# Einführung in NanoBSD

Zusammenfassung

Dieses Dokument stellt Informationen zu den NanoBSD Werkzeugen bereit, die dazu verwendet werden können ein FreeBSD Abbild für eingebettete Systeme zu erstellen, welche auf eine Compact Flash Karte passen (oder andere Massenspeicher).

*Übersetzt von Björn Heidotting.*

---

# Inhaltsverzeichnis

1. Einführung in NanoBSD . . . . . 1
2. NanoBSD Anleitung . . . . . 2

# 1. Einführung in NanoBSD

NanoBSD ist ein Werkzeug welches derzeit von Poul-Henning Kamp entwickelt wird. Es erstellt ein FreeBSD Systemabbild für eingebettete Systeme, die auf eine Compact Flash Karte passen (oder andere Massenspeicher).

Es kann dazu benutzt werden um spezialisierte Installationsabbilder zu bauen, entworfen für die einfache Installation und Wartung von Systemen die als "Computer Appliances" bekannt sind. Computer Appliances haben ihre Hard- und Software fest verbaut, dass bedeutet alle Anwendungen sind vorinstalliert. Die Appliance wird an ein bestehendes Netzwerk angeschlossen und kann mit der Arbeit (fast) sofort beginnen.

Zu den Eigenschaften von NanoBSD gehören:

- Ports und Pakete funktionieren wie in FreeBSD - Jede einzelne Anwendung kann auf dem NanoBSD Abbild installiert und benutzt werden, auf die gleiche Weise wie Sie es aus FreeBSD gewohnt sind.
- Keine fehlende Funktionalität - Wenn es möglich ist, etwas mit FreeBSD zu tun, ist es auch möglich, die gleiche Sache mit NanoBSD zu tun, es sei denn, eine oder mehrere Funktionen wurden ausdrücklich vor dem Bau des NanoBSD Abbilds entfernt.
- Zur Laufzeit ist alles read-only - Es ist sicher den Stromstecker zu ziehen. Es besteht dann keine Notwendigkeit, einen fsck(8) nach einem nicht ordnungsgemäßem Herunterfahren des Systems auszuführen.
- Einfach zu bauen und anzupassen - Unter Verwendung von nur einem Shell-Skript und einer Konfigurationsdatei ist es möglich, ein reduziertes und angepasstes Abbild zu bauen, welches jegliche Reihe von Anforderungen erfüllt.

# 2. NanoBSD Anleitung

## 2.1. Das Design von NanoBSD

Sobald das Abbild auf dem Medium verfügbar ist, kann NanoBSD gebootet werden. Der Massenspeicher ist standardmäßig in drei Teile unterteilt:

- Zwei Abbild Partitionen: `code#1` und `code#2`.
- Die Partition der Konfigurationsdatei, welche zur Laufzeit unter dem /cfg Verzeichnis gemountet werden kann.

Diese Partitionen sind im Allgemeinen read-only.

Die /etc und /var Verzeichnisse sind md(4) (malloc) Speicher.

Die Partition der Konfigurationsdatei besteht unter dem /cfg Verzeichnis. Sie enthält Dateien für das /etc Verzeichnis und wird direkt nach dem Botten read-only eingehangen, weshalb es erforderlich ist geänderte Dateien von /etc zurück nach /cfg zu kopieren falls die Änderungen nach einem Neustart bestehen bleiben sollen.

*Beispiel 1. Dauerhafte Änderungen in /etc/resolv.conf vornehmen*

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

Die /cfg Partition sollte nur während des Bootvorgangs und zu Änderungen an den Konfigurationsdateien gemountet werden.

Die /cfg Partition jederzeit gemountet zu haben ist keine gute Idee, besonders wenn das NanoBSD System auf einem Massenspeicher betrieben wird, der eventuell druch eine große Anzahl von Schreiboperationen nachteilig beeinträchtigt wird (z. B. wenn der Dateisystem-Syncer den Speicher mit Daten überflutet).

## 2.2. Ein NanoBSD Abbild erstellen

Ein NanoBSD Abbild wird über ein einfaches nanobsd.sh Shell-Skript gebaut, das sich unter /usr/src/tools/tools/nanobsd befindet. Das Skript erstellt ein Abbild, welches dann mittels dd(1) auf einen Massenspeicher kopiert werden kann.

Die folgenden Kommandos sind notwendig um ein NanoBSD Abbild zu erstellen:

```
# 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 Wechsel in das Basisverzeichnis des NanoBSD Skripts.

② Den Bauprozess starten.

③ Wechsel in das Verzeichnis, in dem das gebaute Abbild liegt.

④ NanoBSD auf einem Massenspeicher installieren.

## 2.3. Ein NanoBSD Abbild anpassen

Dies ist wahrscheinlich das wichtigste und interessanteste Merkmal von NanoBSD. Hierbei werden Sie auch die meiste Zeit mit der Entwicklung von NanoBSD verbringen.

Der Aufruf des folgenden Kommandos wird nanobsd.sh dazu zwingen, seine Konfiguration aus myconf.nano aus dem aktuellen Verzeichnis zu lesen:

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

Die Anpassung wird auf zwei Arten geschehen:

- Konfigurations-Optionen
- Benutzerdefinierte Funktionen

### 2.3.1. Konfigurations-Optionen

Durch Konfigurationseinstellungen ist es möglich Optionen zu übergeben, die sowohl die `buildworld` und `installworld` Phasen des NanoBSD Bauprozesses betreffen, sowie interne Optionen, die den Haupt-Bauprozess von NanoBSD beeinflussen. Durch diese Optionen ist es möglich, das System so zu reduzieren, dass es mit wenig Platz, etwa 64 MB auskommt. Sie können die Konfigurationsdateien dazu nutzten FreeBSD noch weiter zu trimmen, bis es nur noch aus dem Kernel und zwei oder drei Dateien im Userland besteht.

Die Konfigurationsdatei besteht aus Konfigurations-Optionen, welche die Standardwerte überschreiben.

- `NANO_NAME` - Name des Build (wird verwendet, um die workdir Namen zu konstruieren).
- `NANO_SRC` - Pfad zum Quelltextverzeichnis, das für den Bau des Abbilds verwendet wird.
- `NANO_KERNEL` - Name der Kernelkonfigurationsdatei, die für den Bau des Kernels verwendet wird.
- `CONF_BUILD` - Optionen für die `buildworld` Phase des Bauprozesses.
- `CONF_INSTALL` - Optionen für die `installworld` Phase des Bauprozesses.
- `CONF_WORLD` - Optionen für die `buildworld` und `installworld` Phasen des Bauprozesses.

- `FlashDevice` - Definiert den zu benutzenden Medientyp. Überprüfen Sie die Datei FlashDevice.sub für weitere Informationen.

### 2.3.2. Benutzerdefinierte Funktionen

Mit Hilfe von Shell-Funktionen in der Konfigurationsdatei besteht die Möglichkeit zur Feinabstimmung von NanoBSD. Das folgende Beispiel illustriert das Grundmodell von benutzerdefinierten Funktionen:

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

Ein besseres Beispiel für eine Anpassung ist folgende, welche die Standardgröße des /etc Verzeichnisses von 5 MB auf 30 MB ändert:

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

Es gibt ein paar vordefinierte Standardfunktionen die Sie nutzen können:

- `cust_comconsole` - Deaktiviert getty(8) auf den VGA Geräten (den /dev/ttyv* Gerätedateien) und ermöglicht die Nutzung der seriellen Schnittstelle COM1 als Systemkonsole.
- `cust_allow_ssh_root` - Erlaubt es `root` sich über sshd(8) anzumelden.
- `cust_install_files` - Installiert Dateien aus dem nanobsd/Files Verzeichnis, das einige nützliche Skripte für die Systemverwaltung enthält.

### 2.3.3. Pakete hinzufügen

Durch benutzerdefinierte Funktionen können Pakete zum NanoBSD Abbild hinzugefügt werden. Die nachfolgende Funktion installiert alle Pakete aus /usr/src/tools/tools/nanobsd/packages:

```
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. Beispiel einer Konfigurationsdatei

Ein komplettes Beispiel für eine Konfigurationsdatei zum Erstellen eines benutzerdefinierten NanoBSD Abbilds könnte folgende sein:

```
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. NanoBSD aktualisieren

The Update-Prozess von NanoBSD ist relativ einfach:

1. Erstellen Sie ein neues NanoBSD Abbild.
2. Laden Sie das neue Abbild in eine unbenutzte Partition eines laufenden NanoBSD Systems.

   Der wichtigste Unterschied dieses Schrittes zur ersten NanoBSD Installation besteht darin, das jetzt anstatt der Datei _.disk.full (enthält ein Abbild der gesamten Platte) die Datei _.disk.image (enthält ein Abbild einer einzelnen System-Partition) installiert wird.
3. Neustart, um das System von der neu installierten Partition zu starten.
4. Wenn alles gut geht, ist die Aktualisierung abgeschlossen.
5. Wenn etwas schief läuft, starten Sie wieder in die vorherige Partition (die das alte, funktionierende Abbild enthält) um die System-Funktionalität so schnell wie möglich wieder herzustellen. Beheben Sie alle Probleme des neu gebauten Abbilds, und wiederholen Sie den Vorgang.

Um das neue Abbild auf das laufende NanoBSD System zu installieren, ist es möglich, entweder das updatep1 oder updatep2 Skript im /root Verzeichnis zu verwenden, je nachdem, von welcher Partition das aktuelle System läuft.

In Abhängigkeit davon welche Dienste der Host, der das NanoBSD Abbild anbietet, und welche Art von Transfer bevorzugt wird, bestehen eine von drei zu prüfenden Möglichkeiten:

### 2.4.1. Verwendung von ftp(1)

Wenn die Übertragungsgeschwindigkeit an erster Stelle steht, verwenden Sie dieses Beispiel:

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

### 2.4.2. Verwendung von ssh(1)

Wenn eine sichere Übertragung bevorzugt wird, sollten Sie die Verwendung dieses Beispiels in Betracht ziehen:

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

### 2.4.3. Verwendung von nc(1)

Verwenden Sie dieses Beispiel, wenn auf dem Remote-Host kein ftpd(8) oder sshd(8) Dienst läuft:

1. Zunächst öffnen Sie eine TCP-Listener auf dem Host der das Abbild bereitstellt und zum Client sendet:

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

   > Stellen Sie sicher das der benutzte Port nicht blockiert wird, um eingehende Verbindungen, vom NanoBSD Host durch die Firewall, zu ermöglichen.

2. Verbinden Sie sich zum Host der das Abbild bereitstellt und führen Sie das updatep1 Skript aus:

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