CentOS 丛书目录 — 系统管理 — 网络服务 — 应用部署

Xinetd 和 TCP Wrapper

内容提要

1. 掌握 xinetd 及其配置
2. 掌握 TCP Wrapper 及其配置

扩展网络守护进程 xinetd

xinetd 是新一代的网络守护进程服务程序，提供类似于早期的 inetd+tcp_wrappers 的功能，与之相比 xinetd 更加强大和安全。

xinetd 具有如下的功能：

- 支持对tcp、ucp、RPC服务
- 基于时间段的访问控制
- 功能完备的log功能，即可以记录连接成功也可以记录连接失败的行为
- 能有效的防止DoS攻击（Denial of Services）
- 能限制同时运行的同一类型的服务器数目
- 能限制启动的所有服务器数目
- 能限制log文件大小
- 将某个服务绑定在特定的系统接口上，从而能实现只允许私有网络访问某项服务
- 能实现作为其他系统的代理。如果和IP伪装结合可以实现对内部私有网络的访问

在 CentOS 中 xinetd 是默认被安装的。

xinetd 的配置文件

xinetd 的主配置文件是 /etc/xinetd.conf

```
#
# This is the master xinetd configuration file. Settings in the
# default section will be inherited by all service configurations
# unless explicitly overridden in the service configuration. See
# xinetd.conf in the man pages for a more detailed explanation of
# these attributes.

defaults
{
# The next two items are intended to be a quick access place to
# temporarily enable or disable services.
#
#       enabled         =
#       disabled        =

# Define general logging characteristics.
        log_type        = SYSLOG daemon info
        log_on_failure  = HOST
        log_on_success  = PID HOST DURATION EXIT

# Define access restriction defaults
#
```

```
#       no_access       =
#       only_from       =
#       max_load        = 0
        cps             = 50 10
        instances       = 50
        per_source      = 10

# Address and networking defaults
#
#       bind            =
#       mdns            = yes
        v6only          = no

# setup environmental attributes
#
#       passenv         =
        groups          = yes
        umask           = 002

# Generally, banners are not used. This sets up their global defaults
#
#       banner          =
#       banner_fail     =
#       banner_success  =
}

includedir /etc/xinetd.d
```

最后一行包含了 /etc/xinetd.d 目录下的所有配置文件。

通常，若由 xinetd 监控一项服务，就要在 /etc/xinetd.d 目录下有一个和服务同名的配置文件。

/etc/xinetd.d 目录下的每个文件几乎具有相同的格式：

```
service service-name
{
<attribute>  <operator>      <value> <value> ……
}
```

其中：

- service 是必需的关键字，且属性表必须用大括号括起来
- 每一项都定义了由 service-name 定义的服务。service-name 是任意的，但通常是标准网络服务名，也可增加其他非标准的服务，只要它们能通过网络请求激活，包括 localhost 自身发出的网络请求
- 有很多可以使用的属性（attribute）
- 操作符（operator）可以是 =，+=，或 -=
  - 所有属性可以使用 =，其作用是分配一个或多个值
  - 某些属性可以使用 += 或 -= 的形式，其作用分别是将其值增加到某个现存的值表中，或将其值从现存的值表中删除
- 值（value）是为给定属性设置的参数

xinetd 属性的含义：

属性	含义
disable	由两个值"yes"和"no"，用于设置xinetd是否监控此项服务
socket_type	服务使用的套接子类型，有stream、dgram、raw和seqpacket四个值
protocol	服务使用的协议类型，协议必须是在/etc/protocol文件中定义过的
wait	设置一项服务是以单线程还是多线程的方式运行。如果是yes，则以单线程的方式运行，此时xinetd在启动此服务和客户端建立起一个连接后，将不会再接受新的对此服务的连接请求，直到这个连接结束。如果是no，则以多线程方式运行，xinetd会在为服务建立起一个连接后，继续处理新的连接请求

user	设置运行此服务的用户
instances	设置一项服务能够同时提供的服务数量
per_source	设置每个客户机的最大连接数
server	运行此服务的可执行程序的完整路径名称
server_args	服务的可执行程序的参数
only_from	设置可以访问此服务的客户地址，可以指定主机名称、IP地址、网络地址/子网掩码
no_access	设置不允许访问此服务的客户地址。如果此属性和only_from属性都没有定义，则默认允许任何客户访问服务；如果都定义了，则能更精确地确定客户机地址的属性设置将起作用。例如：在only_from中设置了192.168.，而在no_access种设置了192.168.1.2，则将禁止192.168.1.2访问服务
access_times	设置一项服务可以被访问的时间段。用hour:min-hour:min的形式来指定时间段，hour的范围是0-23，min的范围是0-59
log_type	设置日志记录的文件。有两种形式，SYSLOG（记录到系统日志中）和FILE（记录到指定的文件中
log_on_success	设置如果一个连接成功，在日志中记录哪些内容。有以下几个选项：PID（连接进程的PID），HOST（远程主机地址），USERID（远程用户ID），DURATION（服务会话的持续时间）
log_on_failure	设置如果一个连接失败，在日志中记录哪些内容
redirect	把一个基于tcp协议的服务重定向到另一台主机，此服务器将在客户和另一台主机之间转发数据包。此属性的值是重定向的IP地址和端口。这个设置可用于建立Internet上的客户与内部网络的服务器的连接
bind	把一项服务绑定到一个特定的网络接口上，此属性的值为相应网络接口的IP地址
cps	对连接频率的限制。有两个值，第一个是每秒钟的连接次数，如果超过，则此服务将会被暂时禁用，第二个是禁用此服务的时间，单位是秒

xinetd 配置举例

下面举两个使用 xinetd 实现访问控制配置的例子。

例**1**:

```
$  cat /etc/xinetd.d/telnet
```

```
service telnet
{
      socket_type = stream
      wait= no
      user= root
      server= /usr/sbin/in.telnetd
      only_from = 202.38.85.0/24
      redirect = 192.168.1.15 23
      log_on_success += DURATION HOST USERID
      cps=10 300
}
```

其中:

- 属性 only_from 设置了只允许 202.38.85.0/24 网段内的客户进行 telnet 连接;
- 属性 redirect 把连接重定向到 192.168.1.15 主机的 23 端口;
- 属性 log_on_success 设置了将对每一个成功的连接进行日志记录，包括连接的时间，远程主机名和用户的ID;
- 属性 cps 设置了每秒钟对 telnet 的连接数不能超过 10次，否则将禁用 5分钟。

例**2**:

```
$  cat /etc/xinetd.d/ftp
```

```
service ftp
```

```
{
      socket_type  = stream
      wait = no
      user = root
      server = /usr/sbin/in.ftpd
      server_args = -l
      instances = 10
      per_source = 1
      access_times = 7:00-12:30 13:30-21:00
      only_from = 192.168.1.0/24
}
```

其中:

- 属性 wait 设置为 no，因此 ftp 服务将以多线程的方式运行;
- 属性 instances 设置了最多可以同时建立 10个 ftp 连接;
- 属性 per_source 设置了每个客户机的最大连接数为 1;
- 属性 access_times 设置了每天只有在 7:00-12:30 和 13:30-21:00 两个时间段里允许进行 ftp 连接;
- 属性 only_from 设置了 ftp 服务只对内部网络开放。

TCP Wrappers

TCP Wrapper 是一个应用层的访问控制程序。tcpd 检查收到的网络连接请求并将其与配置文件做比较来决定是否允许其通过。若连接被允许，将运行真实的服务程序（如: in.telnetd 等），若不允许，将丢弃此请求连接的数据包。

TCP Wrappers 的守护进程是 tcpd，一般 tcpd 是默认安装的。

访问控制配置文件和访问控制规则

TCP_wrappers 使用 /etc/hosts.allow 和 /etc/hosts.deny 两个配置文件实现访问控制。

- /etc/hosts.allow 是一个许可表
- /etc/hosts.deny 是一个拒绝表

访问控制规则如下:

1. tcpd 查找 /etc/hosts.allow 和 /etc/hosts.deny，若没有这两个配置文件，或两个配置文件的内容均为空，则允许所有的访问
2. tcpd 若发现这两个文件，首先读取 /etc/hosts.allow ，然后再读取 /etc/hosts.deny，一旦 tcpd 在查询中发现主机与服务相匹配就会终止
   I. 若在 /etc/hosts.allow 中发现主机与服务相匹配，则允许访问并终止规则查询
   II. 若在 /etc/hosts.deny 中发现主机与服务相匹配，则禁止访问并终止规则查询
3. 若在两个文件中均未查找到匹配的项目，则允许访问

访问控制配置文件的基本语法

/etc/hosts.allow 和 /etc/hosts.deny 的基本语法:

- "#"开头的行为注释行
- 可以在行中使用"\"作为续行符
- 其他的规则行均使用如下的语法

```
daemon_list : client_list [ : shell_command ]
```

/etc/hosts.allow 和 /etc/hosts.deny 的基本语法：

- daemon_list — 用逗号间隔的服务列表
  - ALL 表示所有的服务
  - daemon@host 限定某网络接口的服务，用于多宿主机
- client_list — 用逗号间隔的主机表
  - ALL 表示所有的主机
  - 使用 IP 地址，如 12.23.34.45
  - 使用主机名 如 www.abc.com [http://www.proxyserve.net/index.php?q=aHR0cDovL3d3dy5hYmMuY29t]
  - 使用域名，如 .abc.com
  - 使用IP地址段 如 12.23.
  - 使用CIDR 如 192.168.0.0/22
  - 使用client@host 如 osmond@www.abc.com [http://www.proxyserve.net/index.php?q=aHR0cDovL3d3dy5hYmMuY29t]
- shell_command — 执行 Shell 命令，为可选项

在 daemon_list 和 client_list 中还可以使用 EXCEPT 操作符，语法为：

```
list_1 EXCEPT list_2
```

例如：

- All EXCEPT in.ftpd 表示除了 in.ftpd 之外的所有服务
- .foobar.edu EXCEPT terminalserver.foobar.edu 表示 foobar.edu 域中除了 terminalserver.foobar.edu 之外的所有主机

参考

- 可以使用 **man hosts_access** 命令查看 /etc/hosts.allow 和 /etc/hosts.deny 基本语法的详细信息

访问控制配置文件的扩展语法

/etc/hosts.allow 和 /etc/hosts.deny 的扩展语法：

```
daemon_list : client_list : option : option ...
```

其中 daemon_list 与 client_list 的语法与基本语法一致。

/etc/hosts.allow 和 /etc/hosts.deny 的常用扩展选项：

- deny — 拒绝被请求的服务，该选项必须出现在规则的结尾处
- spawn shell_command — 以子进程执行 Shell 命令
- twist shell_command — 执行 Shell 命令，而不是被请求的服务
- setenv name value — 设置执行被请求的服务时的环境变量

扩展语法支持 twist 选项，下面是一个例子：

```
in.ftpd : ALL : twist echo 421 FTP not allowed from %h : deny
```

原本 tcpd 在接收到 ftp 服务请求时将启动 in.ftpd，但由于此处使用了 twist 选项，因此反而执行 echo 命令来显示拒绝信息。

由于扩展语法中支持 deny 选项，因此使用扩展语法时只需配置 /etc/hosts.allow 一个文件。

参考

- 可以使用 **man hosts_options** 命令查看 /etc/hosts.allow 和 /etc/hosts.deny 扩展语法的详细信息
- 显示源文件
- 登录