

II. 系统管理

第 6 章 配置与调整

(翻译中出现的任何问题或错误，请广大读者及时反馈给我：freebsdhandbook@163.com)

6.1 概要

正确地配置系统能充分地减少以后维护和升级系统所需的工作量。本章将描述一些 FreeBSD 系统管理配置的情况。本章也会描述一些用来调整 FreeBSD 系统性能的参数。

读完本章你将了解到这些：

- 为什么和如何在你的硬盘上分配空间，规划和放置文件系统和交换分区。
- `rc.conf` 配置文件和 `/usr/local/etc/rc.d` 启动系统的基础。
- 如何在你网络设备上分配虚拟主机。
- 如何在 `/etc` 上使用不同的配置。
- 如何使用 `sysctl` 变量调节 FreeBSD。
- 如何调整磁盘的性能和修改内核的限制。

在阅读本章之前你应当：

- 了解 UNIX 和 FreeBSD 的基础知识（第 3 章）。
- 熟悉如何与 FreeBSD 的源文件保持同步。
- 内核配置和编译的基础（第 9 章）。

6.2 初步配置

6.2.1 分区设计

6.2.1.1 基础分区

当使用 `disklabel` 或 `sysinstall` 命令划分你的文件系统时，需要记住硬盘驱动器从外磁道传输数据要比从内磁道传输数据来得快，这一点很重要。记住这点，你可以把比较小的，

经常访问的如 `root` 和 `swap` 放在靠近外磁道的地方。可以把比较大的分区如 `/usr` 放在内磁道上。这样做，按照相同的顺序创建分区：`root`, `swap`, `/var`, `/usr`。

你的 `/var` 分区的大小能反映你机器的使用状况。`/var` 主要用来存放：邮箱，打印 spool 和日志文件。邮箱和日志文件可能会达到一个无法预料的数量，这主要取决于在你的系统上有多少用户和你的日志文件可以保存多长时间。如果你想要运行一个邮件服务器，一个超过 G 数量级的 `/var` 分区是必要的。另外，`/var/tmp` 要足够大，以便于能够包含足够的你可能会添加的 packages。

`/usr` 分区保存了支持系统所需的文件和一个叫做 `/usr/local` 的用来保存从 ports 安装文件的子目录。如果你不使用 ports 也不希望把系统源代码保存在机器上，你可以节省超过 1G 的 `/usr` 分区。如果你安装许多 ports，我们建议至少要为 `/usr` 保留 2G 的空间，如果你也想把系统源代码保存在你的机器上，我们建议为 `/usr` 保留 3G 的空间。不要低估了这个分区所需要的空间的大小，它可能会不断增加，让你非常吃惊！当你调整分区的大小时，记住你的系统可能会不断增加对空间的需求。

注意：一些使用 `sysinstall` 的自动默认分区的用户会发现，一段时间以后他们的 `root` 或 `/var` 分区会变得很小。建议尽可能把分区分的大一些。

6.2.1.2 Swap 分区

通常你的交换分区应当是主内存的两倍。例如，如果机器有 128 M 的内存，交换分区应当是 256MB。带有比较少内存的机器可以通过增加交换分区来提高机器的性能。我们并不建议你配置少于 256MB 的 SWAP 分区，你应当记住将来随着内存的扩充，你的 swap 分区也要相应地扩大。当 swap 分区至少是主内存的两倍时，内核的 VM 页面调度程序将被用来调节系统到最佳状态。如果你给你的机器添加更多的内存，配置太少的 swap，会导致在 VM 页面扫描代码时变得效率低下。

最后，在配置有很多 SCSI（或 IDE）磁盘的较大系统中，我们强烈建议你给每一个硬盘都创建一个 SWAP。Swap 分区应当拥有同样的大小。内核可能会处理成任意大小，但内部数据结构则是最大 swap 分区的 4 倍。保持 swap 分区同样的大小，可以允许内核最佳地调度 swap 空间来访问磁盘。不要为此过分担心，交换空间正是 UNIX 系统的长处。

