Kompilacja Kernela Linux

Metoda stara i nowa

Rafał Hrabia
nr indeksu 296583

Informatyka I st.
Uniwersytet Marii Curie-Skłodowskiej
14 czerwca 2022

Spis treści

Rozdział 1

Przygotowanie

W pierwszej kolejności po zalogowaniu się i przejściu do katalogu */usr/src* musimy zdobyć link do paczki z plikami źródłowymi kernela. Z racji tego, że korzystam z wirtualnej maszyny bez interfejsu graficznego posłużę się komputerem, który hostuje maszynę wirtualną. Wchodzimy na stronę i przy użyciu menu kontekstowego kopiujemy do schowka bezpośredni link to pliku(rys. 1.1).

Rysunek 1.1: Kopiowanie linku

Do pobrania pliku jedynym słusznym wyborem będzie komenda *wget*(rys. 1.2).

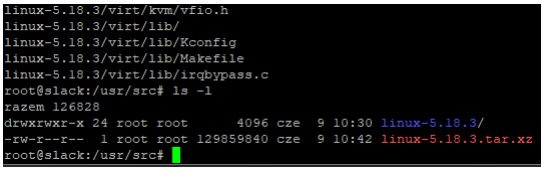
```
root@slack:/usr/src# wget https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.18.3.tar.xz
--2022-06-14 17:24:06--  https://cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.18.3.tar.xz
Translacja cdn.kernel.org (cdn.kernel.org)... 151.101.1.176, 151.101.65.176, 151.101.129.176, ...
Łączenie się z cdn.kernel.org (cdn.kernel.org)|151.101.1.176|:443... połączono.
Żądanie HTTP wysłano, oczekiwanie na odpowiedź... 200 OK
Długość: 129859840 (124M) [application/x-xz]
Zapis do: `linux-5.18.3.tar.xz'

linux-5.18.3.tar.xz        100%[=================================>] 123,84M  1,78MB/s     w 69s

2022-06-14 17:25:15 (1,78 MB/s) - zapisano `linux-5.18.3.tar.xz' [129859840/129859840]

root@slack:/usr/src# ls -l
razem 126824
-rw-r--r-- 1 root root 129859840 cze  9 10:42 linux-5.18.3.tar.xz
root@slack:/usr/src#
```

Rysunek 1.2: Pobieranie kernela

Jako, że pobrany plik jest to plik archiwum używamy na nim komendy *tar -xvpf* (rys 1.3).

Dla przypomnienia (flagi polecenia *tar*):

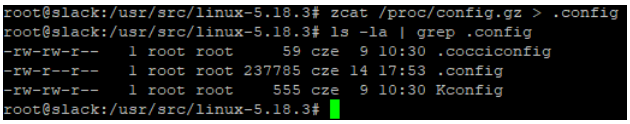
- x – wyodrębnia wymienione pliki
- v – wypisuje nazwy wszystkich plików
- f – określa nazwę pliku archiwum tar
- p - zachowuje ustawienia dostępu do plików



Rysunek 1.3: Rozpakowywanie archiwum

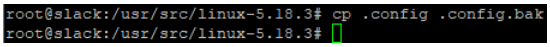
Jak można zauważyć został stworzony nowy folder zawierający pliki z archiwum.

Teraz musimy przekopiować *config* znajdujący się w katalogu */proc* (rys 1.4). Ponieważ plik konfiguracji ma rozszerzenie wskazujące na to, że to archiwum gzip używamy polecenia *zcat*.



Rysunek 1.4: Kopiowanie konfiguracji

Przyda się zrobić kopię zapasową oryginalnego configu. W tym celu kopiujemy plik *.config* do przykładowo pliku *.config.bak* (rys. 1.5).

