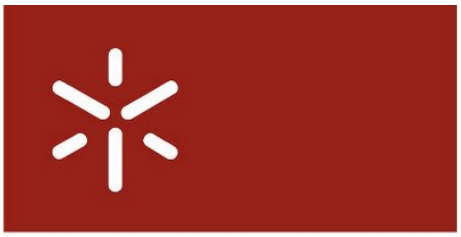
Universidade do Minho

Engenharia de Sistemas de Computação

Mestrado Integrado em Engenharia Informática



Universidade do Minho

Bernardo Silva - a77230

Francisco Lira - a73909

TP2

DTrace - Desenvolvimento de Programas

Conteúdo

1 Ex 1. Open Tracer

A chave para a implementação do Open Tracer baseia-se à volta de dois operadores:

- **?** - Representando o operador condicional devido às independências entre certas flags.
- **&** - Bitwise AND que permite a verificação das flags sendo que sabemos o bit correspondente a cada uma.

```
syscall::openat:entry
{
self->path = copyinstr(arg1);
self->flags = arg2;
}
```

Como se pode observar, no probe de entrada para a função openat, guarda-se o path e as flags, as quais se encontram no terceiro argumento da função, de forma a serem utilizadas quando o probe de retorno é ativado.

```
syscall::openat:return
{
this->return_out = arg0 == -1 ? "UNSUCCESSFUL" : "SUCCESSFUL";

this->flags_out =
    strjoin(self->flags & O_WRONLY ? "O_WRONLY"
                        : self->flags & O_RDWR ? "O_RDWR"
                                               : "O_RDONLY",
            strjoin(self->flags & O_APPEND ? "|O_APPEND"
                                           : "",
                    self->flags & O_CREAT ? "|O_CREAT"
                                          : ""));

    printf("%6d %6d %6d %15s %20s %70s\n",
           pid, uid, gid, this->return_out,
           this->flags_out, self->path);
}
```

No probe de retorno, em primeiro lugar testa-se se o comando foi executado com sucesso ou não. De seguida testa-se a existência ou não das flags com bitwise and, e condicionais, sendo que a existência de certas flags é dependente da inexistência de outras, como por exemplo O_WRONLY, O_RDWR e O_RDONLY. Por outro lado O_APPEND e O_CREAT são completamente independentes das outras flags.

No final de cada probe de retorno imprime-se no ecrã o **pid** (Process ID), **uid** (User ID), **gid** (Group ID), **return_out** (Sucesso ou insucesso da execução), **flags_out** (String correspondente ao estado das flags) e **path** (Path correspondente ao ficheiro que está a ser aberto)

É de notar que se as variáveis path e flags têm um prefixo ->self de forma a manter essas vaiáveis no mesmo thread e não só no probe em si.

2 Ex 2. Process Open Tracer

Utilizando algum do funcionamento do programa anterior, agora queremos obter, de tempo em tempo, informação acerca do process id e do nome de comando, particularmente o número de vezes que o um processo tentou abrir, criar e abrir com sucesso um certo ficheiro e retornar estas estatísticas.

Para este efeito utilizaram-se agregações de forma a registar eficientemente e de fácil acesso a informação que desejamos.

```
syscall::openat:entry
{
    self->flags = arg2;
    @opens[pid,execname] = count();
}
```

No probe de entrada guarda-se então a informação acerca das flags para ser utilizada mais tarde. Também se guarda numa agregação o número de opens realizados utilizando como chave tanto o pid (Process ID) e o execname (Nome do programa).

```
syscall::openat:return
{
    this->create = self->flags & O_CREAT ? 1 : 0;
    @creates[pid,execname] = sum(this->create);

    this->successful = arg0 == -1 ? 0 : 1;
    @successful[pid,execname] = sum(this->successful);
}
```

No probe de retorno testa-se tanto se a flag O_CREAT se encontra ativa como se o programa executou corretamente ou não atribuindo-lhes um valor de 0 ou 1. Seguidamente somam-se os resultados obtidos em mais duas agregações, de forma a serem utilizadas no fim do programa.

```
tick-$1sec
{
    printf("%-20Y\n",walltimestamp);
```

```
    printa("%6d %30s %@6d %@6d %@12d\n",
            @opens,@creates,@successful);
}
```

Por último, a cada 1 segundo é impressa a timestamp corrente e os dados referentes aos opens, crates e successful referente a cada processo e nome do programa.

