

**Universidad de Buenos Aires**

**Facultad de Ingeniería**

**Departamento de Informática**

***Organización de Datos (75.06)***

**Voto Electrónico**

**RSA**

Cuatrimestre y año: 2do Cuatrimestre 2011

Docente a cargo del TP:Nicolás Pablo Fernández Theillet

Grupo: Lamas

Fecha de Entrega: 2011-11-25

Integrantes:

|  |  |  |
| --- | --- | --- |
| ***Padrón*** | ***Nombre*** | ***Email*** |
| 91187 | Gonzalez Durand, Juan Manuel | jmanuel.gonzalez.durand@gmail.com |
| 90762 | Ostrowsky, Gabriel | gaby.ostro@gmail.com |
| 90728 | Schenkelman, Damián | damian.schenkelman@gmail.com |
| 91045 | Torrado, Alejandro | aletorrado@gmail.com |
| 90884 | Zamudio, Gonzalo | ahogadosderazon@gmail.com |

Diseño

En esta sección se explican algunas decisiones de diseño tomadas.

## Longitud máxima claves

Como sabemos el sistema de encriptación hace uso de enteros grandes. Para simplificar el desarrollo del sistema de encriptación, los elementos que componen a las claves (n, e y d) tienen como máximo 64 bits de longitud (tamaño del tipo **long long**). De esta forma, se evita la necesidad de crear un componente para manejo de enteros de mayor longitud o de utilizar una librería externa, ya que las operaciones aritméticas están soportadas por el lenguaje.

## Configuración Tamaño Claves

Se puede configurar la cantidad máxima de bits del valor n a partir del archivo de configuración **config.txt** ubicado en la carpeta **Files**.El formato para el mismo es:

|  |
| --- |
| KeySize,{tamaño en bytes},0,0,0 |

Por ejemplo, el caso siguiente configura el tamaño máximo de n en 8 bytes:

|  |
| --- |
| KeySize,8,0,0,0 |

|  |
| --- |
| **Nota:** Los 0s a la derecha del tamaño no son tenidos en cuenta, pero sirven para poder usar el soporte para configuración implementado en la primer parte del trabajo. |

Los posibles valores para el tamaño en bytes son:

* 2
* 4
* 6
* 8

Si llamamos t al tamaño de la configuración, la implementación asegura que:

## Encriptación

Al momento de encriptar se siguen los siguientes pasos.

1. Sea b el número del bit más significativo de n que vale 1.
2. Define el tamaño del *chunk (chunksize)* como *floor(b / 8)*.
3. Divide el mensaje a encriptar en *chunks* y encripta cada uno por separado, agregandolo al criptograma final.

|  |
| --- |
| **Nota:** Como se puede deducir, el tamaño del criptograma será probablemente mayor al del mensaje, ya que por ejemplo si *chunksize* = 6 y b = 53 se usarán 7 bits para representar 6 caracteres. |

Al momento de des-encriptar, se recibe como parámetro la longitud del criptograma (no del mensaje original) y se revierte el proceso original, es decir se toma *chunksize* + 1 para desencriptar.