11. Хардуерни настройки

• Управление на файлове свързани с устройства (dev);  
• Основни файлове с хардуерна информация в /proc и /sys ;  
• Инструменти за откриване на хардуерна информация (dmesg, lspci, lsusb ) ;  
• Конфигуриране на компоненти и модули на kernel (modinfo, lsmod, insmod, rmmod,  
depmod, modprobe, sysctl) ;

.

**/dev**

/ dev е местоположението на специални или устройствени файлове. Това е много интересна директория, която подчертава един важен аспект на Linux файловата система - всичко е файл или директория. Погледнете през тази директория и трябва да се надяваме да видите hda1, hda2 и т.н., които представляват различните дялове на първото главно устройство на системата. / dev / cdrom и / dev / fd0 представляват вашето CD-ROM устройство и флопи дисково устройство. Това може да изглежда странно, но ще има смисъл, ако сравните характеристиките на файловете с тези на вашия хардуер. И двете могат да се четат и пишат. Вземете например / dev / dsp. Този файл представлява устройството на високоговорителя. Всички данни, записани в този файл, ще бъдат пренасочени към вашия високоговорител. Ако опитате 'cat / boot / vmlinuz> / dev / dsp' (на правилно конфигурирана система), трябва да чуете звук от високоговорителя. Това е звукът на ядрото ви! Файлът, изпратен на / dev / lp0, се отпечатва. Изпращането на данни и четенето от / dev / ttyS0 ще ви позволи да комуникирате с свързано устройство - например вашия модем.

По-голямата част от устройствата са блокови или знакови устройства; съществуват и могат да бъдат създадени и други видове устройства. Като цяло „блоковите устройства“ са устройства, които съхраняват или съхраняват данни, „устройства с характер“ могат да се разглеждат като устройства, които предават или прехвърлят данни. Например дисковите дискове, твърдите дискове и CD-ROM устройствата са всички блокови устройства, докато серийните портове, мишките и паралелните принтерни портове са всички символни устройства. Има някаква схема за наименуване, но в по-голямата част от случаите те са напълно нелогични.

Някои общи файлове с устройства, както и еквивалентните им копия в Windows, които може да искате да запомните, са:

/ dev / ttyS0 (Първи комуникационен порт, COM1)

Първи сериен порт (мишки, модеми).

/ dev / psaux (PS / 2)

PS / 2 връзка за мишка (мишки, клавиатури).

/ dev / lp0 (Първи порт за принтер, LPT1)

Първи паралелен порт (принтери, скенери и др.).

/ dev / dsp (Първо аудио устройство)

Името DSP идва от термина цифров процесор на сигнала, специализиран процесорен чип, оптимизиран за анализ на цифров сигнал. Звуковите карти могат да използват специален DSP чип или могат да изпълняват функциите с няколко дискретни устройства. Други термини, които могат да се използват за това устройство, са дигитализиран глас и PCM.

/ dev / usb (USB устройства)

Тази поддиректория съдържа повечето от възлите на USB устройства. Разпределенията на имена на устройства са доста опростени, така че не е необходимо да се изработва.

/ dev / sda (C: SCSI устройство)

Първо SCSI устройство (HDD, Memory Sticks, външни устройства за съхранение на данни като CD-ROM устройства на лаптопи и др.).

/ dev / scd (D: SCSI CD-ROM устройство)

Първо SCSI CD-ROM устройство.

/ dev / js0 (стандартен джойстик за игра)

Първо устройство за джойстик.

Устройствата са дефинирани по тип, като „блок“ или „знак“, и „голям“ и „малък“ номер. Основният номер се използва за категоризиране на дадено устройство, а малкият номер се използва за идентифициране на конкретен тип устройство. Например, всички IDE устройства, свързани към първичния контролер, имат голям брой 3. Главни и подчинени устройства, както и отделни дялове са допълнително дефинирани чрез използването на малки числа. Това са двете числа, предхождащи датата на следния дисплей:

# ls -l / dev / hd \*

Основният брой както за hda, така и за hdb устройствата е 3. Разбира се, малкият брой се променя за всеки отделен дял. Дефиницията на всяка категория на главния номер може да бъде разгледана чрез разглеждане на съдържанието на файла /usr/src/linux/include/linux/major.h. Устройствата.txt също документират основни и второстепенни номера. Той се намира в директорията / usr / src / linux / Documentation. Този файл определя основните номера. Почти всички устройства с файлове се създават по подразбиране по време на инсталиране. Въпреки това, винаги можете да създадете устройство, използвайки командата mknod или скрипта MAKEDEV, който се намира в самата директория / dev. Устройствата могат да бъдат създадени с тази програма, като предоставят устройството, което ще бъде създадено, типа на устройството (блок или символ) и главните и малките номера. Например, да речем, че случайно сте изтрили / dev / ttyS0 (COM1 под Windows), той може да бъде пресъздаден с помощта на следната команда

# mknod ttyS0 c 4 64

За тези от нас, които са доста мързеливи, можете просто да стартирате скрипта MAKEDEV като такъв

# MAKEDEV \*

което ще създаде всички известни устройства.

Ако е възможно, / dev може също така да съдържа MAKEDEV.local за създаване на файлове на локални устройства.

Като цяло и както се изисква от FSSTND, MAKEDEV ще има разпоредби за създаване на всяко устройство, което може да бъде намерено в системата, а не само тези, които конкретна инсталация се инсталира.

За тези от вас, които се чудят защо Linux използва такава примитивна система за референтни устройства, защото не успяхме да създадем достатъчно усъвършенстван механизъм, който да осигурява достатъчно предимства пред сегашната система, за да постигне широко разпространение.