3 Ex 3. Custom Truss

Para a construção de um script que permita um funcionamento semelhante a um comando truss utilizaram-se as seguintes probes:

```
syscall:::entry
/execname == $$1/
{
    @num[probefunc] = count();
    self->start_time = timestamp;
}
```

Na probe de entrada utiliza-se como condição o execname ser igual ao primeiro argumento do script, garantindo assim apenas system calls provenientes do mesmo.

De seguida, guarda-se numa agregação, utilizando como chave a função que foi chamada, o número de vezes que a mesma foi chamada, e também guardando o timestamp do começo dessa função em particular.

```
syscall:::return
/execname == $$1 && self->start_time != 0/
{
    @time[probefunc] = sum(timestamp-self->start_time);
}
```

Para a saída do probe além de se testar o execname, também se tem de garantir que o tempo de início é diferente de 0, ou seja a probe não foi chamada antes de o script iniciar.

Desta maneira agrega-se o tempo despendido na execução da função, de forma a somar todo o tempo gasto por esta função em particular.

```
dtrace:::END
{
    printa("%-20s %@10d %@10d\n",@num,@time);
}
```

Por fim imprime-se no ecrã os resultados correspondentes às agregações realizadas anteriormente.

4 Exemplos de Output

4.1 Exercício 1

Como não tinha acesso no ambiente Solaris à pasta /tmp criei uma pasta tmp no meu diretório de utilizador.

> cat /etc/inittab > tmp/test

```
  PID    UID    GID          RETURN                FLAGS
  PATH
24891   1010   5000      SUCCESSFUL     O_WRONLY|O_CREAT
tmp/test
24891   1010   5000    UNSUCCESSFUL             O_RDONLY
/var/ld/64/ld.config
24891   1010   5000      SUCCESSFUL             O_RDONLY
/lib/64/libc.so.1
24891   1010   5000      SUCCESSFUL             O_RDONLY
/usr/lib/locale/en_US.UTF-8/en_US.UTF-8
24891   1010   5000      SUCCESSFUL             O_RDONLY
/usr/lib/locale/common/amd64/methods_unicode.so.3
24891   1010   5000    UNSUCCESSFUL             O_RDONLY
/usr/lib/locale/pt_PT.UTF-8/pt_PT.UTF-8
24891   1010   5000    UNSUCCESSFUL             O_RDONLY
/usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_linkers.mo
24891   1010   5000    UNSUCCESSFUL             O_RDONLY
/usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_lib_libc.mo
24891   1010   5000      SUCCESSFUL             O_RDONLY
/etc/inittab
```

> cat /etc/inittab >> tmp/test

```
  PID    UID    GID          RETURN                FLAGS
  PATH
24947   1010   5000      SUCCESSFUL O_WRONLY|O_APPEND|O_CREAT
tmp/test
24947   1010   5000    UNSUCCESSFUL             O_RDONLY
/var/ld/64/ld.config
24947   1010   5000      SUCCESSFUL             O_RDONLY
/lib/64/libc.so.1
24947   1010   5000      SUCCESSFUL             O_RDONLY
/usr/lib/locale/en_US.UTF-8/en_US.UTF-8
24947   1010   5000      SUCCESSFUL             O_RDONLY
/usr/lib/locale/common/amd64/methods_unicode.so.3
24947   1010   5000    UNSUCCESSFUL             O_RDONLY
/usr/lib/locale/pt_PT.UTF-8/pt_PT.UTF-8
```

```
 24947   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_linkers.mo
 24947   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_lib_libc.mo
```

