Kompilacja jądra Linux

Arkadiusz Głąb

Zaczynam, bez względu na metodę od pobrania źródłowej wersji jądra. Robię to ze strony www.kernel.org i wybieram wersję 5.18.5 za pomocą komend:

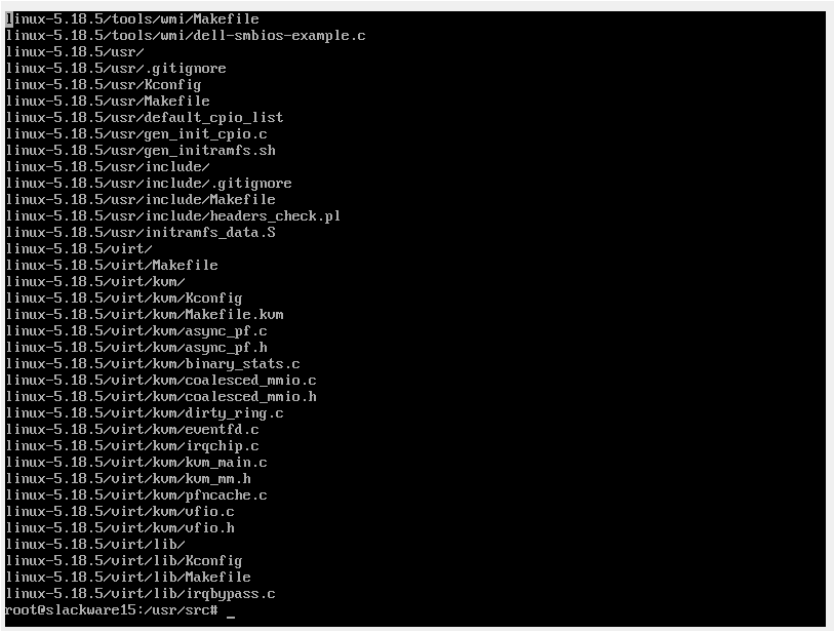
cd /usr/src

i następnie

wget https:/cdn.kernel.org/pub/linux/kernel/v5.x/linux-5.18.5.tar.xz

Działanie komendy:

Następnie rozpakowuje paczkę za pomocą komendy *tar -xvf*. Rozpakowywanie zajęłoby co najmniej 10 stron a4, więc ich wszystkich nie wkleję tylko parę końcowych linijek:



1. Metoda stara – localmodconfig

Obecną konfigurację jądra trzeba skopiować do pliku .config w folderze /usr/src/linux-5.18.5 za pomocą komendy *zcat /proc/config.gz > .config*

```
root@slackware15:/usr/src# zcat /proc/config.gz > .config
root@slackware15:/usr/src# _
```

Następnie w celu generacji pliku przechodzę do folderu /linux-5-18-5 i korzystam z komendy *make localmodconfig* i korzystam z domyślnej konfiguracji akceptując wszystko enterem.

```
root@slackware15:/usr/src# cd linux-5.18.5
root@slackware15:/usr/src/linux-5.18.5# make localmodconfig
  HOSTCC  scripts/basic/fixdep
  HOSTCC  scripts/kconfig/conf.o
  HOSTCC  scripts/kconfig/confdata.o
  HOSTCC  scripts/kconfig/expr.o
  LEX     scripts/kconfig/lexer.lex.c
  YACC    scripts/kconfig/parser.tab.[ch]
  HOSTCC  scripts/kconfig/lexer.lex.o
  HOSTCC  scripts/kconfig/menu.o
  HOSTCC  scripts/kconfig/parser.tab.o
  HOSTCC  scripts/kconfig/preprocess.o
  HOSTCC  scripts/kconfig/symbol.o
  HOSTCC  scripts/kconfig/util.o
  HOSTLD  scripts/kconfig/conf
using config: '/proc/config.gz'
```

Ponownie screenów z akceptowania konfiguracji było zbyt wiele, więc wstawiam tylko przykładowe:

```
  Enable the compressed cache for swap pages by default (ZSWAP_DEFAULT_ON) [N/y/?] n
Common API for compressed memory storage (ZPOOL) [Y/?] y
  Low (Up to 2x) density storage for compressed pages (ZBUD) [Y/?] y
  Up to 3x density storage for compressed pages (Z3FOLD) [Y/n/m/?] y
Memory allocator for compressed pages (ZSMALLOC) [Y/n/m/?] y
  Export zsmalloc statistics (ZSMALLOC_STAT) [N/y/?] n
Defer initialisation of struct pages to kthreads (DEFERRED_STRUCT_PAGE_INIT) [N/y/?] n
Enable idle page tracking (IDLE_PAGE_TRACKING) [N/y/?] n
Support DMA zone (ZONE_DMA) [Y/n/?] y
Support DMA32 zone (ZONE_DMA32) [Y/?] y
Device memory (pmem, HMM, etc...) hotplug support (ZONE_DEVICE) [Y/n/?] y
Unaddressable device memory (GPU memory, ...) (DEVICE_PRIVATE) [Y/n/?] y
Collect percpu memory statistics (PERCPU_STATS) [N/y/?] n
Enable infrastructure for get_user_pages()-related unit tests (GUP_TEST) [N/y/?] n
Read-only THP for filesystems (EXPERIMENTAL) (READ_ONLY_THP_FOR_FS) [N/y/?] n
Anonymous VMA name support (ANON_VMA_NAME) [N/y/?] (NEW)
*
* Core Netfilter Configuration
*
Netfilter ingress support (NETFILTER_INGRESS) [Y/n/?] y
Netfilter egress support (NETFILTER_EGRESS) [Y/n/?] (NEW) _
```

```
  Test functions located in the uuid module at runtime (TEST_UUID) [N/m/y/?] n
  Test the XArray code at runtime (TEST_XARRAY) [N/m/y/?] n
  Perform selftest on resizable hash table (TEST_RHASHTABLE) [N/m/y/?] n
  Perform selftest on siphash functions (TEST_SIPHASH) [N/m/y/?] (NEW)
  Perform selftest on IDA functions (TEST_IDA) [N/m/y/?] n
  Test module loading with 'hello world' module (TEST_LKM) [N/m/?] n
  Test module for compilation of bitops operations (TEST_BITOPS) [N/m/?] n
  Test module for stress/performance analysis of vmalloc allocator (TEST_VMALLOC) [N/m/?] n
  Test user/kernel boundary protections (TEST_USER_COPY) [N/m/?] n
  Test BPF filter functionality (TEST_BPF) [N/m/?] n
  Test blackhole netdev functionality (TEST_BLACKHOLE_DEV) [N/m/?] n
  Test find_bit functions (FIND_BIT_BENCHMARK) [N/m/y/?] n
  Test firmware loading via userspace interface (TEST_FIRMWARE) [N/m/y/?] n
  sysctl test driver (TEST_SYSCTL) [N/m/y/?] n
  udelay test driver (TEST_UDELAY) [N/m/y/?] n
  Test static keys (TEST_STATIC_KEYS) [N/m/?] n
  kmod stress tester (TEST_KMOD) [N/m/?] n
  Test memcat_p() helper function (TEST_MEMCAT_P) [N/m/y/?] n
  Test heap/page initialization (TEST_MEMINIT) [N/m/y/?] n
  Test HMM (Heterogeneous Memory Management) (TEST_HMM) [N/m/y/?] n
  Test freeing pages (TEST_FREE_PAGES) [N/m/y/?] n
  Test floating point operations in kernel space (TEST_FPU) [N/m/y/?] n
  Test clocksource watchdog in kernel space (TEST_CLOCKSOURCE_WATCHDOG) [N/m/y/?] n
#
# configuration written to .config
#
root@slackware15:/usr/src/linux-5.18.5#
```

