السلام عليكم ورحمة الله وبركاته

اخباركم ايه يارب تكونوا بخير ..

programming-fr34ks.net ; linux arabic programming tutorial
author : St0rM-MaN
license : Rippers License , Gnu/Gpl ;
based on : advanced linux programming

OMG!!! Linux Programming !!!

Cool Ain't :D

Visit us (Http://Programming-Fr34ks.Net) ;

friends :

http://securitygurus.net
http://arab4services.com
http://linuxac.org
http://linux-fr34k.com
http://sechost-it.com
/*security rockers */

الموضوع حيتم عرضه علي هيئة سؤال وجواب وبعض القطع للمناقشه .

مقدم من : Programming-Fr34ks.net

برمجة اللينكس الجزء الاول :

الفصل الاول :التعامل مع الملفات

س 1: ماهو الملف بالنسبه لل linux ؟
ج 1:هو كل شيئ علي الجهاز

س 2 : اين توجد الملفات ؟
ج 2:تقع كلها تحت ال root directory ورمزه ال / اي ان اي ملف في جهازك الخاص يقع تحت /
وتحت ال / قد توجد هناك directories اخري تحتوي علي ملفات او directories اخري
مثال علي ذلك

/home/St0rM/Desktop/File.c

تحت المجلد الاصلي / في ال dir المدعي ب home هناك dir اخر يدعي St0rM يحتوي علي dir اخر يدعي Desktop والذي بدوره يحتوي علي الملف المطلوب

س 3 : هل حقيقي ان كل شيئ في ال linux عباره عن ملف اذن ماهو ال dir ؟
ج 3 : اجل حقيقي ان كل شيئ في ال Linux عباره عن ملف حتي اتصالات ال sockets ماهو الا ملف مرفق به عنوان ويتسنخدم في الكتابه والقرائه من هذا ال socket
ال dirs ماهي الا نوع معين من الملفات مرفق به التصريح d
اذا تلاحظ ان ال ملفات في ال Linux كلها لها تصاريح مقسمه الي 4 اجزاء

```
r = read
w = write
x = execute

type users{r/w/x} , groups{ r/w/x} , others{ r/w/x}
```

ال dirs لها نوع d مما يحدد انه نوع خاص من الملفات يحتوي علي اسماء الملفات تحته او مؤشرات لتلك الملفات
لاحظ ان اسم الملف ماهو الا hard link لذلك الملف (تابع لاحقا)

س 4 : هل هناك انواع اخري ؟
ج 4 : اجل كما قلت dirs منها وهناك sockets او pipe او fifo
كلها انواع معينه من الملفات لها طرق خاصه في التعامل

س 5 : لكن كيف توضع او تسجل البيانات في ملف ؟
ح 5 : عن طريق بعض الدوال مثل read , write تتعامل مع الملف علي اساس انه حاويه لتحتوي ال bytes المراد تخزينها سواء ان كانت ارقام او حروف او binary مثل الملفات التنفيذيه لانه ببساطه يتعامل معها علي اساس انها bytes streams اي سيل من ال bytes .

س 6 : ماهو طول اسماء الملفات وماهي قوانين تسمية الملفات علي اي حال ؟
ج 6 : طول الملف قد يصل الي 255 حرف
بالنسبه لقوانين تسميته :
اولا الملف XX يختلف تماما عن xx اي ان هناك تدقيق في اسماء الملفات من نايحة الحروف الكبيره او الصغيره (حساساه لحالة الاسماء)
يمنع عليك وضع الحرف / لانه محجوز بالنسبه لل linux علي اساس انه يقووم بالاشاره الي الدخول الي هذا ال dir اي ان ملف this/is/me.txt
سيتعامل معاه اللينكس علي اساس انه ملف me.txt موجود داخل this/me/
يكفينا من كل هذا الان راجع موضوع الاخ االعزيز ابو عبد الرحمن
سطر الأوامر نظرة عن قرب أكثر

http://linuxac.org/forum/showthread.php?t=1586

الحقوق محفوظه لمجتمع ليونكس العربي الخاضع تحت الرخصه العموميه.

