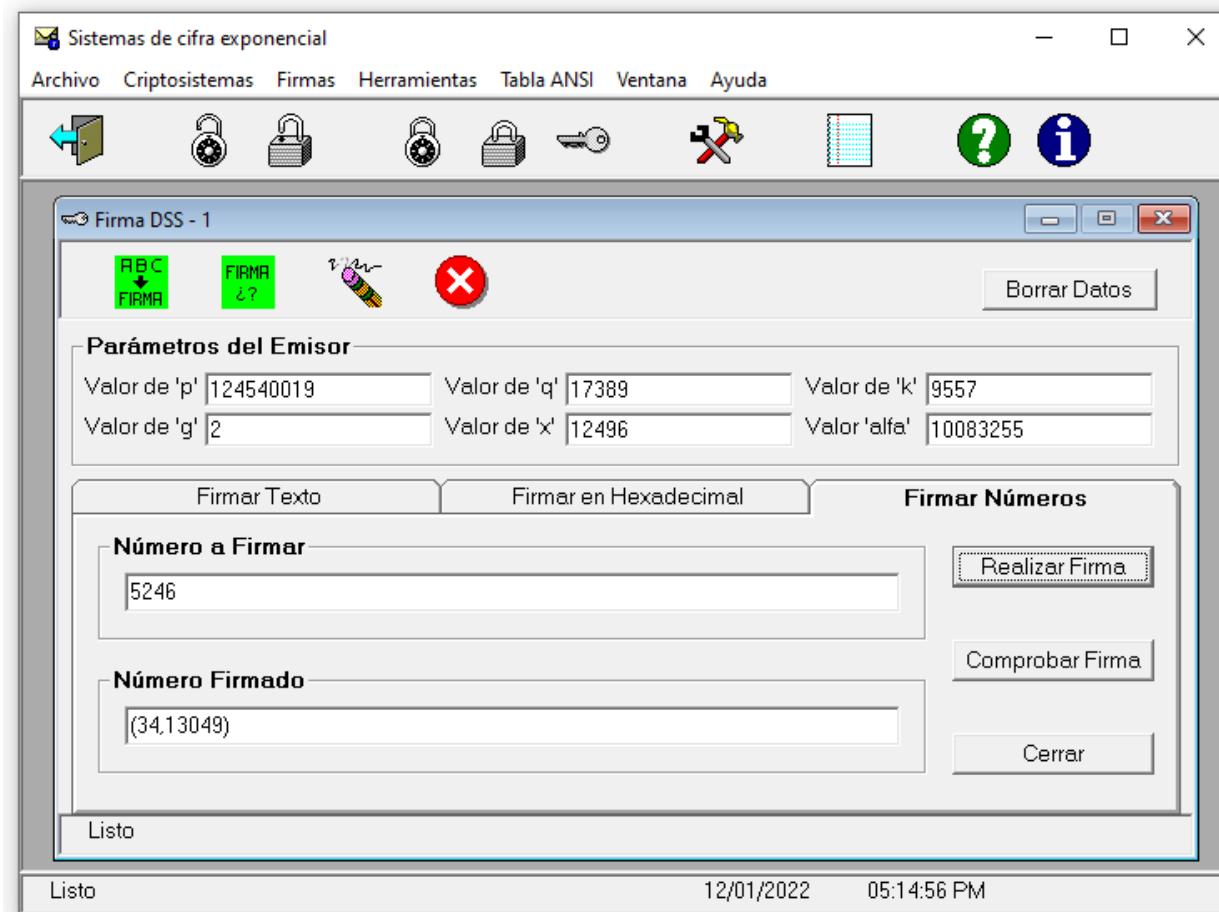
*Key generation.* A selects primes  $p = 124540019$  and  $q = 17389$  such that  $q \mid (p - 1)$ ; here,  $(p - 1)/q = 7162$ . A selects a random element  $\underline{g = 110217528} \in \mathbb{Z}_p^*$  and computes  $\alpha = g^{7162} \bmod p = \underline{10083255}$ . Since  $\alpha \neq 1$ ,  $\alpha$  is a generator for the unique cyclic subgroup of order  $q$  in  $\mathbb{Z}_p^*$ . A next selects a random integer  $a = 12496$  satisfying  $1 \leq a \leq q - 1$ , and computes  $y = \alpha^a \bmod p = 10083255^{12496} \bmod 124540019 = 119946265$ . A's public key is  $(p = 124540019, q = 17389, \alpha = 10083255, y = 119946265)$ , while A's private key is  $a = 12496$ .