Następnie kompiluję bazowe jądro komendą *make -j4 bzImage:*

```
#
root@slackware15:/usr/src/linux-5.18.5#
root@slackware15:/usr/src/linux-5.18.5#
root@slackware15:/usr/src/linux-5.18.5# make -j4 bzImage
```

```
  COPY    usr/initramfs_inc_data
  AS      usr/initramfs_data.o
  AR      usr/built-in.a
  CC      arch/x86/coco/core.o
  CC      init/do_mounts_initrd.o
  AR      arch/x86/coco/built-in.a
  CC      init/initramfs.o
  CC      arch/x86/entry/vdso/vma.o
  CC      init/calibrate.o
  CC      init/init_task.o
  CC      init/version.o
  CC      arch/x86/entry/vdso/extable.o
  CC      arch/x86/entry/vdso/vdso32-setup.o
  LDS     arch/x86/entry/vdso/vdso.lds
  AS      arch/x86/entry/vdso/vdso-note.o
  AR      init/built-in.a
  CC      arch/x86/entry/vdso/vclock_gettime.o
  CC      arch/x86/entry/vdso/vgetcpu.o
  CC      kernel/sched/core.o
  AS      arch/x86/entry/vdso/vsgx.o
  CC      kernel/sched/fair.o
  HOSTCC  arch/x86/entry/vdso/vdso2c
  LDS     arch/x86/entry/vdso/vdso32/vdso32.lds
  AS      arch/x86/entry/vdso/vdso32/note.o
```

```
  AS      arch/x86/boot/compressed/efi_thunk_64.o
  CC      arch/x86/boot/compressed/misc.o
  LZMA    arch/x86/boot/compressed/vmlinux.bin.lzma
  MKPIGGY arch/x86/boot/compressed/piggy.S
  AS      arch/x86/boot/compressed/piggy.o
  LD      arch/x86/boot/compressed/vmlinux
  ZOFFSET arch/x86/boot/zoffset.h
  OBJCOPY arch/x86/boot/vmlinux.bin
  AS      arch/x86/boot/header.o
  LD      arch/x86/boot/setup.elf
  OBJCOPY arch/x86/boot/setup.bin
  BUILD   arch/x86/boot/bzImage
Kernel: arch/x86/boot/bzImage is ready  (#1)
root@slackware15:/usr/src/linux-5.18.5#
```

Oraz moduły jądra komendą *make -j4 modules*

```
root@slackware15:/usr/src/linux-5.18.5# make -j4 modules_
```

```
  LD [M]  net/802/psnap.ko
  LD [M]  net/802/stp.ko
  LD [M]  net/8021q/8021q.ko
  LD [M]  net/ipv6/ipv6.ko
  LD [M]  net/llc/llc.ko
  LD [M]  net/rfkill/rfkill.ko
  LD [M]  net/wireless/cfg80211.ko
  LD [M]  sound/ac97_bus.ko
  LD [M]  sound/core/snd-pcm.ko
  LD [M]  sound/core/snd-timer.ko
  LD [M]  sound/core/snd.ko
  LD [M]  sound/pci/ac97/snd-ac97-codec.ko
  LD [M]  sound/soundcore.ko
  LD [M]  sound/pci/snd-intel8x0.ko
```

Po zbudowaniu trzeba je zainstalować, więc robię to poleceniem *make -j4 modules_install*

```
root@slackware15:/usr/src/linux-5.18.5# make -j4 modules_install
```

```
  INSTALL /lib/modules/5.18.5/kernel/net/802/p8022.ko
  INSTALL /lib/modules/5.18.5/kernel/net/802/psnap.ko
  INSTALL /lib/modules/5.18.5/kernel/net/802/stp.ko
  INSTALL /lib/modules/5.18.5/kernel/net/8021q/8021q.ko
  INSTALL /lib/modules/5.18.5/kernel/net/ipv6/ipv6.ko
  INSTALL /lib/modules/5.18.5/kernel/net/llc/llc.ko
  INSTALL /lib/modules/5.18.5/kernel/net/rfkill/rfkill.ko
  INSTALL /lib/modules/5.18.5/kernel/net/wireless/cfg80211.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/ac97_bus.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/core/snd-pcm.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/core/snd-timer.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/core/snd.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/pci/ac97/snd-ac97-codec.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/pci/snd-intel8x0.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soundcore.ko
  DEPMOD  /lib/modules/5.18.5
```

Następnym krokiem jest skopiowanie plików potrzebnych do uruchomienia jądra:

```
root@slackware15:/usr/src/linux-5.18.5# cp arch/x86/boot/bzImage /boot/vmlinuz-5.18.5-smp
root@slackware15:/usr/src/linux-5.18.5# cp System.map /boot/System.map-5.18.5-smp
root@slackware15:/usr/src/linux-5.18.5# cp .config /boot/config-5.18.5-smp
```

