# Selinux

**1. 简介**

    SELinux带给**[Linux](http://lib.csdn.net/base/linux" \o "Linux知识库" \t "_blank)**的主要价值是：提供了一个灵活的，可配置的MAC机制。

    Security-Enhanced Linux (SELinux)由以下两部分组成：

    1) Kernel SELinux模块(/kernel/security/selinux)

    2) 用户态工具

    SELinux是一个安全体系结构，它通过LSM(Linux Security Modules)框架被集成到Linux Kernel 2.6.x中。它是NSA (United States National Security Agency)和SELinux社区的联合项目。

    SELinux提供了一种灵活的强制访问控制(MAC)系统，且内嵌于Linux Kernel中。SELinux定义了系统中每个【用户】、【进程】、【应用】和【文件】的访问和转变的权限，然后它使用一个安全策略来控制这些实体(用户、进程、应用和文件)之间的交互，安全策略指定如何严格或宽松地进行检查。

    SELinux对系统用户(system users)是透明的，只有系统管理员需要考虑在他的服务器中如何制定严格的策略。策略可以根据需要是严格的或宽松的。

    只有同时满足了【标准Linux访问控制】和【SELinux访问控制】时，主体才能访问客体。

**1.1 DAC与MAC的关键区别(root用户)**

      安 全增强型Linux(SELinux)开始是由NSA（国家安全局）启动并加入到Linux系统中的一套核心组件及用户工具，可以让应用程序运行在其所需的最低权限上。未 经修改过的Linux系统是使用自主访问控制的，用户可以自己请求更高的权限，由此恶意软件几乎可以访问任何它想访问的文件，而如果你授予其root权 限，那它就无所不能了。

      在SELinux中没有root这个概念，安全策略是由管理员来定义的，任何软件都无法取代它。这意味着那些潜在的恶意软件所能造成的损害可以被控制在最小。一般情况下只有非常注重数据安全的企业级用户才会使用SELinux。

[**操作系统**](http://lib.csdn.net/base/operatingsystem)有两类访问控制：自主访问控制（DAC）和强制访问控制（MAC）。标准Linux安全是一种DAC，SELinux为Linux增加了一个灵活的和可配置的的MAC。

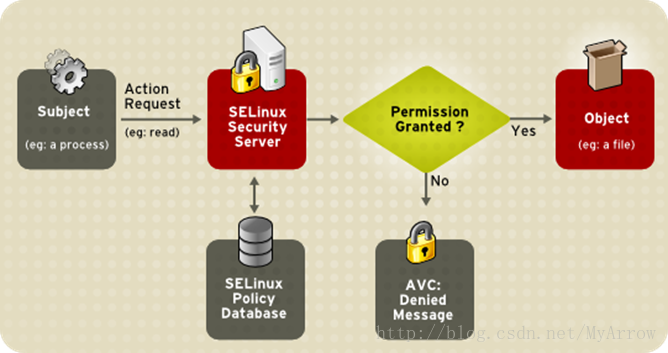
      所有DAC机制都有一个共同的弱点，就是它们不能识别自然人与计算机程序之间最基本的区别。简单点说就是，如果一个用户被授权允许访问，意味着程序也被授权访问，如果程序被授权访问，那么恶意程序也将有同样的访问权。 DAC最根本的弱点是主体容易受到多种多样的恶意软件的攻击，MAC就是避免这些攻击的出路，大多数MAC特性组成了多层安全模型。

      SELinux实现了一个更灵活的MAC形式，叫做类型强制(Type Enforcement)和一个非强制的多层安全形式(Multi-Level Security)。

      在Android4.2中，SELinux是个可选项，谷歌并没有直接取消root权限或其他功能。这是一个为企业级用户或是对隐私数据极为重视的用户提供的选项，普通消费者则完全可以关闭它。

**2. SELinux的运行机制**

    SELinux决策过程如下图所示：



      当一个subject(如: 一个应用)试图访问一个object(如：一个文件)，Kernel中的策略执行服务器将检查AVC (Access Vector Cache), 在AVC中，subject和object的权限被缓存(cached)。如果基于AVC中的数据不能做出决定，则请求安全服务器，安全服务器在一个矩阵中查找“应用+文件”的安全环境。然后根据查询结果允许或拒绝访问，拒绝消息细节位于/var/log/messages中。

