Centro de
Investigación en
Computación

System Call (syscall)

Llamadas al Sistema

API del SO

Course

Operating System (with focus on Security)

Instructor

Acosta Bermejo Raúl et al.

Lecture notes

Instituto
Politécnico
Nacional

Table of contents (outline)

Tabla de contenido

1. Introducción
2. Linux
    1. Lista de system call
3. Mac OS X
4. Windows
5. Revision de algunas llamadas

Introduction

POSIX

POSIX compliant system calls.

- 1003.1 and 1003.1b

System Call / Linux

Llamadas al sistema en Linux

System Call

Llamadas al sistema

- The implementation of system calls in Linux/Windows is varied based on the architecture (like version and 32/64bits), but it can also differ within a given architecture.
- For example, older x86 processors used an **interrupt** mechanism to migrate from user-space to kernel-space, but new IA-32 processors provide instructions that optimize this transition (using sysenter and sysexit instructions).
- Because so many options exist and the end-result is so complicated.

System Call

Llamadas al sistema

x86 interrupt mechanism

- int 0x80 is the **assembly language instruction** that is used to invoke system calls in Linux on x86 (i.e., Intel-compatible) processors.
- A system call is a request in a Unix-like operating system made via a **software interrupt** by an active process for a service performed by the kernel, such as input/output or process creation.
- An interrupt is a signal to the operating system that an event has occurred, and it results in changes in the sequence of instructions executed by the CPU.
- There are numerous **sources of interrupts**, including pressing a key on keyboard, moving the mouse, timers, disk drives, signals originating elsewhere on the network and the loss of electrical power.
- A software interrupt is an interrupt that originates in software, usually by a process in user mode.

System Call

Llamadas al sistema

x86 interrupt mechanism

When a system call is made, you:

1. Store in the process register of the system call number (i.e., the integer assigned to each system call) for that system call,
2. Store any arguments (i.e., input data) for it, and
3. Call the int 0x80 instruction (OS dependant interrupt).

Links

- https://en.wikibooks.org/wiki/X86_Assembly/Interfacing_with_Linux

System Call

Llamadas al sistema

Por default la lista de las llamadas al sistema esta en:

/boot/System.map.###

donde ### es la versión del kernel. También se encuentra una versión “viva” en:

/proc/kallsyms

Adicionalmente las llamadas al sistema estan definidas en el código fuente del kernel lo cual suele estar en:

Carpeta	/usr/include
Programación	<sys/syscall.h>

System Call

Llamadas al sistema

El código de las llamadas al sistema esta dentro de la carpeta del código fuente, el cual suele estar en:

/usr/src/kernels/
/usr/src/Linux-sources-version

System Call

Llamadas al sistema

System call table

- It is defined in Linux kernel source file in diferentes places selon the kernels's version:
  - OLD: "arch/i386/kernel/entry.S ".
  - Kernel 3.0: arch/x86/kernel/syscall_table_32.c
  - Kernel 4.2: arch/x86/syscalls/syscall_64.tbl
  - Kernel 4.10: arch/x86/entry/syscalls/syscall_64.tbl

Actually the syscall table is defined in many files.
For looking for the files you can use:

$ grep -r SYSCALL_DEFINE *

System Call

Llamadas al sistema

Para consultar el código del kernel de Linux en sus diferentes versiones existen varias referencias:

- Linux Cross References
  - http://lxr.free-electrons.com/

Algunas referencias para consultar sólo las Llamadas al sistema son:

- http://syscalls.kernelgrok.com/. Linux con 337 syscall (11/ago/2016).
- http://opensource.apple.com//source/xnu/xnu-1504.3.12/bsd/kern/syscalls.master. MacOSX 429 syscall (11/ago/2016).

Algunas referencias que explican el funcionamiento:

- http://articles.manugarg.com/systemcallinlinux2_6.html
- https://www.ibm.com/developerworks/linux/library/l-system-calls/

System Call (list)

Llamadas al sistema (lista)

1 exit
2 fork
3 read
4 write
5 open
6 close
7 waitpid
8 creat
9 link
10 unlink
11 execve
12 chdir
13 time
14 mknod
15 chmod
16 lchown
17 stat
18 lseek
19 getpid
20 mount
21 umount
22 setuid
23 getuid
24 stime
25 ptrace
26 alarm
27 fstat
28 pause
29 utime
30 access
31 nice
32 sync
33 kill
34 rename
35 mkdir
36 rmdir
37 dup
38 pipe
39 times
40 brk
41 setgid
42 getgid
43 sys_signal
44 geteuid
45 acct
46 umount2
47 ioctl
48 fcntl
49 mpx
50 setpgid
51 ulimit
52 olduname
53 umask
54 chroot
55 ustat
56 dup2
57 getppid
58 getpgrp
59 setsid
60 sigaction
61 sgetmask
62 ssetmask
63 setreuid
64 setregid
65 sigsuspend
66 sigpending
67 sethostname
68 setrlimit
69 getrlimit
70 getrusage
71 gettimeofday
72 settimeofday
73 getgroups
74 setgroups
75 old_select
76 symlink
77 lstat
78 readlink
79 uselib
80 swapon
81 reboot
82 old_readdir
83 old_mmap
84 munmap
85 truncate
86 ftruncate
87 fchmod
88 fchown
89 getpriority
90 setpriority
91 profil
92 statfs
93 fstatfs
94 ioperm
95 socketcall
96 syslog
97 setitimer
98 getitimer
99 sys_newstat
100 sys_newlstat
101 sys_newfstat
102 olduname
103 iopl
104 vhangup
105 idle
106 vm86old
107 wait4
108 swapoff
109 sysinfo
110 ipc
111 fsync
112 sigreturn
113 clone
114 setdomainname
115 uname
116 modify_ldt
117 adjtimex
118 mprotect
119 sigprocmask
120 create_module
121 init_module
122 delete_module
123 personality
124 afs_syscall
125 setfsuid
126 setfsgid
127 sys_llseek
128 getdents
129 select
130 flock
131 msync
132 readv
133 writev
134 sys_getsid
135 fdatasync
136 sysctl
137 mlock
138 munlock
139 mlockall
140 munlockall
141 sched_setparam
142 sched_getparam
143 sched_setscheduler
144 sched_getscheduler
145 sched_yield
146 sched_get_priority_max
147 sched_get_priority_min
148 sched_rr_get_interval
149 nanosleep
150 mremap
151 setresuid
152 getresuid
153 vm86
154 query_module
155 poll
156 nfsservctl
157 setresgid
158 getresgid
159 prctl
160 rt_sigreturn
161 rt_sigaction
162 rt_sigprocmask
163 rt_sigpending
164 rt_sigtimedwait
165 rt_sigqueueinfo
166 rt_sigsuspend
167 pread
168 sys_pwrite
169 chown
170 getcwd
171 capget
172 capset
173 sigaltstack
174 sendfile
175 getpmsg
176 putpmsg
177 vfork

System Call (list)

Llamadas al sistema (lista)

Files

creat()	unlink()
open()	stat()
close()	fstat()
read()	access()
write()	chmod()
lseek()	chown()
dup()	umask()
link()	**ioctl**()

TODO en sistemas like-UNIX son archivos, en particular los periféricos:

- Permite a una aplicación controlar o comunicarse con un **driver de dispositivo**, fuera de los usuales read/write de datos.
- El kernel generalmente envía un ioctl directamente al driver, el cual puede interpretar el número de requerimiento (request code) y datos en cualquier forma requerida por ejemplo:
  - El manejo de los leds del teclado.
  - Instruir a un dispositivo CDROM de eyectar un disco.

System Call (list)

Llamadas al sistema (lista)

Procesos

fork()
getpid(), getppid()
exit()
wait()
pause()
signal()
kill()
pipe()
times()

Creación de nuevos procesos
Exec family
exec(), execl(), execv(), execle(), …, execvp()

Se suele terminar llamando a otra syscall más básica:
sys_execve

System Call

Llamadas al sistema

Comando strace

Para ver las llamadas al sistema que un programa realiza se utiliza principalmente el comando strace aunque existen otras utilerías también.

$ strace archivo-ELF-ejecutable
$ strace /bin/ls

Opciones

$ strace –e open /bin/ls # muestra sólo los syscall open realizados
$ strace –p 1543 # muestra los syscal de un proceso que se está ejecutando
$ strace –c ls # muestra las estadísticas de syscal realizadas

Linux

Inspección del kernel

There are two forms:

1. Using commands (CLI)
2. Reading virtual files
   - Some particular directories.

Linux

Inspección del kernel

El kernel de Linux ha ido evolucionando y actualmente hay varias formas de **ver** (leer y/o modificar) que se tiene. Las principales son:

- Las **carpetas** y sus respectivos archivos:
  - /proc

- Usando las llamadas al sistema (API del SO):
  - Crear un programa específico.
  - Utilizar los comandos específicos de inspección.

Existen varios casos donde la relación carpeta/archivo vs comando es directa, por ejemplo:

- El comando ps lee la carpeta /proc/numero

Linux

Inspección del kernel

La carpeta **/proc**

- /proc/1: Un directorio con información acerca del proceso número 1. Cada proceso tiene un directorio debajo de /proc cuyo nombre es el número de identificación del proceso (PID).
- /proc/cpuinfo: Información acerca del procesador: su tipo, marca, modelo, rendimiento, etc.
- /proc/devices: Lista de controladores de dispositivos configurados dentro del núcleo que está en ejecución.
- /proc/dma: Muestra los canales DMA que están siendo utilizados.
- /proc/filesystems: Lista los sistemas de archivos que están soportados por el kernel.
- /proc/interrupts: Muestra la interrupciones que están siendo utilizadas, y cuantas de cada tipo ha habido.
- /proc/ioports: Información de los puertos de E/S que se estén utilizando en cada momento.
- /proc/kcore: Es una imagen de la memoria física del sistema. Este archivo tiene exactamente el mismo tamaño que la memoria física, pero no existe en memoria como el resto de los archivos bajo /proc, sino que se genera en el momento en que un programa accede a este. (Recuerde: a menos que copie este archivo en otro lugar, nada bajo /proc usa espacio en disco).
- /proc/kmsg: Salida de los mensajes emitidos por el kernel. Estos también son redirigidos hacia syslog.

Linux

Inspección del kernel

La carpeta **/proc**

- /proc/ksyms: Tabla de símbolos para el kernel.
- /proc/loadavg: El nivel medio de carga del sistema; tres indicadores significativos sobre la carga de trabajo del sistema en cada momento.
- /proc/meminfo: Información acerca de la utilización de la memoria física y del archivo de intercambio.
- /proc/modules: Indica los módulos del núcleo que han sido cargados hasta el momento.
- /proc/net: Información acerca del estado de los protocolos de red.
- /proc/self: Un enlace simbólico al directorio de proceso del programa que esté observando a /proc. Cuando dos procesos observan a /proc, obtienen diferentes enlaces. Esto es principalmente una conveniencia para que sea fácil para los programas acceder a su directorio de procesos.
- /proc/stat: Varias estadísticas acerca del sistema, tales como el número de fallos de página que han tenido lugar desde el arranque del sistema.
- /proc/uptime: Indica el tiempo en segundos que el sistema lleva funcionando.
- /proc/version: Indica la versión del núcleo.

Linux

Inspección del kernel

La carpeta **/sys**

Estructura básica:

- block
- bus
- class
- dev
- devices
- Firmware
- Fs
- hypervisor
- **kernel**
- module
- power

System Call / Mac OS X

Llamadas al Sistema

System Call / Mac OS X

Llamadas al sistema

Links

- Mac OS X internals
  - http://dustin.schultz.io/blog/2010/11/15/mac-os-x-64-bit-assembly-system-calls/.

System Call / Mac OS X

Llamadas al sistema

Comando dtruss

- Para ver las llamadas al sistema que un programa realiza en Mac OS X se han creado varios comandos y algunos ya son obsoletos: dtrace, ktrace, dtruss, etc.
- Dtrace fue desarrollado originalmente por Sun Solaris.

Ejemplos y opciones
$ sudo dtruss command
$ sudo dtruss –c command
Imprime al final la cuenta de las syscall

System Call / Windows

Llamadas al Sistema

System Call / Windows

Llamadas al sistema

Links

- https://en.wikipedia.org/wiki/Native_API
- https://github.com/hfiref0x/SyscallTables
- http://j00ru.vexillium.org/syscalls/win32k/32/
- http://j00ru.vexillium.org/syscalls/win32k/64/

System Call / Windows

Llamadas al sistema

There are several options to monitor syscall:

- Windows performance toolkit and Debugging Tools
  - Process monitor from SysInternals
  - Windows API tracing tools (logexts.dll)
- API Monitor from rohitab.
- FileMon, NtTrace
- Commercial software
  - Deviare API Hook $ and Open Source
    - http://www.nektra.com/products/deviare-api-hook-windows/
  - API Monitor
    - http://www.rohitab.com/apimonitor
  - SpyStudio $

System Call / Xen

Llamadas al Sistema

System Call / Xen

Llamadas al sistema

Xen tiene aproximadamente **55** syscall:

1. Manejo de procesos
2. Manejo de memoria
3. Administración: sysctl

Links

- http://xenbits.xen.org/docs/unstable/hypercall/x86_64/include,public,xen.h.html#incontents_hcalls

Libraries

Librerías

libc

Dll

Libraries

Librerías

- Don’t confuse system calls with libc calls.
- Differences?

Is printf() a system call?
Is rand() a system call?

Algunas son muy cercanas:

- open vs fopen
- close vs fclose
- read vs fread, getchar, scanf, fscanf, …, fgets
- write vs fwrite, putchar, ..., fputs
- lseek vs fseek

Libraries

Librerías

- libc
- glibc

Commands

Comandos

Commands

Comandos

Los más usados son:

- strace y sus variantes como ltrace. En Solaris y FreeBSD hay equivalentes (truss).
- Otros son: perf, sysdig.

Strace

Comando

Descripción

Lista las llamadas al sistema ejecutadas por un comando.

Tiene opciones para desplegar la cantidad y el tiempo de las llamadas.

Linux

$ strace –c ls # Lista la frecuencia y el tiempo de las llamadas alsistema

Mac OSX (equivalente a strace)

$ sudo dtruss ls

Review of some Syscall

Revisión de algunas Llamadas al Sistema

ioctl

Control de drivers

Descripción

- Su nombre abrevia la frase **input/output control**.
- ioctl es una llamada de sistema en Unix que permite a una aplicación controlar o comunicarse con un driver de dispositivo, fuera de los usuales read/write de datos.
- Esta llamada se originó en la versión 7 del AT&T Unix.

- Imagine que se tiene un puerto serial conectado a un modem:
  - Lo natural es usar el device file para escribir/leer datos al/del modem.
  - Sin embargo, esto deja abierta la pregunta de que hacer cuando se necesita comunicarse con el puerto serial en si mismo?, por ejemplo para enviar la tasa de velocidad a la cual se quiere enviar/recibir los datos.
  - La respuesta en Unix es usar el syscal especial ioctl. Cada dispositivo puede tener sus propios comandos ioctl.

ioctl

Control de drivers

Prototype

- The **ioctl** syscall takes as parameters:
  - An open file descriptor.
  - A request code number.
  - Either an integer value or a pointer to data.
- The kernel generally dispatches an **ioctl** call straight to the device driver, which can interpret the request number and data in whatever way required.
- The writers of each driver:
  - Document request numbers for that particular driver, and
  - Provide them as constants in the header file (*.h).

Profilers

Depuradores

Software

Comandos

Los programas que se pueden utilizar son:

- oprofile
- Perf
- sysdig

Links

http://oprofile.sourceforge.net/

Application Binary Interface (ABI)

Interfaz Binaria de Aplicaciones

ABI

Introducción

It is compound of:

- Calling conventions
  - Stack management
- Type representations or data representation
- Name mangling

La primera versión fue desarrollada para un sistema Unix tipo System V (SVR4) por lo que el ABI se llama así y Linux lo usa.

ABI

Convenciones de invocaciones (funciones)

Links

- https://en.wikipedia.org/wiki/Calling_convention
- http://www.sco.com/developers/devspecs/abi386-4.pdf

Linux

- https://refspecs.linuxfoundation.org/

Intel

- https://en.wikipedia.org/wiki/X86_calling_conventions
- https://www.uclibc.org/docs/psABI-i386.pdf

MacOSX

- https://developer.apple.com/library/mac/documentation/DeveloperTools/Conceptual/LowLevelABI/000-Introduction/introduction.html

ARM

- http://infocenter.arm.com/help/topic/com.arm.doc.ihi0042f/IHI0042F_aapcs.pdf

Attacks on Syscall

Ataques en Syscall

Attacks on Syscall

Ataques en Syscall

- Using raw system calls instead of normal high level API is obviously unusual and unnecessarily uncomfortable,
- Except if you want to evade some monitoring/hooking software and obtain a basic code obfuscation, in that case you won't use any API but just set up some registers and then calling sysenter.
- The problem here is:
  - On every Windows version, and among each service pack too syscall numbers may vary, and
  - There's no *official* way to obtain the correct values for the system your application is running on.
  - Just the official compiler knows the numbers.

Varios comandos

Depuradores

Varios comandos

- Lstrace
- Lsof
- Auditctl
- ausearch

The end

Contacto

Raúl Acosta Bermejo

http:www.cic.ipn.mx
http://www.ciseg.cic.ipn.mx/

racostab@ipn.mx
racosta@cic.ipn.mx

57-29-60-00
Ext. 56652

Seguimiento a la planeación

Actividades

1. Lectura obligatoria:
   - Quien nos platica el de Tanenbaum?
   - Y de la historia/evolución de SO
2. Lectura de artículos mensuales.
   - Cuales (x persona)?
   - El correo? leyeron enero? Lo dejamos para febrero?
3. .