

NanoBSD 简介

摘要

本篇提供了关于 NanoBSD 工具的介绍信息， 一工具可以用来构建用于嵌入式用的 FreeBSD 系统映像， 以存放在袖珍闪存 (Compact Flash) 或其他大容量存储介质的需要。

目录

1. NanoBSD 简介	1
2. 如何使用 NanoBSD	1

1. NanoBSD 简介

NanoBSD 是 Poul-Henning Kamp 目前正在开发的一工具。 它可以用来构建用于嵌入式用的 FreeBSD 系统映像， 以便配合袖珍闪存 (Compact Flash) 或其他大容量存储介质使用。

一工具也可以用来构建定制的安装映像， 以简化通常称 "计算机设备 (computer appliances)" 的系统的安装和管理工作。 计算机通常在设备中将硬件和软件， 或者言之， 所有的应用程序都是先装好的。 有些可以直接到网络的网中， 并 (几乎是) 立即投入使用。

NanoBSD 提供的功能包括：

- 可以和 FreeBSD 一样使用 Ports 和软件包- 所有的应用程序都可以在 NanoBSD 映像中直接使用， 而方式与 FreeBSD 完全一样。
- 不少功能 - 能使用 FreeBSD 做的任何工作， 都可以在 NanoBSD 中使用， 除非在构建 NanoBSD 映像时， 明确地去它。
- 所有象在运行时均是只读的 - 可以安全地拔掉电源。 在系统非正常之后， 无需运行 `fsck(8)`。
- 便于管理和定制 - 只需使用一个 shell 脚本和一个配置文件， 可以很容易地裁剪和定制于任意需求的映像。

2. 如何使用 NanoBSD

2.1. NanoBSD 的分区

一旦将映像存入介质， 就可以用它来引导 NanoBSD 了。 默认情况下， 大容量存储器会分三个区：

- 两个映像区：`code#1` 和 `code#2`。
- 一个配置文件区， 运行时境中， 可以将其挂接到 `/cfg` 目录下。

这些分区默认情况下以只读方式挂载。

`/etc` 和 `/var` 目录均 `md(4)` (malloc) 。

配置文件分区保存在 `/cfg` 目录。它包含了用于 `/etc` 目录的文件，在之后以只读方式挂载。因此，在需要从 `/etc` 向 `/cfg` 目录复制所执行的、希望在重启保持不变的配置，需要执行一些额外的操作。

例 1. 在 `/etc/resolv.conf` 中执行需要保持的修改

```
# vi /etc/resolv.conf
[...]
# mount /cfg
# cp /etc/resolv.conf /cfg
# umount /cfg
```



只有在系统启动过程中，以及需要修改配置文件的场合，才需要挂载包含 `/cfg` 的那个分区。

在任何时候都保持挂载 `/cfg` 不是一个好主意，特别是当你把 NanoBSD 放在不适合进行大量写操作的分区（由于文件系统的同一进程会定期向系统写一些数据）。

2.2. 构建 NanoBSD 映像

NanoBSD 映像通常使用非常简单的 `nanobsd.sh` shell 脚本来构建的，这个脚本可以在 `/usr/src/tools/tools/nanobsd` 目录中找到。这个脚本建立的映像文件，可以用 [dd\(1\)](#) 工具复制到存储介质上。

构建 NanoBSD 映像所需的命令是：

```
# cd /usr/src/tools/tools/nanobsd ①
# sh nanobsd.sh ②
# cd /usr/obj/nanobsd.full ③
# dd if=_disk.full of=/dev/da0 bs=64k ④
```

① 进入 NanoBSD 构建脚本的主目录。

② 开始构建过程。

③ 进入构建好的映像文件所在的目录。

④ 在存储介质上安装 NanoBSD。

2.3. 定制 NanoBSD 映像

可能是 NanoBSD 最重要，同时也是最感兴趣的功能。同时，在构建 NanoBSD 映像，也是相当耗时的过程。

运行下面的命令将使 `nanobsd.sh` 从当前目录中的 `myconf.nano` 文件读取配置：

```
# sh nanobsd.sh -c myconf.nano
```

定制过程包含：

- 配置
- 定制函数

2.3.1. 配置

通过配置行，可以配置用以 NanoBSD 构建过程中 `buildworld` 和 `installworld` 段的和安装，以及 NanoBSD 的主构建程序中的。通过使用些可以削减系的尺寸，使之能放入 64MB 的存。可以通过些来削减 FreeBSD，直到它只包含内核以及三个用境文件止。

配置文件中包含用以代替默的配置。最重要的句包括：

- `NANO_NAME` - 本次构建的名称 (用于构建工作目的名字)。
- `NANO_SRC` - 用以和构建映像的源的位置。
- `NANO_KERNEL` - 用以内核的配置文件的名字。
- `CONF_BUILD` - 用于 `buildworld` 构建段的。
- `CONF_INSTALL` - 用于 `installworld` 构建段的。
- `CONF_WORLD` - 用以 `buildworld` 和 `installworld` 个构建段的。
- `FlashDevice` - 定所用的介型。要了解一的，参考 `FlashDevice.sub` 文件。

