

# Introduccion a la Ingeniería Inversa

Análisis de malware, cracking de software...

Carlos Ledesma Peña    Fernando Díaz Urbano

Grupo de Desarrollo de Google, Málaga

# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

Estado actual

## Caso práctico

GNU Radio

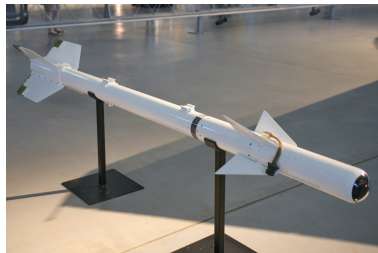
Pager POCSAG

## Para terminar

Conclusiones

Utilidades

## Antecedentes



**Izquierda:** Máquina Enigma  
versión militar alemana

**Arriba:** Misil K-13 soviético  
aire-aire

# Aplicaciones en informática

Este concepto se puede aplicar a diversas aplicaciones, se viene haciendo desde tiempos inmemoriales.

- **Dispositivos electronicos**
- **Programas informáticos**
- **Artilugios de uso cotidiano**

## ¿Qué herramientas necesitamos?

- Conocer la API de Windows(Si el reversing se hace en Windows)
- Debuggers
- Conocimientos en Assembly
- ¡Un cerebro!

## ¿Qué debemos conocer?

- **El entorno**

Sistema operativo, proceso, ejecutable, librería...

- **Los lenguajes**

Ensamblador (x86), bytecode (CIL), scripting (Perl)...

- **Los patrones típicos**

Reconocer un bucle while, una inyección en un proceso...

- **Las herramientas**

Desensamblador, depurador, monitor de recursos...

# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

Estado actual

## Caso práctico

GNU Radio

Pager POCSAG

## Para terminar

Conclusiones

Utilidades

## Análisis estático VS Dinámico

### Estatico:

- **Permite examinar todos los valores de variables y caminos**
- **Encontrar errores que no se manifestarán hasta pasado mucho tiempo**

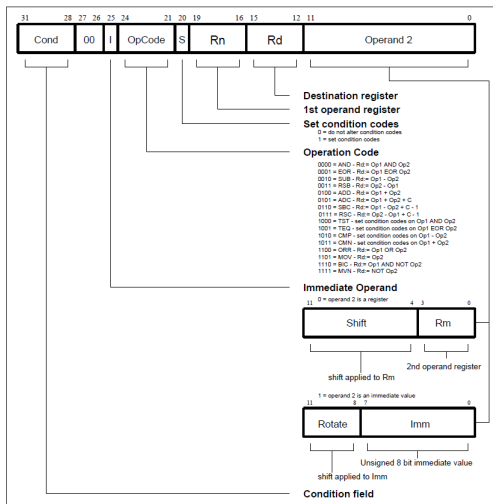
### Dinámico

- **Permite revelar defectos o vulnerabilidades difíciles de encontrar estáticamente**
- **Observar el comportamiento de un ejecutable encriptado**

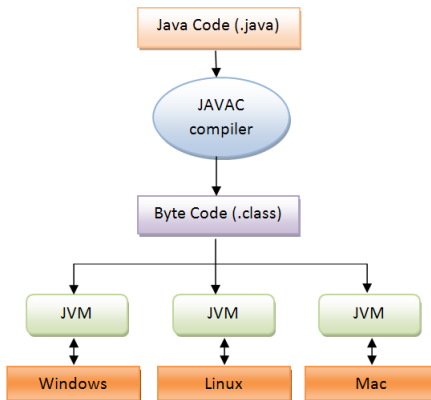
Necesitamos de ambos, para poder llevar a cabo un análisis completo.



# Opcodes



# Bytecode



# WinDBG

Esta herramienta desarrollada por Microsoft, nos permite:

- Debuggear a nivel de Kernel(Ring 0)
- Debuggear el propio Sistema Operativa
- No tiene GUI, funciona a traves de una consola
- Es complicado para principiantes, tedioso

```

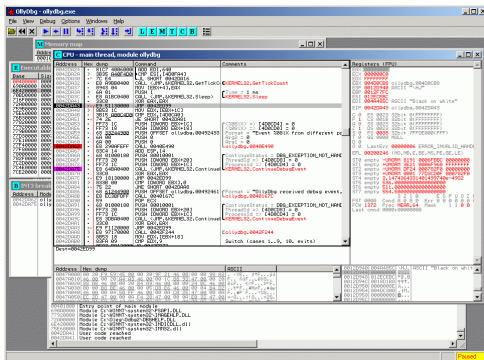
C:\Program Files\Debugging Tools for Windows (x86)\windbg.exe - WinDbg.6.11.0001.404 X86
File Edit View Debug Window Help

Command - "C:\Program Files\Debugging Tools for Windows (x86)\windbg.exe" - WinDbg.6.11.0001.404 X86

ModLoad: 7c900000 7c9b2000 ntdll.dll
ModLoad: 7c800000 7c8f6000 C:\WINDOWS\system32\kernel32.dll
ModLoad: 77d40000 77eb6000 C:\WINDOWS\system32\ADVAPI32.dll
ModLoad: 77e70000 77f03000 C:\WINDOWS\system32\RPCRT4.dll
ModLoad: 77f60000 77f11000 C:\WINDOWS\system32\Secur32.dll
ModLoad: 77f10000 77f59000 C:\WINDOWS\system32\GDI32.dll
ModLoad: 7e410000 7e411000 C:\WINDOWS\system32\USER32.dll
ModLoad: 77c10000 77c88000 C:\WINDOWS\system32\user32.dll
ModLoad: 02000000 02386000 C:\Program Files\Debugging Tools for Windows (x86)\dbgapi.dll
ModLoad: 03000000 03121000 C:\Program Files\Debugging Tools for Windows (x86)\dbgapi.dll
ModLoad: 77e00000 77e38000 C:\WINDOWS\system32\VERSION.dll
ModLoad: 77e00000 77e38000 C:\WINDOWS\system32\USER32.dll
ModLoad: 7c9c0000 7d1d7000 C:\WINDOWS\system32\SHELL32.dll
ModLoad: 77f60000 77f66000 C:\WINDOWS\system32\SHELLAPI.dll
ModLoad: 77340000 774d3000 C:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_
ModLoad: 71b20000 71b32000 C:\WINDOWS\system32\MPF.dll
(a7c f64) Break instruction exception - code 80000003 (first chance)
eax=00191ab4 ebx=7ffdb000 ecx=00000003 edx=00000008 esi=00191f48 edi=00191ab4
eip=7c90120e esp=0006fc20 ebp=0006fc94 iopl=0         nv up ei pl zr na po nc
cs=001b  eip=00120e  ds=0023  esp=0023  ier=003b  gs=0000             efl=00000202
*** ERROR: Symbol file could not be found. Defaulted to export symbols for ntdll.dll
ntdll!DbgBreakPoint:
7c90120e cc                int     3
0:000>
  
```

## Ollydbg

- Posee GUI
- User-level debugging(Ring 3)
- Más sencillo para principiantes



# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

Estado actual

## Caso práctico

GNU Radio

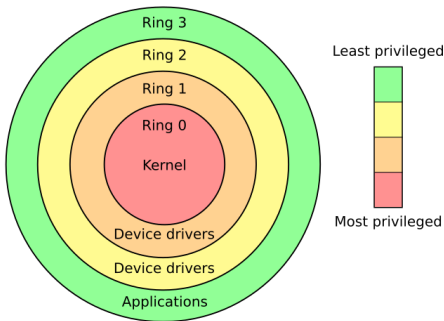
Pager POCSAG

## Para terminar

Conclusiones

Utilidades

# Anillos de protección



## HyperVisor

- **AMD-V:** Pacifica
- **Intel VT-x:** Vanderpool

# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

**Seguridad**

Estado actual

## Caso práctico

GNU Radio

Pager POCSAG

## Para terminar

Conclusiones

Utilidades

## Seguridad

- **2005:** Versión mejorada de *BlueSniper*.
- **2007:** Ataque sobre teclados inalámbricos.
- Ataque sobre el pasaporte europeo.
- **2008:** Michael Ossmann repasa en Black Hat sobre el estado de la seguridad de las radiocomunicaciones, y advierte que SDR accesible es peligroso.
- Ataque sobre el sistema de tarjetas del metro de Boston y de pago remoto en peajes.
- **2009:** Ataque práctico sobre *GSM*.
- **2010:** Lectura de *RFID* a larga distancia.
- **2011:** Ataque sobre *GRPS/EDGE* y *UMTS/HSPA*.



# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

**Estado actual**

## Caso práctico

GNU Radio

Pager POCSAG

## Para terminar

Conclusiones

Utilidades

## Estado actual

- En 2010, Eric Fry se da cuenta de algo extraño al realizar ingeniería inversa a un *driver* de un dispositivo *USB* para recepción *FM* y *DAB+*. Lo que viaja del dispositivo al PC no es audio, sino muestras de la señal en una etapa intermedia entre la señal de radiofrecuencia y el audio.
- En 2012 nace el proyecto *rtl-sdr*, que proporciona una interfaz para usar estos dispositivos como SDR's (sólo recepción, pero muy asequibles).
- Interés en integrar SDR en la comunidad de pentesting, con nuevas herramientas que permiten inyectar paquetes de diversos protocolos al vuelo.

# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

Estado actual

## Caso práctico

GNU Radio

Pager POCSAG

## Para terminar

Conclusiones

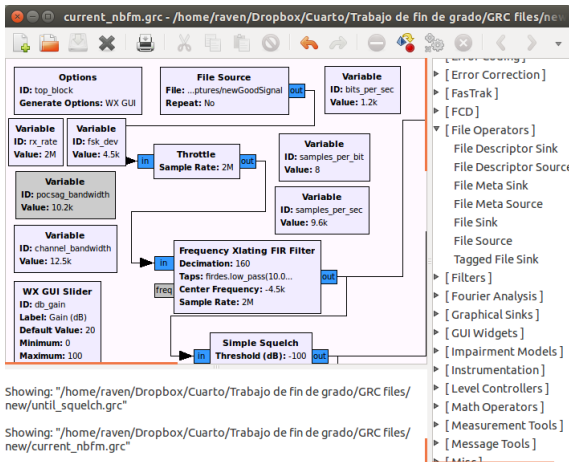
Utilidades

## ¿Qué es GNU Radio?

*GNU Radio* es un entorno de desarrollo *open source* multiplataforma de procesamiento de señales en general, si bien está especializado en SDR, pero no limitado a ello.

- Ofrece una interfaz gráfica, *GRC*, además de las interfaces para *Python* y *C++*. La de *Python* es una envoltura de la de *C++*, y la gráfica una envoltura de la de *Python*.
- La interfaz gráfica sirve para crear diagramas de flujo, con conexiones entre bloques que representan funciones de procesamiento de señales.
- Los bloques pueden ser de entrada o salida, para interactuar con el exterior (parte hardware de SDR, tarjeta de audio, disco duro...), o de entrada y salida, implementando funciones en sí.

## GRC



# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

Estado actual

## Caso práctico

GNU Radio

Pager POCSAG

## Para terminar

Conclusiones

Utilidades

## ¿Qué es un pager POCSAG?



## Al ataque

- Tras escanear en el rango de frecuencias que menciona el *pager* en su parte de atrás, y los avisos que alcanzo a capturar se emiten en la misma frecuencia.
- Construyo el diagrama de flujo (no sin mucho esfuerzo) para decodificar con arreglo al estándar *POCSAG* y efectivamente, se ajusta al estándar. Cada dispositivo tiene un ID, y suena cuando se emite el suyo.
- Modifico el diagrama de flujo y creo un bloque personalizado para *GRC* para imprimir en consola los ID's según se capturan los avisos.



## Dificultades encontradas

- Dominio completamente nuevo para mí, y falta de base sólida a la hora de resolver los problemas (días de diagnóstico por problema).
- *GRC* no está hecho para aprender a base de prueba y error desde el principio, no es fácil saber qué está fallando ni por qué (curva de aprendizaje elevada).
- Limitaciones del hardware, mi portátil usa *USB 2.0*, lo que limita el ancho de banda capturable de una vez, además de no tener potencia de procesamiento suficiente y descartar muestras si se usaban varias operaciones simultáneamente.

# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

Estado actual

## Caso práctico

GNU Radio

Pager POCSAG

## Para terminar

Conclusiones

Utilidades

## Conclusiones

- Desde el punto de vista económico y humano, es necesario invertir en la seguridad de los sistemas informáticos. No sólo se protege de las malas intenciones, sino de las buenas intenciones equivocadas.
- La "seguridad" por oscuridad no es seguridad, si un sistema necesita que su forma de funcionar no sea pública para ser seguro, no es seguro igualmente. *Sólo la clave debe ser desconocida para el resto.*
- No se le ha prestado suficiente atención a la seguridad de las radiocomunicaciones en el pasado, y ahora se dispone de herramientas basadas en SDR que facilitan aprovecharse de sistemas vulnerables. Hay que prestarle atención desde ya.

# Índice

## Introducción

¿Qué es la ingeniería inversa?

Conceptos básicos

## Evolución en el tiempo de la tecnología SDR

Comienzos

Seguridad

Estado actual

## Caso práctico

GNU Radio

Pager POCSAG

## Para terminar

Conclusiones

Utilidades

## Utilidades

- Aprender conocimientos básicos de radio (inquietud personal).
- Incorporar una nueva herramienta de trabajo (es posible que se incorpore en labores de pentesting).
- Servir de guía de inicio rápido a SDR a los investigadores del departamento.
- Obtener el título de Graduado en Ingeniería Informática.

**¡Gracias por vuestra atención!**