# Análise Superficial de Artefatos Usados em Fraudes Bancárias Através do Comando *file* do Unix

**André Gerhard**

**CCE/USP**

**Aritana Pinheiro Falconi**

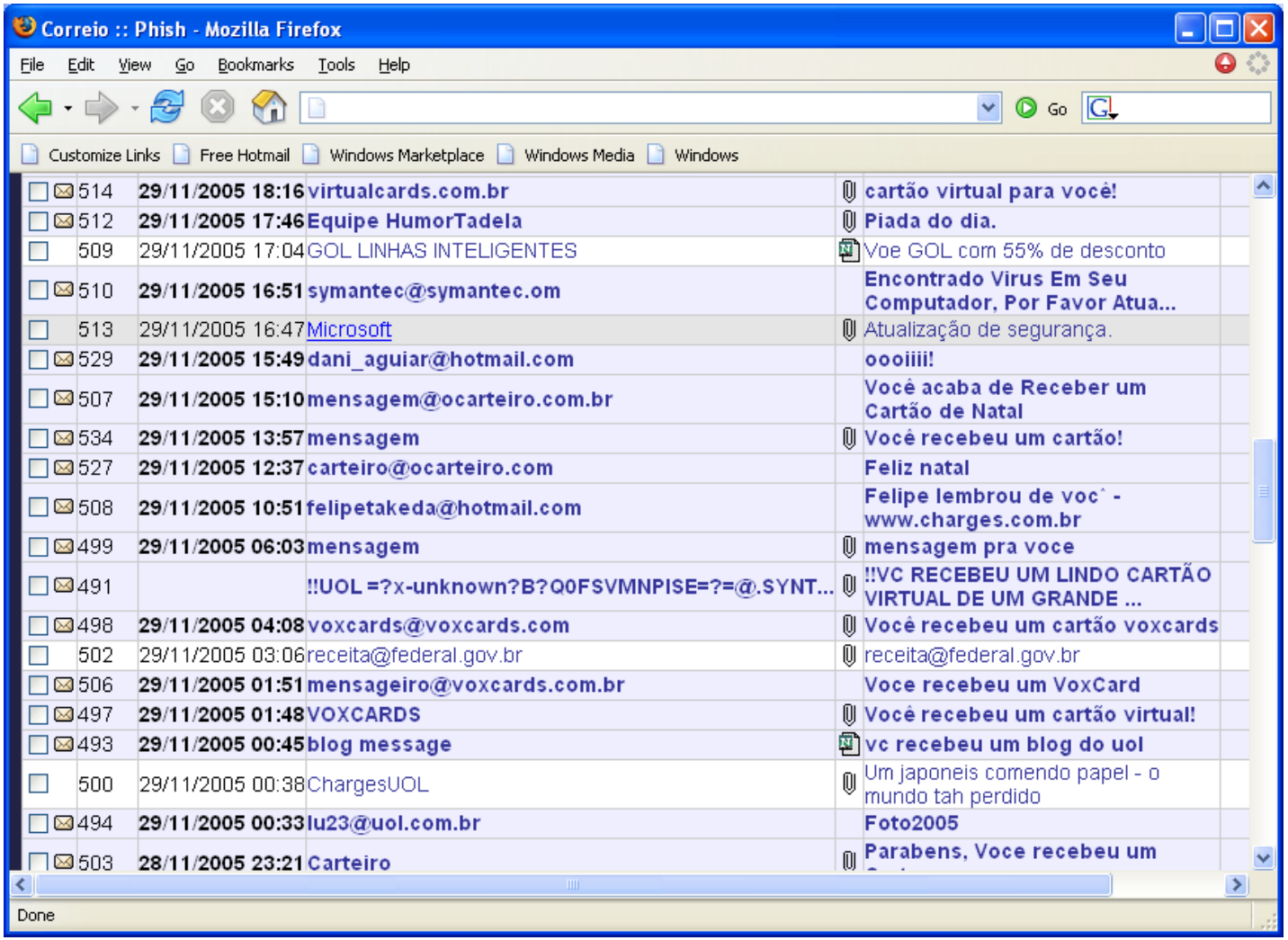
**CGI.br/CERT.br**

**GTS 02.2005**

# Roteiro

- Artefatos em Fraudes Bancárias
- Motivação
- Programa *file*
- Obtenção de Assinaturas
- Resultados
- Conclusão
- Trabalhos Futuros

# Artefatos em Fraudes Bancárias



# Artefatos em Fraudes Bancárias

# Motivação

Dado o grande número de *emails* observado atualmente:

- **Identificar rapidamente o artefato**

Se possível, descobrindo para onde a informação capturada é enviada, quais são os bancos envolvidos, etc.