Teraz trzeba wygenerować ramdysk w katalogu /boot komendą

mkinitrd_command_generator.sh -k 5.18.5

```
root@slackware15:/boot# /usr/share/mkinitrd/mkinitrd_command_generator.sh -k 5.18.5_
```

```
root@slackware15:/boot# /usr/share/mkinitrd/mkinitrd_command_generator.sh -k 5.18.5
#
# mkinitrd_command_generator.sh revision 1.45
#
# This script will now make a recommendation about the command to use
# in case you require an initrd image to boot a kernel that does not
# have support for your storage or root filesystem built in
# (such as the Slackware 'generic' kernels').
# A suitable 'mkinitrd' command will be:
```

Następnie wykorzystuję wygenerowaną komendę:

```
root@slackware15:/boot# mkinitrd -c -k 5.18.5 -f ext4 -r /dev/sda2 -m ext4 -u -o /boot/initrd.gz

49624 blocks
/boot/initrd.gz created.
Be sure to run lilo again if you use it.
```

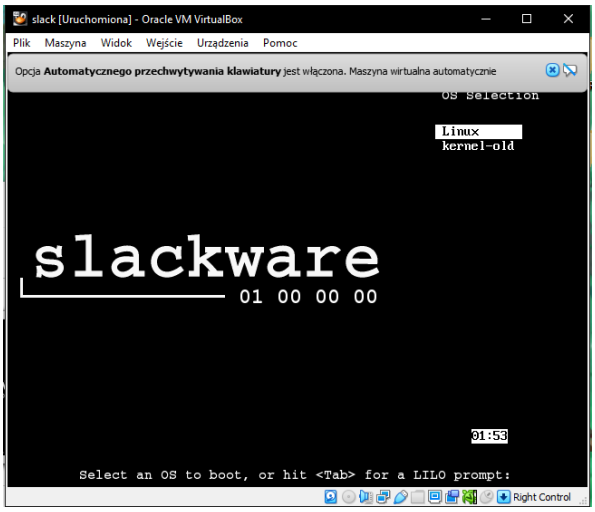
Konfiguruję plik lilo.conf w katalogu /etc/

```
# Linux bootable partition config begins
image = /boot/vmlinuz
  root = /dev/sda2
  label = Linux
  read-only

image = /boot/vmlinuz-5.18.5-smp
  root = /dev/sda2
  initrd = /boot/initrd.gz
  label = "kernel-old"
  read-only
# Linux bootable partition config ends

^G Help      ^O Write Out  ^W Where Is   ^K Cut
```

Sprawdzenie czy został dodany komendą *lilo*

```
root@slackware15:~# lilo
Warning: LBA32 addressing assumed
Added Linux  *
Added kernel-old  +
One warning was issued.
root@slackware15:~# _
```

Restart, by zobaczyć wynik:



Po uruchomieniu kernel-old

```
Welcome to Linux 5.18.5 x86_64 (tty1)

slackware15 login: root
Password:
Last login: Sun Jul  3 06:19:58 on tty1
Linux 5.18.5.
root@slackware15:~# _
```

2. Metoda nowa - streamline_config.pl

Jest to metoda, do której wykorzystuje skrypt *streamline_config.pl* znajdujący się w katalogu linux-5-18-5/scripts/kconfig.

Zaczynam, więc od stworzenia pliku konfiguracyjnego:

```
root@slackware15:~# zcat /proc/config.gz > .config
```

Następnie używam wyżej wymienionego skryptu:

```
root@slackware15:/usr/src/linux-5.18.5# ./scripts/kconfig/streamline_config.pl > config_strip
root@slackware15:/usr/src/linux-5.18.5# ./scripts/kconfig/streamline_config.pl > config_strip
using config: '.config'
root@slackware15:/usr/src/linux-5.18.5# _
```

Teraz korzystam z komendy *make oldconfig*

```
root@slackware15:/usr/src/linux-5.18.5# make oldconfig
```