Към днешна дата (от версия на ядрото 2.4), най-добрият опит е направен от Richard Gooch от CSIRO. Нарича се devfsd и вече няколко години е част от ядрото. Той е санкциониран от разработчиците на ядрото и самият Линус, а подробностите за неговата реализация могат да бъдат намерени на адрес / usr / src / linux / Documentation / filesystems / devfs / README. По-долу е извадка от този документ.

Devfs е алтернатива на "истинския" характер и блокира специални устройства в кореновата файлова система. Драйверите на ядрото на устройството могат да регистрират устройства по име, а не по големи и малки числа. Тези устройства ще се появят автоматично в devfs, с каквато и да е собственост и защита по подразбиране, които драйверът е посочил. Демон (devfsd) може да се използва за отмяна на тези настройки по подразбиране. Devfs е в ядрото от 2.3.46.

ЗАБЕЛЕЖКА, че devfs е изцяло незадължително. Ако предпочитате старите дискови устройства, просто оставете CONFIG\_DEVFS\_FS = n (по подразбиране). В този случай нищо няма да се промени. Също така имайте предвид, че ако активирате devfs, настройките по подразбиране са такива, че се поддържа пълна съвместимост със старите имена на устройства.

Има два аспекта на devfs: единият е пространството за имена на базовото устройство, което е пространство от имена точно като всяка монтирана файлова система. Другият аспект е кодът на файловата система, който предоставя изглед на пространството на имената на устройствата. Причината да направя разлика е, че devfs може да се монтира много пъти, като всеки монтиран елемент показва същото пространство за имена на устройства. Направените промени са глобални за всички монтирани файлови системи на devfs. Също така, защото пространството на имената на devfs съществува без никакви devfs mounts, можете лесно да монтирате кореновата файлова система, като се позовавате на запис в пространството на имената на devfs.

Цената на devfs е малко увеличение на размера на кода на ядрото и използването на паметта. Около 7 страници код (някои от тях в \_\_init раздели) и 72 байта за всеки запис в пространството от имена. Скромната система има само няколко стотни устройства, така че това струва още няколко страници. Сравнете това с предложението да поставите / dev на ramdisc.

На една типична машина цената е под 0,2 процента. При скромна система с 64 МБ RAM, цената е под 0,1%. Обвиненията за "bloatware", изравнени в devfs, не са оправдани.

# /PROC/SYS/

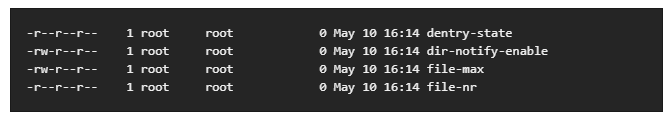
Директорията / proc / sys / е различна от другите в / proc /, защото не само предоставя информация за системата, но и позволява на системния администратор незабавно да активира и деактивира функциите на ядрото.

**Внимание**

**Бъдете внимателни, когато променяте настройките на производствената система, като използвате различните файлове в директорията / proc / sys /. Промяната на грешната настройка може да направи ядрото нестабилно, което изисква рестартиране на системата.**

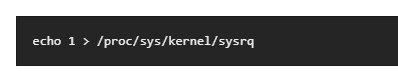
**Поради тази причина се уверете, че опциите са валидни за този файл, преди да се опитате да промените някоя стойност в / proc / sys /.**

Добър начин да определите дали даден файл може да бъде конфигуриран, или ако той е предназначен само за предоставяне на информация, е да го изброите с опцията -l в командния ред. Ако файлът е записваем, той може да се използва за конфигуриране на ядрото. Например частичен списък на / proc / sys / fs изглежда по следния начин:



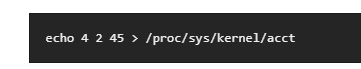
В тази малка обява за файловете dir-notify-enable и file-max можете да въведете и запишете дали можете да използвате за конфигуриране на ядрото. Другите файлове предлагат само обратна информация за текущите настройки.

Промяната на стойността в / proc / sys / файл се повтаря чрез повторение на новата стойност на файла. Например, за да активирате ключа за системна заявка за работа на ядре, въведете командата:



Това означава, че стойността на системата е от 0 (изключено) до 1 (включено).

Лко За да изпратите правилно нови стойности към тях, поставете интервал между всяка стойност, подадена с командата echo, както е посочена в този пример:

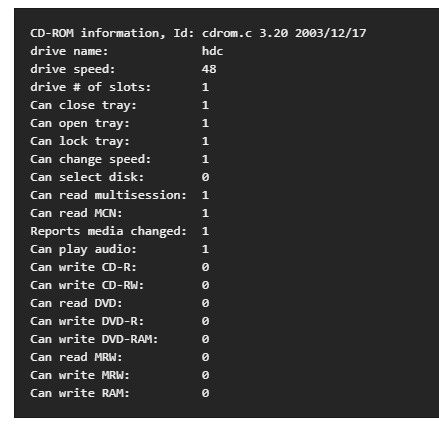


Директорията / proc / sys / съдържа няколко поддиректории, които контролират различни аспекти на работещо ядро.

/ Ргос / сис / сътрудничество /

Тази директория предоставя параметри за конкретни устройства в системата. Повечето системи имат поне две директории, cdrom / и raid /. Персонализираните ядра могат да имат други директории, като например parport /, което дава възможност за споделяне на един паралелен порт между множество драйвери на устройства.

Директорията cdrom / съдържа файл наречен info, който разкрива редица важни параметри на CD-ROM:



Този файл може бързо да бъде сканиран, за да открие качествата на неизвестен CD-ROM. Ако в системата има няколко CD-ROM диска, на всяко устройство се дава собствена колона с информация.

Различни файлове в / proc / sys / dev / cdrom, като autoclose и checkmedia, могат да се използват за управление на CD-ROM на системата. Използвайте командата echo, за да активирате или деактивирате тези функции.