*Signature generation.* To sign  $m$ , A selects a random integer  $k = 9557$ , and computes  $r = (\alpha^k \bmod p) \bmod q = (10083255^{9557} \bmod 124540019) \bmod 17389 = 27039929 \bmod 17389 = 34$ . A then computes  $k^{-1} \bmod q = 7631$ ,  $h(m) = 5246$  (the hash value has been contrived for this example), and finally  $s = (7631)\{5246 + (12496)(34)\} \bmod q = 13049$ . The signature for  $m$  is the pair  $(r = 34, s = 13049)$ .

*Signature verification.* B computes  $w = s^{-1} \bmod q = 1799$ ,  $u_1 = w \cdot h(m) \bmod q = (5246)(1799) \bmod 17389 = 12716$ , and  $u_2 = rw \bmod q = (34)(1799) \bmod 17389 = 8999$ . B then computes  $v = (\alpha^{u_1} y^{u_2} \bmod p) \bmod q = (10083255^{12716} \cdot 119946265^{8999} \bmod 124540019) \bmod 17389 = 27039929 \bmod 17389 = 34$ . Since  $v = r$ , B accepts the signature.  $\square$

Se llega a igual  
 $\alpha = 10083255$   
con  $g = 2$  (valor típico) que con  $g = 110217528$

# Resolución firma DSA con ExpoCript



$$\alpha = g^{(p-1)/q} \bmod p$$

Donde  $(p-1)/q = 124.540.018/17.389 = 7.162$

Con  $g = 2$

$$\alpha = 2^{7.162} \bmod 124.540.019 = 10.083.255$$

Con  $g = 110.217.528$

$$\alpha = 110.217.528^{7.162} \bmod 124.540.019 = 10.083.255$$

(Compruébalo con SAMCript)

# Ataque con éxito a la firma de Elgamal

- Repasar la lección 10.11 Algoritmos de firma RSA y de Elgamal. Alicia tiene las siguientes claves:  $p_A = 521$ ,  $a = 133$ ,  $\alpha_A = 7$ ,  $K_{\text{púb}A} = 7^{133} \text{ mod } 521 = 150$ . Y va a firmar las funciones hash de dos documentos diferentes con la misma constante  $h = 33$ , que cumple con la condición de la firma  $\text{mcd}(33, 520) = 1$  y por lo tanto  $\text{inv}(33, 520) = 457$
- Las firmas de esos dos documentos serán:
  - Para  $h(M) = 375$  obtendrá la firma  $(r, s)$  y para  $h(M') = 404$  la firma  $(r, s')$
  - $r = \alpha_A^h \text{ mod } p_A = 7^{133} \text{ mod } 521 = 339$
  - Para que el ataque prospere, este valor  $r$  debería ser impar y aquí 339 lo cumple. Por lo tanto,  $r(s' - s)$  podría entregar un número impar y tener inverso en  $p_A - 1$ , que siempre será par
  - $s = [h(M) - a * r] * \text{inv}(h, p_A - 1) \text{ mod } p_A - 1$
  - $s = [375 - 133 * 339] * \text{inv}(33, 520) \text{ mod } 520 = -44.712 * 457 \text{ mod } 520 = 16$
  - $s' = [h(M') - a * r] * \text{inv}(h, p_A - 1) \text{ mod } p_A - 1$
  - $s' = [404 - 133 * 339] * \text{inv}(33, 520) \text{ mod } 520 = -44.683 * 457 \text{ mod } 520 = 269$
  - $r(s' - s) = 339 * (269 - 16) = 85.767 \text{ mod } 520 = 487$
- Como  $\text{mcd}(487, 520) = 1$  entonces  $\text{inv}(487, 520) = 63$  y el ataque prosperará. Se calcula ahora  $[h(M)s' - h(M')s] * \text{inv}[r(s' - s), p_A - 1] \text{ mod } p_A - 1 = [375 * 269 - 404 * 16] * 63 \text{ mod } 520 = (100.875 - 6.464) * 63 \text{ mod } 520 = 133$
- Y se ha encontrado la clave privada de Alicia  $a = 133$