I zatwierdzam cała konfigurację enterem

```
Test bitmap_*() family of functions at runtime (TEST_BITMAP) [N/m/y/?] n
Test functions located in the uuid module at runtime (TEST_UUID) [N/m/y/?] n
Test the XArray code at runtime (TEST_XARRAY) [N/m/y/?] n
Perform selftest on resizable hash table (TEST_RHASHTABLE) [N/m/y/?] n
Perform selftest on siphash functions (TEST_SIPHASH) [N/m/y/?] (NEW)
Perform selftest on IDA functions (TEST_IDA) [N/m/y/?] n
Perform selftest on priority array manager (TEST_PARMAN) [N/m/?] n
Test module loading with 'hello world' module (TEST_LKM) [M/n/?] m
Test module for compilation of bitops operations (TEST_BITOPS) [N/m/?] n
Test module for stress/performance analysis of vmalloc allocator (TEST_VMALLOC) [N/m/?] n
Test user/kernel boundary protections (TEST_USER_COPY) [M/n/?] m
Test BPF filter functionality (TEST_BPF) [M/n/?] m
Test blackhole netdev functionality (TEST_BLACKHOLE_DEV) [N/m/?] n
Test find_bit functions (FIND_BIT_BENCHMARK) [N/m/y/?] n
Test firmware loading via userspace interface (TEST_FIRMWARE) [M/n/y/?] m
sysctl test driver (TEST_SYSCTL) [N/m/y/?] n
udelay test driver (TEST_UDELAY) [M/n/y/?] m
Test static keys (TEST_STATIC_KEYS) [N/m/?] n
kmod stress tester (TEST_KMOD) [N/m/?] n
Test memcat_p() helper function (TEST_MEMCAT_P) [N/m/y/?] n
Perform selftest on object aggreration manager (TEST_OBJAGG) [N/m/?] n
Test heap/page initialization (TEST_MEMINIT) [N/m/y/?] n
Test HMM (Heterogeneous Memory Management) (TEST_HMM) [N/m/y/?] n
Test freeing pages (TEST_FREE_PAGES) [N/m/y/?] n
Test floating point operations in kernel space (TEST_FPU) [N/m/y/?] n
Test clocksource watchdog in kernel space (TEST_CLOCKSOURCE_WATCHDOG) [N/m/y/?] n

configuration written to .config
```

Przechodzę teraz do kompilacji jądra make -j4 bzImage

```
  LD      .tmp_vmlinux.kallsyms2
  KSYMS   .tmp_vmlinux.kallsyms2.S
  AS      .tmp_vmlinux.kallsyms2.S
  LD      vmlinux
  SYSMAP  System.map
  SORTTAB vmlinux
  CC      arch/x86/boot/edd.o
  CC      arch/x86/boot/main.o
  CC      arch/x86/boot/version.o
  AS      arch/x86/boot/compressed/head_64.o
  VOFFSET arch/x86/boot/compressed/../voffset.h
  CC      arch/x86/boot/compressed/cmdline.o
  CC      arch/x86/boot/compressed/error.o
  OBJCOPY arch/x86/boot/compressed/vmlinux.bin
  RELOCS  arch/x86/boot/compressed/vmlinux.relocs
  CC      arch/x86/boot/compressed/early_serial_console.o
  CC      arch/x86/boot/compressed/kaslr.o
  CC      arch/x86/boot/compressed/ident_map_64.o
  CC      arch/x86/boot/compressed/idt_64.o
  AS      arch/x86/boot/compressed/idt_handlers_64.o
  AS      arch/x86/boot/compressed/mem_encrypt.o
  CC      arch/x86/boot/compressed/pgtable_64.o
  CC      arch/x86/boot/compressed/sev.o
  CC      arch/x86/boot/compressed/acpi.o
  CC      arch/x86/boot/compressed/misc.o
  LZMA    arch/x86/boot/compressed/vmlinux.bin.lzma
  MKPIGGY arch/x86/boot/compressed/piggy.S
  AS      arch/x86/boot/compressed/piggy.o
  LD      arch/x86/boot/compressed/vmlinux
  ZOFFSET arch/x86/boot/zoffset.h
  OBJCOPY arch/x86/boot/vmlinux.bin
  AS      arch/x86/boot/header.o
  LD      arch/x86/boot/setup.elf
  OBJCOPY arch/x86/boot/setup.bin
  BUILD   arch/x86/boot/bzImage
Kernel: arch/x86/boot/bzImage is ready  (#2)
root@slackware15:/usr/src/linux-5.18.5#
```