Ако RAID поддръжката е компилирана в ядрото, директорията / proc / sys / dev / raid / става достъпна с поне два файла в нея: speed\_limit\_min и speed\_limit\_max. Тези настройки определят ускорението на RAID устройствата за I / O интензивни задачи, като например ресинжиране на дисковете.

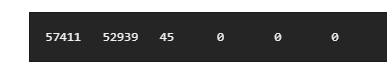
#### **/proc/sys/fs/**

Тази директория съдържа масив от опции и информация, отнасящи се до различни аспекти на файловата система, включително информация за квоти, файлови манипулатори, inode и dentry.

Директорията binfmt\_misc / се използва за осигуряване на поддръжка на ядрото за различни двоични формати.

Важните файлове в / proc / sys / fs / включват:

dentry-state - Предоставя състоянието на кеша на директорията. Файлът изглежда по следния начин:



Първият номер показва общия брой вписвания в кеш директорията, а вторият - броя на неизползваните записи. Третият номер показва броя на секундите между момента на освобождаване на директорията и кога може да бъде възстановена, а четвъртият измерва страниците, които понастоящем се изискват от системата. Последните две числа не се използват и показват само нули.

file-max - Изброява максималния брой файлови манипулации, които ядрото разпределя. Повишаването на стойността в този файл може да разреши грешки, причинени от липсата на налични файлови манипулатори.

file-nr - Показва броя на разпределените файлови манипулатори, използваните файлови манипулатори и максималния брой файлови манипулатори.

overflowgid и overflowuid - Определя съответно фиксирания идентификатор на група и потребителски идентификатор за използване с файлови системи, които поддържат само 16-битови идентификатори на групи и потребители.

#### **/proc/sys/kernel/**

Тази директория съдържа различни конфигурационни файлове, които пряко засягат работата на ядрото. Някои от най-важните файлове включват:

acct - Контролира спирането на процеса на отчитане въз основа на процента свободно пространство, налично във файловата система, съдържаща дневника. По подразбиране файлът изглежда по следния начин:



Първата стойност диктува процента на свободното пространство, необходимо за възобновяване на регистрацията, докато втората стойност задава праговия процент на свободното пространство при спиране на регистрирането. Третата стойност задава интервала, в секунди, че ядрото избира файловата система, за да види дали регистрацията трябва да бъде спряна или възобновена.

ctrl-alt-del - Контролира дали Ctrl + Alt + Delete елегантно рестартира компютъра с init (0) или принуждава незабавното рестартиране, без да синхронизира мръсните буфери с диска (1).

domainname - конфигурира името на системния домейн, например example.com.

exec-shield - Конфигурира функцията Exec Shield на ядрото. Exec Shield осигурява защита срещу определени типове атаки с препълване на буфера.

За този виртуален файл има две възможни стойности:

0 - Деактивира Exec Shield.

1 - Позволява Exec щит. Това е стойността по подразбиране.

**важно**

**Ако системата изпълнява чувствителни към сигурността приложения, които са стартирани, докато Exec Shield е забранена, тези приложения трябва да се рестартират, когато Exec Shield е активиран, за да може Exec Shield да влезе в сила.**

**hostname - конфигурира името на хоста на системата, като www.example.com.**

**hotplug - Конфигурира помощната програма, която ще се използва, когато системата е открила промяна в конфигурацията. Това се използва предимно с USB и Cardbus PCI. Стойността по подразбиране на / sbin / hotplug не трябва да се променя, освен ако не се тества нова програма, за да изпълни тази роля.**

**modprobe - Задава местоположението на програмата, използвана за зареждане на модули на ядрото. Стойността по подразбиране е / sbin / modprobe, което означава, че kmod го призовава да зареди модула, когато нишката на ядрото извика kmod.**

**msgmax - Задава максималния размер на всяко съобщение, изпратено от един процес в друг и е настроено на 8192 байта по подразбиране. Бъдете внимателни, когато повишавате тази стойност, тъй като съобщенията между чакащите процеси се съхраняват в незаменяема памет на ядрото. Всяко увеличение на msgmax би увеличило RAM изискванията за системата.**

**msgmnb - Задава максималния брой байтове в една опашка за съобщения. По подразбиране е 16384.**

**msgmni - Задава максималния брой идентификатори на опашката за съобщения. По подразбиране е 4008.**

**osrelease - Изброява номера на съобщение на ядрото на Linux. Този файл може да бъде променен само чрез промяна на източника на ядрото и прекомпилиране.**

**ostype - Показва вида на операционната система. По подразбиране този файл е настроен на Linux и тази стойност може да бъде променена само чрез промяна на източника на ядрото и прекомпилиране.**

**overflowgid и overflowuid - Определя съответно фиксирания идентификатор на група и потребителски идентификатор за използване със системни повиквания на архитектури, които поддържат само 16-битови идентификатори на групи и потребители.**

**panic - Определя броя на секундите, които ядрото отлага за рестартиране, когато системата изпита паника на ядрото. По подразбиране стойността е зададена на 0, което изключва автоматичното рестартиране след паника.**

**printk - Този файл контролира различни настройки, свързани с отпечатване или регистриране на съобщения за грешки. Всяко съобщение за грешка, съобщено от ядрото, е свързано с него, което определя важността на съобщението. Стойностите на loglevel се разделят в този ред:**

**0 - Спешно ядро. Системата е неизползваема.**

**1 - Сигнал за ядрото. Трябва незабавно да се предприемат действия.**

**2 - Състоянието на ядрото се счита за критично.**

**3 - Общо състояние на грешката на ядрото.**

**4 - Общо предупреждение за ядрото.**

**5 - Известие на ядрото за нормално, но съществено условие.**

**6 - Информационно съобщение на ядрото.**

**7 - Съобщения на ниво ядро.**

**Четири стойности се намират в файла printk:**



**Всяка от тези стойности определя различно правило за справяне с съобщения за грешки. Първата стойност, наречена конзола loglevel, определя най-ниския приоритет на съобщенията, отпечатани в конзолата. (Обърнете внимание, че колкото по-нисък е приоритетът, толкова по-висок е номерът на loglevel.) Втората стойност задава стандартния loglevel за съобщения без прикрепен към тях изричен loglevel. Третата стойност определя най-ниската възможна конфигурация на loglevel за конзолния loglevel. Последната стойност задава стойността по подразбиране за конзолния loglevel.**

