CentOS 7.3 Xinetd服务的安装与配置

[](http://blog.51cto.com/13525470)

[小左先森](http://blog.51cto.com/13525470)2人评论[4003人阅读2018-01-14 15:54:06](javascript:;)

**一、Linux守护进程与初始化进程**

**1. 什么是守护进程**

Linux服务器的主要任务就是为本地或远程用户提供各种服务。通常Linux系统上提供服务的程序是由运行在后台的守护进程（Daemon）来执行。一个实际运行中的Linux系统一般会有多个这样的程序在运行。这些后台守护进程在系统开机后就运行了，并且在时刻地监听前台客户地服务请求，一旦客户发出了服务请求，守护进程便为它们提供服务。Windows系统中的守护进程被称为“服务”。

按照服务类型，守护进程可以分为如下两类：

        系统守护进程：如crond(周期任务)、rsyslogd(日志服务)、cpus等；

        网络守护进程：如sshd、httpd、xinetd（托管）等。

**2. 什么是守护进程**

系统初始化进程是一个特殊的的守护进程，其PID为1，它是所有其他守护进程的父进程或者祖先进程。也就是说，系统上所有的守护进程都是由系统初始化进程进行管理的（如启动、停止等）。

在Linux的发展历史过程中，使用过3种Linux初始化系统。

**SysVinit**

        为 UNIX System V 系统创建的；

        RHEL/CentOS 5及之前的版本一直使用。

**Upstart**

        由Ubuntu创建的；

        RHEL/CentOS 6 使用Upstart。

**Systemd**

        先进的初始化系统；

        RHEL/CentOS 7使用Systemd。

**二、Linux独立启动和超级守护进程**

如果用两个比喻来形容两类守护进程的话，一般会用银行的业务处理窗口来类比：

**独立启动守护进程**：银行里有一种单服务的窗口，像取钱，存钱等窗口，这些窗口边上始终会坐着一个人，如果有人来取钱或存钱，可以直接到相应的窗口去办理，这个处理单一服务的始终存在的人就是独立启动的守护进程。

**超级守护进程**：银行里还有一种窗口，提供综合服务，像汇款，转账，提款等业务；这种窗口附近也始终坐着一个人（xinet），她可能不提供具体的服务，提供具体服务的人在里面闲着聊天啊，喝茶啊，但是当有人来汇款时他会大声喊一句，小王，有人汇款啦，然后里面管汇款的小王会立马跑过来帮忙办完汇款业务。其他的人继续聊天，喝茶。这些负责具体业务的人我们就称之为超级守护进程。当然可能汇款人会有一些规则，可能不能往北京汇款，他就会提早告诉xinet，所以如果有人来汇款想汇往北京的话，管理员就直接告诉他这个我们这里办不到的，于是就根本不会去喊汇款员了，相当于提供了一层管理机制。