2.3.2. 定制函数

通过在配置文件中使用 shell 函数可以一微 NanoBSD。下面的例子展示了定制函数的基本模式：

```
cust_foo () (
    echo "bar=baz" > \
        ${NANO_WORLDDIR}/etc/foo
)
customize_cmd cust_foo
```

下面是一个更近的例子，它将默的 /etc 目尺寸，从 5MB 整 30MB：

```
cust_etc_size () (
    cd ${NANO_WORLDDIR}/conf
    echo 30000 > default/etc/md_size
)
customize_cmd cust_etc_size
```

除此之外，有几个默的定制函数：

- `cust_comconsole` - 在 VGA 上禁止 `getty(8)` (/dev/ttyv* 点) 并用串口 COM1 作系控制台。
- `cust_allow_ssh_root` - 允 `root` 通 `sshd(8)` 登。
- `cust_install_files` - 从 `nanobsd/Files` 目中安装文件，包含一些用的系管理脚本。

2.3.3. 安装软件包

通过添加自定义的函数，可以在 NanoBSD 添加软件包。下面的函数会添加位于 `/usr/src/tools/tools/nanobsd/packages` 的全部软件包：

```
install_packages () (  
  mkdir -p ${NANO_WORLDDIR}/packages  
  cp /usr/src/tools/tools/nanobsd/packages/* ${NANO_WORLDDIR}/packages  
  chroot ${NANO_WORLDDIR} sh -c 'cd packages; pkg_add -v *;cd ..;'   
  rm -rf ${NANO_WORLDDIR}/packages  
)  
customize_cmd install_packages
```

2.3.4. 配置文件示例

下面是一个用于构建定制的 NanoBSD 映像的完整例子：

```
NANO_NAME=custom
NANO_SRC=/usr/src
NANO_KERNEL=MYKERNEL
NANO_IMAGES=2
```

```
CONF_BUILD='
NO_KLDLOAD=YES
NO_NETGRAPH=YES
NO_PAM=YES
'
```

```
CONF_INSTALL='
NO_ACPI=YES
NO_BLUETOOTH=YES
NO_CVS=YES
NO_FORTRAN=YES
NO_HTML=YES
NO_LPR=YES
NO_MAN=YES
NO_SENDMAIL=YES
NO_SHAREDOCS=YES
NO_EXAMPLES=YES
NO_INSTALLLIB=YES
NO_CALENDAR=YES
NO_MISC=YES
NO_SHARE=YES
'
```

```
CONF_WORLD='
NO_BIND=YES
NO_MODULES=YES
NO_KERBEROS=YES
NO_GAMES=YES
NO_RESCUE=YES
NO_LOCALES=YES
NO_SYSCONS=YES
NO_INFO=YES
'
```

FlashDevice SanDisk 1G

```
cust_nobeastie() (
    touch ${NANO_WORLDDIR}/boot/loader.conf
    echo "beastie_disable=\"YES\"" >> ${NANO_WORLDDIR}/boot/loader.conf
)
```

```
customize_cmd cust_comconsole
customize_cmd cust_install_files
customize_cmd cust_allow_ssh_root
customize_cmd cust_nobeastie
```

2.4. 更新 NanoBSD

更新 NanoBSD 相对而言很简单：

1. 和之前一样建新的 NanoBSD 映像文件。
2. 将新的映像放入正在运行的 NanoBSD 中的一个未用的分区。

与之前最初安装 NanoBSD 的情况相比，唯一最重要的区别在于这次不使用 `_.disk.full` 文件（它包含整个映像），而安装 `_.disk.image` 映像（多个文件中，只包含一个系统分区）。

3. 重新引导，并从新安装的分区中启动。
4. 如果一切顺利的，升级工作就完成了。
5. 如果生了任何问题，你可以从先前的分区（其中包含了旧的、可用的映像），来尽可能快地恢复系统功能。接下来可以修正新版本的版本中存在的问题，并重试前述。

要在正在运行的 NanoBSD 系统中安装新的映像，可以使用位于 `/root` 目录的 `updatep1` 或 `updatep2` 脚本，具体使用哪一个脚本，取决于正在运行的系统位于那个分区。

随提供新 NanoBSD 映像所提供的服务，以及采用的方法的不同，可以参考并使用下列三种方式之一：

2.4.1. 使用 `ftp(1)`

如果速度是第一要，采用下面的例子：

```
# ftp myhost
get _.disk.image "| sh updatep1"
```

2.4.2. 使用 `ssh(1)`

如果更倾向于安全，参考下面的例子：

```
# ssh myhost cat _.disk.image.gz | zcat | sh updatep1
```

2.4.3. 使用 `nc(1)`

如果程序主机既不提供 `ftp(1)` 服务，也不提供 `sshd(8)` 服务：

1. 开始，在提供映像的主机上打开 TCP 监听，并令其将映像文件发送给客户机：

```
myhost# nc -l 2222 < _.disk.image
```



所使用端口没有通过防火墙阻止来自 NanoBSD 客户机的连接请求。

2. 连接到提供新映像服务的主机，并运行 `updatep1` 脚本：

```
# nc myhost 2222 | sh updatep1
```