**random / directory - Показва редица стойности, свързани с генерирането на случайни числа за ядрото.**

**sem - Конфигурира настройките на семафора в ядрото. Семафор е System V IPC обект, който се използва за контрол на използването на даден процес.**

**shmall - Задава общата сума на споделената памет, която може да се използва едновременно в системата, в байтове. По подразбиране тази стойност е 2097152.**

**shmmax - Задава най-големия размер на сегмента на споделената памет, разрешен от ядрото. По подразбиране тази стойност е 33554432. Въпреки това ядрото поддържа много по-големи стойности от това.**

**shmmni - Задава максималния брой сегменти на общата памет за цялата система. По подразбиране тази стойност е 4096.**

**sysrq - Активира ключа за системни заявки, ако тази стойност е настроена на нещо различно от нула (0), по подразбиране.**

**Ключът System Request позволява незабавно въвеждане на ядрото чрез прости комбинации от клавиши. Например, ключът за системна заявка може да се използва за незабавно изключване или рестартиране на системата, синхронизиране на всички монтирани файлови системи или изхвърляне на важна информация към конзолата. За да инициирате ключ за системна заявка, въведете Alt + SysRq + код за системни заявки. Заменете кода на системния заявка с един от следните кодове на системните заявки:**

**r - Изключва суровия режим за клавиатурата и го задава на XLATE (режим с ограничена клавиатура, който не разпознава модификатори като Alt, Ctrl или Shift за всички клавиши).**

**k - Убива всички процеси, активни във виртуална конзола. Също наричан Secure Access Key (SAK), той често се използва, за да се провери, че подканата за влизане се задейства от init, а не от копие на троянски кон, създадено да улавя потребителски имена и пароли.**

**b - Рестартира ядрото, без първо да демонтира файловите системи или синхронизиращите дискове, свързани към системата.**

**c - Прекъсва системата без първо да демонтира файловите системи или синхронизиращите дискове, свързани към системата.**

**o - Изключва системата.**

**s - Опит за синхронизиране на дискове, свързани с системата.**

**u - Опитвате се да демонтирате и подновявате всички файлови системи като само за четене.**

**p - Извежда всички флагове и регистри в конзолата.**

**t - Извежда списък на процесите към конзолата.**

**m - Извежда статистически данни за паметта в конзолата.**

**0 до 9 - Задава нивото на регистъра за конзолата.**

**e - убива всички процеси с изключение на init, използвайки SIGTERM.**

**i - Убива всички процеси с изключение на init, използвайки SIGKILL.**

**l - Убива всички процеси с помощта на SIGKILL (включително init). Системата е неизползваема след издаването на този код на системния ключ за заявка.**

**h - Показва помощния текст.**

**Тази функция е най-полезна, когато се използва ядрото на разработката или когато системата се замразява.**

**Внимание**

**Функцията Ключ за системните заявки се счита за риск за сигурността, тъй като конзолата без надзор предоставя на нападателя достъп до системата. Поради тази причина тя е изключена по подразбиране.**

**Вижте /usr/share/doc/kernel-doc-kernel\_version/Documentation/sysrq.txt за повече информация относно ключа за системна заявка.**

**tainted - Показва дали е зареден не-GPL модул.**

**0 - Не се зареждат не-GPL модули.**

**1 - Зареден е поне един модул без GPL лиценз (включително модули без лиценз).**

**2 - Поне един модул е ​​бил натоварен с командата insmod -f.**

**threads-max - Задава максималния брой нишки, които да се използват от ядрото, със стойност по подразбиране 2048.**

**version - Показва датата и часа на последното компилиране на ядрото. Първото поле в този файл, като # 3, се отнася до броя пъти, в които ядрото е построено от базата източник.**

**E.3.9.4. / Proc / сис / нето /**

**Тази директория съдържа поддиректории, отнасящи се до различни теми за работа в мрежа. Различни конфигурации по време на компилирането на ядрото правят различни директории достъпни тук, като ethernet /, ipv4 /, ipx / и ipv6 /. Чрез промяна на файловете в тези директории системните администратори могат да коригират мрежовата конфигурация на работеща система.**

**Като се има предвид голямото разнообразие от възможни мрежови опции, достъпни с Linux, се обсъждат само най-често срещаните / proc / sys / net / директории.**

**Директорията / proc / sys / net / core / съдържа различни настройки, които контролират взаимодействието между ядрото и мрежовите слоеве. Най-важните от тези файлове са:**

**message\_burst - Задава максималния брой нови предупредителни съобщения, които да бъдат записани в регистъра на ядрото в интервала от време, определен от message\_cost. Стойността по подразбиране на този файл е 10.**

**В комбинация с message\_cost, тази настройка се използва за налагане на ограничение на скоростта за предупредителни съобщения, записани в регистъра на ядрото от мрежовия код и за смекчаване на атаките с отказ на услуга (DoS). Идеята на DoS атаката е да бомбардира целевата система с искания, които генерират грешки и или да попълват дискови дялове с регистрационни файлове, или да изискват всички ресурси на системата да се справят с регистрирането на грешки.**

**Настройките в message\_burst и message\_cost са предназначени да бъдат модифицирани въз основа на приемливия риск на системата в сравнение с необходимостта от цялостно регистриране. Например, като зададете message\_burst на 10 и съобщение\_кост до 5, вие позволявате на системата да записва максималния брой от 10 съобщения на всеки 5 секунди.**

