

# Fedora jako NFS-readonly-root

## Start Fedory Core w pracy NFS-readonly-root - obraz initrd\*

Krzysztof Kozłowski<sup>†</sup>

8 maja 2005 roku

### Spis treści

<b>1</b>	<b>Wstęp</b>	<b>2</b>
<b>2</b>	<b>Ogólny zarys czynności</b>	<b>2</b>
<b>3</b>	<b>Jak podnieść Fedorę z korzeniem z NFS-u?</b>	<b>2</b>
3.1	Podnoszenie - PXE . . . . .	2
3.2	Initrd - pierwszy system plików . . . . .	3
3.2.1	Struktura . . . . .	3
3.2.2	Urządzenia /dev . . . . .	3
3.2.3	Pliki wykonywalne i biblioteki . . . . .	3
3.3	Procedura startowa - krok po kroku . . . . .	4
3.3.1	pre-init . . . . .	4
3.3.2	LinuxRC - skrypt startowy . . . . .	4
3.3.3	Kilka zmian w dalszej procedurze startowej . . . . .	6
3.4	Tworzenie initrd . . . . .	6
3.4.1	System plików . . . . .	7
3.4.2	Zapisanie initrd . . . . .	8
<b>4</b>	<b>Moje skrypty</b>	<b>9</b>
<b>5</b>	<b>Do zrobienia</b>	<b>9</b>
<b>6</b>	<b>Literatura</b>	<b>9</b>

---

\*Dokument udostępniany bezpłatnie. Możliwe jest jego dowolne rozprowadzanie, ale bez zmiany zawartości. Copyright (C) 2005 Krzysztof Kozłowski

<sup>†</sup>Autor jest studentem informatyki na wydziale Elektrycznym Politechniki Warszawskiej.

Kontakt: k.kozlowski [at] acn.waw.pl, strona: <http://acn.waw.pl/koziol>

## 1 Wstęp

Celem projektu było przygotowanie Fedory Core (od wersji 3) do bezdyskowej pracy na korzeniu tylko do odczytu montowanym poprzez NFS (NFS-readonly-root). Wykonanie wiąże się zarówno z przygotowaniem własnego initrd ("*boot loader initialized RAM disk*") oraz z modyfikacją zainstalowanej już Fedory w celu dostosowania jej do takiej pracy. Tekst ten będzie opisem zarówno etapu przygotowania initrd jak i procedury startowej Fedory Core (w ogólności - systemu operacyjnego Linux).

## 2 Ogólny zarys czynności

1. Minimalna instalacja Fedory Core 3/4.
2. Przeniesienie zainstalowanej Fedory na serwer, z którego będzie udostępniana.
3. Przygotowanie initrd do podnoszenia Fedory poprzez PXE i montowania swojego korzenia z katalogu NFS.
4. Przygotowanie plików Fedory na serwerze do pracy na zasadzie NFS-readonly-root.

## 3 Jak podnieść Fedorę z korzeniem z NFS-u?

### 3.1 Podnoszenie - PXE

Podnoszenie systemu realizuje PXE (pxelinux), którego konfiguracja jest w zasadzie bardzo podobna do ISOLINUX. W pliku pxelinux.conf należy dopisać :

```
# --- Fedora Core 3 - stateless (NFS-readonly-root)
# volt - 194.29.146.3
label fc3
kernel fedora/2.92/diskless/vmlinuz
append initrd=fedora/2.92/diskless/initrd.img root=/dev/ram0
    ramdisk_size=32000 rw vga=788 lang=pl
    nfsdir=194.29.146.3:/nfs/redhat/2.92/diskless
```

Gdzie :

1. *kernel* jest względną ścieżką do pliku jądra Linuksa
2. *append* jest listą parametrów dołączanych podczas podnoszenia jądra

3. *initrd* jest względną ścieżką do obrazu *initrd.img*
  4. *root* jest urządzeniem, pod którym zostanie zamontowany pierwszy system plików
  5. *ramdisk\_size* jest wielkością początkowego RAM-dysku (minimum wielkość obrazu *initrd*).
  6. *nfsdir* jest pełną ścieżką do zasobu NFS z korzeniem Fedory (wymagane adres IP dla serwera, gdyż w trakcie montowania zasobu NFS nie będzie jeszcze działało mapowanie nazw DNS)
- Można też dodać inne parametry, które będą sprawdzane przez skrypt LinuxRC podczas startu maszyny. Aktualnie obsługuje on *debug* (uaktywnia dodatkowe komunikaty i uruchomienia */bin/bash* w trakcie wykonywania skryptu linuxRC) oraz *nfs\_root\_rw* (próbuję zamontować korzeń z NFS-u w trybie read-write).