6.2.1.3 为什么要分区？

为什么要分区？为什么不创建一个大的根分区？我并不介意大小问题！有很多原因证实这不是一个好主意。首先，每个分区有不同的操作特性，把它们分开可以允许文件系统去自动调节自己以适应那些特性。例如，根和 */usr* 分区通常是读得比较频繁，写得比较少，而象 */var* 和 */var/tmp* 则读写都比较频繁。

6.3 核心配置

负责系统配置信息的主要是 */etc/rc.conf*。这个文件包含了配置文件很宽的范围。在系统启动时主要被用来配置系统。它的名字直接表明了这点；配置信息一般是 *rc** 这样类型的文件。一个系统管理员应当在 *rc.conf* 文件中建立一个记录来修改 */etc/default/rc.conf* 的默认配置。默认文件不应当被逐字地拷到 */etc*。所有系统的任何变化将被记录在 *rc.conf* 文件中。由于 *rc.conf* 可以被 *sh* 命令打开阅读，所以完成这点很简单。例如：

rc.conf:

- *rc.conf.site*
- *hostname= "node15.webcompany.com"*
- *network_interfaces= "fxp0 lo0"*
- *ifconfig_fxp0= "inet 10.1.1.1"*

rc.conf.site:

- *default_router= "10.1.1.254"*
- *saver= "daemon"*
- *blanktime= "100"*

Rc.conf.site 文件会被分发给每一个使用 *eg.rsync* 的系统，而 *rc.conf* 文件仍保持独立。通过使用 *sysinstall* 或 *make world* 来升级系统不会覆盖 *rc.conf* 文件，所以系统配置信息不会被丢失。

6.4 应用程序配置

典型的，已安装的应用程序有它自己的配置文件，有它自己的语法。这些文件与基本系统相分离是很重要的，以至于它们能被 package 管理工具很好地定位和管理。另外，这些文件被安装在 `/usr/local/etc` 中。一个应用程序有许多配置文件，一个子目录将被创建以支持它们。

通常，当一个 port 或 package 被安装时，默认的文件也会被安装。这些通常可以通过 `.default` 后缀来辨别。如果不存在应用程序的配置文件，他们会通过拷贝 `.default` 文件来创建。例如，这儿是 `/usr/local/etc/apache`:

```
·      rw-r--r--  1 root  wheel   2184 May 20  1998 access.conf
·      rw-r--r--  1 root  wheel   2184 May 20  1998 access.conf.default
·      rw-r--r--  1 root  wheel   9555 May 20  1998 httpd.conf
·      rw-r--r--  1 root  wheel   9555 May 20  1998 httpd.conf.default
·      rw-r--r--  1 root  wheel  12205 May 20  1998 magic
·      rw-r--r--  1 root  wheel  12205 May 20  1998 magic.default
·      rw-r--r--  1 root  wheel   2700 May 20  1998 mime.types
·      rw-r--r--  1 root  wheel   2700 May 20  1998 mime.types.default
·      rw-r--r--  1 root  wheel   7980 May 20  1998 srm.conf
·      rw-r--r--  1 root  wheel   7933 May 20  1998 srm.conf.default
```

可以看到只有 `srm.conf` 文件已发生了变化。一个后来的 `apache` 的升级不会修改这个变化的文件。

6.5 启动服务

对一个系统来说，支持许多服务是很普通的。这些可能是用不同的形式来启动，每一个都有不同的长处。

一个 port 或 packages collection 安装软件通常把一个脚本放在 `/usr/local/etc/rc.d` 中，它可以在系统启动的时候被自动调用，在系统关闭的时候自动停止。这是一种我们推荐使用的启动服务的方法。这些脚本会作为安装 package 的一部分被注册，当 package 被删除的时候它也会被删除。在 `/usr/local/etc/rc.d` 中，一个普通的启动脚本是这样的：

```
#!/bin/sh

echo -n ' FooBar '

case "$1" in
start)
/usr/local/bin/foobar

;;

stop)
kill -9 ' cat /var/run/foobar.pid '

;;

*)

echo " Usage: ' basename $0 ' {start|stop} " >&2

exit 64

;;

esac

exit 0
```

