IKO31204
Pemrograman Sistem
Jilid 7: Proc File System

Fakultas Ilmu Komputer - Universitas Indonesia
Sep 2012

topik

proc file system

referensi

Explore Linux /proc File System
http://www.thegeekstuff.com/2010/11/linux-proc-file-system/

The C10K problem
http://www.kegel.com/c10k.html

Linux Kernel Procfs Guide
www.stillhq.com/pdfdb/000445/data.pdf

referensi

Linux Kernel Module Programming Guide (Chapter 5)
http://www.tldp.org/LDP/lkmpg/2.6/html/lkmpg.html

Kernel Documentation Source Code
<src kernel>/Documentation/proc.txt

Procfs Analysis
http://www.nsa.gov/research/_files/selinux/papers/slinux/node57.shtml

Apa itu procfs ?

Proc Filesystem adalah filesystem khusus di UNIX yang menyajikan informasi tentang proses dan sistem informasi lainnya dalam struktur file secara hirarkis

Apa guna procfs ?

Menyediakan metode yang lebih nyaman dan standar untuk secara dinamis mengakses data proses yang diselenggarakan di kernel dibandingkan dengan metode akses langsung ke memori kernel

Seluruh aktifitas sistem terpetakan sebagai berkas pada /proc/

Procfs bagi administrator

- Viewing Statistical Information
- Finding out Hardware Information
- Modifying Runtime Parameters
- Viewing and Modifying Network and Host Parameters
- Memory and Performance Information
- etc

Procfs bagi administrator

- Viewing Statistical Information
- Finding out Hardware Information
- Modifying Runtime Parameters
- Viewing and Modifying Network and Host Parameters
- Memory and Performance Information
- etc

Peta Procfs

Lokasi Mount point: /proc/

/proc /proc proc rw 0 0

- Subdirektori berdasarkan angka:
  Mewakili setiap proses (PID) yang terjadi di dalam sistem

- Subdirektori tdk b'dasarkan angka:
  Menyediakan informasi sistem

Procfs berdirektori “PID”

Lokasi: /proc/{PID}/

cmdline – command line of the command.
environ – environment variables.
fd – Contains the file descriptors which is linked to the appropriate files.
limits – Contains the information about the specific limits to the process.
exe – Link to executable of the process.
root – Link to the root directory of the process.
dll

Cth: Melihat variabel program google-chrome dgn /proc

ps -aef | grep mozilla

```
root 32558 32425 8  22:53 pts/1  00:01:23  /usr/bin/mozilla
```

ls -al /proc/32558

```
total 0
-r--r--r--    1 root  root              0 Dec 25 22:59 cmdline
-r--r--r--    1 root  root              0 Dec 25 22:59 cpu
lrwxrwxrwx    1 root  root              0 Dec 25 22:59 cwd ->
/proc/
-r--------    1 root  root              0 Dec 25 22:59 environ
lrwxrwxrwx    1 root  root              0 Dec 25 22:59 exe ->
/usr/bin/mozilla*
dr-x------    2 root  root              0 Dec 25 22:59 fd/
-r--r--r--    1 root  root              0 Dec 25 22:59 maps
-rw-------    1 root  root              0 Dec 25 22:59 mem
-r--r--r--    1 root  root              0 Dec 25 22:59 mounts
lrwxrwxrwx    1 root  root              0 Dec 25 22:59 root -> //
-r--r--r--    1 root  root              0 Dec 25 22:59 stat
…..............
```

Procfs informasi sistem

Lokasi: /proc/

/proc/cpuinfo – information about CPU,
/proc/meminfo – information about memory,
/proc/loadvg – load average,
/proc/partitions – partition related info,
/proc/version – linux version
/proc/sys/ - subdir berisi editable kernel variables

dll

Berinteraksi dengan Kernel (I)

Lokasi: /proc/sys/

Sebagian besar file dalam /proc adalah **read-only,** sebagian kecil bisa **read-write**.

Menulis ke file ini dapat mengubah keadaan kernel dan harus dilakukan dengan hati-hati.