# Ataque fallido a la firma de Elgamal

- Alicia tiene las siguientes claves:  $p_A = 521$ ,  $a = 133$ ,  $\alpha_A = 7$ ,  $K_{\text{púbA}} = 7^{133} \text{ mod } 521 = 150$ . Y va a firmar las funciones hash de dos documentos diferentes con la misma constante  $h = 83$ , que cumple con la condición de la firma puesto que  $\text{mcd}(83, 520) = 1$  y entonces  $\text{inv}(83, 520) = 307$
- Las firmas de esos dos documentos serán:
  - Para  $h(M) = 375$  obtendrá la firma  $(r, s)$  y para  $h(M') = 404$  la firma  $(r, s')$
  - $r = \alpha_A^h \text{ mod } p_A = 7^{83} \text{ mod } 521 = 428$
  - Como  $r = 428$  es par, entonces  $r(s' - s)$  será siempre un número par. Puesto que  $p-1$  es par, nunca existirá el inverso  $[r(s' - s), p_A - 1]$  y el ataque no podrá prosperar
  - $s = [h(M) - a * r] * \text{inv}(h, p_A - 1) \text{ mod } p_A - 1$
  - $s = [375 - 133 * 428] * \text{inv}(83, 520) \text{ mod } 520 = -56.549 * 307 \text{ mod } 520 = 177$
  - $s' = [h(M') - a * r] * \text{inv}(h, p_A - 1) \text{ mod } p_A - 1$
  - $s' = [404 - 133 * 428] * \text{inv}(83, 520) \text{ mod } 520 = -56.520 * 307 \text{ mod } 520 = 240$
  - $r(s' - s) = 428(177 - 240) = -26.964 \text{ mod } 520 = 76$
- Como  $\text{mcd}(76, 520) = 4$ , no existe el  $\text{inv}[r(s' - s), p_A - 1]$  y el ataque no prospera porque no se podrá ahora calcular la expresión  $[h(M)s' - h(M')s] * \text{inv}[r(s' - s), p_A - 1] \text{ mod } p_A - 1$
- El ataque ha fallado y no se descubre la clave privada de Alicia

# Seguridad en firmas DSA e ECDSA

- Al igual que sucedía con la firma de Elgamal, aquí la constante k deberá ser siempre un valor aleatorio y diferente
- Repetir dicho valor en dos o más firmas, permitiría a un atacante encontrar la clave privada del sujeto realizando operaciones matemáticas simples con los datos públicos de la firma
- Este ataque a DSA y ECDSA (Elliptic Curve Digital Signature Algorithm) ocurrió en diciembre de 2010 cuando el grupo *fail0verflow* reconoció haber descubierto la clave privada usada por Sony para firmar el software de la consola PlayStation 3
- Sony no usó valores aleatorios k diferentes para cada firma
  - PS3 hacked through poor cryptography implementation:  
<https://arstechnica.com/gaming/2010/12/ps3-hacked-through-poor-implementation-of-cryptography/>
  - Breaking ECDSA Elliptic Curve Cryptography: <https://www.youtube.com/watch?v=-UcCMjQab4w>
  - PS3 Private Key discovered fail0verflow: [https://www.youtube.com/watch?v=LP1t\\_pzxKyE](https://www.youtube.com/watch?v=LP1t_pzxKyE)

# Conclusiones de la Lección 10.12

- El algoritmo DSA, Digital Signature Algorithm, permite una firma digital con un primo  $q$  de un tamaño igual al hash usado en la firma, pero con la misma seguridad que la firma de Elgamal con un primo  $p$ , de miles de bits
- La firma DSA es el par  $(r, s)$ , muy similar a la firma de Elgamal
- Al igual que en la firma de Elgamal, la firma DSA sobre un mismo documento deberá tener siempre un valor diferente
- La comprobación de la firma se hace sobre  $r$ , un valor en el que no interviene el valor del mensaje  $M$  o  $h(M)$
- Las firmas de Elgamal y DSA pueden atacarse si no se usan adecuadamente los parámetros de diseño, en especial el uso de un valor aleatorio  $h$  en la firma de Elgamal y  $k$  en DSA que deben ser diferentes en cada firma. Si no lo son, en algunos casos el ataque podría desvelar la clave secreta del firmante