**message\_cost - Задава стойността на всяко предупредително съобщение, като дефинира интервал от време за message\_burst. Колкото по-висока е стойността, толкова по-вероятно е предупредителното съобщение да бъде игнорирано. Стойността по подразбиране на този файл е 5.**

**netdev\_max\_backlog - Задава максималния брой пакети, позволени за опашка, когато определен интерфейс получава пакети по-бързо, отколкото ядрото може да ги обработи. Стойността по подразбиране за този файл е 1000.**

**optmem\_max - Конфигурира максималния размер на спомагателния буфер, разрешен за сокет.**

**rmem\_default - Задава размера на буфера за приемане на буфер по подразбиране в байтове.**

**rmem\_max - Определя максималния размер на буфера за приемане на сокет в байтове.**

**wmem\_default - Задава размера на буфера за изпращане на сокет по подразбиране в байтове.**

**wmem\_max - Определя максималния размер на буфера за изпращане на сокети в байтове.**

**Директорията / proc / sys / net / ipv4 / съдържа допълнителни мрежови настройки. Много от тези настройки, използвани едно с друго, са полезни за предотвратяване на атаки в системата или при използване на системата като рутер.**

**Внимание**

**Погрешна промяна на тези файлове може да повлияе на отдалечената връзка със системата.**

Следва списък на някои от по-важните файлове в директорията / proc / sys / net / ipv4 /:

icmp\_echo\_ignore\_all и icmp\_echo\_ignore\_broadcasts - Позволява на ядрото да игнорира ICMP ECHO пакети от всеки хост или само тези, произхождащи от излъчвани и мултикаст адреси. Стойност 0 позволява на ядрото да реагира, докато стойността на 1 игнорира пакетите.

ip\_default\_ttl - задава стойността по подразбиране Time To Live (TTL), която ограничава броя на прескачанията, които пакетът може да направи, преди да достигне целта си. Увеличаването на тази стойност може да намали производителността на системата.

ip\_forward - Разрешава интерфейси в системата за препращане на пакети. По подразбиране този файл е настроен на 0. Задаването на този файл на 1 позволява пренасочване на мрежови пакети.

ip\_local\_port\_range - Определя диапазона от портове, които да се използват от TCP или UDP, когато е необходим локален порт. Първият номер е най-ниският порт, който се използва, а вторият номер определя най-високия порт. Всички системи, които очакват да изискват повече портове от стандартните 1024 до 4999, трябва да използват диапазон от 32768 до 61000.

tcp\_syn\_retries - Осигурява ограничение за броя пъти, през които системата предава SYN пакет при опит за свързване.

tcp\_retries1 - Задава броя на разрешените повторни предавания, които се опитват да отговорят на входяща връзка. По подразбиране 3.

tcp\_retries2 - Задава броя на разрешените препредавания на TCP пакети. По подразбиране 15.

Файлът /usr/share/doc/kernel-doc-kernel\_version/Documentation/networking/ip-sysctl.txt съдържа списък с файлове и опции, налични в / proc / sys / net / ipv4 / и / proc / sys / net / ipv6 / директории. Използвайте командата sysctl -a, за да изброите параметрите във формата на ключа sysctl.

В директорията / proc / sys / net / ipv4 / съществуват редица други директории и всеки покрива различен аспект на мрежовия стек. Директорията / proc / sys / net / ipv4 / conf / позволява всеки системен интерфейс да бъде конфигуриран по различни начини, включително използването на настройки по подразбиране за неконфигурирани устройства (в / proc / sys / net / ipv4 / conf / default / subdirectory ) и настройки, които отменят всички специални конфигурации (в / proc / sys / net / ipv4 / conf / all / поддиректорията).

Директорията / proc / sys / net / ipv4 / neigh / съдържа настройки за комуникация с хост, директно свързан със системата (наречен съсед на мрежата) и също така съдържа различни настройки за системи, повече от един хоп.

Маршрутизирането през IPV4 също има своя собствена директория / proc / sys / net / ipv4 / route /. За разлика от conf / и neigh /, директорията / proc / sys / net / ipv4 / route / съдържа спецификации, които се отнасят за маршрутизиране с всякакви интерфейси в системата. Много от тези настройки, като max\_size, max\_delay и min\_delay, се отнасят за контролиране на размера на маршрутизирания кеш. За да изчистите кеша за маршрутизация, запишете всяка стойност в флъш файла.

Допълнителна информация за тези директории и възможните стойности за техните конфигурационни файлове могат да бъдат намерени в:



#### **/proc/sys/vm/**

Тази директория улеснява конфигурирането на подсистемата за виртуална памет (VM) на ядрото на Linux. Ядрото прави широко и интелигентно използване на виртуалната памет, която обикновено се нарича swap space.

В директорията / proc / sys / vm / обикновено се намират следните файлове:

block\_dump - Конфигурира блоковия I / O отстраняване на грешки, когато е активиран. Всички операции за четене / запис и блокиране на замърсявания, които се правят на файловете, се записват съответно. Това може да бъде полезно, ако диагностицирате дисковото въртене и се върти надолу за запазване на батерията на лаптопа. Всички изходи, когато е активирана block\_dump, могат да бъдат извлечени чрез dmesg. Стойността по подразбиране е 0.