Observa-se que essa informação não está disponível tão facilmente assim .... Quem faz o artefato, busca esconder esses dados, tornar o programa menor.

# Motivação

- Um único comando
- Ambiente Unix
- Utilização em outras ferramentas

- Análise posterior do artefato (manual ou estática)
- Extrair o artefato original, retirando camadas de proteção (packers & archivers)

# Packer / Archiver

- Packer
  - Comprime e criptografa (ofusca) executáveis
  - Geralmente um único arquivo
  - Exemplos: UPX, ASPack
- Archiver
  - Comprime apenas
  - Um ou mais arquivos
  - Exemplos: Zip, RAR

# File

**“Determines file type using magic numbers”**

- Baseia-se num arquivo de regras denominado *magic*
- Disponível em qualquer Unix

- *magic* default não identifica a maioria dos Packers/Archivers

➔ Criação de regras que permitam identificar esses formatos

# File - Versões

- ftp://www.astron.com/pub/file (Christos Zoulas)

  File 4.09 (06/04/2004): bug, regra “(0x3c)” não funciona
  File 4.14 (25/06/2005): indirect offset/search mode
  File 4.16 (17/10/2005): última versão

- Versões atuais:

  OpenBSD 3.8: 4.09
  FreeBSD 6.0: 4.12
  Debian stable: 4.12
  Fedora 4: 4.15

# Arquivo *magic*

Linhas: *offset* *type* *test* *message*

```
0                       string    MZ          MS-DOS executable (EXE)
>24                     string    @           \b, OS/2 or Windows
>>(0x3c)                string    PE\0\0      \b, MS Windows PE
>>>(0x3c+0x238)         string    .armp       \b, ARM Packed
```

# Obtenção de Assinaturas

Metodologia:

- Inspeção do arquivo
- TrIDscan

- Validação com o antivírus Kaspersky

# Inspeção do Executável

Windows: **Portable Executable File Format (PE)**

Estrutura: DOS Header, DOS Stub, **PE File Header**, Image Optional Header, **Section Table**, Sections.

- **0x3C**: PE Header Pointer
- **0xF8**: Primeira seção (relativo ao PE Header Pointer)
- **40 bytes**: Tamanho de cada seção na tabela

No exemplo anterior: 0x238 = 0xF8 + 8*40

# Inspeção do Executável

```
0000000: 4d5a 5000 0200 0000 0400 0f00 ffff 0000  MZP.............
0000010: b800 0000 0000 0000 4000 1a00 0000 0000  ........@.......
0000020: 0000 0000 0000 0000 0000 0000 0000 0000  ................
0000030: 0000 0000 0000 0000 0000 0000 0002 0000  ................
0000040: ba10 000e 1fb4 09cd 21b8 014c cd21 9090  ........!..L.!..
0000050: 5468 6973 2070 726f 6772 616d 206d 7573  This program mus
0000060: 7420 6265 2072 756e 2075 6e64 6572 2057  t be run under W
0000070: 696e 3332 0d0a 2437 0000 0000 0000 0000  in32..$7........
0000080: 0000 0000 0000 0000 0000 0000 0000 0000  ................

0000200: 5045 0000 4c01 0300 5c01 e13c 0000 0000  PE..L...\..<....
0000210: 0000 0000 e000 0f01 0b01 0500 00a0 0000  ................

00002e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00002f0: 0000 0000 0000 0000 5550 5830 0000 0000  ........UPX0....
0000300: 0030 0100 0010 0000 0000 0000 0004 0000  .0..............
```

**Regra:**

**>>>(0x3c.s+0xF8)  string  UPX0  \b, UPX Packed**

# *file* – dificuldades

Regras de acesso indireto são limitadas:

- Em alguns formatos, a identificação do tipo encontra-se no final do arquivo, depende do seu tamanho (Por exemplo: Zip, RAR)

➔ Mesmo assim, é possível obter assinaturas que permitam identificar esses formatos: identificando pontos (bytes) em comum em vários arquivos do mesmo tipo