这个脚本随着系统的启动而被呼叫，随着系统的关闭而停止。当一个连接被一个相配的 port 接收到时，一些服务会被 `inetd` 所调用。这个对邮件阅读服务器来说使用比较普遍（POP, IMAP 等）。这些服务可以通过编辑 `/etc/inetd.conf` 文件来启用。可以看看 `inetd` 命令的说明就可以了。

一些额外的系统功能不太可能会被隐藏在 `/etc/rc.conf` 中。这些通常能够使用命令来调用它们。就象在 FreeBSD 3.1 中，没有默认的 `/etc/rc.local`。如果它被系统管理员创建，它将不可能以普通的形式得到尊重。注意 `rc.local` 通常被作为是最后一个访问的记录；如果有一个比较好的地方能启动一个服务，就是在那儿。

注意：不要把任何命令都放在 `/etc/rc.conf` 中。要启动后台程序，或在启动时运行任何命令，可以在 `/usr/local/etc/rc.d` 中添加一行脚本。

使用 `cron` 程序来启动系统服务也是可以的。这种方法有很多优势，不仅仅是 `cron` 能运行这些进程，这些服务也可以被没有 `root` 权限的用户来启动和操作。

这利用了 `cron` 的一个非正式文件的特性；时钟的形式可能会被“`@reboot`”这种形式所取代，当系统启动以后，`cron` 程序被很快执行，这将导致工作暂停运行。

6.6 虚拟主机

FreeBSD 的一个非常普通的应用是虚拟主机功能，一台服务器可以虚拟成多台服务器来提供网络服务。这只需要分配多个网络地址给一个简单的接口就可以实现。

一个给定的网络接口有一个“`real`”地址，也会有很多“`alias`”地址。这些别名通常被添加到 `/etc/rc.conf` 中。

一个接口“`fxp0`”的别名记录是这样的：

```
ifconfig_fxp0_alias0="inet xxx.xxx.xxx.xxx netmask xxx.xxx.xxx.xxx"
```

注意：别名记录必须用 `alias0` 启动，然后按顺序向上处理，如 `_alias1`, `_alias2` 等。配置进程将在第一个丢失的数字时停止。别名的子网掩码的计算是很重要的，但幸运的是它非常简单。对于一个给定的接口，必须有一个正确反映网络的子网掩码的地址。

例如：假设 `fxp0` 接口连接到两个网络，`10.1.1.0` 的子网掩码是 `255.255.255.0`，而 `202.0.75.16` 的子网掩码是 `255.255.255.240`。我们要求系统显示从 `10.1.1.1` 到 `10.1.1.5` 和 `202.0.75.17` 到 `202.0.75.20`。

要正确配置适配器可以这样做：

```
ifconfig_fxp0="inet 10.1.1.1 netmask 255.255.255.0"
```

```
ifconfig_fxp0_alias0="inet 10.1.1.2 netmask 255.255.255.255"
```

```

ifconfig_fxp0_alias1= "inet 10.1.1.3 netmask 255.255.255.255 "
ifconfig_fxp0_alias2= "inet 10.1.1.4 netmask 255.255.255.255 "
ifconfig_fxp0_alias3= "inet 10.1.1.5 netmask 255.255.255.255 "
ifconfig_fxp0_alias4= "inet 202.0.75.17 netmask 255.255.255.240 "
ifconfig_fxp0_alias5= "inet 202.0.75.18 netmask 255.255.255.255 "
ifconfig_fxp0_alias6= "inet 202.0.75.19 netmask 255.255.255.255 "
ifconfig_fxp0_alias7= "inet 202.0.75.20 netmask 255.255.255.255 "

