LINUX SERVER ADMINISTRATION

DOCUMENTATIE CURS

Ghid compilare kernel

De cele mai multe ori compilarea kernelului se realizeaza cu ajutorul unui ghid de compilare. Exista mai multe astfel de ghiduri si anume pasi care trebuie urmati pentru compilarea kernelului. Toate acestea au in comun aceiasi pasi importanti.

Ghidul de compilare prezentat mai jos a fost testat cu succes pe majoritatea distributiilor de Linux. In exemplul nostru compilarea s-a realizat pentru o distributie FEDORA. Puteti urma aceiasi pasi identici sau doar cu mici modificari pentru orice alta distributie.

Etape compilare kernel

Etapa 1
Download ultima versiune de kernel de la www.kernel.org, decomprimarea si dezarhivarea sa intr-un director la alegere. Dupa decomprimarea arhivei ne mutam in directorul nou creat.

Important

Toate operatiile si comenziile care vor urma se vor executa din directorul care contine sursele kernelului. Pana la ultima etapa si anume restartarea calculatorului ramanem in acest director.
In exemplul acestui curs sursele kernelului se downloadeaza in /home/dan, iar in urma decomprimarii arhivei cu sursele rezulta directorul /home/dan/linux-2.6.25.12. Toate comenzile vor fi executate din directorul /home/dan/linux.2.6.25.12
In functie de versiunea kernelului si de directoarele din propriul sistem, caile absolute si relative pot diferi. Acestea trebuie ajustate corespunzator.

Exemplu

```
1. cd /home/dan
2. wget -c http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.25.12.tar.bz2
3. tar -xjvf linux-2.6.25.12.tar.bz2
```

Etapa 2 (Aceasta etapa este optionala, dar foarte recomandata)
Fiind cea mai importanta componenta a sistemului, este aproape obligatoriu sa verificam semnatura digitala a arhivei. Aceasta operatie garanteaza faptul ca avem o versiune de kernel originala (nemodificata si downloadata fara erori).

Pentru verificarea semnaturii digitale a unui fisier avem nevoie de urmatoarele:
1. Fisierul a carui semnatura digitala se verifica (in acest caz arhiva care contine sursele kernelului)
2. Fisierul care reprezinta semnatura digitala. In acest exemplu acesta este linux-2.6.25.12.tar.bz2.sign. Acesta se obtine din acelasi loc din care se downloadeaza kernelul (www.kernel.org)
3. Cheia publica a celui care a semnat fisierul si anume producatorul kernelului.
4. Un program care verifica semnatura digitala si care este invariabil gpg.

Nota

Intreaga teorie legata de criptografie si semnaturi digitale este prezentata in cursul "Advanced Linux & Infosec" disponibil de asemenea online.

Exemplu

1. Download semnatura digitala
```
wget http://www.kernel.org/pub/linux/kernel/v2.6/linux-2.6.25.12.tar.bz2.sign
```

2. Importarea chei publice cu care s-a semnat. Fiecare cheie are un ID unic.
```
gpg --keyserver wwwkeys.pgp.net --recv-keys 517d0f0e
```

3. Verificare
```
gpg --verify /home/dan/linux-2.6.25.12.tar.bz2.sign /home/dan/linux-2.6.25.12.tar.bz2
```

Etapa 3
Kernelul foloseste un fisier de configurare numit .config. Acesta se numeste "kernel configuration starting point".
La acest pas copiem fisierul de configurare .config al kernelului care ruleaza in directorul cu sursele kernelului abia downloadat si care se doreste a fi compilat.

Fisierul .config pentru kernelul care ruleaza se gaseste cel mai probabil in directorul /usr/src/kernels/VERSIUNE_KERNEL_CARE_RULEAZA sau in directorul /boot

Nota

Pe Fedora Core 9 fisierul de configurare al kernelului care ruleaza este /boot/config-2.6.25-14.fc9.i686 pentru versiunea 2.6.25.14

Nota

Compilarea unui nou kernel presupune pornirea de la o baza de facilitati si anume cele oferite de kernelul care ruleaza. Practic pentru noul kernel adaugam facilitati noi sau renuntam la facilitati in raport cu "running kernel". Acesta este motivul pentru care avem nevoie de .config al kernelului care ruleaza.

Daca .config nu exista in directoarele specificate mai sus, se poate downloada un kernel in format RPM pentru distributia si arhitectura respectiva si i se foloseste fisierul .config.