# TrID / TrIDScan

**“TrID è un utility per l'identificazione di binary files”**

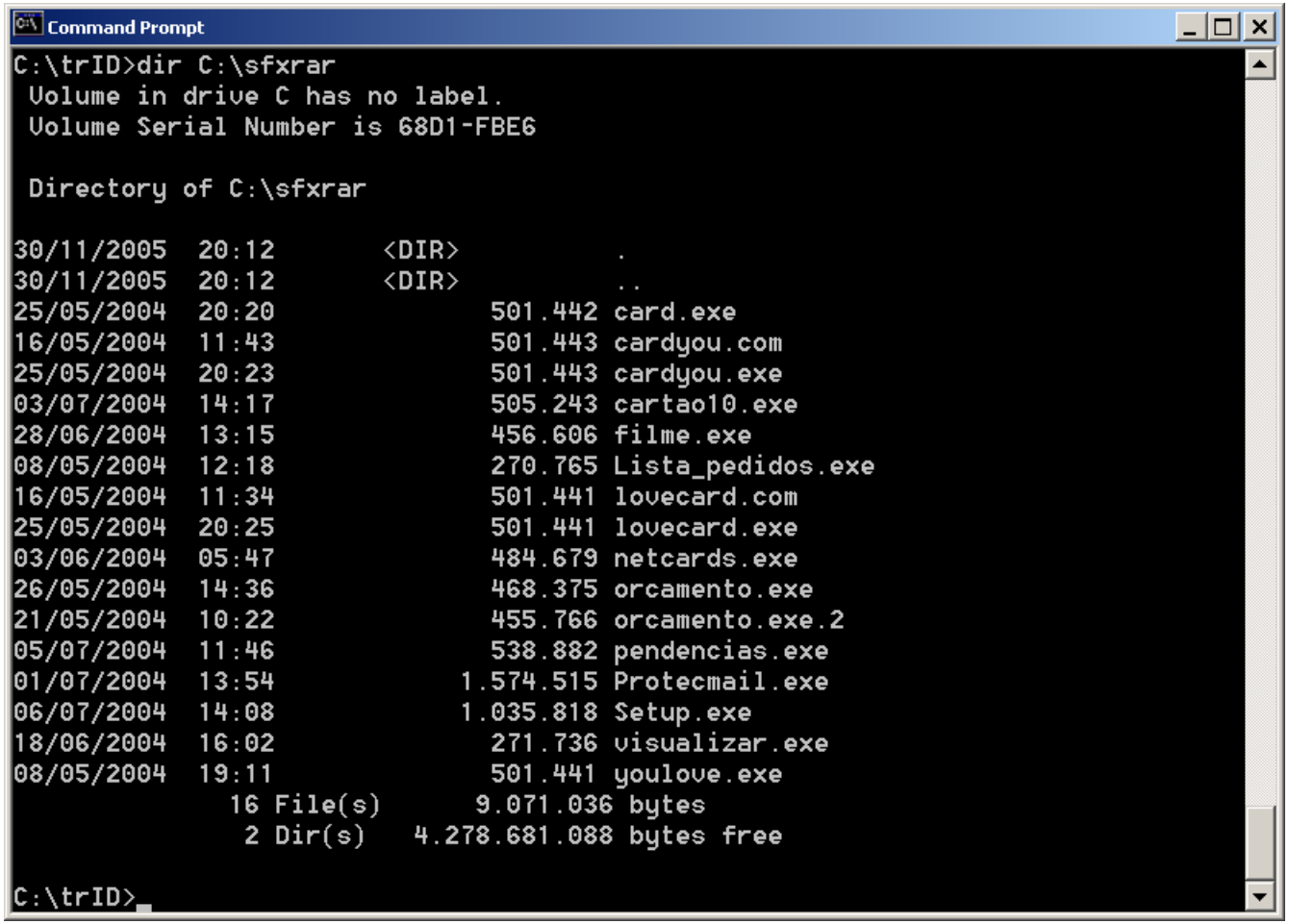
http://mark0.net/soft-tridscan-e.html (Marco Pontello)