س 7 : كيف اذن نتعامل مع تلك الملفات عن طريق البرمجه ؟
ج 7 : هناك بعض الدوال التي تسمح لك بالتعامل مع الملفات من اول فتح الملف الي اغلاقه مرورا بالكتابه او القرائه او حتي تعديل البيانات الخاصه بالملف في حد ذاته

س 8 : كيف اذن وماهي تلك الدوال ؟
ج 8 : اولا دعنا نناقش كيف نفتح ملف وماذا يعني بفتح ملف
فتح ملف ليس بالضروره يعني فتحه للكتابه او للقارئه او حتي ايا منهما
فتح ملف يعني انك قد قمت بتضمين المعلومات الخاصه بهذا الملف في جدول العنواين الخاصه بالعمليه المرفقه بالبرنامج الخاص بك
كل برنامج يعمل علي اللينكس له مساحه محجوزه من الذاكره تدعي memory addresses يوضع فيها كل شيئ خاص بالبرنامج من عدد العمليات ومن عدد ال files descriptors المفتوحه (تعرف معناها لاحقا) وهكذا
يمكنك استخدام طريقتين للتعامل مع الملفات
طريقه stander خاصه بكل الانظمه و portable ايضا تدعي ب
C standard files manipulating functions
وطريقه اخري خاصه باللينكس عن طريق فتح files descriptor

ال file descriptor ماهو الا رقم مرفق به الملف المفتوح (لاحقا تتعرف عليه)

اولا : C standard files manipulating functions

تحتوي ال c علي data structure او ماتسمي بمنشئة بيانات خاصه بالتعامل مع الملفت
تدعي ب FILE تحتوي تلك المنشاه علي عدة عناصر للتعامل مع الملف
من ضمنها اين نوجد حاليا في الملف , تصريح الملف , ...;

س 9:كيف نتعامل مع تلك العناصر؟
عن طريق بعض الدوال تتعرف عليها الان ;

س 10 : كيف نقوم بفتح ملف عن طريق تلك الدوال ؟
اول شيئ يجب عليك فعله هو تحضير تلك المنشئه
لنبدا بكتابة اول سطر في الكود

```
#include<stdio.h>

int main(void)
{
   FILE* file_to_open;
```

قمنا بتعريف مؤشر الي structure من النوع file (لا تستغرب معرفه في ال stdio ومحوله بواسطة typedef)
اصبح لدينا الان data structure لنتعامل مع الملف عن طريقها

كيف اذن نقوم بفتح الملف ؟

```
#include <stdio.h>

   FILE *
   fopen(const char * restrict path, const char * restrict mode);
```

داله ترجع مؤشر الي ال FILE structure المرد التطبيق عليها
اول شيئ يمرر الي الداله مكان وجود الملف

ثاني شيئ طريقة فتح الملف وكيفية التعامل معه
عن طريق mode تستطيع فتح ملف اما للقرائه او للكتابه او للدمج

r , w , a

لاحظ انه في حالة فتح ملف لللقرائه فقط يجب ان يكون الملف موجود مسبقا او تفشل الداله في فتحه
واذا كان الملف موجود مسبقا وتم فتحه للكتابه يتم تدمير محتوياته واذا لم يكن موجود اصلا يتم انشائه
واذا فتح للدمج ولم يكن موجود مسبقا يتم انشائه اما اذا كان موجود يتم ارجاع اخر مكان كتب فيه الي ال FILE structure (راجع تعريفها)
اما اذا استخدمت

r+ , w+ , a+

الاولي تعني للقرائه و الكتابه بدون تدمير محتويات
الثانيه تعني فتح الملف للقرائه والكتابه واذا لم يكن موجود انشئه اما اذا كان موجود دمر محتوايته ثم افتحه للتعامل معه .
الثالثه تعني افتح الملف للدمج والقرائه اذا كان موجود اشر الي نهاية الملف للقرائه او الكتابه اذا لم يكن موجود قم بانشائه واشر الي البدايه .
لاحظ انه في حالة الفشل الداله ترجع NULL ويمكنك استخدام perror
يجب عليك اغلاق الملف بعد الانتهاء منه , سوف يغلق باي حال من الاحوال عند انتهاء البرنامج من العمل ولكن افعل ما عليك فعله .

```
Int fclose(FILE *file);
```

مثال علي ذلك :

```
#include<stdio.h>
#include<unistd.h>
#include<errno.h>

int main(void)
{
   FILE* file_to_open;

   file_to_open = fopen("textfile" , "r+");

   if(file_to_open == NULL)
   {
            perror("fopen()");
            _exit(1);
   }
       fclose(file_to_open);
   return 0;
}
```

مفهوم وواضح علي ماظن .