**3. SELinux伪文件系统**

*/selinux/*伪文件系统kernel子系统通常使用的命令，它类似于*/proc/*伪文件系统。系统管理员和用户不需要操作这部分。*/selinux/*目录举例如下：

**[cpp]** [view plain](http://blog.csdn.net/myarrow/article/details/9856095/) [copy](http://blog.csdn.net/myarrow/article/details/9856095/)

1. -rw-rw-rw-  1 root root 0 Sep 22 13:14 access
2. dr-xr-xr-x  1 root root 0 Sep 22 13:14 booleans
3. --w-------  1 root root 0 Sep 22 13:14 commit\_pending\_bools
4. -rw-rw-rw-  1 root root 0 Sep 22 13:14 context
5. -rw-rw-rw-  1 root root 0 Sep 22 13:14 create
6. --w-------  1 root root 0 Sep 22 13:14 disable
7. -rw-r--r--  1 root root 0 Sep 22 13:14 enforce
8. -rw-------  1 root root 0 Sep 22 13:14 load
9. -r--r--r--  1 root root 0 Sep 22 13:14 mls
10. -r--r--r--  1 root root 0 Sep 22 13:14 policyvers
11. -rw-rw-rw-  1 root root 0 Sep 22 13:14 relabel
12. -rw-rw-rw-  1 root root 0 Sep 22 13:14 user

   如cat enforce其值可能如下：

   1： enforcing mode

   0： permissive mode

**4. SELinux配置文件**

    SELinux配置文件(configuration)或策略文件(policy)位于*/etc/*目录下。

**4.1 /etc/sysconfig/selinux配置文件**

     /etc/sysconfig/selinux是一个符号链接，真正的配置文件为：*/etc/selinux/config*

     配置SELinux有如下两种方式：

      1) 使用配置工具：Security Level Configuration Tool (system-config-selinux)

      2) 编辑配置文件 (/etc/sysconfig/selinux).  
  
      /etc/sysconfig/selinux中包含如下配置选项：

     1) 打开或关闭SELinux

     2) 设置系统执行哪一个策略(policy)

     3) 设置系统如何执行策略(policy)

**4.2 配置文件选项**

**4.2.1 SELINUX**

        SELINUX=enforcing|permissive|disabled —定义SELinux的高级状态  
        • enforcing — The SELinux security policy is enforced.  
        • permissive — The SELinux system prints warnings but does not enforce policy.  
        • disabled — SELinux is fully disabled. SELinux hooks are disengaged from the kernel and the pseudo-file system is unregistered.

**4.2.2 SELINUXTYPE(安全策略)**

         SELINUXTYPE=targeted|strict — 指定SELinux执行哪一个策略  
         • targeted — 只有目标网络daemons保护。每个daemon是否执行策略，可通过system-config-selinux进行配置。保护常见的网络服务，为SELinux默认值。  
         可使用如下工具设置每个daemon的布尔值：

         1) getsebool -a： 列出SELinux的所有布尔值

         2) setsebool： 设置SELinux布尔值，如：setsebool -P dhcpd\_disable\_trans=0，-P表示即使用reboot之后，仍然有效。  
  
         • strict — 对SELinux执行完全的保护。为所有的subjects和objects定义安全环境，且每一个Action由策略执行服务器处理。提供符合Role-based-Access Control(RBAC)之policy，具备完整的保护功能，保护网络服务、一般指令及应用程序。

**4.2.3 SETLOCALDEFS**

         SETLOCALDEFS=0|1 — 控制如何设置本地定义(users and booleans)。

         • 1：这些定义由load\_policy控制，load\_policy来自于文件/etc/selinux/<policyname>

         • 0：由semanage控制

**4.3 /etc/selinux/目录**

      /etc/selinux/是存放所有策略文件和主要配置文件的目录。其例子如下：   

**[cpp]** [view plain](http://blog.csdn.net/myarrow/article/details/9856095/) [copy](http://blog.csdn.net/myarrow/article/details/9856095/)

1. -rw-r--r--  1 root root  448 Sep 22 17:34 config
2. drwxr-xr-x  5 root root 4096 Sep 22 17:27 strict
3. drwxr-xr-x  5 root root 4096 Sep 22 17:28 targeted

**5. SELinux工具**

1) /usr/sbin/setenforce — 修改SELinux运行模式，例子如下：

         • setenforce 1 — SELinux以强制(enforcing)模式运行  
         • setenforce 0 — SELinux以警告(permissive)模式运行