Ferramenta Windows, código proprietário ☹

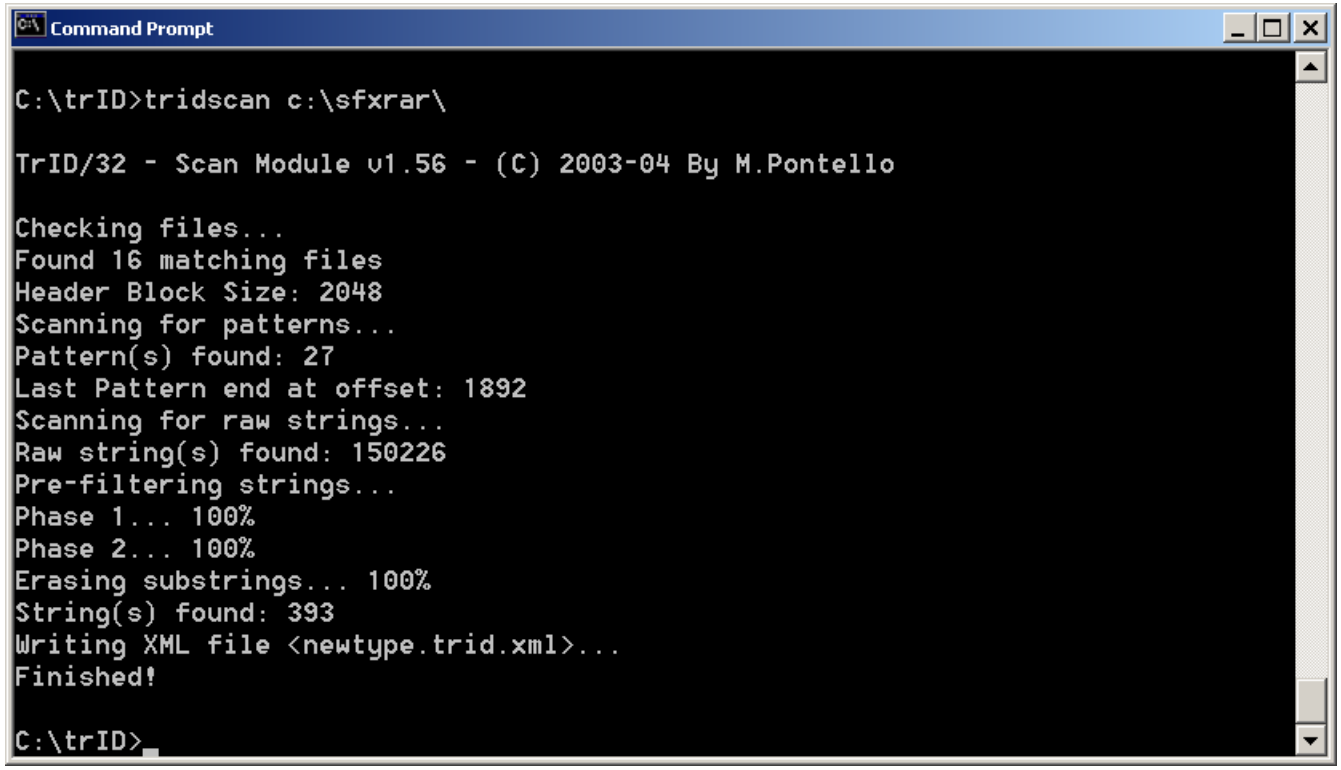
Comparação de vários arquivos com tipo *previamente conhecido*, identificando pontos em comum entre eles