كيف اذن ننفذ عمليات علي ذلك الملف , كيف نكتب بداخله او نقرا منه ؟
عدة دوال مستخدمه في ذلك مثال

fgets , fputs , fprintf , fscanf , fputc , fgetc ;

اولا

```
char *
 fgets(char * restrict str, int size, FILE * restrict stream);
```

تقوم بالكتابه داخل السلسله str بالحجم المطلوب size من ال stream

ال stream هو الملف المطلوب
تقرا سطر واحد فقط لاغير لانه عندما تجد ان السطر الحالي انتهي (ضغط enter) تقوم بانهاء القرائه وتخزين ماتم قرائته في str والانتهاء .

لاحظ انه stdin هو احدي ال streams المفتوحه دائما معك .

مثال لقرائة اول سطر من الملف

/etc/services

```
#include<stdio.h>
#include<unistd.h>
#include<errno.h>

int main(void)
{
   FILE* file_to_open;
   char buffer[81]; /*enough to hold one line*/

   file_to_open = fopen("/etc/services" , "r"); /*open for read and write if exist*/

   if(file_to_open == NULL)
   {
            perror("fopen()");
            _exit(1);
   }

   fgets(buffer , 250 , file_to_open);
   printf("File /etc/services first line is\n"
          "%s",buffer);
   fclose(file_to_open);

   return 0;
}
```

ماذا اذا اردنا الكتابه داخل ملف ؟

```
#include <stdio.h>

   int
   fputs(const char *str, FILE *stream);
```

قم بكتابة السلسله str بداخل ال stream .

```
#include<stdio.h>

#include<unistd.h>

#include<errno.h>
```

```
int main(void)
{
   FILE* file_to_open;
   char buffer[81]; /*enough to hold one line*/
   file_to_open = fopen("test" , "w"); /*open for read and write if exist*/
   strcpy(buffer , "Hello Files\n"); /*copy our string */
   if(file_to_open == NULL)
   {
                perror("fopen()");
                _exit(1);
   }
   fputs(buffer , file_to_open); /*write it */
   printf("File test file has line \n"
             "%s",buffer);
   fclose(file_to_open);

   return 0;
}
```

استكشف الباقي بنفسك . لاحظ انه بعد كتابتك النص اصبح file_to_read يؤشر الي نهاية الملف اذا كنت تحتاج ان تقوم بالكتابه مره اخري استخدم rewind();

```
void
   rewind(FILE *stream);
```

اذا اردت مرعرفة موقعك في الملف بالتحديد استخدم

```
   long
   ftell(FILE *stream);
```

اذا اردت تغير موقع في الملف استخدم

```
int

    fseek(FILE *stream, long offset, int whence);
```

كود كامل يشرح ذلك يكتب سلسله داخل ملف ويقوم بعمل *rewind* للملف ثم يقرئها ثم يقوم بعمل *rewind* مره اخري ويحرك المؤشر بواسطة *fseek* الي نهاية الملف ويخربك مكانك في الملف باتحديد (عدد ال *bytes* السابقه)

```
#include<stdio.h>

#include<unistd.h>

#include<errno.h>

#include<string.h>

int main(void)

{

   FILE* file_to_open;

   char buffer[81]; /*enough to hold one line*/

   char buf[81]; /*enough to read one line*/

   unsigned long int pos;

   file_to_open = fopen("test.txt" , "w+"); /*open for read and write if exist*/

   strcpy(buffer , "Hello Files\n"); /*copy our string */

   if(file_to_open == NULL)

   {

                perror("fopen()");

                _exit(1);

   }

   fputs(buffer , file_to_open); /*write it */

   rewind(file_to_open);

   pos = ftell(file_to_open);

   printf("Current position %lu\n",pos);

   fgets(buf , 81 , file_to_open);

   printf("File test has %s\n",buf);
```

```
    rewind(file_to_open);

    fseek(file_to_open , SEEK_END , 0 ); /* reacg end of file */

    pos = ftell(file_to_open);

    printf("File test size %lu\n",pos);

    fclose(file_to_open);



    return 0;

}
```

وبكده انتهينا من جزء ال standard ;

جزء ال file descriptors اكثر متعه وسهل جدا لو فهمت الي فات.

كل مافي الامر انك بتعرف متغير عادي جدا int وبتستخدم الداله open عشان تفتح الملف !

كل الي بيحصل ان الداله open بتفتح حاجه اسمها file descriptor

عباره عن رقم مرفق به الملف يتستخدم للتلاعب بالملف من قرائة وكتابه و .. يعتبر كمؤشر لجدول مرفق في العمليه كما سبق وان اخبرتكم

هذا الجدول يحتوي علي ال file descriptors الموجوده بداخل تلك العمليه ويحتوي علي اماكن ال memory الموجود بداخلها الملف ومعلومات عن الملف

تشبه FILE ..

س 11 : كيف تحصل علي file descriptor خاص بملف ؟

عن طريق الداله open

```
#include <fcntl.h>


    int
    open(const char *path, int flags, mode_t mode);

     int open(const char path , int flags);
```

flags = O_CREAT , O_EXCL , O_TRUNC ,O_SHLOCK , O_EXLOCL , O_RDONLY , O_WRONLY , O_RDWR , O_APPEND

لاحظ ان ال modes اختياريه .

O_CREAT = انشئ الملف

O_EXCL = اذا كان الملف موجود لاتق بانشائه وارجع خطا

O_TRUCN = اذا كان موجود قم بتدمير محتوياته وانشئه

P_SHLOCK , O_EXLOCK = سوف اقوم بشرحهم في اجزاء اخري

O_RDONLY = افتح للقرائه فقط

O_WRONLY = للكتابه فقط

O_APPEND = للدمج

O_RDWR = اقرا واكتب

ماهي ال modes ? هي عباره عن permissions تمرر الي الداله حين الحاجه مثلا اذا اردت ان يكون ال access مسموح به فقط لل users دون اي شخص اخر

S_IRWXU , S_IRUSR , S_IWUSR , S_IXUSR (users); = read/write , read , write , execute

S_IRWXG , S_IRGRP , S_IWGRP , S_IXGRP (groups); = read/write , read , write , execute

S_IROTH , S_IROTH , S_IWOTH , S_IXOTH (others); = read/write , read , write , execute

س 12:كيف تقوم بدمج كل تلك ال modes وال flags مع بعض لتحصل علي نتيجه مطلوبه ؟

عن طريق استخدام ال bitwise وعمل | oring لتلك ال flags او ال modes

مثال فتح ملف اذا كان موجود اخرج اذا لم يكن موجود قم بانشائه وافتحه للكتابه فقط ومسموح ايضا للكتابه لل users فقط (هل تري قدرة التحكم العجيبه)

```
int fd;

fd = open("test.txt" , O_CREAT | O_EXCL , I_RWUSR);
```

اذا رادت فقتحه وتدمير محتوياته وكل الناس لها قدرة الكتابه والقرائه ؟

```
int fd;

fd = open("test.txt" , O_CREAT | O_TRUCN , I_RWUSR | I_RWGRP , | I_RWOTH);
```

كيف تقوم بقفله اذن ؟

```
#include <unistd.h>


    int
    close(int d);
```

تقوم فقط بقفل ال file descriptor اي قفل التعامل مع الملف في العمليه الحاااااااااليه .

هناك خدعه جميله تستخدم في حالة كتابة بيانات لاتود المستخد ان يراها او يعرف بوجودها اطلاقا .

يمكن ان تقوم بانشاء ملف ثم حذفه !!! كيف ؟ اذن كيف تتعامل معه ؟ عن طريق ال file descriptor الم اقل لك انه يشير الي المكان في الذاكره ؟ واالم اقل لك ان اسم الملف ماهو الا hard link لهذا الملف ! بالتحديد مايحدث ان الداله unlink() تقوم بحذف هذا ال link ! اذن كيف يحذف الملف نهائيا ؟

عندما تنتهي كل ال file descriptors وال hard links الموجوده داخل النظام باكمله من التعامل مع هذا الملف .

لاحظ انه open ترجع -1 في حالة الفشل لان ال file descriptor هي ارقام موجبه فقط . الفشل قد يكون لاي سبب ليس لديك تصريح او لايمكن فتح الملف او او او . استخدم perror لمعرفة السبب.

س 13 : كيف تقرا وتكتب داخل ملف ؟

read , write وانتهي الامر .

```
#include <sys/types.h>

#include <unistd.h>

ssize_t
read(int d, void *buf, size_t nbytes);
```

قم بالقرائه من ال file descriptor المشار اليه بوساطة d

اكتب في buf ايا كان نوعه من الحجم nbytes ما تم قرائته من d

لاحظ انه nbytes تستخدم فقط اذا كنت تمرر مؤشر او سلسله مثلا

مثلا اذا اردت القرائه من الملف المرفق في d كيف ؟

```
#include<stdio.h>

#include<unistd.h>

#include<fcntl.h>

#include<sys/types.h>

int main(void)

{



    int fd;

    char buf[80];

    size_t bytes_read;



    fd = open("test.txt" , O_RDONLY);



    bytes_read = read(fd , buf , sizeof(buf));
```

```
    if(bytes_read < 0)
    {
                perror("read()");
                _exit(1);
    }

    buf[bytes_read] = '\x0';

    printf("%s with length %d\n",buf , (int)bytes_read);

    close(fd);
}
```

لاحظ ان read ترجع ماقد تم قرائته بالفعل من ال bytes

اذا كانت اقل من ال 0 اذن قد حدث خطا ما ونتحقق منه بواسطة perror

لاحظ انه ليس هناك ضمان ان السلسله مقفوله بوساطة ال terminating character

لذا نتاكد من ذلك بقفلها نحن عن طريق القيمه المرجعه من read

س14 : كيف تكتب اذن داخل ملف ؟

مثل السابق ولكن عن طريق write .

```
  ssize_t
    write(int d, const void *buf, size_t nbytes);
```

ببساطه اكتب داخل fd مايشار اليه من buf .

مثال اكتب ال "Hello World"

```
#include<stdio.h>
#include<string.h>
#include<unistd.h>
```

```
#include<fcntl.h>

#include<sys/types.h>

int main(void)

{

    int fd;

    char buf[80];

    size_t bytes_write;

    fd = open("test.txt" , O_WRONLY);

    strcpy(buf , "Hello World");

    bytes_write = write(fd , buf , strlen(buf));

    if(bytes_write < 0)

    {

                perror("read()");

                _exit(1);

    }


    printf("%s with length %d has been written\n",buf , (int)bytes_write);

    close(fd);

}
```

س15: كيفية اذن معرفة مكانك داخل الملف او التحرك داخل الملف ؟

```
 #include <unistd.h>


     off_t
     lseek(int fildes, off_t offset, int whence);
```

كما سبق اضف offest الي whence لكي تذهب الي مكانك المطلوب مثلا اذهب 10 bytes من المكان الحالي

```
lseek(fd , 10 , SEEK_CUR);
```

ااو اجعله يؤشر االي بعد اول 10 bytes

```
lseek(fd , 10 , SEEK_SET);
```

راجع ال man pages .

لاحظ انها بترجع مكانك في ال file بدل ftell ووجع الدماغ لا هي بتديك مكانك .

وبكده انتهينا من جزئية التعامل مع الملفات .

ملخص :

كل شيئ في الليوونكس عباره عن ملف

يتعامل الليونكس مع الملفات علي انها سيل من ال bytes

يمكنك استعامل طريقتين في التعامل مع الملفات

الطريقه الثانيه تجعلك في تحكم اكبر مع الملفات الخاصه باللينكس .

الفصل الاول انتهي .

ماذا بعد ؟

التعامل مع ال dirs والتحقق من الملفات وتغير المعلومات الخاصه بها .