## 3.2 Initrd - pierwszy system plików

### 3.2.1 Struktura

Initrd jest to w praktyce system plików ext2fs z podstawową strukturą katalogów. Samą strukturę (drzewo katalogów i wymagane pliki) można zaczerpnąć ze standardowego *initrd* dla jądra Linuksa generowanego poprzez program *mkinitrd*<sup>1</sup>

Wymagane katalogi, to: *bin*, *dev*, *etc*, *lib*, *modules*, *proc*, *sbin*, *sys*, *sysroot* (nazwa dowolna - tu będzie zamontowany zasób NFS), *tmp*, *var*, przy czym zależnie od skryptu startującego można obejść się bez części z nich.

### 3.2.2 Urządzenia /dev

Fedora Core 3 i 4 korzystają z *udev*, więc w katalogu */dev* wymagają tylko kilku podstawowych urządzeń : *console*, *null*, *zero*, *sys tty*, *tty[0-6]*, *ram[01]*. Tworzy się je przy pomocy programu *mknod*.

### 3.2.3 Pliki wykonywalne i biblioteki

Teoretycznie podczas startu systemu powinniśmy korzystać tylko z programów zbudowanych statycznie - niewymagających dynamicznych bibliotek. Jednakże nie wszystkie potrzebne narzędzia standardowo znajdują się w takiej postaci, a ewentualne dołączenie bibliotek nie stanowi problemu.

---

<sup>1</sup>**Uwaga!** Program ten nie tworzy systemu plików ext2fs, ale skompresowane archiwum CPIO ("magic cpio").

Wszystkie dynamicznie skompilowane programy należy potraktować *ldd* - bez trudu wskazał on jakie biblioteki są wymagane przez dany program. Te biblioteki należy dołączyć do obrazu *initrd*.

### 3.3 Procedura startowa - krok po kroku

#### 3.3.1 pre-init

Niektóre dokumentacje Linuksa podają, że w przypadku obecności skryptu *linuxrc* w korzeniu systemu plików jest on wykonywany. Jak się okazało, Fedora postępowała inaczej. Wpierw szuka programu */bin/init*, a w przypadku jego braku szuka */bin/sh*<sup>2</sup>. I właśnie pod */bin/sh* podpinamy skrypt *linuxrc*, który przygotowuje środowisko do dalszego podnoszenia systemu.

#### 3.3.2 LinuxRC - skrypt startowy

To jest sedno procedury startowej. Skrypt ten wywoływany jest po załadowaniu jądra i wykonaniu przez niego podstawowych czynności (wykrywanie sprzętu itp.). Przygotuje całe środowisko, aby system mógł przejść w kolejny tryb (*initdefault*). Interpretować go powinien program w rodzaju *ash* czy *nash* - Fedora 3 posiada do tego celu statycznie zbudowany *ash.static*. Posiada on funkcjonalność typowej powłoki - w odróżnieniu od *nash* używanego w typowym *initrd* budowanym przez *mkinitrd*, który nie obsługuje zmiennych. Niestety Fedora Core 4 nie zawierała *ash.static*, więc trzeba było posłużyć się *bash-em*.

Czynności jakie wykonuje ten skrypt (odnośnie szczegółów odsyłam do skryptu *linuxrc\_kk*).

**/proc i /sysfs** Zamontowanie systemu */proc* oraz */sysfs*

**/dev** Zamontowanie */dev* - system plików *tmpfs*. Wystartowanie *udev*.

**Załadowanie modułów** Należy teraz załadować wszystkie potrzebne moduły, tj.:

1. lokalne systemy plików (*ext3.ko*, *jbd.ko*)
2. NFS i RPC (*sunrpc.ko*, *lockd.ko*, *nfs.ko* - istotna kolejność)

---

<sup>2</sup>Potwierdza to bardzo szczegółowy opis działania jądra z serii 2.4 - patrz *Literatura* pozycja nr 6

3. Karta sieciowa - wykrywana automatycznie poprzez *lspci* i tablicę *pcitable* (w Fedorze znajduje się w `/usr/share/hwdata/pcitable` - trzeba skopiować na *initrd*). Często wystarczy załadowanie modułu *mii.ko* oraz *e100.ko*.