Exemplo: RAR SFX

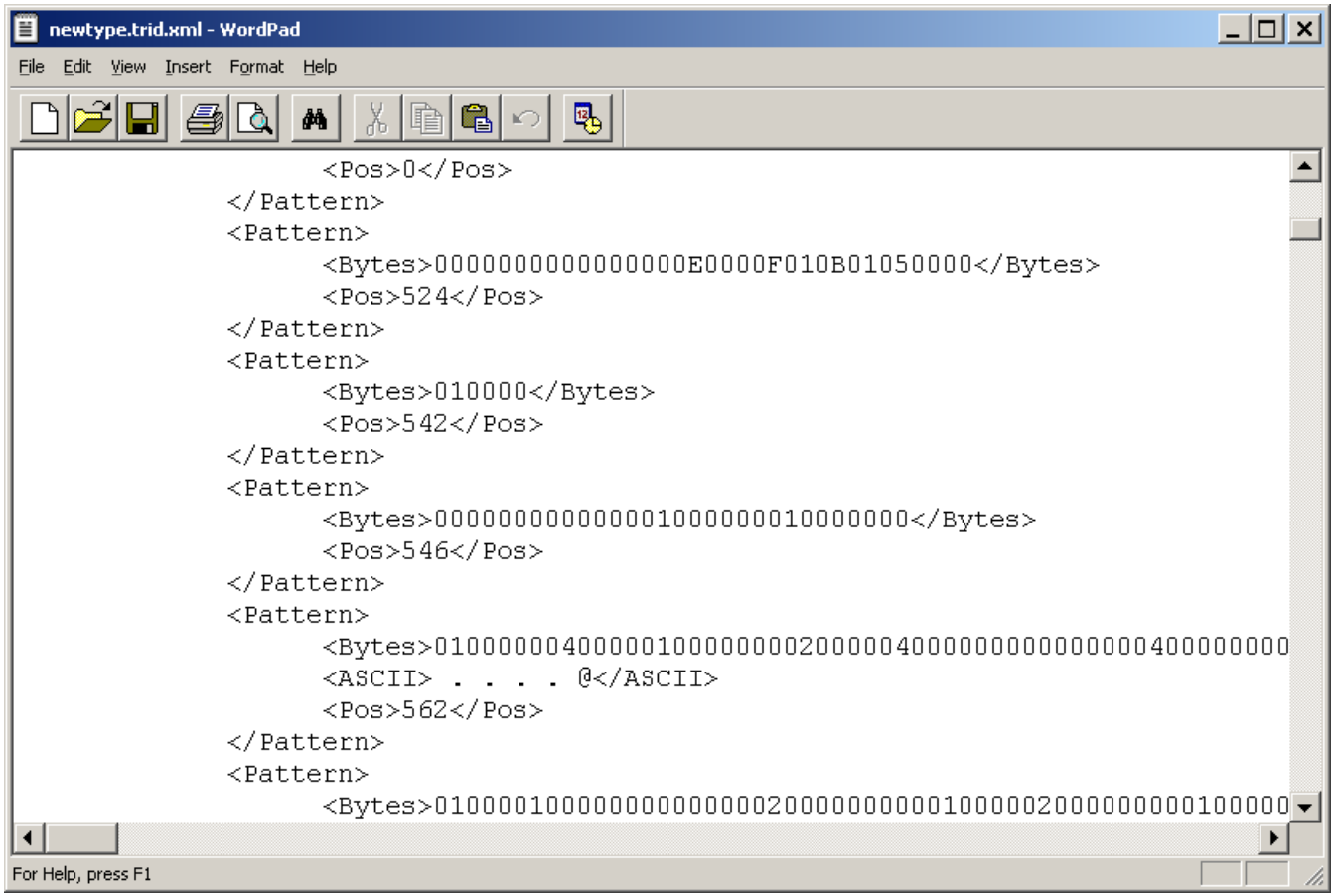
# TrIDScan - Exemplo



# TrIDScan - Exemplo



# TrIDScan - Exemplo



# TrIDScan - Exemplo

1. Escolher alguns padrões identificados pelo trIDScan
2. Criar a regra:

```
>542          string  \x01\x00\x00
>>814         string  \x01\x00\x00
>>>1723       string  \x00
>>>>1773      string  \x00\x00
>>>>>1892     string  \x00          \b, RAR SFX Archived
```

3. Testar a regra em novas amostras

- Não é necessário usar todos os padrões !
- Cuidado com falsos positivos
- Contexto: análise de artefatos

# Assinaturas (*magic*)

```
# .EXE formats and packers/archivers (Andre Gerhard, agerhard@usp.br;
# Aritana Falconi, falconi@cert.br)
#
0                   string  MZ                      MS-DOS executable (EXE)
>0x14               string  FSG!                    \b, FSG Packed
>24                 string  @           \b, OS/2 or Windows
>>(0x3c)            string  PE                      \b, MS Windows PE
>>>(0x3c.s+0xF8)    string    UPX0                  \b, UPX Packed
>>>(0x3c.s+0xF8)    string    .Upack                \b, UPack Packed
>>>(0x3c.s+0xF8)    string    pec1                  \b, PECompact Packed
>>>(0x3c.s+0x110)   string    PEC2                  \b, PE_Patch.PECompact Packed
>>>(0x3c.s+0x198)   string    .perplex  \b, PE_Patch.UltraProtect Packed
>>>(0x3c.s+0x238)   string    .perplex  \b, PE_Patch.UltraProtect Packed
>>>(0x3c.s+0x238)   string    .armp                 \b, ARM Packed
```

# Assinaturas (cont.)

```
>>>(0x3c.s+0xF8)    string    .petite            \b, Petite Packed
>>>(0x3c.s+0x120)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x148)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x170)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x198)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x1c0)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x1e8)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x210)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x238)   string    .petite            \b, Petite Packed
>>>(0x3c.s+0x120)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x148)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x170)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x198)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x1c0)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x1e8)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x210)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x238)   string    .aspack            \b, ASPack Packed
>>>(0x3c.s+0x260)   string    .aspack            \b, ASPack Packed
```

# Assinaturas (cont.)

```
>>>1969              string    \x00\x6a\x00\x6a\x00          \b, ZIP SFX Archived
>>>0x400             string    \x56\x57\x8b\x74\x24          \b, SFX CAB Archived
>>>0x400             string    \x31\xc0\x40\x8b\x4c          \b, SFX Maker (ZIP) Archived
>>>0x400             string    \x55\x8b\xec\x81\xec          \b, XCeed Packager (ZIP) Archived
>>>0x1000            string    \x55\x8b\xec\x81\xec          \b, XCeed Packager (ZIP) Archived
>>>0x9c00            string    Rap!                  \b, RAP SFX Archived
>>>0xC004            string  Target                  \b, StubbieMan Archived
>>>542               string  \x01\x00\x00
>>>>814              string  \x01\x00\x00
>>>>>1723            string  \x00
>>>>>>1773           string  \x00\x00
>>>>>>>1892          string  \x00        \b, RAR SFX Archived

# ZIP archives (Greg Roelofs, c/o zip-bugs@wkuvx1.wku.edu)
0               string  PK\003\004              ZIP Archived
# Alternate ZIP string (amc@arwen.cs.berkeley.edu)
0               string  PK00PK\003\004          ZIP Archived
```

# Exemplo – *file* original

```
$ file *

007.exe:                    MS-DOS executable (EXE), OS/2 or Windows
2turma.exe:                 MS-DOS executable (EXE), OS/2 or Windows
Atualizacao.scr:            MS-DOS executable (EXE), OS/2 or Windows
Cartao.exe:                 MS-DOS executable (EXE), OS/2 or Windows
amor.exe:                   MS-DOS executable (EXE), OS/2 or Windows
cadastro_cpf.exe:           MS-DOS executable (EXE), OS/2 or Windows
cardlove.exe:               MS-DOS executable (EXE), OS/2 or Windows
cardsterra.exe:             MS-DOS executable (EXE), OS/2 or Windows
cartao.exe:                 MS-DOS executable (EXE), OS/2 or Windows
```

# Exemplo – novas assinaturas

```
$ file -m ~/magic *

007.exe:            MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE, ASPack
    Packed
2turma.exe:         MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE, UPX Packed
Atualizacao.scr:    MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE,
    PE_Patch.PECompact Packed
Cartao.exe:         MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE,
    PE_Patch.UltraProtect Packed
amor.exe:           MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE, ARM Packed
cadastro_cpf.exe:   MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE, Petite
    Packed
cardlove.exe:       MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE, PECompact
    Packed
cardsterra.exe:     MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE, RAR
    Archived
cartao.exe:         MS-DOS executable (EXE), OS/2 or Windows, MS Windows PE, UPack
    Packed
```

# Resultados

| *Compactador* | *Quantidade* | *%* |
|---|---|---|
| **PE_Patch.PECompact** | 604 | 38.59 |
| **UPX** | 228 | 14.57 |
| **Petite** | 211 | 13.48 |
| **ASPack** | 133 | 8.50 |
| **UPack** | 130 | 8.31 |
| **ZIP** | 68 | 4.34 |
| **RAR** | 23 | 1.69 |
| **PECompact** | 7 | 0.44 |
| **PE_Patch.UltraProtect** | 7 | 0.44 |
| **ARM** | 1 | 0.00 |
| **FSG** | 1 | 0.00 |
| **StubbieMan** | 1 | 0.00 |
| **Assinaturas com mais de uma resposta** | 70 | 4.47 |
| **Subtotal** | 1484 | 94.82 |
| **Sem assinaturas conhecidas** | 50 | 3.19 |
| **Falso positivo (diferente do Kaspersky)** | 31 | 1.98 |
| **Total** | 1565 | 100 |

# Resultados (Kaspersky)

| *Compactador* | *Ocorrências* | *%* |
|---|---|---|
| **PECompact** | 763 | 21.61 |
| **PE_Patch.PECompact** | 753 | 21.33 |
| **PecBundle** | 753 | 21.33 |
| **UPX** | 295 | 8.35 |
| **Petite** | 244 | 6.91 |
| **Upack** | 179 | 5.07 |
| **ASPack** | 165 | 4.67 |
| **ZIP** | 88 | 2.49 |
| **RAR** | 81 | 2.29 |
| **PE_Patch** | 77 | 2.18 |
| **Ezip** | 21 | 0.59 |
| **UltraProtect** | 20 | 0.57 |
| **PE_Patch.UltraProtect** | 20 | 0.57 |
| **Expressor** | 10 | 0.28 |
| **NSPack** | 8 | 0.23 |
| **TeLock** | 8 | 0.23 |
| **ASProtect** | 6 | 0.17 |
| **YodaProt** | 4 | 0.11 |
| **PE_Patch.Morphine** | 4 | 0.11 |

| *Compactador* | *Ocorrências* | *%* |
|---|---|---|
| **Morphine** | 4 | 0.11 |
| **FSG** | 3 | 0.08 |
| **PE_Patch.Upolyx** | 3 | 0.08 |
| **Embedded EXE** | 2 | 0.06 |
| **MewBundle** | 2 | 0.06 |
| **PE-Crypt.XorPE** | 2 | 0.06 |
| **CAB** | 2 | 0.06 |
| **SoftComp** | 2 | 0.06 |
| **StubbieMan** | 2 | 0.06 |
| **MEW** | 2 | 0.06 |
| **Yoda** | 2 | 0.06 |
| **PEBundle** | 1 | 0.03 |
| **Embedded** | 1 | 0.03 |
| **PKLite32** | 1 | 0.03 |
| **SpisSFX** | 1 | 0.03 |
| **WinKript** | 1 | 0.03 |
| **Exe32Pack** | 1 | 0.03 |
| **Total** | 3531 | 100 |

# Resultados (Kaspersky)

Trojan-Spy.Win32.Bancos.ha
Trojan-Spy.Win32.Banker.ahy
Trojan-Spy.Win32.Bancos.u
Trojan-Spy.Win32.Banker.add
Trojan-Spy.Win32.Banker.abg
Trojan-Spy.Win32.Banker.aho
Trojan-Downloader.Win32.Delf.qz
Trojan-Spy.Win32.Banker.cv
Trojan-Downloader.Win32.VB.ji
Trojan-Spy.Win32.Banker.akc

# Exemplo: File 4.16

```
0               string   MZ
>0x18          leshort  >0x3f
>>(0x3c.l)     string   PE\e0\e0     PE executable (MS-Windows)
# search for the PE section called ".idata"...
>>>&0xf4       search/0x140 .idata
# ...and go to the end of it, calculated from start+length;
# these are located 14 and 10 bytes after the section name
>>>>(&0xe.l+(-4)) string PK\e3\e4  \eb, ZIP self-extracting archive
```

*magic* default: regras para PECompact, UPX e ZIP SFX

# Conclusão

- Desenvolvido um conjunto de assinaturas para o file
- Utilização em conjunto com outros programas

- Problemas:
  - Entendimento dos diversos formatos
  - Descompactadores
    - Ferramentas “hacker”
    - Windows

  - Kaspersky não disponibiliza os arquivos intermediários

# Trabalhos Futuros

- Novas assinaturas & refinamento
- File 4.16+

- Motivação para desenvolvimento de novas ferramentas (em Unix)
- Descompactador genérico – emulação

# Referências

- Analise estática de programas
- Tutorial GTS 01.2005
- Black Hat Conferences
- Virus Bulletin
- Fabricantes de antivírus (Kaspersky, Ewido)
- http://handlers.dshield.org/pbueno/

# Perguntas ?

Mais detalhes:

André Gerhard (agerhard@usp.br)
Aritana Pinheiro Falconi (falconi@cert.br)