Забележка

Ако block\_dump е активиран едновременно с отстраняването на грешки в ядрото, разумно е да се спре демона на klogd, тъй като той генерира грешна дискова активност, причинена от block\_dump.

dirty\_background\_ratio - Стартира фоновото записване на мръсни данни на този процент от общата памет, чрез демонлукс pdflush. Стойността по подразбиране е 10.

dirty\_expire\_centisecs - Определя кога замърсените данни в паметта са достатъчно възрастни, за да могат да бъдат изписвани. Данните, които са били замърсени в паметта по-дълго от този интервал, се изписват следващия път, когато демонът на pdflush се събуди. Стойността по подразбиране е 3000, изразена в стотни от секундата.

dirty\_ratio - Стартира активното записване на мръсни данни на този процент от общата памет за генератора на мръсни данни чрез pdflush. Стойността по подразбиране е 20.

dirty\_writeback\_centisecs - Определя интервала между pdflush daemon wakeups, който периодично записва мръсни данни в паметта на диск. Стойността по подразбиране е 500, изразена в стотни от секундата.

laptop\_mode - Намалява броя на необходимите дискове, като държи диска колкото е възможно по-дълъг, като по този начин пести енергията на батерията на лаптопите. Това увеличава ефективността чрез комбиниране на всички бъдещи I / O процеси заедно, намаляване на честотата на въртене нагоре. Стойността по подразбиране е 0, но автоматично се активира при използване на батерия на лаптоп.

Тази стойност се управлява автоматично от демон acpid, след като потребителят е уведомен за захранването на батерията. Не са необходими потребителски модификации или взаимодействия, ако лаптопът поддържа спецификацията ACPI (Advanced Configuration and Power Interface).

За повече информация вижте следната инсталирана документация:

/usr/share/doc/kernel-doc-kernel\_version/Documentation/laptop-mode.txt

max\_map\_count - Конфигурира максималния брой области на паметта, които процесът може да има. В повечето случаи стойността по подразбиране е 65536.

min\_free\_kbytes - Принуждава Linux VM (мениджър на виртуална памет) да поддържа минимален брой килобайти безплатно. VM използва този номер, за да изчисли стойността на страница\_мин за всяка lowmem зона в системата. Стойността по подразбиране е по отношение на общата памет на устройството.

nr\_hugepages - Показва текущия брой конфигурирани hugetlb страници в ядрото.

За повече информация вижте следната инсталирана документация:

/usr/share/doc/kernel-doc-kernel\_version/Documentation/vm/hugetlbpage.txt

nr\_pdflush\_threads - Показва броя на демоните pdflush, които се изпълняват в момента. Този файл е само за четене и не трябва да се променя от потребителя. При тежки I / O товари, стойността по подразбиране на две се увеличава от ядрото.

overcommit\_memory - Конфигурира условията, при които се приема или отхвърля искане за голяма памет. Налични са следните три режима:

0 - Ядрото изпълнява евристична памет при обработване на обекти, като оценява количеството налична памет и неуспешните заявки, които са явно невалидни. За съжаление, тъй като паметта се разпределя по-скоро чрез евристичен, отколкото с точен алгоритъм, тази настройка понякога позволява претоварване на наличната памет в системата. Това е настройката по подразбиране.

1 - Ядрото не изпълнява памет за обработка на обвързването. При тази настройка, потенциалът за претоварване на паметта се увеличава, но също така и производителността за задачи с интензивна памет (като тези, изпълнявани от някой научен софтуер).

2 - Ядрото не изпълнява никаква заявка за памет, която би накарала общото адресно пространство да надвишава сумата на разпределеното пространство за обмен и процента на физическата оперативна памет, указан в / proc / sys / vm / overcommit\_ratio. Тази настройка е най-подходяща за тези, които желаят по-малък риск от презареждане с памет.

Забележка

Тази настройка се препоръчва само за системи, чиито зони са по-големи от физическата памет.

overcommit\_ratio - Указва процента на физическия RAM, който се разглежда, когато / proc / sys / vm / overcommit\_memory е зададено на 2. Стойността по подразбиране е 50.

page-cluster - Задава броя на страниците, прочетени в един опит. Стойността по подразбиране 3, която всъщност се отнася до 16 страници, е подходяща за повечето системи.

swappiness - определя колко машина трябва да сменя. Колкото по-висока е стойността, толкова повече се извършва размяната. Стойността по подразбиране, като процент, е настроена на 60.

Цялата документация, базирана на ядрото, може да бъде намерена в следното локално инсталирано местоположение:

/ usr / share / doc / kernel-doc-kernel\_version / Документация /, която съдържа допълнителна информация.

Отстраняване на грешки в хардуера на Linux

dmesg, lspci, lsusb

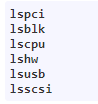
Тази статия ще се съсредоточи върху предоставянето на преглед на различни команди и съвети на Linux, които могат да се използват за отстраняване на грешки в хардуерните проблеми на компютъра ви. Тъй като тази статия е насочена към начинаещи, няма да навлизаме в дълбочина по всяка тема. Затова си струва да се отбележи, че можете да направите много повече с всеки един от тези инструменти, отколкото описаното тук.

     Някои от темите, които се въвеждат включват: Модули, Драйвери, Bootloader на GRUB, монтаж на твърд диск, init процеси, дневници на ядрото и различни други команди за отстраняване на грешки.

     ПРЕДУПРЕЖДЕНИЕ: Редица идеи, обсъждани в тази статия, могат да доведат до загуба на данни, ако не ги използвате правилно. Ако сте начинаещ, моля, внимавайте, когато използвате команди, които използват буутлоудъра, форматирането на твърдия диск, разделянето на файлове и т.н.

Общи

     Понякога ще искате да видите списък на хардуера, който в момента е във вашия компютър. Всяка от следните команди може да се използва за получаване на много подробна информация за вашия хардуер. Всяка команда трябва да ви напомня за командата 'ls', която изброява файловете. Например, 'lspci' се използва за изброяване на информация за PCI устройства, 'lsblk' се използва за изброяване на блокови устройства, 'lscpu' се използва за изброяване на процесори, 'lshw' изброява обща информация за хардуера, 'lsusb' изброява информация за USB устройства \ t и 'lsscsi' изброява информация за SCSI устройства.



