linux下使用binfmt_misc设定不同二进制的打开程序

转载

whatday 于 2019-03-07 13:52:08 发布 6109 收藏 9

在Windows平台上，可以绑定拥有特定扩展名的文件，使用特定的程序打开。比如，PDF文件就使用Acrobat Reader打开。这样做确实极大的方便了用户的使用体验。

其实，在Linux平台上，也提供了类似的功能，甚至从某种意义上来说更加的强大。Linux的内核从很早开始就引入了一个叫做Miscellaneous Binary Format（binfmt_misc）的机制，可以通过要打开文件的特性来选择到底使用哪个程序来打开。比Windows更加强大的地方是，它不光光可以通过文件的扩展名来判断的，还可以通过文件开始位置的特殊的字节（Magic Byte）来判断。

如果要使用这个功能的话，首先要绑定binfmt_misc，可以通过以下命令来绑定：

```
mount binfmt_misc -t binfmt_misc /proc/sys/fs/binfmt_misc
```

这样绑定的话，系统重新启动之后就失效了。如果想让系统每次启动的时候都自动绑定的话，可以往/etc/fstab文件中加入下面这行：

```
none  /proc/sys/fs/binfmt_misc binfmt_misc defaults 0 0
```

绑定完之后，就可以通过向/proc/sys/fs/binfmt_misc/register（这个文件只能写不能读）文件中写入一行匹配规则字符串来告诉内核什么样的程序要用什么样的程序打开（一般使用echo命令）。这行字符串的格式如下：

```
:name:type:offset:magic:mask:interpreter:flags
```

每个字段都用冒号“:”分割。某些字段拥有默认值，或者只在前面字段被设置成了某个特定值后才有效，因此可以跳过某些字段的设置，但是必须保留相应的冒号分割符。各个字段的意义如下：

1）name：这个规则的名字，理论上可以取任何名字，只要不重名就可以了。但是为了方便以后维护一般都取一个有意义的名字，比如表示被打开文件特性的名字，或者要打开这个文件的程序的名字等；

2）type：表示如何匹配被打开的文件，只可以使用“E”或者“M”，只能选其一，两者不可共用。“E”代表只根据待打开文件的扩展名来识别，而“M”表示只根据待打开文件特定位置的几位魔数（Magic Byte）来识别；

3）offset：这个字段只对前面type字段设置成“M”之后才有效，它表示从文件的多少偏移开始查找要匹配的魔数。如果跳过这个字断不设置的话，默认就是0；

4）magic：它表示真正要匹配的魔数，如果type字段设置成“M”的话；或者表示文件的扩展名，如果type字段设置成“E”的话。对于匹配魔数来说，如果要匹配的魔数是ASCII码可见字符，可以直接输入，而如果是不可见的话，可以输入其16进制数值，前面加上“\x”或者“\\x”（如果在Shell环境中的话。对于匹配文件扩展名来说，就在这里写上文件的扩展名，但不要包括扩展名前面的点号（“.”），且这个扩展名是大小写敏感的，有些特殊的字符，例如目录分隔符正斜杠（“/”）是不允许输入的；

5）mask：同样，这个字段只对前面type字段设置成“M”之后才有效。它表示要匹配哪些位，它的长度要和magic字段魔数的长度一致。如果某一位为1，表示这一位必须要与magic对应的位匹配；如果对应的位为0，表示忽略对这一位的匹配，取什么值都可以。如果是0xff的话，即表示全部位都要匹配，默认情况下，如果不设置这个字段的话，表示要与magic全部匹配（即等效于所有都设置成0xff）。还有同样对于NUL来说，要使用转义（\x00），否则对这行字符串的解释将到NUL停止，后面的不再起作用；

6）interpreter：表示要用哪个程序来启动这个类型的文件，一定要使用全路径名，不要使用相对路径名；

7）flags：这个字段可选，主要用来控制interpreter打开文件的行为。比较常用的是‘P’（请注意，一定要大写），表示保留原始的argv[0]参数。这是什么意思呢？默认情况下，如果不设置这个标志的话，binfmt_misc会将传给interpreter的第一个参数，即argv[0]，修改成要被打开文件的全路径名。当设置了‘P’之后，binfmt_misc会保留原来的argv[0]，在原来的argv[0]和argv[1]之间插入一个参数，用来存放要被打开文件的全路径名。比如，如果想用程序/bin/foo来打开/usr/local/bin/blah这个文件，如果不设置‘P’的话，传给程序/bin/foo的参数列表argv[]是["/usr/local/bin/blah", "blah"]，而如果设置了‘P’之后，程序/bin/foo得到的参数列表是["/bin/foo", "/usr/local/bin/blah", "blah"]。

除了以上的规则之外，还有一些额外的限制条件：

1）每一行匹配规则字符串的长度不能超过1920个字符；

2）魔数（Magic Byte）必须在文件头128个字节内，也就是说offset+sizeof(magic)不能超过128；

3）interpreter字段的长度不能超过127个字符。

每次成功写入一行规则，都会在/proc/sys/fs/binfmt_misc/目录下，创建一个名字为输入的匹配规则字符串中"name"字段的文件。

通过读取这个文件的内容，可以知道这条匹配规则当前的状态：

```
cat /proc/sys/fs/binfmt_misc/<name>
```

而通过向这个文件中写入0或1，可以关闭或打开这条匹配规则，而写入-1表示彻底删除这条规则：

```
echo 0 > /proc/sys/fs/binfmt_misc/<name>    # Disable the match
echo 1 > /proc/sys/fs/binfmt_misc/<name>    # Enable the match
echo -1 > /proc/sys/fs/binfmt_misc/<name>   # Delete the match
```

不过前提是必须要切换到root用户才能有权限写入。

在/proc/sys/fs/binfmt_misc/目录下，还缺省存在一个叫做status的文件，通过它可以查看和控制整个binfmt_misc的状态，而不光是单个匹配规则。

可以查看当前binfmt_misc是否处于打开状态：

```
cat /proc/sys/fs/binfmt_misc/status
```

也可以通过向它写入1或0来打开或关闭binfmt_misc：

```
echo 0 > /proc/sys/fs/binfmt_misc/status    # Disable binfmt_misc
echo 1 > /proc/sys/fs/binfmt_misc/status    # Enable binfmt_misc
```

如果想删除当前binfmt_misc中的所有匹配规则，可以向其传入-1：

```
echo -1 > /proc/sys/fs/binfmt_misc/status    # Disable all matches
```

---

binfmt_misc文档翻译

Kernel Support for miscellaneous (your favourite) Binary Formats v1.1
内核对五花八门的二进制格式的支持
==========

This Kernel feature allows you to invoke almost (for restrictions see below)
every program by simply typing its name in the shell.
该内核允许你调用几乎任何一个程序，你仅仅需要在shell中键入他的名字。
This includes for example compiled Java(TM), Python or Emacs programs.
这包括例如被编译之后的java，python或者是emacs程序。

To achieve this you must tell binfmt_misc which interpreter has to be invoked
with which binary. Binfmt_misc recognises the binary-type by matching some bytes
at the beginning of the file with a magic byte sequence (masking out specified
bits) you have supplied. Binfmt_misc can also recognise a filename extension
aka '.com' or '.exe'.
为了达到这个目的，你必须告诉binfmt_misc你需要调用哪一个二进制格式的解释器。Binfmt_misc
程序通过匹配文件开始的一些二进制序列（以一些特殊的二进制位来标识）来识别是哪一种二进制
类型。Binfmt_misc也能够识别一个文件的扩展名例如'.com'或者是'.exe'.

First you must mount binfmt_misc:
mount binfmt_misc -t binfmt_misc /proc/sys/fs/binfmt_misc
首先你需要挂载binfmt_misc
mount binfmt_misc -t binfmt_misc /proc/sys/fs/binfmt_misc

To actually register a new binary type, you have to set up a string looking like
:name:type:offset:magic:mask:interpreter: (where you can choose the ':' upon
your needs) and echo it to /proc/sys/fs/binfmt_misc/register.
为了能够真正的注册一个新的二进制类型，你应该设置一个字符串看上去像:name:type:offset
:magic:mask:interpreter:(你可以根据你的需要选择':')并且将他广播到/proc/sys/fs/binfmt_misc/resgister中

Here is what the fields mean:
下面是这些参数的意思：
- 'name' is an identifier string. A new /proc file will be created with this
  name below /proc/sys/fs/binfmt_misc
- 'name'是定义字符串.一个新的/proc文件将在/proc/sys/fs/binfmt_misc下以他的名字被创建

- 'type' is the type of recognition. Give 'M' for magic and 'E' for extension.
- 'type' 是识别的类型。'M'代表着二进制序列和'E'是扩展

- 'offset' is the offset of the magic/mask in the file, counted in bytes. This
  defaults to 0 if you omit it (i.e. you write ':name:type::magic...')
- 'offset'是在文件中二进制学列的偏移量，以位计数。如果你这样写':name:type::magic...',
  那么它默认是0。

- 'magic' is the byte sequence binfmt_misc is matching for. The magic string
  may contain hex-encoded characters like \x0a or \xA4. In a shell environment
  you will have to write \\x0a to prevent the shell from eating your \.
  If you chose filename extension matching, this is the extension to be
  recognised (without the '.', the \x0a specials are not allowed). Extension
  matching is case sensitive!
- 'magic'是binfmt_misc要匹配的位序列。这个位序列可能包含16进制编码的字符例如\x0a
  或者是 \xA4.在一个shell环境中，你必须写\\x0a来避免shell吞掉你的\。如果你选择
  使用文件扩展名开匹配的话，下面是一些小规则(不能有'.',\x0a是不允许的).并且扩展名
  匹配是大小写敏感的。

- 'mask' is an (optional, defaults to all 0xff) mask. You can mask out some
  bits from matching by supplying a string like magic and as long as magic.
  The mask is anded with the byte sequence of the file.
- 'mask'是一个标识(可选的，默认所有的都是0xff)。通过提供一个类似于magic或者是和magic
一样长的字符串来从正在匹配的文件中标识出一些位。

- 'interpreter' is the program that should be invoked with the binary as first
  argument (specify the full path)
- 'interpreter' 应该是被二进制文件首先被调用的参数(特别是全路径)。

There are some restrictions:
同时还有一些显示:
- the whole register string may not exceed 255 characters　　整体的寄存器字符串不应该超过255个字
符
- the magic must reside in the first 128 bytes of the file, i.e.　　位序列一定要放在文件的前128位上，例如
offset+size(magic)一定要少于128
  offset+size(magic) has to be less than 128
- the interpreter string may not exceed 127 characters　　解释器字符串不能超过127个字符

To use binfmt_misc you have to mount it first. You can mount it with
"mount -t binfmt_misc none /proc/sys/fs/binfmt_misc" command, or you can add
a line "none /proc/sys/fs/binfmt_misc binfmt_misc defaults 0 0" to your
/etc/fstab so it auto mounts on boot.
为了能够使用binfmt_misc你必须先去挂载她。你可以使用命令"mount -t binfmt_misc

none /proc/sys/fs/binfmt_misc"来挂载她，或者是你可以在/etc/fstab中添加一行
"none /proc/sys/fs/binfmt_misc binfmt_misc defaults 0 0",这样他就会自动在
启动的时候挂载。

You may want to add the binary formats in one of your /etc/rc scripts during
boot-up. Read the manual of your init program to figure out how to do this
right.
你可能想要在启动的时候在你的一个/etc/rc脚本文件中添加一个二进制格式。阅读你的启动
程序手册来弄清楚如何正确的做这些事情。

Think about the order of adding entries! Later added entries are matched first!
请思考一些添加的顺序，后添加的会先被匹配。

A few examples (assumed you are in /proc/sys/fs/binfmt_misc):
几个小例子（假设你在/proc/sys/fs/binfmt_misc中）

- enable support for em86 (like binfmt_em86, for Alpha AXP only):
  echo
':i386:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\x03:\xff\xff\xff\xff\xff\xfe\xfe\xff\xff\xff\
> register
  echo
':i486:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\x06:\xff\xff\xff\xff\xff\xfe\xfe\xff\xff\xff\
> register
启动对em86的支持(类似于binfmt_em86，仅仅对Alpha AXP适用)
echo
':i386:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\x03:\xff\xff\xff\xff\xff\xfe\xfe\xff\xff\xff\
> register
  echo
':i486:M::\x7fELF\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x02\x00\x06:\xff\xff\xff\xff\xff\xfe\xfe\xff\xff\xff\
> register

- enable support for packed DOS applications (pre-configured dosemu hdimages):
  echo ':DEXE:M::\x0eDEX::/usr/bin/dosexec:' > register
启动对DOS应用程序的支持
echo ':DEXE:M::\x0eDEX::/usr/bin/dosexec:' > register

- enable support for Windows executables using wine:
  echo ':DOSWin:M::MZ::/usr/local/bin/wine:' > register
启动使用wine对windows可执行程序的支持
echo ':DOSWin:M::MZ::/usr/local/bin/wine:' > register

For java support see Documentation/java.txt
对于java支持，请查看Documentation/java.txt文件

You can enable/disable binfmt_misc or one binary type by echoing 0 (to disable)
or 1 (to enable) to /proc/sys/fs/binfmt_misc/status or /proc/.../the_name.
Catting the file tells you the current status of binfmt_misc/the entry.
你可以通过向/proc/sys/fs/binfmt_misc/status 或者是 /proc/.../the_name广播0(关闭)
或者是1(启动)来启动/关闭binfmt_misc或者是一个二进制类型。查看文件能够告诉你binfmt_misc
/the entry的当前状态。

You can remove one entry or all entries by echoing -1 to /proc/.../the_name
or /proc/sys/fs/binfmt_misc/status.
你可以通过向/proc/.../the_name或者是/proc/sys/fs/binfmt_misc/status广播-1来移除
一个记录或者是所有的记录。

HINTS:
提示：
======

If you want to pass special arguments to your interpreter, you can
write a wrapper script for it. See Documentation/java.txt for an
example.
如果你想要向你的解释器传送特殊的参数，你可以向他写一个脚本文件。例如
你可以查看Documentation/java.txt文件。

Your interpreter should NOT look in the PATH for the filename; the
kernel passes it the full filename to use. Using the PATH can cause
unexpected behaviour and be a security hazard.
你的解释器不应该查看文件名字的路径。（这句话不是很明白）内核会将文件全名
传送给她使用。使用PATH可能会导致不可预料的问题并且会成为一个安全危险。

There is a web page about binfmt_misc at
http://www.tat.physik.uni-tuebingen.de/~rguenth/linux/binfmt_misc.html
这里有一个有关binfmt_misc的网页:
http://www.tat.physik.uni-tuebingen.de/~rguenth/linux/binfmt_misc.html