    为了关闭SELinux，你可以修改配置文件：/etc/selinux/config或/etc/sysconfig/selinux  
  
  
2) /usr/sbin/sestatus -v — 显示系统的详细状态，例子如下：

**[cpp]** [view plain](http://blog.csdn.net/myarrow/article/details/9856095/) [copy](http://blog.csdn.net/myarrow/article/details/9856095/)

1. SELinux status:                 enabled
2. SELinuxfs mount:                /selinux
3. Current mode:                   enforcing
4. Mode from config file:          enforcing
5. Policy version:                 21
6. Policy from config file:        targeted
8. Process contexts:
9. Current context:                user\_u:system\_r:unconfined\_t:s0
10. Init context:                   system\_u:system\_r:init\_t:s0
11. /sbin/mingetty                  system\_u:system\_r:getty\_t:s0

3) /usr/bin/newrole — 在一个新的context或role中运行一个新的shell

4) /sbin/restorecon — 通过为适当的文件或安全环境标记扩展属性，设置一个或多个文件的安全环境

5) /sbin/fixfiles — 检查或校正文件系统中的安全环境**[数据库](http://lib.csdn.net/base/mysql" \o "MySQL知识库" \t "_blank)**

6) getsebool — getsebool -a：查看所有布尔值

7) setsebool — 参数-P，永久性设置

8) chcon 修改文件、目录的安全上下文  
      chcon –u[user]  
      chcon –r[role]  
      chcon –t[type]   
      chcon –R  递归

**6. 类型强制的安全上下文(Type Enforcement Security Context)**

    安全上下文是一个简单的、一致的访问控制属性，在SELinux中，类型标识符是安全上下文的主要组成部分，由于历史原因，一个进程的类型通常被称为一个域（domain），"域"和"域类型"意思都一样，我们不必苛刻地去区分或避免使用术语域，通常，我们认为【域】、【域类型】、【主体类型】和【进程类型】都是同义的，即都是安全上下文中的“TYPE”。

    SELinux对系统中的许多命令做了修改，通过添加一个-Z选项显示客体和主体的安全上下文。

    1) 系统根据PAM子系统中的pam\_selinux.so模块设定登录者运行程序的安全上下文；  
    2) 文件的Security Contex规则如下：

        • rpm包安装的：会根据rpm包内记录来生成安全上下文；

        • 手动创建的文件：会根据policy中规定的来设置安全上下文；

        • cp：会重新生成安全上下文；

        • mv：安全上下文则不变。

    3) id -Z   
        显示了你的shell的安全上下文；  
    4) ps -Z

        检查进程的安全上下文；  
    5) ls -Z  
        检查文件、目录的安全上下文；

**6.1 安全上下文格式**

      所有操作系统访问控制都是以关联的客体和主体的某种类型的访问控制属性为基础的。在SELinux中，访问控制属性叫做安全上下文。所有客体（文件、进程间通讯通道、套接字、网络主机等）和主体（进程）都有与其关联的安全上下文，一个安全上下文由三部分组成：用户、角色和类型标识符。常常用下面的格式指定或显示安全上下文：

      USER：ROLE：TYPE[LEVEL[：CATEGORY]]

      安全上下文中的用户和角色标识符除了对强制有一点约束之外对类型强制访问控制策略没什么影响，对于进程，用户和角色标识符显得更有意义，因为它们是用于控制类型和用户标识符的联合体，这样就会与Linux用户账号关联起来；然而，对于客体，用户和角色标识符几乎很少使用，为了规范管理，客体的角色常常是object\_r，客体的用户常常是创建客体的进程的用户标识符，它们在访问控制上没什么作用。

      标准Linux安全中的用户ID和安全上下文中的用户标识符之间的区别，就技术而论，它们是正交标识符，分别用于标准的和安全增强的访问控制机制，这两者之间的任一相互关联都是通过登陆进程按照规范严格规定的，而不是通过SELinux策略直接强制实施的。

**6.1.1 USER**

         1) user identity：类似Linux系统中的UID，提供身份识别，用来记录身份；安全上下文的一部分；  
         2) 三种常见的 user:

              • user\_u ：普通用户登录系统后的预设；  
             • system\_u ：开机过程中系统进程的预设；  
             • root ：root 登录后的预设；

        3) 在 targeted policy 中 users 不是很重要；  
        4) 在strict policy 中比较重要，所有预设的 SELinux Users 都是以 “\_u” 结尾的，root 除外。

**6.1.2 ROLE**

        1) 文件、目录和设备的role：通常是 object\_r；  
        2) 程序的role：通常是 system\_r；  
        3) 用户的role：targeted policy为system\_r； strict policy为sysadm\_r、staff\_r、user\_r；用户的role，类似系统中的GID，不同角色具备不同的的权限；用户可以具备多个role；但是同一时间内只能使用一个role；

        4) 使用基于RBAC(Roles Based Access Control) 的strict和mls策略中，用来存储角色信息

**6.1.3 TYPE**

        1) type：用来将主体(subject)和客体(object)划分为不同的组，给每个主体和系统中的客体定义了一个类型；为进程运行提供最低的权限环境；  
        2) 当一个类型与执行中的进程相关联时，其type也称为domain；  
        3) type是SElinux security context 中最重要的部位，是 SELinux Type Enforcement 的心脏，预设值以\_t结尾；

        LEVEL和CATEGORY：定义层次和分类，只用于mls策略中  
             • LEVEL：代表安全等级,目前已经定义的安全等级为s0-s15,等级越来越高  
             • CATEGORY：代表分类，目前已经定义的分类为c0-c1023

**6.2 对比SELinux和标准Linux的访问控制属性**

      在标准Linux中，主体的访问控制属性是与进程通过在内核中的进程结构关联的真实有效的用户和组ID，这些属性通过内核利用大量工具进行保护，包括登陆进程和setuid程序，对于客体（如文件），文件的inode包括一套访问模式位、文件用户和组ID。以前的访问控制基于读/写/执行这三个控制位，文件所有者、文件所有者所属组、其他人各一套。

      在SELinux中，访问控制属性总是安全上下文三人组(用户：角色：类型)形式，所有客体和主体都有一个关联的安全上下文。需要特别指出的是，因为SELinux的主要访问控制特性是类型强制，安全上下文中的类型标识符决定了访问权。

      注意：SELinux是在标准Linux基础上增加了类型强制(TE: Type Enforcement)，这就意味着标准Linux和SELinux访问控制都必须满足先要能访问一个客体，例如：如果我们对某个文件有SELinux写入权限，但我们没有该文件的w许可，那么我们也不能写该文件。下表总结了标准Linux和SELinux之间访问控制属性的对比：

|  |  |  |
| --- | --- | --- |
|  | 标准Linux | SELInux |
| 进程安全属性 | 真实有效的用户和组ID | 安全上下文 |
| 客体安全属性 | 访问模式、文件用户和组ID | 安全上下文 |
| 访问控制基础 | 进程用户/组ID和文件的访问模式， 此访问模式基于文件的用户/组ID | 在进程类型和文件类型 之间允许的许可 |

**6.3 小结**

      1) 系统中每个文件、目录、网络端口等都被指定一个安全上下文，policy 则给出各安全上下文之间的作用规则。  
      2) SELinux根据policy及security context规则来决定存取行为是否可执行；  
      3) Subject(主体)：系统进程，比如/usr/sbin/httpd；  
      4) Object(客体)：被存取的项目，比如File、Directory、IP、Socket等；

**7. 类型强制(TE)访问控制**

     在SELinux中，所有访问都必须明确授权，SELinux默认不允许任何访问，不管Linux用户/组ID是什么。这就意味着在SELinux中，没有默认的超级用户了，与标准Linux中的root不一样，通过指定主体类型（即域）和客体类型使用allow规则授予访问权限，allow规则由四部分组成：

     • 源类型（Source type(s) ） 通常是尝试访问的进程的域类型  
     • 目标类型（Target type(s) ） 被进程访问的客体的类型  
     • 客体类别（Object class(es)） 指定允许访问的客体的类型  
     • 许可（Permission(s)） 象征目标类型允许源类型访问客体类型的访问种类  
     举例如下：

**[cpp]** [view plain](http://blog.csdn.net/myarrow/article/details/9856095/) [copy](http://blog.csdn.net/myarrow/article/details/9856095/)

1. allow user\_t bin\_t : file {read execute getattr};