```

6.7 配置文件

6.7.1 /etc 规划

在配置信息中有很多的目录。这些包括：

/etc	一般的系统配置信息：这儿的数据是系统指定的。
/etc/default	系统配置文件的默认版本。
/etc/mail	额外的 sendmail 配置，其他 MTA 配置文件。
/etc/ppp	programs.user-和 kernel -ppp 程序的配置。
/etc/namedb	Bind 数据的默认定位。通常启动文件是定位在这儿，在/var/db 中参考其他数据的一个指示。
/usr/local/etc	安装应用程序的配置文件。可以参考每个应用程序的子目录。
/usr/local/etc/rc.d	安装应用程序的启动/停止的脚本。
/var/db	稳定的系统指定的数据文件: bind 区域文件，数据库文件等等。

6.7.2 主机名

6.7.2.1 /etc/resolv.conf

/etc/resolv.conf 描述了 FreeBSD 如何访问 internet 的域名系统 (DNS)。最普通的 resolv.conf 的记录是：

nameserver	要查询的域名服务器的 IP 地址。服务器按照顺序查询。
search	搜索域名的列表。这通常是由本地的域名决定的。
domain	本地域名。

一个典型的 `resolv.conf`:

```
search foobar.com

nameserver 147.11.1.11

nameserver 147.11.100.30
```

`Dhclient` 通常会把 DHCP 服务器接收到的信息重写 `resolv.conf`。

6.7.2.2 `/etc/hosts`

`/etc/hosts` 是一个文本数据库。它是用来联合 DNS 和 NIS 给 IP 地址的影射表提供名字。本地的电脑通过一个 LAN 进行连接，可能会作为一个简单的命名目的而放在这儿，以代替设置一个名称服务器。另外，`/etc/hosts` 能被用来提供一个本地 internet 名称的记录，减少搜索普通访问名称的需求。

```
# $FreeBSD$

#

# Host Database

# This file should contain the addresses and aliases

# for local hosts that share this file.

# In the presence of the domain name service or NIS, this file may

# not be consulted at all; see /etc/nsswitch.conf for the resolution order.

#

#

::1          localhost localhost.my.domain myname.my.domain

127.0.0.1    localhost localhost.my.domain myname.my.domain

#
```

Imaginary network.

```
#10.0.0.2          myname.my.domain myname
```

```
#10.0.0.3          myfriend.my.domain myfriend
```

```
#
```

According to RFC 1918, you can use the following IP networks for

private nets which will never be connected to the Internet:

```
#
```

```
#      10.0.0.0      -   10.255.255.255
```

```
#      172.16.0.0    -   172.31.255.255
```

```
#      192.168.0.0   -   192.168.255.255
```

```
#
```

In case you want to be able to connect to the Internet, you need

real official assigned numbers. PLEASE PLEASE PLEASE do not try

to invent your own network numbers but instead get one from your

network provider (if any) or from the Internet Registry (ftp to

rs.internic.net, directory ' /templates ').

```
#
```

/etc/hosts 的简单格式：

```
[Internet address] [official hostname] [alias1] [alias2] ...
```

例如：

```
10.0.0.1 myRealHostname.foobar.com myRealHostname foobar1 foobar2
```

6.7.3 日志文件配置

6.7.3.1 syslog.conf

syslog.conf 是 syslogd 程序的配置文件。它指出记录到日志文件的 syslog 信息的类型。

```
# $FreeBSD$

#

# Spaces ARE valid field separators in this file. However,
# other *nix-like systems still insist on using tabs as field
# separators. If you are sharing this file between systems, you
# may want to use only tabs as field separators here.
# Consult the syslog.conf manpage.

*.err;kern.debug;auth.notice;mail.crit /dev/console

*.notice;kern.debug;lpr.info;mail.crit;news.err /var/log/messages

security.* /var/log/security

mail.info /var/log/maillog

lpr.info /var/log/lpd-errs

cron.* /var/log/cron

*.err root

*.notice;news.err root

*.alert root

*.emerg *

# uncomment this to log all writes to /dev/console to /var/log/console.log
```

```
#console.info                                /var/log/console.log

# uncomment this to enable logging of all log messages to /var/log/all.log

#*. *                                         /var/log/all.log

# uncomment this to enable logging to a remote loghost named loghost

#*. *                                         @loghost

# uncomment these if you're running inn

# news.crit                                   /var/log/news/news.crit

# news.err                                    /var/log/news/news.err

# news.notice                                /var/log/news/news.notice

!startslip

.                                              /var/log/slip.log

!ppp

.                                              /var/log/ppp.log
```

6.7.3.2 newsyslog.conf

Newsyslog.conf 是 newsyslog 程序的配置文件。一个被 cron newsyslog 安排来运行的程序决定了什么时候日志文件要求重新存档或整理。Logfile 会被改为 logfile.1, logfile.1 会被改为 logfile.2 等等。另外, 日志文件会用 **gzip** 格式进行存档。它们是这样命名的: logfile.0.gz, logfile.1.gz 等等。

Newsyslog.conf 指出了哪个文件需要被管理, 有多少需要被保存, 什么时候他们会被调用。当他们达到一定大小或到一个适当的周期时, 日志文件需要被重新整理。

```
# configuration file for newsyslog

# $FreeBSD$

#
```

# logfile	[owner:group]	mode	count	size	when [ZB]	[/pid_file]	[sig_num]
/var/log/cron		600	3	100	*	Z	
/var/log/amd.log		644	7	100	*	Z	
/var/log/kerberos.log		644	7	100	*	Z	
/var/log/lpd-errs		644	7	100	*	Z	
/var/log/maillog		644	7	*	@T00	Z	
/var/log/sendmail.st		644	10	*	168	B	
/var/log/messages		644	5	100	*	Z	
/var/log/all.log		600	7	*	@T00	Z	
/var/log/slip.log		600	3	100	*	Z	
/var/log/ppp.log		600	3	100	*	Z	
/var/log/security		600	10	100	*	Z	
/var/log/wtmp		644	3	*	@01T05	B	
/var/log/daily.log		640	7	*	@T00	Z	
/var/log/weekly.log		640	5	1	\$W6D0	Z	
/var/log/monthly.log		640	12	*	\$M1D0	Z	
/var/log/console.log		640	5	100	*	Z	

6.7.4 sysctl.conf

Sysctl.conf 看起来象 rc.conf。它的值以这种形式来设置：variable=value。指定的值需要在进入多用户模式时才能被设置。在这种模式中不是所有的变量都能被设置。

一个 sysctl.conf 关闭引起重大错误产生的日志,然后让 Linux 程序知道他们真的运行在 FreeBSD 下面。

```
kern.logsigexit=0      # Do not log fatal signal exits (e.g. sig 11)

compat.linux.osname=FreeBSD

compat.linux.osrelease=4.3-STABLE
```

6.8 用 sysctl 进行调整

sysctl 是一个允许你对一个运行着的 FreeBSD 系统进行修改的接口。这包括许多 TCP/IP 堆栈和虚拟内存系统的高级选项，它通常可以为一个有经验的系统管理员提高系统的性能。能够使用 sysctl 来阅读和设置超过 5 百个系统变量。

基于这点，sysctl 起到两个功能：阅读和修改系统设置。去看看所有可读的变量：

```
% sysctl -a
```

阅读一个详细的变量，例如，kern.maxproc:

```
% sysctl kern.maxproc
kern.maxproc: 1044
```

设置一个特殊的变量，使用=选项：

```
# sysctl kern.maxfiles=5000
kern.maxfiles: 2088 -> 5000
```

Sysctl 变量的设置通常不是字符，数字，就是布尔类型的。一个布尔类型的树 1 代表 yes, 0 代表 no。

6.9 调整磁盘

6.9.1 Sysctl 变量

6.9.1.1 vfs.vmiodirenable

`vfs.vmiodirenable` `sysctl` 变量默认为 0 (off)，也可以被设置成 0 或 1。很多目录是比较小，只使用一小片段（典型的：1k），甚至更少（典型的：512 bytes）。然而，当在默认的模式下操作时，即使你有很多内存，缓存器只缓存一些固定的目录。打开这个 `sysctl` 可以允许缓存器使用 VM 页面缓存来缓存目录。这样的优势是所有的内存都能被缓存目录所利用。不利的是最小的用来缓存目录的核心内存要大于 512 bytes（典型的是：4K）。如果你运行有大量文件处理的服务时，我们建议把这个选项打开。这样的服务包括 web 缓存、大邮件系统和新闻系统。打开这项服务通常不会降低系统的性能，只是会浪费一点内存，但你要仔细检查一下。

6.9.1.2 `hw.ata.wc`

FreeBSD 通常把 IDE 写入缓存关闭掉。这可以减少写入磁盘时需要的带宽。基本上，当写入完成后，IDE 基本上就没事了。由于 IDE 写入缓存被打开，IDE 驱动器将不再按顺序把数据写入到硬盘中。当磁盘处于比较大的负荷时，它们通常能缓冲写入的数据。不幸的是，这样会丢失很多性能，最好还是改回默认值。你应当通过观察 `hw.ata.wc` `sysctl` 变量来检查你的系统的默认情况。如果 IDE 的写入缓存被关闭，你可以在内核变量中把它改回 1 就可以把它打开。这必须在启动时从引导程序过程中进行。在内核启动之后再这样做就没有用了。

6.9.2 `SoftUpdates`

`Tunefs` 能被用来很好地调整文件系统。可以这样做：

```
# tunefs -n enable /filesystem
# tunefs -n disable /filesystem
```

一个文件系统当被挂上的时候不能使用 `tunefs` 进行修改。在单用户模式下，在所有分区都没有被挂上之前，起用 `SoftUpdates` 是最佳时机。

通过使用内存缓冲器，`SoftUpdates` 能够极大地提高文件的性能，只要是文件创建和删除。我们建议在你的所有文件系统上都打开 `SoftUpdates`。你应当清楚两点：第一，`SoftUpdates` 能在系统出现错误时保证系统的一致性，但在升级了物理磁盘后可能只需要几秒钟。如果你的系统崩溃了，你可能会丢失很多的工作。第二，`SoftUpdates` 可以推迟文件系统块的释放时间。如果你有一个接近满了的文件系统，对它作一个主升级，`make installworld`，可能会超出运行空间，从而引起升级失败。

6.10 调整内核限制

6.10.1 文件/进程限制

6.10.1.1 kern.maxfiles

kern.maxfiles 可以根据你系统的要求加大或减小。这个变量指出了在你系统上描述的最大数量文件。可以用 dmesg 来显示文件描述符的情况。

每一个打开的文件，套接字，或 fifo 使用的文件描述符，都依赖于当前运行的服务的种类和数量。

kern.maxfile 的默认值可以在你的内核配置中通过 maxusers 选项来指定。kern.maxfiles 可以按比例地增加 maxuser 的值。

6.10.2 网络限制

NMBCLUSTERS 内核配置选项指出了系统可用的网络 mbufs 的数量。一个具有大量负载的服务器如果 Mbufs 比较少，就会影响 FreeBSD 的性能。每一个 cluster 大概需要 2K 的内存，所以 1024 个 cluster 就需要保留 2MB 的内存给网络缓存。如果你的服务器超过一个并发连接，那每个连接需要吃掉一个 16k 的接收缓存和一个 16K 的发送缓存，你需要提供 32MB 的内存给网络缓存以确保 web 服务器的稳定。一个最笨的计算方法是乘以 2，所以 $32\text{MB} \times 2 = 64\text{MB} / 2\text{K} = 32768$ 。