# ESTUDIO DE PROTECCIONES BASICO PARA PRINCIPIANTES (También llamado cracking)

**Impartido por Ratón**

Nota

Cada capitulo ira acompañado de su crackme correspondiente.No facilitare paginas de donde bajarse herramientas ni enlaces a paginas de crackers, la intención es que busquéis en Internet todo lo necesario. Seguro que encontráis mas paginas de herramientas, tutoriales y utilidades relacionadas con este tema que las que yo pueda deciros.
Con esto solo quiero fomentar vuestro interés, además se que la búsqueda os proporcionara gratas sorpresas.
A todos un saludo.

## Con este capitulo adjunto dos manuales sobre Olly Debugger de The Pope

## Capitulo 0

## Objetivo

Prepararse minimamente para el curso y ver de qué va el rollo.

## Introducción

Esta idea surge con el fin de rellenar el hueco que bajo mi punto de vista falta en el foro de HackxCrack sobre el tema Cracking.
Este pretende ser un cursillo muy básico de aproximación al estudio de las protecciones en los programas comerciales examinando sus puntos vulnerables.
Esta dirigido a personas que nunca hayan tenido contacto con este tema y explicado de la mejor manera que pueda explicarlo un principiante (o sea yo).
Lo explicare como a mi me hubiera gustado que me lo explicaran, por tanto será explicado de la manera más llana y sencilla posible explicando todo de forma detallada (siempre dentro de mis posibilidades).
Será tratado partiendo de crackmes, pequeños programas construidos para simular la protección de los programas comerciales.
También veremos algún programa shareware que tenga relación con el tema, sin modificar el programa pues seria un acto ilegal, observaremos el programa cargado en memoria sin realizar cambio alguno en el ejecutable, solo como complemento y ejemplo practico de alguno de los capítulos del curso.
Esta pensado para complementarse con tutoriales e información difundida por la red, pues como digo trataremos lo mas básico y si tenéis ansias de saber no os bastara solo con la información que se incluya en cada capitulo.
Constara en principio de 10 capítulos comprimidos en formato rar para descargar, con sus correspondientes crackmes y material de apoyo si fuera necesario.
Acepto críticas, ayudas y correcciones.
Espero animar con esto a que colabore gente con más y más amplios conocimientos que yo en la materia (cosa no muy difícil, por cierto).
Si alguien desea unirse al proyecto será bienvenido.
Para cualquier pregunta sobre el tema habilitare un hilo en el foro. No contestare

preguntas sobre este tema en ningún otro hilo con el fin de centrar todo lo que tenga que ver con el tema crack en un solo hilo para facilitar vuestras consultas.
La intención no es solo utilizar este hilo para el curso sino agrupar en el cualquier pregunta relacionada con el tema crack.

A la hora de explicar utilizare distintos términos para nombrar la misma cosa, por ejemplo: programa – ejecutable ó cadena – String.
Esto lo haré para que veáis las distintas formas de llamar a la misma cosa y os aclaréis al leer tutoriales de otros crackers.
Click o clickear será la acción de pulsar el botón izquierdo del ratón, click derecho será la acción de pulsar el botón derecho del ratón, doble click será la acción de pulsar 2 veces el botón izquierdo del ratón.
Ya iréis pillando mi forma de explicar (espero).

## Herramientas necesarias.

Será importante hacerse con ciertas herramientas.
Os daré una base sobre la que partir y vosotros mismos iréis adaptándoos a lo que necesitéis según vayáis aprendiendo.
En principio seria interesante que consiguierais poco a poco estos programas:

Necesario para el capitulo 1 Olly Debugger (debugger) El curso se basa al 90% en este programa. Imprescindible.

Siguientes capítulos W32Dasm disassembler (desemsamblador)
DeDe Delphi decompiler (descompilador de ejecutables programados en delphi)
PE Explorer (editor de recursos)
Exescope (editor de recursos)
Peid (detector de protecciones de ejecutables)
Pe scan (detector de protecciones de ejecutables)
Procdump (desempacador genérico de protecciones de ejecutables)
Ida pro (desemsamblador)
Smartcheck (debugger visual basic)
Resource hacker (editor de recursos)
Patch FX (creador de cracks para distribuir)

La intuición, la paciencia y el empeño esta en vuestras manos.

Si necesitáramos algún programa específico que no estuviera en esta lista (un editor hexadecimal por ejemplo) lo diré con antelación.

Procurad conseguir la versión mas reciente de los programas así como sus plugins si los tuvieran.

Aparte no estaría de más que mirarais algún tutorial de ensamblador para ver las instrucciones más comunes.
No es necesario aprender a programar en ensamblador, lo que necesitemos saber se ira aprendiendo poco a poco.

De momento aquí os dejo una pequeña tabla con instrucciones en lenguaje ensamblador sacada de diversos libros y paginas Web.
También una tabla de conversión ASCII – Hexadecimal.
Para que toméis contacto con el Olly Debugger incluyo dos manuales (muy buenos manuales, mirad la carpeta correspondiente a este capitulo) de The POPE (gracias The POPE).

No es necesario estudiar el lenguaje ensamblador, pero estaría bien que le echarais un vistazo por lo menos a las instrucciones de salto.

## ENSAMBLADOR

### Registros de la CPU

La CPU tiene 14 registros internos, cada uno de 16 bits. Los primeros cuatro, AX, BX, CX, y
DX son registros de uso general y también pueden ser utilizados como registros de 8 bits,
para utilizarlos como tales es necesario referirse a ellos como por ejemplo: AH y AL, que
son los bytes alto (high) y bajo (low) del registro AX. Esta nomenclatura es aplicable también a los registros BX, CX y DX.

Los registros son conocidos por sus nombres específicos:

AX Acumulador
BX Registro base
CX Registro contador
DX Registro de datos
DS Registro del segmento de datos
ES Registro del segmento extra
SS Registro del segmento de pila
CS Registro del segmento de código

BP Registro de apuntadores base
SI Registro índice fuente
DI Registro índice destino
SP Registro del apuntador de la pila
IP Registro de apuntador de siguiente instrucción
F Registro de banderas

Instrucciones lógicas.

Son utilizadas para realizar operaciones lógicas sobre los operandos.

AND
NEG
NOT
OR
TEST
XOR

Instrucciones aritméticas.

Se usan para realizar operaciones aritméticas sobre los operandos.

ADC
ADD
DIV
IDIV
MUL
IMUL
SBB
SUB

Operaciones CALL y RET

La instrucción CALL transfiere el control a un procedimiento llamado, y la instrucción RET
regresa del procedimiento llamado al procedimiento original que hizo la llamada. RET debe
ser la ultima instrucción en un procedimiento llamado. Los formatos generales para CALL y

RET son:

(etiqueta) CALL (procedimiento)
(etiqueta) RET (inmediato)

El código objeto particular que CALL y RET generan depende de si la operación implica un
procedimiento NEAR (cercano) o un procedimiento FAR (lejano).

Instrucciones para ciclos

LOOP Transfieren el flujo del proceso, condicional o

incondicionalmente, a un destino repitiéndose esta acción hasta que el contador sea cero.

LOOP
LOOPE
LOOPNE

Instrucciones de conteo

Se utilizan para decrementar o incrementar el contenido de los
contadores.

DEC
INC

Instrucciones de comparación Son usadas para comparar operandos, afectan al contenido de las
banderas.

CMP
CMPS (CMPSB) (CMPSW)

Instrucciones de banderas

Afectan directamente al contenido de las banderas.

CLC
CLD
CLI
CMC
STC
STD
STI

Instrucciones de salto

Son utilizadas para transferir el flujo del proceso al operando indicado.

JMP
JA (JNBE)
JAE (JNBE)
JB (JNAE)
JBE (JNA)
JE (JZ)
JNE (JNZ)
JG (JNLE)
JGE (JNL)
JL (JNGE)
JLE (JNG)
JC
JNC
JNO
JNP (JPO)
JNS
JO
JP (JPE)
JS

Instrucción JMP
Propósito: Salto incondicional
Sintaxis:
JMP destino
Esta instrucción se utiliza para desviar el flujo de un programa sin tomar en cuenta las

condiciones actuales de las banderas ni de los datos.

Instrucción JA (JNBE)

Propósito: Brinco condicional
Sintaxis:
JA Etiqueta
Después de una comparación este comando salta si está arriba o salta si no está abajo o si

no es igual.
Esto significa que el salto se realiza solo si la bandera CF esta desactivada o si la

bandera ZF esta desactivada (que alguna de las dos sea igual a cero).

Instrucción JAE (JNB)
Propósito: salto condicional
Sintaxis:
JAE etiqueta
Salta si está arriba o si es igual o salta si no está abajo.
El salto se efectúa si CF esta desactivada.

Instrucción JB (JNAE)
Propósito: salto condicional
Sintaxis:
JB etiqueta
Salta si está abajo o salta si no está arriba o si no es igual.
Se efectúa el salto si CF esta activada.

Instrucción JBE (JNA)
Propósito: salto condicional
Sintaxis:
JBE etiqueta
Salta si está abajo o si es igual o salta si no está arriba.
El salto se efectúa si CF está activado o si ZF está activado (que cualquiera sea igual a

1).

Instrucción JE (JZ)
Propósito: salto condicional
Sintaxis:
JE etiqueta
Salta si es igual o salta si es cero.
El salto se realiza si ZF está activada.

Instrucción JNE (JNZ)

Propósito: salto condicional
Sintaxis:
JNE etiqueta
Salta si no es igual o salta si no es cero.
El salto se efectúa si ZF está desactivada.
Instrucción JG (JNLE)
Propósito: salto condicional, se toma en cuenta el signo.
Sintaxis:
JG etiqueta
Salta si es más grande o salta si no es menor o igual.
El salto ocurre si ZF = 0 u OF = SF.

Instrucción JGE (JNL)
Propósito: salto condicional, se toma en cuenta el signo.
Sintaxis:
JGE etiqueta
Salta si es más grande o igual o salta si no es menor que.
El salto se realiza si SF = OF

Instrucción JL (JNGE)
Propósito: salto condicional, se toma en cuenta el signo.
Sintaxis:
JL etiqueta
Salta si es menor que o salta si no es mayor o igual.
El salto se efectúa si SF es diferente a OF.

Instrucción JLE (JNG)
Propósito: salto condicional, se toma en cuenta el signo.
Sintaxis:
JLE etiqueta
Salta si es menor o igual o salta si no es más grande.
El salto se realiza si ZF = 1 o si SF es diferente a OF

Instrucción JC
Propósito: salto condicional, se toman en cuenta las banderas.
Sintaxis:
JC etiqueta
Salta si hay acarreo.
El salto se realiza si CF = 1

Instrucción JNC
Propósito: salto condicional, se toma en cuenta el estado de las banderas.
Sintaxis:

JNC etiqueta
Salta si no hay acarreo.
El salto se efectúa si CF = 0.

Instrucción JNO
Propósito: salto condicional, se toma en cuenta el estado de las banderas.
Sintaxis:
JNO etiqueta
Salta si no hay desbordamiento.
El salto se efectúa si OF = 0.

Instrucción JNP (JPO)
Propósito: salto condicional, toma en cuenta el estado de las banderas.
Sintaxis:
JNP etiqueta
Salta si no hay paridad o salta si la paridad es non.
El salto ocurre si PF = 0.

Instrucción JNS
Propósito: salto condicional, toma en cuenta el estado de las banderas.
Sintaxis:
JNP etiqueta
Salta si el signo esta desactivado.
El salto se efectúa si SF = 0.

Instrucción JO
Propósito: salto condicional, toma en cuenta el estado de las banderas.
Sintaxis:
JO etiqueta
Salta si hay desbordamiento (overflow).
El salto se realiza si OF = 1.

Instrucción JP (JPE)
Propósito: salto condicional, toma en cuenta el estado de las banderas.
Sintaxis:
JP etiqueta
Salta si hay paridad o salta si la paridad es par.
El salto se efectúa si PF = 1.

Instrucción JS
Propósito: salto condicional, toma en cuenta el estado de las banderas.
Sintaxis:
JS etiqueta

Salta si el signo está prendido.
El salto se efectúa si SF = 1.

Instrucciones de transferencia
Son utilizadas para mover los contenidos de los operandos. Cada instrucción se puede usar

con diferentes modos de direccionamiento.

MOV
MOVS (MOVSB) (MOVSW)

Instrucciones de carga

Son instrucciones específicas de los registros. Son usadas para cargar en algo en registro
bytes o cadenas de bytes.

LODS (LODSB) (LODSW)
LAHF
LDS
LEA
LES

Instrucciones de la pila

Estas instrucciones permiten el uso de la pila para almacenar y extraer datos.

POP
POPF
PUSH
PUSHF

TABLA DE CONVERSION

| ASCII | Hex | SÃmbolo | ASCII | Hex | SÃmbolo | ASCII | Hex | SÃmbolo | ASCII | Hex | S |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | NUL | 16 | 10 | DLE | 32 | 20 | (espacio) | 48 | 30 | |
| 1 | 1 | SOH | 17 | 11 | DC1 | 33 | 21 | ! | 49 | 31 | |
| 2 | 2 | STX | 18 | 12 | DC2 | 34 | 22 | " | 50 | 32 | |
| 3 | 3 | ETX | 19 | 13 | DC3 | 35 | 23 | # | 51 | 33 | |
| 4 | 4 | EOT | 20 | 14 | DC4 | 36 | 24 | $ | 52 | 34 | |
| 5 | 5 | ENQ | 21 | 15 | NAK | 37 | 25 | % | 53 | 35 | |
| 6 | 6 | ACK | 22 | 16 | SYN | 38 | 26 | & | 54 | 36 | |
| 7 | 7 | BEL | 23 | 17 | ETB | 39 | 27 | ' | 55 | 37 | |
| 8 | 8 | BS | 24 | 18 | CAN | 40 | 28 | ( | 56 | 38 | |
| 9 | 9 | TAB | 25 | 19 | EM | 41 | 29 | ) | 57 | 39 | |
| 10 | A | LF | 26 | 1A | SUB | 42 | 2A | * | 58 | 3A | |
| 11 | B | VT | 27 | 1B | ESC | 43 | 2B | + | 59 | 3B | |
| 12 | C | FF | 28 | 1C | FS | 44 | 2C | , | 60 | 3C | |
| 13 | D | CR | 29 | 1D | GS | 45 | 2D | - | 61 | 3D | |
| 14 | E | SO | 30 | 1E | RS | 46 | 2E | . | 62 | 3E | |
| 15 | F | SI | 31 | 1F | US | 47 | 2F | / | 63 | 3F | |

| ASCII | Hex | SÃmbolo | ASCII | Hex | SÃmbolo | ASCII | Hex | SÃmbolo | ASCII | Hex | S |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 64 | 40 | @ | 80 | 50 | P | 96 | 60 | ` | 112 | 70 | |
| 65 | 41 | A | 81 | 51 | Q | 97 | 61 | a | 113 | 71 | |
| 66 | 42 | B | 82 | 52 | R | 98 | 62 | b | 114 | 72 | |
| 67 | 43 | C | 83 | 53 | S | 99 | 63 | c | 115 | 73 | |
| 68 | 44 | D | 84 | 54 | T | 100 | 64 | d | 116 | 74 | |
| 69 | 45 | E | 85 | 55 | U | 101 | 65 | e | 117 | 75 | |
| 70 | 46 | F | 86 | 56 | V | 102 | 66 | f | 118 | 76 | |
| 71 | 47 | G | 87 | 57 | W | 103 | 67 | g | 119 | 77 | |
| 72 | 48 | H | 88 | 58 | X | 104 | 68 | h | 120 | 78 | |
| 73 | 49 | I | 89 | 59 | Y | 105 | 69 | i | 121 | 79 | |
| 74 | 4A | J | 90 | 5A | Z | 106 | 6A | j | 122 | 7A | |
| 75 | 4B | K | 91 | 5B | [ | 107 | 6B | k | 123 | 7B | |
| 76 | 4C | L | 92 | 5C | \ | 108 | 6C | l | 124 | 7C | |
| 77 | 4D | M | 93 | 5D | ] | 109 | 6D | m | 125 | 7D | |
| 78 | 4E | N | 94 | 5E | ^ | 110 | 6E | n | 126 | 7E | |
| 79 | 4F | O | 95 | 5F | _ | 111 | 6F | o | 127 | 7F | |

Bueno espero que mientras leíais esta introducción estuvierais bajando el OllY, si es así podemos pasar al siguiente capitulo y empezar a trastear con el primer crackme.

Gracias

A todos los crackers y programadores de los cuales he aprendido y sigo aprendiendo.

A los creadores de crackmes

En especial y sin menospreciar a nadie a Ricardo Narvaja por su aportación y su trabajo sobre el estudio de las protecciones y sus tutoriales en castellano y a Makkakko por sus tutoriales con Olly Debugger (Recomendados 100%).
A ti que me estas leyendo.

Ratón Enero 2004