     这个例子显示了TE allow规则的基础语法，这个规则包含了两个类型标识符：源类型(或主体类型或域)user\_t，目标类型(或客体类型)bin\_t。标识符file是定义在策略中的客体类别名称(在这里，表示一个普通的文件)，大括号中包括的许可是文件客体类别有效许可的一个子集，这个规则解释如下：

     拥有域类型user\_t的进程可以读/执行或获取具有bin\_t类型的文件客体的属性。

     SELinux allow规则如之前的例子在SELinux中实际上都是授予访问权的，真正的挑战是如何保证数以万计的访问正确授权，只授予必须的权限，实现尽可能的安全。

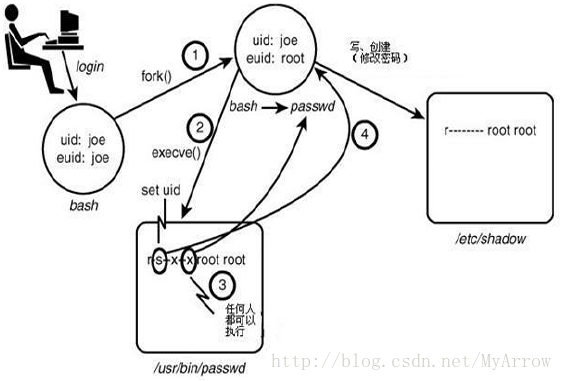
**7.1 标准Linux安全中的setuid程序**

      精通用户joe想安全地修改现有的密码问题，Linux解决这个问题的方法是通过给passwd赋一个setuid值，使其执行时具有root权限，如果你在一个普通Linux系统上列出密码文件，你看到的会是：

**[cpp]** [view plain](http://blog.csdn.net/myarrow/article/details/9856095/) [copy](http://blog.csdn.net/myarrow/article/details/9856095/)

1. # ls -l /usr/bin/passwd
2. -rwsr-xr-x. 1 root root 41292 Sep  7  2012 /usr/bin/passwd

       这里注意两件事，第一个是在所有者权限的x位置被设置为s了，这就是所谓的setuid位，意思是任何执行这个文件的进程，它的有效UID（即用户ID）将会被改为文件所有者。这里，root是文件所有者，因此当执行密码程序时实际上将会以root用户的ID运行。其执行过程如下图所示：



       从上面的分析中可以看出，passwd以root权限的身份运行， 它可以访问系统的任何资源，这给系统带来了安全问题，其实它只需要访问shadow及其相关的文件就可以了。而且shadow只需要接受passwd的访问即可。这在标准Linux中是无法做到的，而TE(类型强制)可实现此功能。

**8. 基于角色的访问控制**

    SELinux也提供了一种基于角色的访问控制（RBAC），SELinux的RBAC特性是依靠类型强制建立的，SELinux中的访问控制主要是通过类型实现的，角色基于进程安全上下文中的角色标识符限制进程可以转变的类型，如此，策略编写器可以创建一个角色，允许它转变为一套域类型（假设类型强制规则允许转变），从而定义角色的限制。

**9. SELinux中的多级安全(Multi-Level Security)**

    类型强制(Type Enforcement)无疑是SELinux引入的最重要的强制访问控制（MAC）机制，然而，在某些情况下，主要是保密控制应用程序的一个子集，传统的多级安全（MLS）MAC与类型强制一起使用显得更有价值，在这些情况下，SELinux总是包括某种格式的MLS功能，MLS特性是可选的，在SELinux的两个MAC机制中，它通常不是最重要的那个，对大多数安全应用程序而言，包括许多非保密数据应用程序，类型强制是最适合的安全增强的机制，尽管如此，MLS对部分应用程序还是增强了安全性。

     在大多数SELinux策略中，敏感度(s0，s1，...)和范畴(c0，c1，...)使用通配名，将它留给用户空间程序和程序库，以指定有意义的用户名。(例如：s0可能与UNCLASSIFIED 关联，s1可能与SECRET关联)  
     为了支持MLS，安全上下文被扩展了，包括了安全级别，如：

**[cpp]** [view plain](http://blog.csdn.net/myarrow/article/details/9856095/) [copy](http://blog.csdn.net/myarrow/article/details/9856095/)

1. user:role:type:sensitivity[:category,...] [-sensitivity[:category,...]]

     例子如下所示：

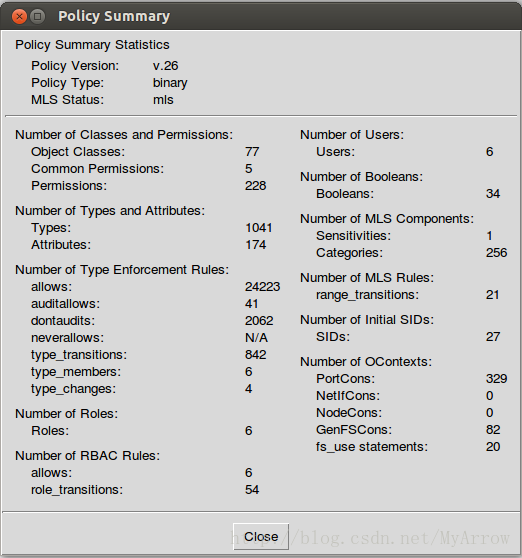
**[cpp]** [view plain](http://blog.csdn.net/myarrow/article/details/9856095/) [copy](http://blog.csdn.net/myarrow/article/details/9856095/)

1. root@luohj-**virtual**-machine:~# ps -aZ
2. LABEL                                        PID   TTY   TIME    CMD
3. unconfined\_u:system\_r:insmod\_t:s0-s0:c0.c255 4940 pts/0 00:00:00 passwd

     注意MLS安全上下文至少必须有一个安全级别（它由单个敏感度和0个或多个范畴组成），但可以包括两个安全级别，这两个安全级别分别被叫做低（或进程趋势）和高（或进程间隙），如果高安全级别丢失，它会被认为与低安全级别的值是相同的（最常见的情况），实际上，对于客体和进程而言，低和高安全级别通常都是相同的，通常用于进程的级别范围被认为是受信任的主体（即进程信任降级信息）或多层客体，如一个目录，它又包括了不同安全级别的客体。为了使描述简单，假设所有的进程和客体都只有一个安全级别。

**10. 策略分析工具apol**

      apol(即analyze policy【分析策略】)工具是一个成熟的SELinux策略分析工具，它位于setools工具包中。使用它打开policy.xx文件即可分析所有的相关策略。xx为策略编译器(checkpolicy)的版本号。



**11. 小结**

      SELinux访问控制是基于与所有系统资源(包括进程)关联的安全上下文的，安全上下文包括三个组件：用户、角色和类型标识符。类型标识符是访问控制的主要基础。

      在SELinux中，访问控制的主要特性是类型强制，在主体(即进程)与客体之间通过指定allow规则(主体的类型【也叫做域类型】是源，客体的类型是目标)进行访问授权，访问被授予特定的客体类别，为每个客体类别设置细粒度的许可。

      类型强制的一个关键优势是它可以控制哪个程序可能运行在给定的域类型上，因此，它允许对单个程序进行访问控制（比起用户级的安全控制要安全得多了），使程序进入另一个域（即以一个给定的进程类型运行）叫做域转变，它是通过SELinux的allow规则紧密控制的，SELinux也允许通过type\_transition 文件使域转变自动发生。

      SELinux在访问控制安全上下文中不直接使用角色标识符，相反，所有的访问都是基于类型的，角色用于关联允许的域类型，这样可以设置类型强制允许的功能组合到一起，将用户作为一个角色进行认证。

      SELinux提供了一个可选的MLS访问控制机制，它提供了更多的访问限制，MLS特性依靠TE机制建立起来的，MLS扩展了安全上下文的内容，包括了一个当前的（或低）安全级别和一个可选的高安全级别。

参考：<http://www.centos.org/docs/5/html/Deployment_Guide-en-US/selg-overview.html>

<http://wenku.baidu.com/view/df89fe235901020207409c49.html>

<http://wenku.baidu.com/view/fed9abb569dc5022aaea002d.html>

[http://selinux.sourceforge.net](http://selinux.sourceforge.net/) Homepage for the SELinux community.

<http://www.nsa.gov/selinux/> Homepage for the NSA SELinux development team. Many resources are available in HTML and PDF formats. Although many of these links are not SELinux specific, some concepts may apply.

<http://fedora.redhat.com/docs/> Homepage for the Fedora documentation project, which contains Fedora Core specific materials that may be more timely, since the release cycle is much shorter.

<http://wenku.baidu.com/view/4d26594fc850ad02de804189.html>

原文地址http://blog.csdn.net/myarrow/article/details/9856095/