Fisierul de configurare al kernelului este format din optiunile care vor fi incluse in kernel, una pe cate o linie urmate de y (yes) daca se compileaza in imaginea kernelui sau m daca se compileaza ca modul.

Exemplu

1. cp /boot/config-2.6.25-14.fc9.i686 /home/dan/linux-2.6.25.12
2. cd /home/dan/linux-2.6.25.12
3. mv config-2.6.25-14.fc9.i686 .config

Etapa 4
Executam comanda: make oldconfig

Atentie ! Aceasta comanda precum si urmatoarele se executa din directorul cu sursele kernelului.

Rularea comenzii determina aparitia unei interfete text care foloseste fisierul de configurare de la kernelul vechi (.config) pe care il modifica astfel incat sa introduca noile optiuni din kernelul nou care se compileaza, in vechiul fisier de configurare.

Etapa 5
Executam comanda: make menuconfig

Comanda make gconfig reprezinta o alternativa grafica la make menuconfig
Rularea comenzii determina aparitia unei interfete text sau grafice pentru a alege optiunile ce vor fi incluse in kernel la compilare.

Important

Acesta este momentul in care trebuie sa adaugam optiunile pe care le dorim in kernel, sa le scoatem pe cele in plus etc. Pentru compilarea kernelului in cele mai optime conditii este nevoie de o cunoastere foarte buna a hardware-ului hostului.

Etapa 6
Executam comanda: make

In acest moment are loc compilarea kernelului si a modulelor, rezultand imaginea comprimata a acestuia numita bzImage. Aceasta operatie poate sa dureze mult (1-2 ore).

Etapa 7
Executam: cp System.map /boot/System.map-2.6.25.12
ln -sf /boot/System.map-2.6.25.12 /boot/System.map

Un simbol reprezinta un nume de variabila sau un nume de functie.
System.map reprezinta un tabel cu echivalenta dintre numele simbolurilor si adresele acestora din memorie si este folosit de kernel.

Etapa 8
Executam: make modules_install
Comanda de mai sus instaleaza modulele.

Etapa 9
Executam: cp arch/x86/boot/bzImage /boot/vmlinuz-2.6.25.12
In functie de arhitectura directorul x86 poate sa fie i386.

bzImage vine de la "big zImage" - Imaginea kernelului comprimata cu algoritmul zlib
Cu toate ca numele nu este important, prin conventie se foloseste denumirea de vmlinuz sau bzImage pentru imaginea binara comprimata a kernelului.

Nota

Intreg kernelul se gaseste intr-un singur fisier si anume fisierul bzImage sau vmlinuz. Sursele kernelului nu mai sunt necesare si pot fi sterse.

Etapa 10
Executam: /sbin/mkinitrd /boot/initrd-2.6.25.12.img 2.6.25.12 pentru RedHad/Fedora/SuSE sau mkinitramfs -o /boot/initrt-2.6.25.12.img 2.6.25.12 pentru Ubuntu/Kubuntu/etc

Exemplu: Se doreste suport pentru RAID, LVM sau SATA sau pentru alte functionalitati low level. Comanda de mai sus creaza un RAM Disk care contine modulele necesare la butarea sistemului.

Nota

Primul argument al comenzii mkinitrd este fisierul de tip RAM Disk care se va crea. Al doilea argument si anume 2.6.25.12 este numele directorului din /lib/modules in care se gasesc modulele noului kernel. Automat in fata argumentului 2 se pune /lib/modules/

Etapa 11
Se modifica bootloaderul si anume grub.conf pentru a adauga noul kernel.

Daca se foloseste Grub2 se ruleaza: update-grub2

Nota

Kernelul compilat nu este inca folosit de sistem. Acesta va fi folosit doar dupa restartarea calculatorului si alegerea sa din lista de kernele din meniul Grub. Referirea la vechiul kernel din fisierul de configurare al Grub-ului NU se sterge !

Important

Posibilitatea ca sistemul sa nu mai buteze datorita unei erori sau datorita lipsei unei optiuni incluse in kernel este destul de mare. Este extrem de important sa nu stergem vechiul kernel (cel butabil) pentru a buta cu el in caz de nevoie.
Eroarea fatala care poate sa apara este KERNEL PANIC.

Compilare kernel

Durata: 13.23 min
Marime: 11.86MB