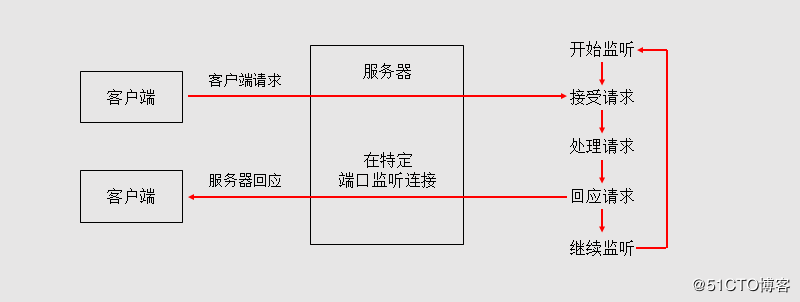
针对这种窗口还存在多线程和单线程的区别：

多线程：将所有用户的要求都提上来，里面的人都别闲着了，都一起干活吧；

单线程：大家都排好队了，一个一个来，里面的人同一时间只有一个人在工作。  
  
**三、Linux守护进程运行方式**

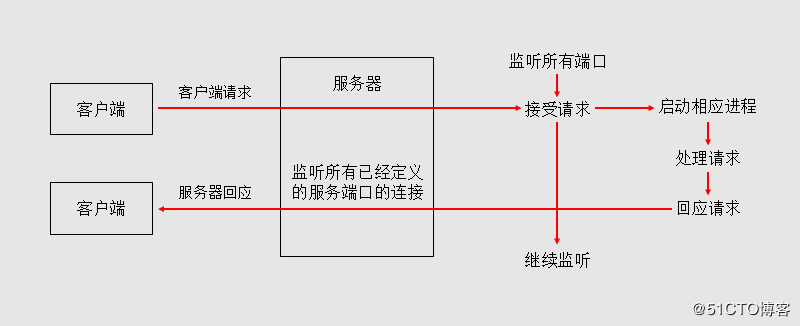
**1. 独立运行（stand-alone）的守护进程**

独立运行的守护进程由init脚本负责管理，所有独立运行的守护进程的脚本在/etc/rc.d/init.d/目录下。系统服务都是独立运行的守护进程，包括syslogd和cron等。独立运行的守护进程的工作方式称做stand-alone，它是UNIX传统的C/S模式的访问模式。



**2．xinetd模式运行独立的守护进程**

从守护进程的概念可以看出，对于系统所要通过的每一种服务，都必须运行一个监听某个端口连接所发生的守护进程，这意味着资源浪费。为了解决这个问题，Linux引进了"网络守护进程服务程序"的概念。也就是xinted（extended internet daemon）。xinetd能够同时监听多个指定的端口，在接受用户请求时，它能够根据用户请求的端口的不同，启动不同的网络服务进程来处理这些用户请求。可以把xinetd看做一个管理启动服务的管理服务器，它决定把一个客户请求交给哪个程序处理，然后启动相应的守护进程。xinetd无时不在运行并监听它所管理的所有端口上的服务。当某个要连接它管理的某项服务的请求到达时，xinetd就会为该服务启动合适的服务器。



**四、Xinetd托管服务简介**

Xinetd 托管服务。又名：超级守护进程，可以把一些小服务放到xinetd里进行托管。拖管后的好处就是可以使用xinetd强大的参数来控制这些服务，并且增强安全性。（比如一个小服务没有一些控制功能，但支持xinetd拖管，你就可以拖管并使用xinetd的参数来控制它)。

Xinetd提供类似于inetd + TCP Wrappers的功能，但是更加强大和安全。后面xinetd已经取代了inetd，并且提供了访问控制、加强的日志和资源管理功能。

TCP Wrappers是一个应用层的访问控制程序，其原理是在服务器向外提供的TCP服务上包裹一层安全检测机制。外来的连接请求首先要通过这层安全检测，获得认证之后才能被系统服务接收。TCP Wrappers的功能有两种实现方式：一种是由tcpd守护进程实现的，常被用于inetd + TCP Wrappers的系统中（如FreeBSD等）；另一种是通过每种服务程序调用libwrap.so链接库实现的，即libwrap.so库支持的网络服务程序都能使用TCP Wrappers来实现访问控制，常用于xinetd + TCP Wrappers的系统中（如CentOS等）。

在CentOS中，TCP Wrappers一般是默认安装的，若未安装成功，可以使用如下命令安装：

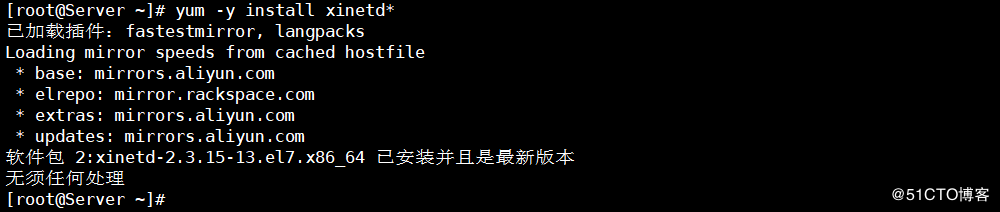
|  |
| --- |
| #  yum  –y  install  tcp\_wrappers                 --安装tcp\_wrappers |

**五、Xinetd 服务的搭建与配置**

**1. 配置xinetd的方法**

安装xinetd服务

|  |
| --- |
| #  yum  -y  install  xinetd\*                            --使用yum安装 |



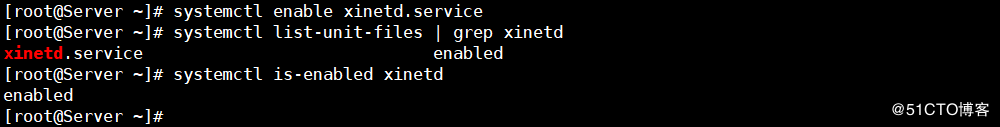
xinetd服务的主配置文件： /etc/xinetd.conf     --保持默认即可

用于存放被托管的服务的目录：/etc/xinetd.d/

|  |
| --- |
| #  systemctl  enable  xinetd.service             --开机启用xinetd服务 |

查看服务是否开机启动：

|  |
| --- |
| #  systemctl  list-unit-files                                --列出各种服务开机是否启动  #  systemctl  list-unit-files  |  grep  xinetd       --列出xinetd服务是否开机启动  #  systemctl  is-enabled  xinetd                       --列出xinetd服务是否开机启动 |



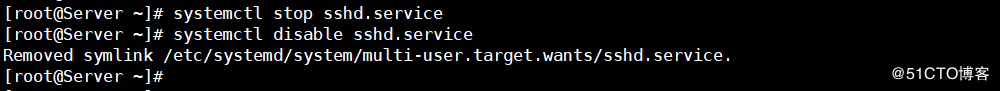
**2. 配置实例**

**例1：以sshd为例，把sshd拖管到xinetd下**

sshd服务也有配置文件，为/etc/ssh/sshd\_config；但此配置文件功能有限，可以选择拖管sshd服务，来实现额外的功能。

托管前先关闭ssh本身提供的服务

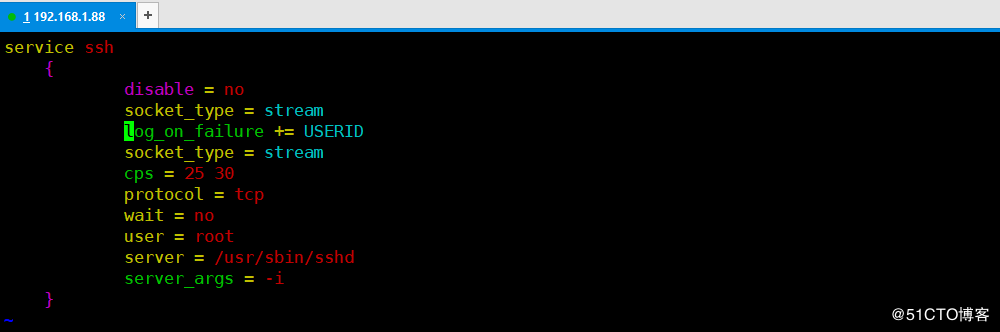
|  |
| --- |
| #  systemctl  stop  sshd.service                    --关闭ssh服务 #  systemctl  disable  sshd.service                --设置为开机不启动（注意：如果你关闭xinetd之后，需要把ssh服务重新启动，不然你不能通过ssh远程连接） |



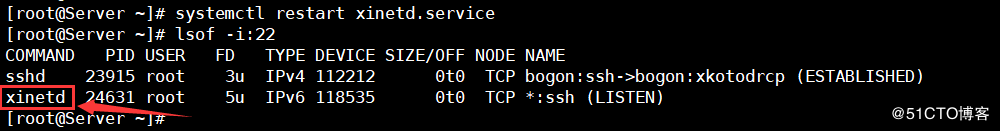
下面开始配置ssh被xinetd托管

|  |
| --- |
| #  vim  /etc/xinetd.d/ssh                             --默认是没有这个文件的，需要创建 |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | service ssh                             # 代表被托管服务的名称  {          disable = no                    # 是否禁用托管服务，no表示开启托管服务          log\_on\_failure += USERID        # 设置失败时，UID添加到系统登记表          socket\_type = stream            # socket连接方式，这个是属于本地方式，对ssh无效          server\_args = --daemon          # 设置服务启动时需要的参数          cps = 25 30                     # 每秒25个入站连接，如果超过限制，则等待30秒。主要用于对付拒绝服务\*\*\*          protocol = tcp                  # 代表ssh走的是tcp协议连接          wait = no                       # 是否并发，这个参数对ssh无效          user = root                     # 以什么用户进行启动          server = /usr/sbin/sshd         # 被托管服务的启动脚本          server\_args = -i                # 启动脚本的参数  } |



|  |
| --- |
| #  systemctl  restart  xinetd.service    --重新启动xinetd服务  #  lsof  -i:22                                        --ssh的默认端口为22,可以通过/etc/services查看各个服务的端口    --查看22端口的进程，由sshd变为了xinetd就表示你拖管成功了 |

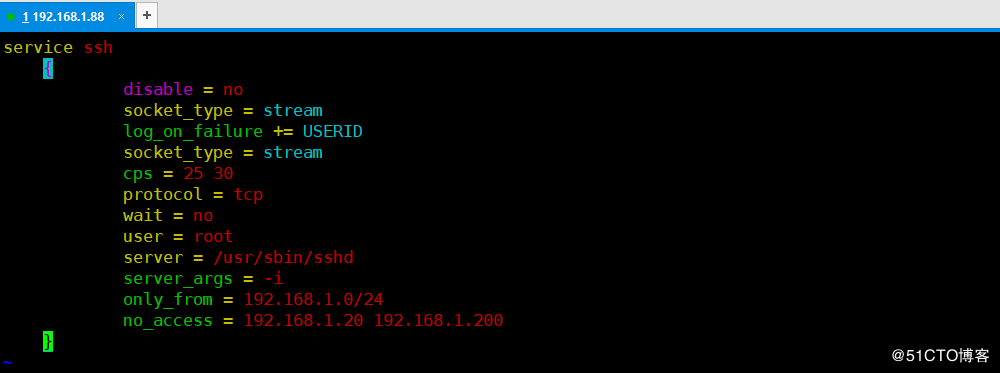


**例2：（上面的例子并没有看出拖管与不拖管的区别）**

下面在例1 的基础上加上对IP或网段的访问控制功能，这个是ssh服务本身不具备的功能（除非写iptables)

|  |
| --- |
| #  vim  /etc/xinetd.d/ssh |

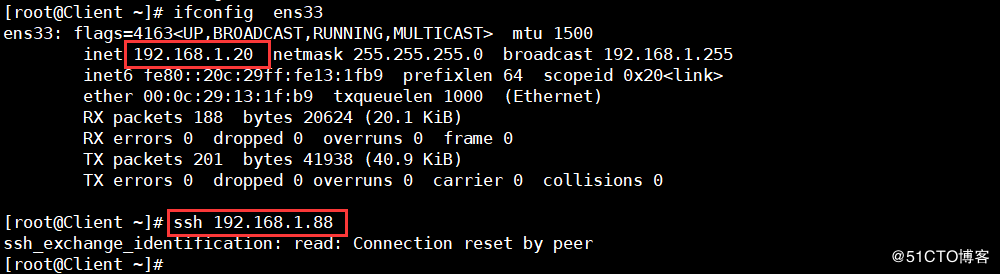
|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | service ssh  {          disable = no          log\_on\_failure += USERID          socket\_type = stream          server\_args = --daemon          cps = 25 30          protocol = tcp          wait = no          user = root          server = /usr/sbin/sshd          server\_args = -i          only\_from = 192.168.1.0/24              # 表示允许1网段访问          no\_access = 192.168.1.20 192.168.1.200  # 表示只能1网段访问，但1网段里的20和200这两IP不能访问  } |



|  |
| --- |
| #  systemctl  restart  xinetd.service    --重新启动xinetd服务 |

修改后重新启动xinetd服务再去客户端进行测试

|  |
| --- |
| #  ifconfig   ens33                              --查看客户端IP  #  ssh  192.168.1.88                           --ssh连接服务器 |



IP为192.168.1.88的拒绝客户端远程连接。因为在服务器里面做了xinetd托管设置：192.168.1.20的IP不能连接服务器。

**例3：在例2 的基础再加一些功能(man xinetd.conf)**

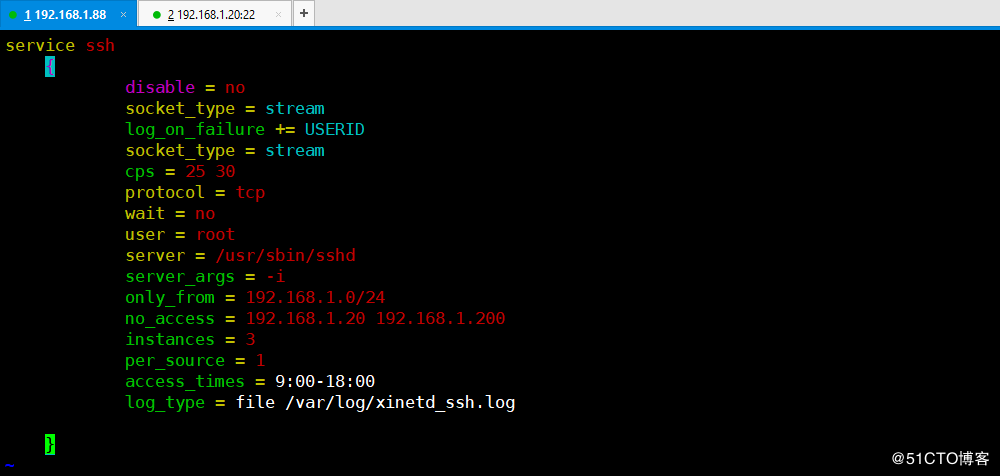
    1、控制这个服务最多只能3个连接，每个源IP只能1个连接；

    2、控制只能9：00到18：00才能ssh连接；

    3、指定日志记录到/var/log/xinetd\_ssh.log里。

|  |
| --- |
| #  vim  /etc/xinetd.d/ssh |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | service ssh  {          disable = no          log\_on\_failure += USERID          socket\_type = stream          server\_args = --daemon          cps = 25 30          protocol = tcp          wait = no          user = root          server = /usr/sbin/sshd          server\_args = -i          only\_from = 192.168.1.0/24          no\_access = 192.168.1.20 192.168.1.200          instances = 3                                   # 最大连接数为3          per\_source = 1                                  # 每个源IP只能有1个连接          access\_times = 9:00-18:00                       # 只能9：00到18：00才能ssh连接          log\_type = file /var/log/xinetd\_ssh.log         # 指定日志记录到/var/log/xinetd\_ssh.log里  } |



|  |
| --- |
| #  systemctl  restart  xinetd.service    --重新启动xinetd服务 |

修改好以后大家在客户端进行测试，修改时间，查看log\_type指定的路径下是否有xinetd\_ssh.log。这里就不一一举列验证了。

**例4：修改ssh服务的连接端口**

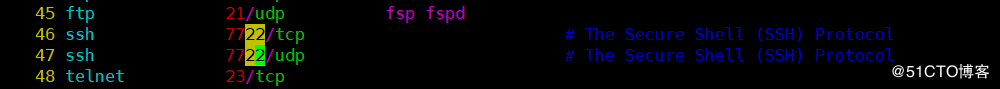
**（1）修改ssh服务的连接端口**

|  |
| --- |
| #  vim  /etc/xinetd.d/ssh |

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | service ssh  {          disable = no          log\_on\_failure += USERID          socket\_type = stream          server\_args = --daemon          cps = 25 30          protocol = tcp          wait = no          user = root          server = /usr/sbin/sshd          server\_args = -i          only\_from = 192.168.1.0/24          no\_access = 192.168.1.20 192.168.1.200          instances = 3          per\_source = 1          access\_times = 9:00-18:00          log\_type = file /var/log/xinetd\_ssh.log          port = 7722                             # 指定ssh的连接端口为7722  } |

**（2）将ssh服务的端口改为7722**

|  |
| --- |
| #  vim  /etc/services                                                                                                   --修改ssh服务端口    46 ssh             7722/tcp                          # The Secure Shell (SSH) Protocol          --将ssh的tcp端口改为7722     47 ssh             7722/udp                          # The Secure Shell (SSH) Protocol         --将ssh的udp端口改为7722 |



**注意：**这个端口7722不能和其他服务的端口相同，可以在/etc/services文件中查找是否有只有ssh的端口为7722。

端口改了以后，客户端访问是要使用ssh 192.168.1.88  -p  7722，但我自己连自己不用加-p 7722，因为我默认就是用7722来连接。这也就是说/etc/services里的端口也决定了你做为客户端去访问别人的默认端口。

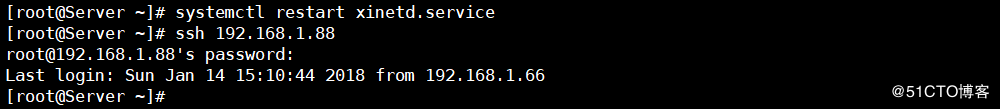
重启xinetd服务后通过lsof  -i:7722来查看ssh是否被托管

**（3）配置文件改动后需要重启xinetd服务**

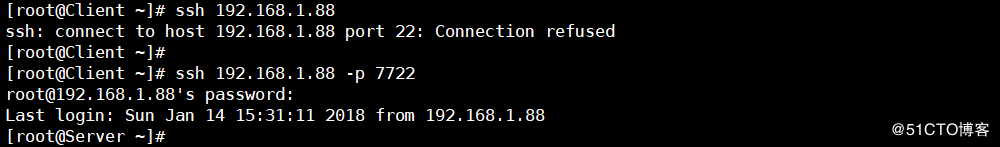
|  |
| --- |
| #  systemctl  restart  xinetd.service              --重新启动xinetd服务 |

**（4）验证以上配置的参数是否生效**

服务器：自己连接自己，不需加端口号

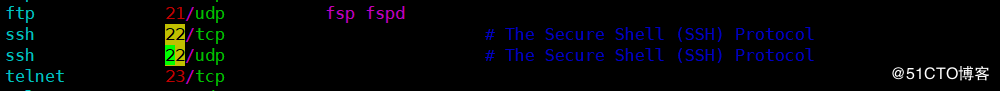


客户端：需要加端口号



**（5）测试完后将端口改回22**

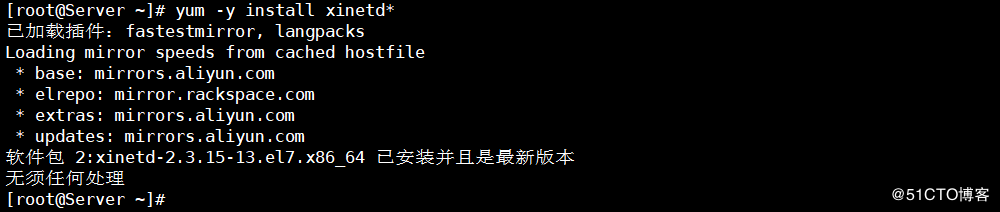
|  |
| --- |
| #  vim  /etc/services                                                                                               --修改ssh服务端口    46 ssh             22/tcp                          # The Secure Shell (SSH) Protocol          --将ssh的tcp端口改回22     47 ssh             22/udp                          # The Secure Shell (SSH) Protocol         --将ssh的udp端口改回22  #  systemctl  restart  xinetd.service                                                                       --重新启动xinetd服务 |



**六、Xinetd 时间同步**

由于硬件的原因，机器或多或少的根标准时间对不上，一个月的误差几秒到几分钟不等。对于服务器来说时间不准，会有很多麻烦。那么我们就使用Xinetd搭建时间同步服务器。

|  |
| --- |
| #  yum  -y  install  xinetd\*                            --使用yum安装 |



|  |
| --- |
| #  vim  /etc/xinetd.d/time-dgram        --udp   6         disable         = no                                --将yes改为no |

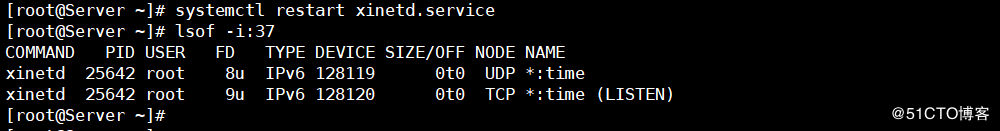
image.png

|  |
| --- |
| #  vim  /etc/xinetd.d/time-stream        --tcp   6         disable         = no                                --将yes改为no |

image.png

|  |
| --- |
| #  systemctl  restart  xinetd.service                 --重新启动xinetd服务 |

|  |
| --- |
| #  lsof  -i:37 |



客户端要同步服务器时间，用下面的命令就可以了

|  |
| --- |
| #  rdate  -s  192.168.1.88                                --这里写服务器的IP |

总结：如果你想实现你公司服务器的所有时间与北京时间一致。可以用一台能上外网的服务器去同步公网上的时间服务器，然后把这台也配置成时间服务器，其它的机器通过内网定时同步就可以了。

©著作权归作者所有：来自51CTO博客作者小左先森的原创作品，如需转载，请注明出处，否则将追究法律责任