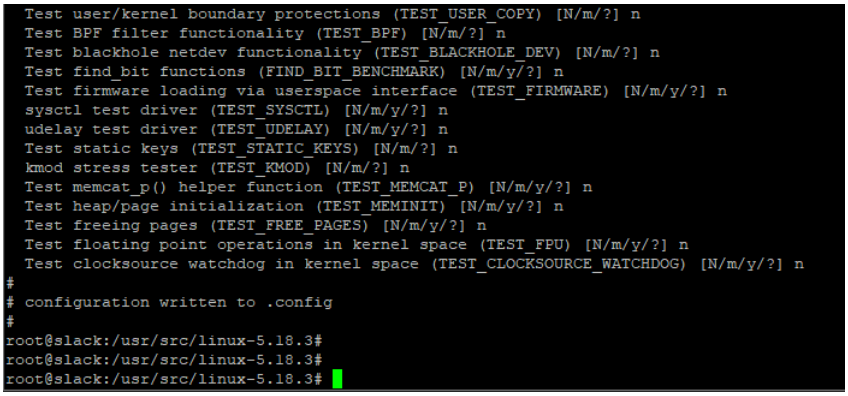


Rysunek 1.5: Kopia zapasowa

Rozdział 2

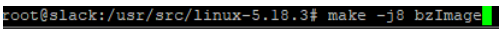
Metoda stara

Przyszła pora na wywołanie polecenia *make localmodconfig*. Po pomyślnym wykonaniu jesteśmy informowani o nadpisaniu konfiguracji do pliku *.config* (rys. 2.1).



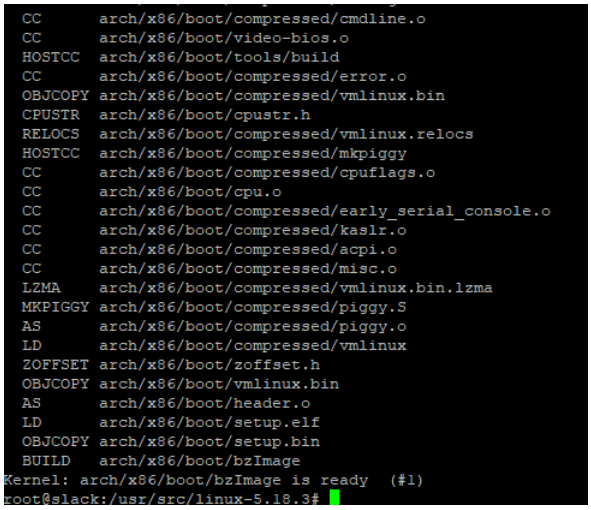
Rysunek 2.1: Wynik polecenia *make localmodconfig*

Nastała wielka chwila - kompilacja jądra. W tym celu trzeba znowu użyć polecenia *make*. Tym razem użyjemy parametru *-j< liczba >* co pozwoli nam uruchomić kompilację kernela na wielu rdzeniach(lub wątkach?). W moim przypadku użyję liczby 8.



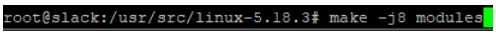
Rysunek 2.2: Kompilowanie kernela

Kompilacja kernela przebiegła w zawrotnym czasie 4 minut i 45 sekund (mniej więcej).



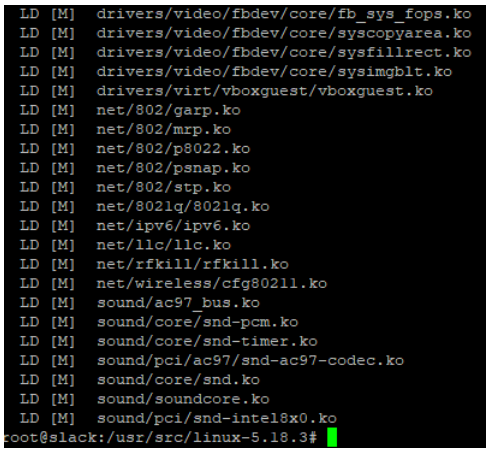
Rysunek 2.3: Kompilowanie kernela - wynik

Kompilację modułów zleca się również komendą make, ale zamieniamy parametr pozycyjny *bzImage* na *modules*.



Rysunek 2.4: Kompilowanie modułów

Budowanie modułów zakończyło się zaskakująco szybko - w niecałe 36 sekund (tak, stałem ze stoperem w ręce). Pytanie czy tak powinno być? Czy to może zasługa Intel Core i9? Podejrzewam, że "stara metoda" ograniczyła ilość modułów do niezbędnego minimum. Przyjmijmy więc, że wszystko jest w porządku i kontynuujmy.



Rysunek 2.5: Kompilowanie modułów - wynik

Rozdział 3

Metoda nowa

Rozdział 4

Wnioski