> cat /etc/inittab | tee tmp/test

```
  PID    UID    GID          RETURN                FLAGS
  PATH
24986   1010   5000    UNSUCCESSFUL             O_RDONLY
/var/ld/64/ld.config
 24986   1010   5000      SUCCESSFUL             O_RDONLY
 /lib/64/libc.so.1
 24986   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/en_US.UTF-8
 24986   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/common/amd64/methods_unicode.so.3
 24986   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/pt_PT.UTF-8/pt_PT.UTF-8
 24986   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_linkers.mo
 24986   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_lib_libc.mo
 24986   1010   5000      SUCCESSFUL             O_RDONLY
 /etc/inittab
 24987   1010   5000    UNSUCCESSFUL             O_RDONLY
 /var/ld/64/ld.config
 24987   1010   5000      SUCCESSFUL             O_RDONLY
 /lib/64/libc.so.1
 24987   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/en_US.UTF-8
 24987   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/common/amd64/methods_unicode.so.3
 24987   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/pt_PT.UTF-8/pt_PT.UTF-8
 24987   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_linkers.mo
 24987   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_lib_libc.mo
 24987   1010   5000      SUCCESSFUL     O_WRONLY|O_CREAT
 tmp/test
```

> cat /etc/inittab | tee -a tmp/test

```
  PID    UID    GID          RETURN                FLAGS
```

```
   PATH
25023   1010   5000    UNSUCCESSFUL             O_RDONLY
/var/ld/64/ld.config
 25023   1010   5000      SUCCESSFUL             O_RDONLY
 /lib/64/libc.so.1
 25023   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/en_US.UTF-8
 25023   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/common/amd64/methods_unicode.so.3
 25023   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/pt_PT.UTF-8/pt_PT.UTF-8
 25023   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_linkers.mo
 25023   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_lib_libc.mo
 25023   1010   5000      SUCCESSFUL O_WRONLY|O_APPEND|O_CREAT
 tmp/test
 25022   1010   5000    UNSUCCESSFUL             O_RDONLY
 /var/ld/64/ld.config
 25022   1010   5000      SUCCESSFUL             O_RDONLY
 /lib/64/libc.so.1
 25022   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/en_US.UTF-8
 25022   1010   5000      SUCCESSFUL             O_RDONLY
 /usr/lib/locale/common/amd64/methods_unicode.so.3
 25022   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/pt_PT.UTF-8/pt_PT.UTF-8
 25022   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_linkers.mo
 25022   1010   5000    UNSUCCESSFUL             O_RDONLY
 /usr/lib/locale/en_US.UTF-8/LC_MESSAGES/solaris_lib_libc.mo
 25022   1010   5000      SUCCESSFUL             O_RDONLY
 /etc/inittab
```

4.2 Exercício 2

```
a77230@solaris:~/TP2$ ./openattrace2.d
##########################################
#              Openat Tracer             #
##########################################
   PID                           CMD    OPEN CREATE  SUCCESSFUL
2020 Jun  9 13:07:10
   911                       sstored      46     28          46
2020 Jun  9 13:07:11
   804                           fmd       1      0           0
 24405                 openattrace2.d       2      0           2
   911                       sstored      62     38          62
2020 Jun  9 13:07:12
   804                           fmd       1      0           0
 24405                 openattrace2.d       2      0           2
   911                       sstored      74     44          74
2020 Jun  9 13:07:13
   804                           fmd       1      0           0
 24405                 openattrace2.d       2      0           2
   911                       sstored      94     54          94
2020 Jun  9 13:07:14
   804                           fmd       1      0           0
 24405                 openattrace2.d       2      0           2
   911                       sstored      94     54          94
```

4.3 Exercício 3

```
a77230@solaris:~/TP2$ ./customstrace.d ls
##############################################
#               Custom STrace                #
##############################################
System Call                    #   Time(ns)
^C
rexit                          1          0
getpid                         1       3550
sysconfig                      1       3853
lwp_private                    1       4598
sigpending                     1       5946
getrlimit                      1       6341
systeminfo                     1      14805
write                          1      33168
setcontext                     2      15489
getdents                       2      45198
mmapobj                        2     146882
ioctl                          3      33596
mmap                           3      37978
close                          4      21147
```

```
resolvepath                     4     132887
brk                             5      33466
memcntl                         5      85453
fstatat                         7      90773
openat                          8     261661
```

5 Possíveis melhorias

No primeiro exercício obtem-se um problema por vezes em que o probe não consegue identificar o argumento da função correspondente às flags e retorna um erro.