# Lectura recomendada (1/2)

- Digital Signature Algorithm (Wikipedia)
  - [https://en.wikipedia.org/wiki/Digital\\_Signature\\_Algorithm](https://en.wikipedia.org/wiki/Digital_Signature_Algorithm)
- Schnorr signature (Wikipedia)
  - [https://en.wikipedia.org/wiki/Schnorr\\_signature](https://en.wikipedia.org/wiki/Schnorr_signature)
- Handbook of Applied Cryptography, Alfred Menezes, Paul van Oorschot y Scott Vanstone, 1997, Fifth Printing August 2001 (descarga gratuita)
  - <https://cacr.uwaterloo.ca/hac/>
- SAMCript: Software de Aritmética Modular para Criptografía, María Nieto Díaz y Jorge Ramió Aguirre, 2018
  - [https://www.criptored.es/software/sw\\_m001t.htm](https://www.criptored.es/software/sw_m001t.htm)

# Lectura recomendada (2/2)

- ExpoCrip: Software para Generación de Claves, Cifra y Firma RSA, ElGamal y DSS, Olga Mariana González Ming y Jorge Ramió Aguirre, 2003
  - [https://www.criptored.es/software/sw\\_m001l.htm](https://www.criptored.es/software/sw_m001l.htm)

# Class4crypt 10.13 (estudio personal)

## Módulo 10. Criptografía asimétrica

### Lección 10.13. Criptografía con curvas elípticas ECC

- Como se comenta en el vídeo de esta última clase, Class4crypt se plantea como objetivo llegar hasta la lección 10.2, sin abordar entre otros temas el uso de las curvas elípticas en la criptografía asimétrica, con un total de 71 lecciones
- Esto significa que hemos estudiado más del 95% de los temarios impartidos en asignaturas de criptografía en universidades y, con seguridad, un 100% en lo que respecta a los ejercicios y preguntas que suelen aparecer en los exámenes
- Más que suficiente para la posible continuidad del proyecto como Class4crypt2 en donde se aborde la resolución de ejercicios y test típicos de examen
- Te dejo a continuación referencias y bibliografía de interés en ECC

# Bibliografía recomendada (1/2)

- Píldora formativa Thoth nº 49, “¿Por qué pueden utilizarse las curvas elípticas para cifrar?”, Víctor Gayoso Martínez y Luis Hernández Encinas, 2018
  - <https://www.youtube.com/watch?v=vi2wvAQsy-A>
- Guion píldora formativa Thoth nº 49, “¿Por qué pueden utilizarse las curvas elípticas para cifrar?”, Víctor Gayoso Martínez y Luis Hernández Encinas, 2018
  - <https://www.criptored.es/thoth/material/texto/pildora049.pdf>
- Criptografía con curvas elípticas, MOOC Crypt4you, Josep M. Miret, Javier Valera y Magda Valls, 2015
  - <https://www.criptored.es/crypt4you/temas/ECC/leccion1/leccion1.html>
- Criptografía de curva elíptica, Wikipedia
  - [https://es.wikipedia.org/wiki/Criptograf%C3%ADA\\_de\\_curva\\_el%C3%ADptica](https://es.wikipedia.org/wiki/Criptograf%C3%ADA_de_curva_el%C3%ADptica)

# Bibliografía recomendada (2/2)

- Elliptic-curve cryptography, Wikipedia
  - [https://en.wikipedia.org/wiki/Elliptic-curve\\_cryptography](https://en.wikipedia.org/wiki/Elliptic-curve_cryptography)
- Criptografía con curvas elípticas, Llorenç Huguet Rotger, Josep Rifà Coma et Juan Gabriel Tena Ayuso, PID\_00200952
  - [https://www.exabyteinformatica.com/uoc/Informatica/Criptografia\\_avanzada/Criptografia\\_avanzada\\_\(Modulo\\_4\).pdf](https://www.exabyteinformatica.com/uoc/Informatica/Criptografia_avanzada/Criptografia_avanzada_(Modulo_4).pdf)
- Criptografía con curvas elípticas, Víctor Gayoso Martínez, Luis Hernández Encinas et Agustín Martín Muñoz, 2018
  - <https://www.casadellibro.com/libro-criptografia-con-curvas-elipticas/9788400104320/8928850>



Una inadecuada implementación de la criptografía hace que, en vez de atacarla, sea más eficiente esquivarla