Universidad Nacional ARTURO JAURETCHE

Sistemas Operativos 2

Laboratorio 3
Criptografía V1.0

Objetivos

Los objetivos de este laboratorio son:

- Familiarizarse con la esteganografía
- Familiarizarce con la utilización de llaves públicas y privadas..

Breve descripción

Se utilizarán técnicas de desencriptación de una imágen con un texto encriptado mediante la técnica de “Esteganografía”. Se realizará una posterior encriptación con llave publica y se debe realizar el código para descifrarlo con una llave privada

Herramientas

Para la realización del laboratorio se utilizarán dos ejemplos de código, que se describen en las secciones siguientes y una imagen de linux con las librerías necesarias pre-instaladas:

A continuación se describen los datos necesarios para su operación:
La imagen de linux es en formato OVA (para ser utilizada en VirtualBox 3.0 o superior)

Imagen de linux:

Versión de linux Debian 7 sin entorno grafico.
Usuario: linuxtest
password: 123456
Usuario: root
password: 123456

Carpetas compartidas

En virtualbox, definir como carpeta compartida el sitio en donde se guardarán los fuentes en la máquina host. El nombre de la carpeta es indistinto, pero cuando se realice la configuración de VB, para carpetas compartidas, el recurso se debe llamar “share”

script de montaje: mount.sh
carpeta de montaje /home/linuxtest/mnt
recurso compartido host: share

Librerías instaladas

Librerias necesarias instaladas: gcc, gcc-multilib

Ejercicios de ejemplo

Para realizar esta práctica se proveen dos ejemplos:

Imagen.c para entender como se opera sobre imágenes BMP

Crypt.c: para realizar un algorítmo RSA en C

Imagen.c

ejemplo de manipulación de una imagen en formato BMP.

Las imágenes BMP están compuestas por 4 partes bien definidas.

- Código de cabecera
- Metadatos con información del archivo
- Metadatos con información de la imágen
- Datos binarios planos de la imagen en RGB (24 bits)

El código de cabecera debe ser 0x4D42 en hexa.

La estructura de los metadatos de información tiene las siguientes instancias:

```
uint32_t size;          /* Tamaño del archivo */
uint16_t resv1;        /* Reservado */
uint16_t resv2;        /* Reservado */
uint32_t offset;       /* Offset hasta hasta los datos de imagen */
```

La estructura para los metadatos de la imágen tiene las siguientes instancias

```
uint32_t headersize;     /* Tamaño de la cabecera */
uint32_t width;   /* Ancho */
uint32_t height;  /* Alto */
uint16_t planes;  /* Planos de color (Siempre 1) */
uint16_t bpp;            /* bits por pixel */
uint32_t compress;       /* compresión */
uint32_t imgsize;        /* tamaño de los datos de imagen */
uint32_t bpmx;    /* Resolución X en bits por metro */
uint32_t bpmy;    /* Resolución Y en bits por metro */
uint32_t colors;  /* colores usados en la paleta */
uint32_t imxtcolors;     /* Colores importantes. 0 si son todos */
```

La sección de la imagen está compuesta por los valores consecutivos que forman los tonos Red, Green y Blue (RGB) respectivemente. Cáda uno de los tonos tienen 8 bits (1 byte)

La trama se debería ver de la siguiente forma:
RGBRGBRGBRGBRGB.....

En este ejemplo se reemplaza uno de los valores por 0x00 o 0xFF de tal forma que quede de la siguiente forma:
RG0RG0RG0RG0.....

El resultado es el siguiente:

Imagen original

Imagen procesada

Crypt.c

Realiza -la encriptación de un valor fijo:

```
// Message to be encrypted
double msg = 12;
```

Mediante el algorítmo RSA visto en clase con una llave publica:

Se buscan dos N° primos grandes p y q, y se obtiene n = p.q

```
// Two random prime numbers
double p = 3;
double q = 7;
double n = p*q;
```

Se buscan dos números e y d que contemplen la siguiente propiedad:

- e.d mod [(p-1).(q-1)] = 1

```
// Finding other part of public key.
// e stands for encrypt
double e = 2;
double phi = (p-1)*(q-1);
while (e < phi)
{
    // e must be co-prime to phi and
    // smaller than phi.
    if (gcd(e, phi)==1)
        break;
    else
        e++;
}
```

```
    // Private key (d stands for decrypt)
    // choosing d such that it satisfies
    // d*e = 1 + k * totient
    int k = 2;  // A constant value
    double d = (1 + (k*phi))/e;
```

Se requiere de la función:

```
// Returns gcd of a and b
int gcd(int a, int h)
{
    int temp;
    while (1)
    {
        temp = a%h;
        if (temp == 0)
          return h;
        a = h;
        h = temp;
    }
}
```

Se representa el texto llano por medio de un M tal que $0 \leq M \leq n-1$ de forma tal que el encriptado resulta:

$C = M.e \bmod(n)$

```
    // Encryption c = (msg ^ e) % n
    double c = pow(msg, e);
    c = fmod(c, n);
    printf("\nEncrypted data = %lf", c);
```

y el desencriptado es:

$M = C.d \bmod(n)$

```
    // Decryption m = (c ^ d) % n
    double m = pow(c, d);
    m = fmod(m, n);
    printf("\nOriginal Message Sent = %lf", m);
```

La clave pública es el par (e,n) y la privada el par (d,n).

NOTA: este código se compila con

```
gcc Crypt.c -o Crypt -lm
```

-lm es para poder utilizar las librerías de matematica.

Tareas a realizar por el alumno

Se entregará una imagen nueva en BMP, la cual contendrá un texto encriptado en el byte “B”.

Por ejemplo:

RG**M** RG**E** RG**N** RG**S** RG**A** RG**J** RG**E**

1) Se deberá realizar un código que desencripte dicho mensaje utilizando. El alímno deberá utilizar técnicas creativas para asilar la sección de la imágen que tiene texto y la que no y poder limitar el análisis.

2) Realizar un codigo para encriptar el texto con una llame pública y luego realizar otro código para desencriptarlo con una privada.

Formato de entrega

- Se debe entregar informe que contenga.
    - Carátula con nombre y apellido del alumno, fecha y título del trabajo práctico.
    - Descripción de realización de cada punto de ser necesario con capturas de pantalla de los resultados.
    - Conclusiones
- La entrega es individual, en formato PDF y se debe adjuntar un paquete comprimido con los códigos realizados (comentados).