Karena sangat banyak file yang dapat dikonfigurasi, biasanya kegiatan konfigurasi ini disebut sebagai "**tweaking kernel params**"

Berinteraksi dengan Kernel (II)

Cth modif variabel pada: /proc/sys/kernel/

```
$ hostname
machinename.domainname.com

$ cat /proc/sys/kernel/domainname
domainname.com

$ cat /proc/sys/kernel/hostname
machinename

$ echo "new-machinename"  > /proc/sys/kernel/hostname

$ hostname
new-machinename.domainname.com
```

Berinteraksi dengan Kernel (III)

Cth modif variabel pada: /proc/sys/net/

This will hide your machine in the network as it disables answers to icmp_echos. The host will not respond to ping queries from other hosts.

```
# echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

# ping machinename.domainname.com
no answer from machinename.domainname.com
```

To turn it back to default behavior, do

```
# echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_all
```

Menyimpan hasil konfigurasi

Konfigurasi dapat disimpan pada berkas /etc/sysctl.conf

Cth:

```
# echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

SAMA dengan

# echo “net.ipv4.icmp_echo_ignore_all” > /etc/sysctl.conf
# sysctl -p /etc/sysctl.conf
```

Sysctl.conf selalu dipanggil oleh BOOT SCRIPT sehinga konfigurasi dapat di “restore” ulang saat awal booting kernel

Cara membuat procfs

1. include proc_fs.h

```
#include <linux/proc_fs.h>
```

2. membuat file proc

```
struct proc_dir_entry* create_proc_entry(const char* name,
mode_t mode, struct proc_dir_entry* parent);
```

3. mendaftarkan fungsi pelayan untuk kegiata read/write file proc (saat init module)

```
struct proc_dir_entry* entry;
entry->read_proc = read_proc_foo;
entry->write_proc = write_proc_foo;
```

Cth module helloworld

1. mendaftarkan /proc/helloworld
   saat init_module()
2. memproses panggilan
   terhadap /proc/helloworld
   a. memproses READ, atau
   b. memproses WRITE
3. men-delete file /proc/helloworld
   saat cleanup_module()

membuat file /proc/helloworld (1)

```
-------------------- START FILE helloworld.c -------------------------
#include <linux/module.h>          /* Specifically, a module */
#include <linux/kernel.h>          /* We're doing kernel work */
#include <linux/proc_fs.h>         /* Necessary because we use the proc fs */

#define procfs_name "helloworld"

struct proc_dir_entry *Our_Proc_File;

int procfile_read(char *buffer, char **buffer_location, off_t offset, int buffer_length, int *eof,
void *data)    {

        int ret;
        printk(KERN_INFO "procfile_read (/proc/%s) called\n", procfs_name);

        if (offset > 0) {
                /* we have finished to read, return 0 */
                ret = 0;
        } else {
                /* fill the buffer, return the buffer size */
                ret = sprintf(buffer, "HelloWorld!\n");
        }
        return ret;
}
```

membuat file /proc/helloworld (1)

```
int init_module()    {
        Our_Proc_File = create_proc_entry(procfs_name, 0644, NULL);
        if (Our_Proc_File == NULL) {
                remove_proc_entry(procfs_name, &proc_root);
                printk(KERN_ALERT "Error: Could not initialize /proc/%s\n", procfs_name);
                return –ENOMEM;
        }
        Our_Proc_File–>read_proc          = procfile_read;
        Our_Proc_File–>owner         = THIS_MODULE;
        Our_Proc_File–>mode          = S_IFREG  |  S_IRUGO;
        Our_Proc_File–>uid                = 0;
        Our_Proc_File–>gid                = 0;
        Our_Proc_File–>size               = 37;
        printk(KERN_INFO "/proc/%s created\n", procfs_name);
        return 0; /* everything is ok */
}

void cleanup_module()       {
        remove_proc_entry(procfs_name, &proc_root);
        printk(KERN_INFO "/proc/%s removed\n", procfs_name);
}

-------------------- END FILE helloworld.c -------------------------
```

Percobaan

mkdir helloworld
cd helloworld
vi helloworld.c
(copy paste seluruh helloworld.c di atas)
vi Makefile
(buat Makefile untuk meng-compile module di atas)
make
insmod helloworld.ko
dmesg
(liat apa yang terjadi)
ls -al /proc/helloworld
cat /proc/helloworld

NOTE: coba buat procfs yang dapat di write

tanya jawab