Имайте предвид, че изходът, който получавате от горните команди, ще съдържа само хардуера, ако е правилно открит. Има различни проблеми, които могат да доведат до неразкриване на хардуера. Освен това, понякога хардуерът може да се разпознае погрешно като хардуер от друг доставчик.

      Ето друг вариант на командата 'lspci' по-горе, която ще бъде по-подробна и ще отпечатва допълнителна информация, включително имена на драйвери и модули, свързани с устройства:



И накрая, можете да използвате тази команда, която отпечатва повече информация, отколкото вероятно може да обработвате:



Една добра причина да използвате тази команда е да я стартирате и да пренесете резултатите във файл, където те могат да бъдат изпратени на някой друг за доклад за грешка. Това ще им позволи да прегледат каквито и да е подробности, от които се нуждаят, за да отстраняват грешки в някой друг хардуер.

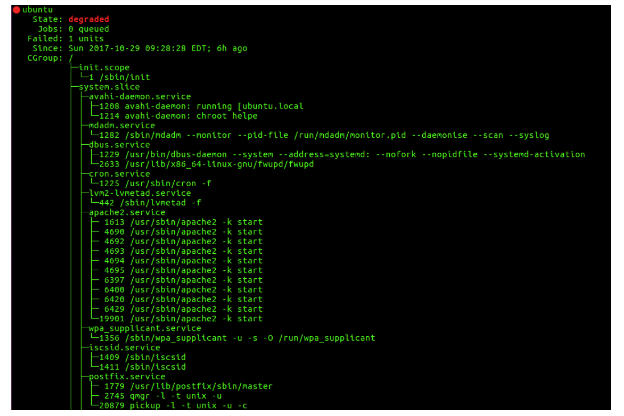


Init процеси

      Важна част от разбирането на начина, по който работи вашата система, е да знаете какъв init процес използвате. Един популярен init процес е systemd. За да разберете дали използвате systemd, изпълнете тази команда:



Ако не използвате systemd, ще видите нещо като 'command not found'. Ако е така, вероятно ще видите нещо подобно:



Горният изход показва състоянието на редица услуги, стартирани чрез systemd.

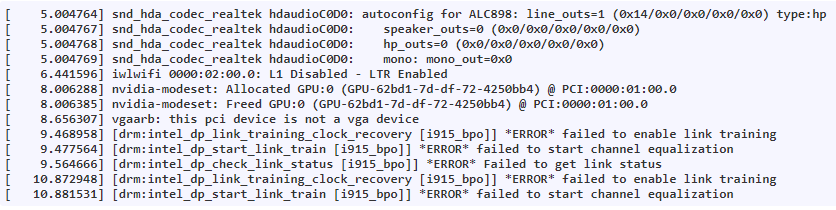
      Тъй като вашият init процес е първото нещо, което започва след стартирането на ядрото, то се занимава с стартирането на редица услуги, някои от които ще взаимодействат директно с вашия хардуер. Ето защо, ако се опитвате да отстраните грешки, които изглеждат като хардуерен проблем, вероятно е добра идея да проверите състоянието на всяка услуга, която е стартирана от вашия init процес, за да се уверите, че функционира правилно.

# Dmesg

Можете да използвате тази команда:



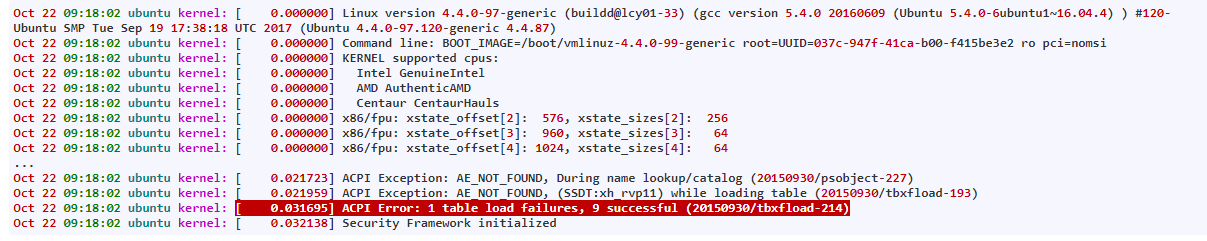
за да ви покаже всички скорошни диагностични съобщения на ядрото. Ето един пример за някои случайни съобщения, които виждам в момента на моя лаптоп (куп неща са счупени, които не съм успял да разгледам):



Тези съобщения представляват още един източник на информация, който можете да използвате за отстраняване на грешки, свързани с хардуера. Често хардуерните проблеми, открити от ядрото след зареждане, се добавят към изхода на командата dmesg.

Отстраняване на грешки в последователността за стартиране

      Когато компютърът се зарежда, в /var/log/kern.log се съхранява дневник на всичко, което се случва. Ето пример за неща, които виждам в дневника си на ядрото:



Ако някога имате проблеми с някой от хардуерните си устройства в Linux, добра идея е да прегледате съдържанието на този регистрационен файл, за да видите дали има съобщения за грешка, свързани с устройството, с което имате проблеми. Този дневник е много подробен и на практика често ще виждате някои съобщения за грешки или предупреждения, които не изискват непременно да правите нещо за тях.

     В допълнение към регистъра на ядрото, може да е полезно да промените настройките на GRUB, така че да можете да виждате съобщенията на ядрото на екрана, когато компютърът ви се опитва да стартира. Това може да се направи, като се уверите, че настройките за "тихи" и "изпръсквам" са премахнати от вашата конфигурация на GRUB. Това ще бъде обсъдено по-подробно по-долу в тази статия.

