Tvorba modulů I. - Úvod

Vložil/a dr34m [1], 6 Srpen, 2007 - 08:00

- GNU/Linux a BSD [2]
- Programming [3]

Myslím, že většina těch zdatnějších uživatelů GNU/Linuxu si už kompilovala kernel, se kterým se samozřejmě kompilují i nějaké ty moduly. Co jsou to moduly? Jak se vytvářejí, teda programují, a vlastně téměř všechno okolo nich se dozvíte právě v tomto seriálu, jehož úvodní díl právě čtete.

Co je dobré vědět?

Předpokládám alespoň minimální znalosti jazyka C, případně C++, základní informace o tom, co je kernel a jak se kompiluje. Taktéž použití kernelu řady 2.6.x, u mě konkrétně kernel-2.6.22, protože právě pro něj píšu tento seriál, a proto mě nemusíte obtěžovat připomínkami, když vám něco nebude fungovat tak, jak by mělo, a používate přitom řadu 2.4. Na začátek by to stačilo a dále se vám ve zkratce pokusím popsat to, co ten modul vlastně je a povíme si také, jaké druhy kernelu můžeme používat.

Co je to modul?

Modul, resp. moduly, jsou části kódu napsané v programovacím jazyku - většinou je to C, ve kterém je napsaný i samotný kernel, ale některé věci jsou v něm samozřejmě v assembleru, které mohou být podle potřeby načítané, případně odstraněné z kernelu za jeho běhu. Jednoduše se dá říct, že moduly rozšiřují funkcionalitu kernelu bez toho, abyste museli restartovat počítač a tím načítat nový kernel do paměti. Za jejich výhodu je možné považovat to, že nemusíte mít všechno zakompilované do kernelu, tím pádem má jeho obraz (image) menší velikost, čímž zabírá méně místa v paměti. Samozřejmě, že ne všechno může být zkompilované jako modul. Jako příklad uvedu podporu pro IDE řadiče disků. To znamená, že pokud byste podporu pro ně v kernelu zkompilovali jako modul, systém vám nenaběhne a uvidíte jen známý "kernel panic", protože moduly se načítají až potom, co proběhne *init* a stejně tak potom, co se připojí hlavní kořenový adresář, tzn. že podporu pro použitý fs (file system) na něm musíte mít zakompilovanou přímo do kernelu. Některé věci, jako např. ALSA, se doporučují kompilovat jako modul. Existují dva typy kernelu:

1. monolitický kernel (monolithic kernel) - co nejvíce věcí je zakompilovaných přímo do kernelu.
2. modulární kernel (modular kernel) - jen nejnutnější věci, jako už zmíněný IDE řadič nebo fs, jsou přímo v kernelu a všechno ostatní je jako modul.

Většina dnešních distribucí používá právě modulární kernel, protože se jeví jako výhodnější řešení. Kdykoliv totiž můžete daný modul, když ho nepotřebujete používat, odstranit příkazem:

rmmod "nazev_modulu"

Je to určitě výhodnější, než znovu kompilovat celý kernel. Další důvod, proč používat moduly je ten, že když dojde v komunikaci mezi driverem a HW k nějaké chybě a daný driver pro obsluhu tohoto HW máte přímo v kernelu, může dojít k pádu systému, což se v případě, kdy budete mít daný driver zkompilovaný jako modul, nemusí stát, resp. následky této chyby budou mít menší dopad na systém, než kdyby byl driver přímo v kernelu, i když výjimky potvrzují pravidlo :). Na úvod toho, co jsou moduly, by to stačilo. Dále si povíme to, jak pracovat s už zkompilovanými moduly a jak si vlastně umí naše moduly najít cestu ke kernelu.

CC-BY-SA Security-Portal.cz | secured by paranoid sense | we hack to learn

Moduly a kernel

Jaké moduly máte právě načtené, se dozvíte příkazem

lsmod

který zpracovává výpis ze souboru **/proc/modules**. Ptáte se, jak si tyto moduly umí najít cestu ke kernelu, resp. jak ví odkud se mají načítat?

Když kernel potřebuje funkcionalitu, která není přímo v něm, modul/součást kernelu nazvaný/á **kmod** - který se ve starších verzích kernelu nazýval **kerneld** - spustí/inicializuje program **modprobe** a ten daný modul najde a načte. Načte ho buď *podle jeho názvu*, teda např.:

nvidia - modulární driver pro moji graf. kartu
bttv - ovladač pro moji TV kartu

nebo si ho vyhledá *podle unikátního identifikátoru*, což v našem případě pro modul **nvidia** bude **char-major-195** a pro **bttv** je to **char-major-81**. Pokud **modprobe** volá modul *podle unikát. identifikátoru*, jeho název zjišťuje zpravidla ze souboru **/etc/modprobe.conf**, což však u vás nemusí platit, protože každá distribuce ho může mít na jiném místě. To, kde sa nachází právě u vás, si už musíte zjistit vy sami. Když tento soubor najde, vyhledá v něm řádek začínající *aliasem*, teda jakýmsi ukazatelem na daný identifikátor, který označuje modul. U mě bude tento řádek pro modul 'bttv' vypadat následovně:

alias char-major-81-* bttv

a podle něho **modprobe** ví, že právě ten identifikátor (char-major-81-*) ukazuje na můj modul **bttv.ko** a žádný jiný. Ješte jsem zapomněl zmínit, že moduly, které máme už zkompilované, se nacházejí standardně v adresáři **/lib/modules/'verzia-kernelu'/** a mají příponu **.ko** (což však zatím není tak podstatné) a v tomto adresáři jsou umístěné další soubory a podadresáře, a právě v nich se nacházejí naše moduly. Tyto podadresáře jsou vytvářené při kompilovaní kernelu a jsou to např. **kernel, fs**:

kernel - zde se nacházejí moduly, které přímo souvisí s kernelem a vašim HW, teda např. ovladače graf. karty, zvuk. karty, zákl. desky apod.
fs - moduly, které souvisí - jako už samotný název napovídá - s nějakým file systémem. Já tu mám např. modul 'fuse', který slouží k tomu, abych mohl používat ntfs-3g driver a mít připojené ntfs oddíly v rw (read-write) módu.

Některé moduly na sobě závisí. Příklad: používáte balík **lm_sensors**, teda program

sensors

k zjištění teploty, otáček procesoru apod. Abyste ho mohli využívat, potřebujete mít v kernelu zakompilovanou podporu pro čip, který máte na vaší základní desce. U mě je to **i2c_viapro**. Tento modul je však závislý na jiném, resp. obsluhuje ho další modul, protože používá symboly (proměnné nebo funkce), které jsou definované v tomto případě v modulu, který se nazývá **i2c_core**. O to, jaké moduly jsou na sobě navzájem závislé, se stará opět **modprobe**, který si prohlédne soubor nacházející se v už zmíněném adresáři **/lib/modules/'verze-kernelu'/modules.dep**, což v mém případě bude **/lib/modules/2.6.21-gentoo-r2/modules.dep** a právě z něho zjistí, jestli musí být načtené nějaké moduly před tím (i2c_core), než bude načtený modul, který požadujeme (i2c_viapro). Tento soubor se vytváří příkazem:

depmod -a

a obsahuje závislosti mezi vašimi zkompilovanými moduly. Aby načítání modulů proběhlo ve správném pořadí, tedy aby se nejdříve načetl modul 'i2c_core', **modprobe** si na to zavolá další program a to **insmod**, který nám toto zajistí. Kdybyste si chtěli načíst modul 'i2c_viapro', musíte spustit nasledující příkazy:

CC-BY-SA Security-Portal.cz | secured by paranoid sense | we hack to learn

insmod /lib/modules/'verze-kernelu'/kernel/drivers/i2c/i2c-core.ko ; insmod /lib/modules/'verze-kernelu'/kernel/drivers/i2c/busses/i2c-viapro.ko

Toto řešení mi však přijde "trošku" zdlouhavé a popravdě i zbytečné, protože stačí, když jednoduše zadáte tento příkaz:

modprobe i2c-viapro

a 'modprobe' si sám zjistí závislosti pro daný modul ze souboru **/lib/modules/'verzia-kernelu'/modules.dep**, zavolá program **insmod** a ví tedy, v jakém pořadí mají být dané moduly načtené.

Rozdíl mezi **insmod** a **modprobe** je teda jasný. Zatím, co byste chtěli použít pro načtení modulů insmod, musíte si zjistit správné pořadí modulů a stejně tak úplnou cestu k nim, 'modprobe' si 'insmod' zavolá sám a všechnu tu zdlouhavou práci vykoná za vás.

Výpis všech vámi zkompilovaných modulů (načtených i nenačtených) získate příkazem:

modprobe -l

kde se dozvíte taktéž umístění a přípony modulů.

K závěru jen tolik, že linuxové distribuce obsahují nástroje **modprobe, insmod** a **depmod** v balíku *module-init-tools*. Předtím se tento balík nazýval *modutils*.

Závěr

Vím, že jste možná čekali víc a možná to bylo pro vás trochu nudné a mysleli jste si, že se tu dozvíte, jak se takový modul vytváří, ale cílem tohoto úvodního článku bylo obeznámit vás s používáním modulů, dát vám základní informace o nich a hlavně vysvětlit, jak a na co nám moduly slouží. Doufám, že se mi to alespoň z malé časti podařilo a už teď vám mohu říct, že pokud se vám třeba jen trochu líbil tento článek, příště se máte na co těšit. V dalším pokračování našeho seriálu si napíšeme a rozebereme náš první jednoduchý modul s už "zažitým" názvem **Hello World**. Určitě tam bude i něco navíc, ale o tom bude až druhý díl.

Názory, postřehy, připomínky a otázky postujte do komentářů, příp. se ozvěte na můj mail.

Poděkování: **warriant**ovi - za korekci překlepů a za kompletní překlad do češtiny ;)

» powered by #skola [4]

URL článku: https://security-portal.cz/clanky/tvorba-modul%C5%AF-i-%C3%BAvod

Odkazy:
[1] https://security-portal.cz/users/dr34m
[2] https://security-portal.cz/category/tagy/gnu/linux-bsd
[3] https://security-portal.cz/category/tagy/programming
[4] http://skola.security-portal.cz/

CC-BY-SA Security-Portal.cz | secured by paranoid sense | we hack to learn