**Sieć** Podniesienie interfejsu sieciowego *eth0* poprzez DHCP. Tworzony jest w tym celu plik *dhclient.conf* i wywoływany `/bin/dhclient`.

**Korzeń z NFS** Kolejny etap to wyszukanie wśród zmiennych środowiskowych zmiennej *NFSDIR* (podawana jest podczas podnoszenia jądra w części *append*) i zamontowanie drzewa Fedory w podkatalogu `/sysroot` (nazwa samego miejsca montowania dowolna).

**RAM-dysk** Tworzony jest dynamiczny RAM-dysk w `/sysroot/var`. Wykorzystywany jest system plików *tmpfs* (obecny od jądra 2.4). Ten RAM-dysk będzie musiał pomieścić wszystkie generowane przez użytkownika pliki jak i katalogi `/etc` oraz `/tmp`.

**Przeniesienie /dev** Należy przenieść teraz stare drzewo `/dev` do podkatalogu `/sysroot/dev`. Wykona to polecenie *mount* z parametrem *-move*:

```
/bin/mount --move /dev /sysroot/dev
```

Tu pojawia się istotna kwestia uprawnień urządzeń `/dev/null` (**0666**) oraz `/dev/console` (**0660**). W przypadku tworzenia *initrd* na systemie innym niż Linux nie można było zmienić uprawnień tego pliku komendą *chmod* (a *mknode* nie obsługuje parametru *-mode=0660*), więc musi przeprowadzić to skrypt *linuxrc*.

**Przemontowanie roota** Przechodzimy do katalogu `/sysroot` z korzeniem z i uruchamiamy program *pivot\_root* wedle składni :

```
/bin/pivot_root . initrd
```

gdzie *initrd*, to podkatalog w `/sysroot`, w którym znajdzie się później nasz obecny *initrd* (dzięki temu po zmianie korzenia dalej będziemy mieć dostęp do starego). Demontujemy po tym stary `/proc` i `/sys` (teraz jako podkatalogi `/initrd`) i zabijamy proces *dhclient* (jego PID znajduje się w `/initrd/tmp/dhclient.pid`).

**Przygotowanie /var i /tmp** Uruchamiany jest specjalny skrypt *rc.prepare\_var\_tmp*, którego zadaniem jest skopiowanie z katalogu `/template/var` struktury drzewa `/var` na nasz RAM-dysk. `/tmp` musi istnieć tworzony jako dowiązanie symboliczne do `/tmp` → `/var/tmp`.

**Przygotowanie /etc** Katalog /etc również kopiowany jest (poprzez skrypt *rc.prepare\_etc*) z szablonu /template/etc do /var/etc. W korzeniu musi istnieć dowiązanie /etc → /var/etc. Należy zwrócić uwagę, że daje to katalog /etc read-write. W przypadku zastosowania opcji "read-only" (tylko do odczytu) należy pamiętać o pliku /etc/mtab, który musi być zapisywalny. Realizuje się to poprzez utworzenie pliku /proc/mounts w czystym /proc drzewa Fedory na serwerze i utworzenie dołączenia symbolicznego /template/etc/mtab na tamten plik. W naszym przypadku nie jest to konieczne, gdyż /etc będziemy mieć zapisywalne.

**Całkowite ucięcie initrd** Teraz całkowicie demontujemy wszystkie pozostałości initrd, czyli /initrd/dev oraz sam /initrd. Demontaż tego ostatniego przeprowadza się specjalną komendą z wykorzystaniem *chroot* oraz uruchomieniem *init*.

```
exec /usr/sbin/chroot . sh -c 'umount /initrd; exec /sbin/init' \  
<dev/console >dev/console 2>&1
```

**Powrót** Wywołany *init* przechodzi w tryb "sysinit", a następnie w "initdefault".

### 3.3.3 Kilka zmian w dalszej procedurze startowej

Przed wszystkim należy zapobiec standardowemu podnoszeniu interfejsów sieciowych. Nasz główny interfejs mamy już uruchomiony i w przypadku jego ponownej reinicjalizacji tracimy połączenie z serwerem NFS i całe podnoszenie systemu staje w miejscu. To pokazuje jak ważne jest poprawne zainicjowanie sieci na samym początku.

Mogą pojawić się problemy ze ścieżkami, np. w skryptach inicjalizujących sieć (/etc/sysconfig/network-scripts) było dowiązanie symboliczne do pliku /sbin/ifup, ale względne (../../sbin/ifup). Rozwiązać to można poprzez utworzenie w katalogu /var linków symbolicznych - zarówno do /bin jak i /sbin.

## 3.4 Tworzenie initrd

Przygotowałem skrypt *make\_initrd* (patrz koniec dokumentu), który może działać zarówno na zainstalowanej już Fedorze jak i FreeBSD z drzewem Fedory umiejscowionym gdzieś w podkatalogach. Skrypt ten w pierw należy wyedytować i ustawić szereg podstawowych zmiennych :

1. *RESOURCE\_DIR* - katalog ze skryptem oraz z wykorzystywanymi przez niego plikami jak *linuxrc\_kk* i *rc.prepare\**

2. *FEDORA\_ROOT\_DIR* - katalog z drzewem Fedory (w przypadku pracy na tejże Fedorze zmienna ma być pusta)
3. *IS\_FREEBSD* - wybór systemu na jakim pracujemy ("no" to Linux)
4. *OUTDIR* - katalog, w którym zostanie umieszczony generowany obraz
5. *ETC\_DIR* - względne wobec korzenia drzewa Fedory przeniesienie /etc (np. *template/etc* lub *etc*)
6. *TMP\_MOUNT* - tymczasowe zamontowanie obrazu (najlepiej w /tmp lub w katalogu domowym użytkownika)
7. *EGG* oraz *ARCH* - jeśli skrypt wywołujemy z FreeBSD to wersja jądra i architektura Fedory
8. *COMPRESS\_MODULES* - czy kompresować moduły w paczkę CPIO.GZ

Wywołanie programu :

```
# make_initrd względna_lokalizacja_initrd.img [--nocomp]
```

(ostatni parametr "*-nocomp*" wyłączy kompresję wynikowego obrazu)

### 3.4.1 System plików

**Tworzenie** Initrd to system plików ext2fs. Do jego utworzenia potrzebny będzie *dd* oraz *mke2fs*, a uzyskamy to komendami :

```
dd if=/dev/zero of=$NAZWA_OBRAZU bs=1M count=$ROZMIAR_OBRAZU
mke2fs -q -F -m 0 -b 1024 $NAZWA_OBRAZU
```

Rozmiar obrazu jest bardzo istotnym parametrem - będzie to de facto rozmiar początkowego RAM-dysku, więc musi ewentualnie uwzględnić miejsce na wszystkie pliki generowane przez skrypt *linuxrc*.

Zamontowanie utworzonego obrazu w podkatalogu w Linuksie :

```
mount -t ext2 -o loop $NAZWA_OBRAZU $MIEJSCE_MONTOWANIA
```

(w przypadku FreeBSD należy skorzystać w pierw z *mdconfig* i później *mount* na danym urządzeniu *md*)

**Drzewo katalogów** W zamontowanym obrazie tworzymy strukturę katalogów i kopiujemy podstawowe pliki wykonywalne (wszystkie, które zostaną użyte przez skrypt *linuxrc*) do */sbin* (katalog */bin* jest linkiem symbolicznym do */sbin*) jak : *nash*, *insmod.static*, *udev.static*, *ash.static*. Potrzebne będą też dynamicznie linkowane binaria jak : *mkdir*, *mknod*, *kill* oraz *killll*, *ls*, *sort*, *awk*, *grep*, *cut* (wymagane przez skrypt wykrywający typ karty sieciowej), *mv*, *cp*, *bash*, *cpio* i *g(un)zip*, *mount* i *umount*, *pivot\_root*, *chroot*, *dhclient*, *route*, *ifconfig*, *lspci*.

**/lib** Wszystkie potrzebne biblioteki poprawnie wskazał program *ldd* - je również należy skopiować (do */lib*)

**/etc** Tworzymy */etc*, w którym powinny znaleźć się : *udev/udev.conf*, *fstab*, *passwd*, *shadow*, *group* (pliki z kontami i grupami wystarczy żeby zawierały konto roota) i *dhclient.conf*.

**/dev** Jak zostało już wymienione w katalogu */dev* powinno znaleźć się tylko kilka podstawowych urządzeń : *console*, *null*, *zero*, *systty*, *tty[0-6]*, *ram[01]*.

**Moduły** Należy przygotować również moduły dla jądra. Istotna jest oczywiście zgodność wersji.

**/var** W katalogu */var* wystarczy tylko obecność pustego pliku dla *dhclient* - */var/lib/dhcp/dhclient.leases*.

**Skrypty** W korzeniu *initrd* tworzymy nasz skrypt startowy *linuxrc* i tworzymy dowiązanie symboliczne pliku */bin/sh* do niego. Oczywiście musi być wykonywalny. Kopiujemy również pozostałe skrypty do odpowiednich lokalizacji : *dhclient-script*, *rc.prepare\_var\_tmp* i *rc.prepare\_etc*. Należy pamiętać, że wszystkie skrypty wykonywane w drzewie *initrd*, nie mogą być interpretowane przez */bin/sh*, gdyż to jest link do *linuxrc*. W zamian można skorzystać z *ash.static* lub *bash*.

### 3.4.2 Zapisanie *initrd*

Na koniec należy wykonać kilka komend *sync*, aby opróżnić bufor zapisu, odmontować *initrd* i ewentualnie można spakować go GZIP-em (wszystkie praktycznie jądra od dawna obsługują spakowany *initrd*, aczkolwiek testy na wielu różnych wersjach nie były przeprowadzane).

## 4 Moje skrypty

Aktualne skrypty w kozikowym projekcie " *Fedora Core - NFS-readonly-root*" znaleźć można pod adresem :

[http://acn.waw.pl/koziol/projekty/fedora\\_nfs\\_readonly\\_root/](http://acn.waw.pl/koziol/projekty/fedora_nfs_readonly_root/)

O obejmuje to m.in.:

1. *make\_initrd* - przygotowanie obrazu initrd
2. *linuxrc\_kk* - skrypt linuxrc (wykorzystywany przez *make\_initrd* do zbudowania initrd.img)
3. *dhclient-script\_kk* - skrypt dla *dhclient* (j.w.)
4. *rc.prepare\_etc* - przygotowanie /etc (j.w.)
5. *rc.prepare\_var\_tmp* - przygotowanie /var (j.w.)
6. *prepare\_diskless\_root* - przygotowanie drzewa Fedory już znajdującego się na serwerze do serwowania maszynom diskless

## 5 Do zrobienia

1. Poprawa błędów podczas podnoszenia Fedory (np. Kudzu, ssh).
2. Przygotowanie skryptu ułatwiającego instalację oraz aktualizację oprogramowania w drzewie Fedory na serwerze FreeBSD (w ogólności dowolnym NIX).
3. X-y...
4. Uwarzenie specjalnego piwka na okazję zakończenia pracy ;)))... jasne :)...

## 6 Literatura

1. " *Build the initrd image*" -  
<http://www.faqs.org/docs/evms/x3834.html>
2. " *Building a root filesystem*" -  
[http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html\\_single/Bootdisk-HOWTO.html#BUILDROOT](http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/html_single/Bootdisk-HOWTO.html#BUILDROOT)
3. " *Network Booting of Diskless Systems*" -  
<http://hepwww.ph.qmul.ac.uk/l1calo/sweb/software/diskless.txt>

4. "*InitRD Howto Examples*" -  
<http://www.linux-boot.net/InitRD/Howto/>
5. "*Fedora - Stateless Linux*" -  
<http://fedora.redhat.com/projects/stateless/>
6. "*Linux Kernel 2.4 Internals*" -  
[http://www.faqs.org/docs/kernel\\_2\\_4/lki-1.html](http://www.faqs.org/docs/kernel_2_4/lki-1.html)
7. "*The Linux BootPrompt-HowTo*" -  
<http://www.tldp.org/HOWTO/BootPrompt-HOWTO.html>