Драйвери и модули

     Ако прекарате някакво количество време в отстраняване на грешки в хардуерните устройства на вашата Linux машина, накрая ще срещнете концепцията за драйвери и модули. Разликата между драйверите и модулите е малко объркваща и не е добре дефинирана, но ще опиша разликата между двете:

     Драйверът е просто всяко парче код, което се грижи за ниското ниво на взаимодействие с конкретното хардуерно устройство, с което трябва да се свържете. Драйверите могат да се нуждаят или да не работят в изключително привилегирован режим, за да изпълняват специални инструкции, специфични за хардуера, или могат да извършват произволен брой операции на ниско ниво, които взаимодействат с вашия твърд диск, клавиатура, видео карта и др.

     Модулът на ядрото на Linux е пакет от код, който може да бъде добавен и премахнат от ядрото по време на изпълнение. Често, но не винаги, модулът на ядрото на Linux всъщност е драйвер. Затова, ако сте просто начинаещ в Linux, вероятно е ОК, да предположим, че „модулите и драйверите са едно и също нещо“. Когато научите повече, ще станете по-наясно с техните различия. Ето дискусия, която говори повече за разликата между модулите на Linux ядрото и драйверите.

     Можете да използвате командата



за да покажете подробна информация за вашите устройства. Този изход ще изброи модулите на ядрото, които могат да се използват с всяко устройство, и ще изброи текущия драйвер, който се използва за всяко устройство, ако е приложимо.

      Ако искате да видите всички възможни модули на ядрото, които съществуват във вашата система, можете да използвате тази команда:



Горната команда ще изброи всички налични модули, включително тези, които в действителност не са заредени.

      Можете също да видите информация за модула, разглеждаща съдържанието на '/ proc / modules':

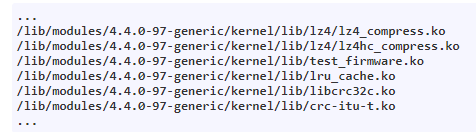


Ако искате да видите къде се съхраняват модулите на ядрото на Linux, можете да проверите '/ lib / modules'. Тази папка ще съдържа папки за всяка версия на ядрото. Ако машината ви е преминала през много обновявания на ядрото, вероятно ще видите много папки тук.

      Например, тук е команда, която ще ми покаже всички модули, които бяха активирани с моето ядро версия 4.4.0-97:



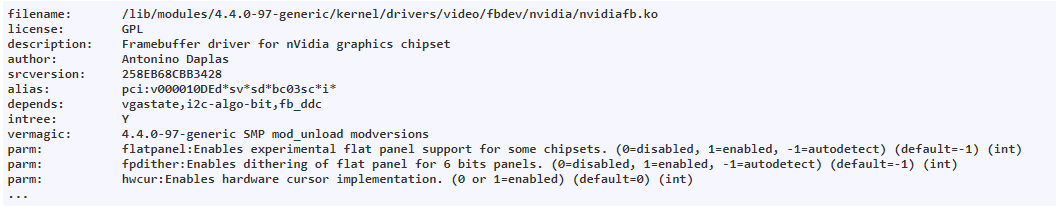
Имайте предвид, че вероятно ще трябва да замените „4.4.0-97-generic“ с нещо друго, освен ако нямате тази версия на ядрото. Ето пример за изход, който виждам от тази команда:



Ако искате да научите информация за конкретен модул на ядрото, можете да използвате командата 'modinfo' с име на модул като аргумент. Например, нека научим повече за модул, наречен „nvidiafb“, който виждам в изхода на командата „lspci -v“:



Ето резултатът, който виждам от горната команда:



И накрая, полезно е също така да знаете, че можете да добавяте и премахвате модули по време на изпълнение с помощта на командата 'modprobe'. За повече информация изпълнете:

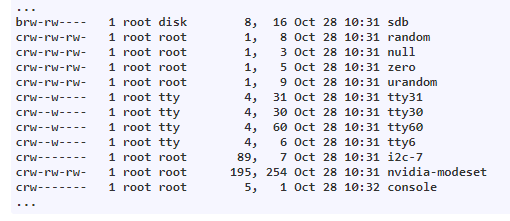


# /dev

Директорията '/ dev' е специална, тъй като съдържа файлови представяния на много устройства във вашата система. Ако изпълните следната команда:



ще видите нещо като следното:



всеки от тези редове представлява „файл“, с който може да се взаимодейства по много от същите начини, както би могъл да бъде и нормалният файл (с някои изключения). Например, има специално устройство, което може да се използва за генериране на случайни числа. Можете да четете от това устройство, както бихте чели от всеки файл със следната команда:

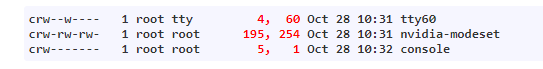


и резултатът от тази команда е:



Можете дори да използвате такива техники, за да отворите целия си твърд диск като файл (често като "/ dev / sda" или "/ dev / sdb") и да го напишете. Въпреки това, ако направите това, ще ви трябват разрешения за корен и по-добре да знаете какво правите в противен случай, вероятно ще направите системата си неподлежаща на зареждане!

      Нещо друго, което трябва да забележите от командата по-горе, са главните и малките числа:



Тези главни и малки номера се появяват там, където иначе бихте видели номер, за да посочите размера на файла за обикновен файл. По принцип всеки основен номер идентифицира драйвера, свързан с това устройство. Един водач може да контролира няколко устройства, всеки от които ще бъде назначен различен малък номер. Ето връзката с още няколко подробности за големи и малки числа. Също така имайте предвид, че 'c' в горния изход показва, че устройството е устройство за символи (а не блок устройство).

      Често можете да отидете в обратната посока и да започнете с големите малки двойки числа и да получите имената на устройствата с помощта на информация, намираща се в '/ sys / dev / block /' или '/ sys / dev / char /' в зависимост от това дали устройството е блок или символ устройство. Например, преди това видяхме, че на моята машина „конзолата“ на устройството имаше главен номер 5 и второстепенен номер 1. Ако изпълнявам следната команда:



резултатът на машината ми е:

