

Implementation of System Call in Minix.

Edit callnr.h and use the spot 69 to define MYCALL

```
minix (before syscall) [Running]
# cd /usr/src/include/minix/
# nano callnr.h _
```

```
minix (before syscall) [Running]
GNU nano 2.2.4      File: /usr/src/include/minix/callnr.h      Modified

#define IOCTL          54
#define FCNTL          55
#define FS_READY       57
#define EXEC           59
#define UMASK          60
#define CHROOT         61
#define SETSID         62
#define GETPGRP        63
#define ITIMER          64
#define GETGROUPS_0     65
#define SETGROUPS_0     66
#define GETMCONTEXT    67
#define SETMCONTEXT    68
#define MYCALL          69
/* Posix signal handling. */
#define SIGACTION       71
#define SIGSUSPEND      72
#define SIGPENDING       73
#define SIGPROCMASK     74
#define SIGRETURN       75

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Edit [/usr/src/servers/pm/table.c](#) to map a new function in this server; the function do_mycall()

```
minix (before syscall) [Running]
# nano /usr/src/servers/pm/table.c_
```

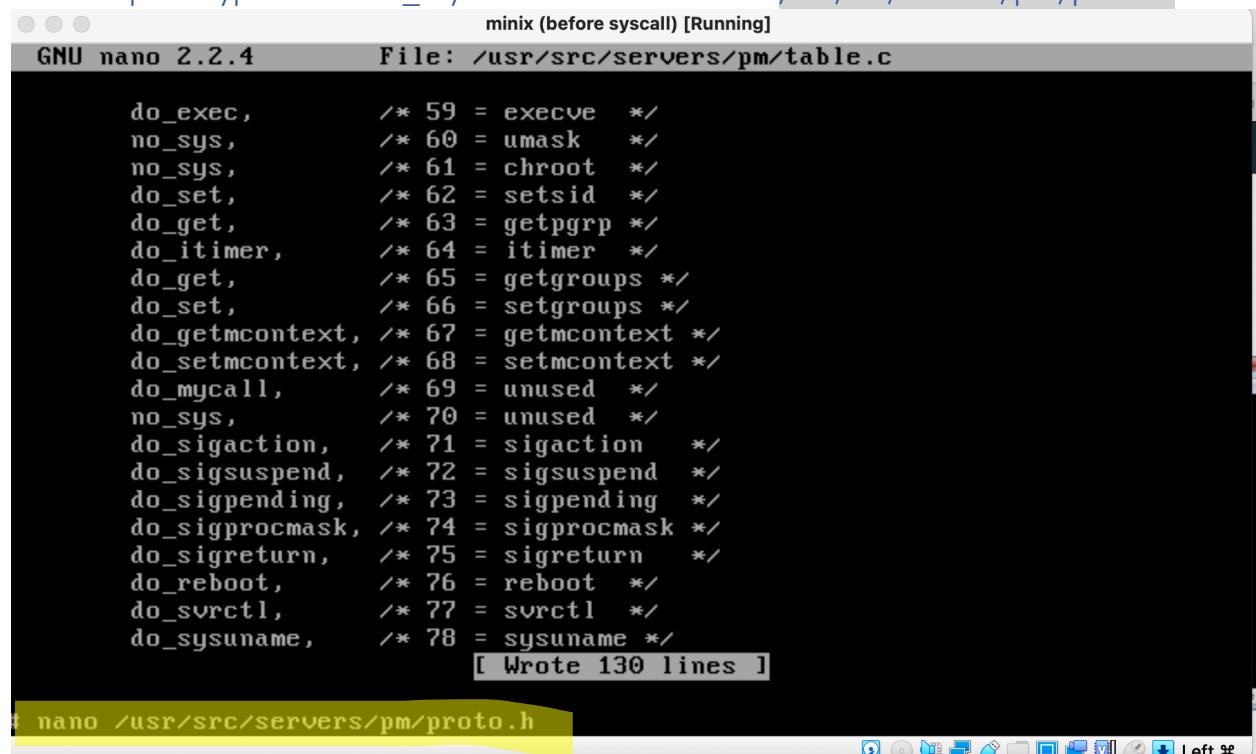
```
minix (before syscall) [Running]
GNU nano 2.2.4          File: /usr/src/servers/pm/table.c          Modified

do_exec,      /* 59 = execve */
no_sys,       /* 60 = umask */
no_sys,       /* 61 = chroot */
do_set,       /* 62 = setsid */
do_get,       /* 63 = getpgrp */
do_itimer,    /* 64 = itimer */
do_get,       /* 65 = getgroups */
do_set,       /* 66 = setgroups */
do_getmcontext, /* 67 = getmcontext */
do_setmcontext, /* 68 = setmcontext */
do_mycall,    /* 69 = unused */
no_sys,       /* 70 = unused */
do_sigaction, /* 71 = sigaction */
do_sigsuspend, /* 72 = sigsuspend */
do_sigpending, /* 73 = sigpending */
do_sigprocmask, /* 74 = sigprocmask */
do_sigreturn, /* 75 = sigreturn */
do_reboot,    /* 76 = reboot */
do_svrcctl,   /* 77 = svrcctl */
do_sysuname,  /* 78 = sysuname */

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^U Next Page  ^U UnCut Text ^T To Spell

```

Add the prototype of the do_mycall function in the file /usr/src/servers/pm/proto.h

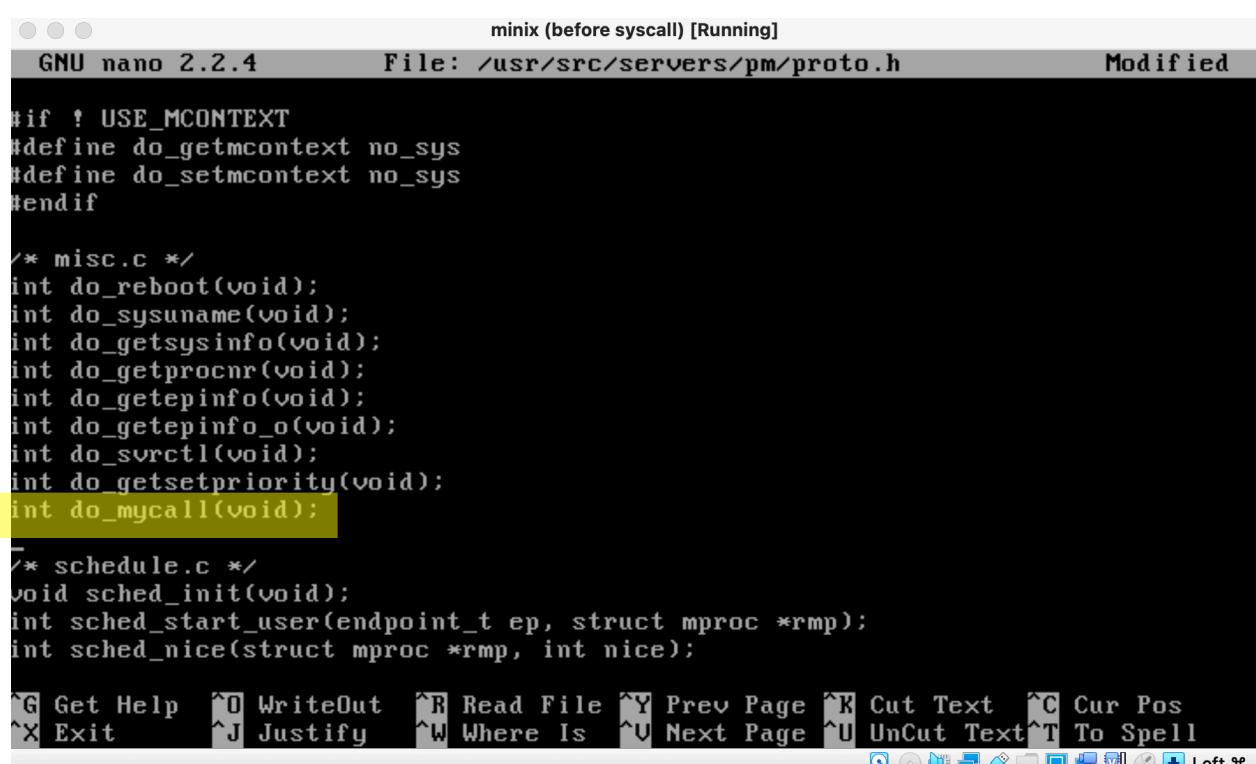


```
minix (before syscall) [Running]
GNU nano 2.2.4      File: /usr/src/servers/pm/table.c

do_exec,          /* 59 = execve */
no_sys,           /* 60 = umask */
no_sys,           /* 61 = chroot */
do_set,           /* 62 = setsid */
do_get,           /* 63 = getpgrp */
do_itimer,        /* 64 = itimer */
do_get,           /* 65 = getgroups */
do_set,           /* 66 = setgroups */
do_getmcontext,   /* 67 = getmcontext */
do_setmcontext,   /* 68 = setmcontext */
do_mycall,        /* 69 = unused */
no_sys,           /* 70 = unused */
do_sigaction,     /* 71 = sigaction */
do_sigsuspend,    /* 72 = sigsuspend */
do_sigpending,    /* 73 = sigpending */
do_sigprocmask,   /* 74 = sigprocmask */
do_sigreturn,     /* 75 = sigreturn */
do_reboot,        /* 76 = reboot */
do_svrcctl,       /* 77 = svrctl */
do_sysuname,      /* 78 = sysuname */

[ Wrote 130 lines ]
```

nano /usr/src/servers/pm/proto.h



```
minix (before syscall) [Running]
GNU nano 2.2.4      File: /usr/src/servers/pm/proto.h      Modified

#if ! USE_MCONTEXT
#define do_getmcontext no_sys
#define do_setmcontext no_sys
#endif

/* misc.c */
int do_reboot(void);
int do_sysuname(void);
int do_getsysinfo(void);
int do_getprocnr(void);
int do_getepinfo(void);
int do_getepinfo_o(void);
int do_svrcctl(void);
int do_getsetpriority(void);
int do_mycall(void);

/* schedule.c */
void sched_init(void);
int sched_start_user(endpoint_t ep, struct mproc *rmp);
int sched_nice(struct mproc *rmp, int nice);

^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^U Next Page  ^U UnCut Text ^T To Spell
```

Create the function do_mycall() in the file </usr/src/servers/pm/misc.c>.

```
minix (before syscall) [Running]
# nano /usr/src/servers/pm/misc.c

minix (before syscall) [Running]
GNU nano 2.2.4          File: /usr/src/servers/pm/misc.c          Modified
uts_val.sysname,
NULL,                  /* No bus */                      /* No bus */
};

#endif ENABLE_SYSCALL_STATS
unsigned long calls_stats[NCALLS];
#endif

/*=====
 *   do_mycall
 *=====
int do_mycall(){
    printf("Hello World! First System call!\n");
    return 0;
}

=====*/
*                         do_sysuname                         *
=====*/
int do_sysuname()
[ Cancelled ]
^G Get Help  ^O WriteOut  ^R Read File  ^Y Prev Page  ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

In the directory </usr/src/servers/pm> execute the make command to compile the files

```
minix (before syscall) [Running]
# cd /usr/src/servers/pm
```

```
minix (before syscall) [Running]
# cd /usr/src/servers/pm
# make
compile pm/main.o
compile pm/forkexit.o
compile pm/break.o
compile pm/exec.o
compile pm/time.o
compile pm/alarm.o
compile pm/signal.o
compile pm/utility.o
compile pm/table.o
compile pm/getset.o
compile pm/misc.o
compile pm/profile.o
compile pm/schedule.o
compile pm/mcontext.o
compile pm/trace.o
link pm/pm
#
```

Create a user library. Go to `/usr/include` and create a file called `mycalllib.h`

```
minix (before syscall) [Running]
# cd /usr/include
# -
```

```
minix (before syscall) [Running]
# nano mycalllib.h
```

Inside this library, execute the system call 69 (MYCALL)

The screenshot shows a terminal window titled "minix (before syscall) [Running]". At the top, there are three small circular icons. Below the title, the text "GNU nano 2.2.4" is displayed, followed by "File: mycalllib.h" and "Modified". The main area contains the following C code:

```
#include <lib.h>
#include <unistd.h>

int mycall(){
    message m;
    return (_syscall(PM_PROC_NR, MYCALL, &m));
}
```

At the bottom of the terminal window, there is a menu bar with various keyboard shortcuts for file operations like Get Help, WriteOut, Read File, Prev Page, Cut Text, Cur Pos, Exit, Justify, Where Is, Next Page, UnCut Text, To Spell.

Compile the OS, inside [/usr/src/releasetools/](#) execute #make hdboot

The screenshot shows a terminal window titled "minix (before syscall) [Running]". At the top, there are three small circular icons. The main area contains the following command:

```
# cd /usr/src/releasetools/
```

Below the command, there is a single "#".

#make hdboot

After the make hdboot, sync and shutdown to restart minix.

```
minix (before syscall) [Running]

install ==> acpi
    install /usr/sbin/acpi
install ==> virtio_blk
    install /sbin/virtio_blk
install ==> virtio_net
    install /usr/sbin/virtio_net
    install /etc/system.conf.d/virtio_net
install ==> ramdisk
install ==> memory
    install /usr/sbin/memory
git: not found
Done.

# sync
# shutdown

Broadcast message from root@10.0.2.15 (console)
Tue Oct 12 21:06:34 2021...

The system will shutdown NOW

Local packages (down): sshd done.
Sending SIGTERM to all processes ...
MINIX will now be shut down ...
```

Create a test program. Inside /home, create a C file with calling mycall() function of the mycalllib.h

```
minix (before syscall) [Running]

Password:
You have the Auto capture keyboard option turned on. This will cause the Virtual Machine to automatically capture the keyboard every time the VM window is focused. You can change this setting in the VM settings.

Copyright (c) 2012, Vrije Universiteit, Amsterdam, The Netherlands

To install additional packages, run 'pkгин' .

To install packages from the online package repository, if you have a working network connection from MINIX: first do a 'pkгин update' to update the list of available packages, and then do a 'pkгин' to get a list of commands. For example, 'pkгин install vim' installs the 'vim' package, and 'pkгин available' will list all available packages.

To install packages from the installation CD: same, but use pkгин_cd .
To switch to the online repository, do 'pkгин update' again. To install all packages, do pkгин_all .

MINIX 3 supports multiple virtual terminals. Just use ALT+F1, F2, F3 and F4 to navigate among them.

For more information on how to use MINIX 3, see the wiki:
http://wiki.minix3.org.

# cd /home
# nano test.c_
```

minix (before syscall) [Running]
GNU nano 2.2.4 File: test.c Modified

```
#include <mycalllib.h>

int main(){
    mycall();
    return 0;
}
```

G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos
X Exit ^J Justify ^W Where Is ^U Next Page ^U UnCut Text ^T To Spell

Your output should be the printf that you wrote in do_mycall():

minix (before syscall) [Running]

```
# clang test.c
# ./a.out
Hello World! First System call!
```