Tym razem kompilacja jądra zajęła znacznie dłużej(ok. pół godziny, a poprzednio około 15 minut). Tak jak poprzednio teraz trzeba skompilować moduły komendą *make -j4 modules*

```
  LD [M]  sound/soc/snd-soc-core.ko
  LD [M]  sound/soc/sof/intel/snd-sof-acpi-intel-byt.ko
  LD [M]  sound/soc/sof/intel/snd-sof-intel-atom.ko
  LD [M]  sound/soc/sof/intel/snd-sof-intel-hda-common.ko
  LD [M]  sound/soc/sof/intel/snd-sof-acpi-intel-bdw.ko
  LD [M]  sound/soc/sof/intel/snd-sof-intel-hda.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-apl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-cnl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-icl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-tgl.ko
  LD [M]  sound/soc/sof/intel/snd-sof-pci-intel-tng.ko
  LD [M]  sound/soc/sof/snd-sof-acpi.ko
  LD [M]  sound/soc/sof/snd-sof-pci.ko
  LD [M]  sound/soc/sof/snd-sof-probes.ko
  LD [M]  sound/soc/sof/snd-sof-utils.ko
  LD [M]  sound/soc/sof/snd-sof.ko
  LD [M]  sound/synth/emux/snd-emux-synth.ko
  LD [M]  sound/soc/sof/xtensa/snd-sof-xtensa-dsp.ko
  LD [M]  sound/usb/6fire/snd-usb-6fire.ko
  LD [M]  sound/synth/snd-util-mem.ko
  LD [M]  sound/usb/bcd2000/snd-bcd2000.ko
  LD [M]  sound/usb/caiaq/snd-usb-caiaq.ko
  LD [M]  sound/usb/hiface/snd-usb-hiface.ko
  LD [M]  sound/usb/line6/snd-usb-line6.ko
  LD [M]  sound/usb/line6/snd-usb-pod.ko
  LD [M]  sound/usb/line6/snd-usb-podhd.ko
  LD [M]  sound/usb/misc/snd-ua101.ko
  LD [M]  sound/usb/line6/snd-usb-toneport.ko
  LD [M]  sound/usb/line6/snd-usb-variax.ko
  LD [M]  sound/usb/snd-usb-audio.ko
  LD [M]  sound/usb/snd-usbmidi-lib.ko
  LD [M]  sound/usb/usx2y/snd-usb-us122l.ko
  LD [M]  sound/usb/usx2y/snd-usb-usx2y.ko
  LD [M]  sound/virtio/virtio_snd.ko
  LD [M]  virt/lib/irqbypass.ko
  LD [M]  sound/x86/snd-hdmi-lpe-audio.ko
root@slackware15:/usr/src/linux-5.18.5#
```

Z jakiegoś powodu trwało to ponad 3 godziny, jednak zakończyło się sukcesem.

Następnie instaluje moduły komendą make *modules_install*

```
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/intel/snd-sof-intel-hda-common.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/intel/snd-sof-intel-hda.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/intel/snd-sof-pci-intel-apl.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/intel/snd-sof-pci-intel-cnl.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/intel/snd-sof-pci-intel-icl.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/intel/snd-sof-pci-intel-tgl.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/intel/snd-sof-pci-intel-tng.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/snd-sof-acpi.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/snd-sof-pci.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/snd-sof-probes.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/snd-sof-utils.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/snd-sof.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soc/sof/xtensa/snd-sof-xtensa-dsp.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/soundcore.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/synth/emux/snd-emux-synth.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/synth/snd-util-mem.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/6fire/snd-usb-6fire.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/bcd2000/snd-bcd2000.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/caiaq/snd-usb-caiaq.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/hiface/snd-usb-hiface.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/line6/snd-usb-line6.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/line6/snd-usb-pod.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/line6/snd-usb-podhd.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/line6/snd-usb-toneport.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/line6/snd-usb-variax.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/misc/snd-ua101.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/snd-usb-audio.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/snd-usbmidi-lib.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/usx2y/snd-usb-us1221.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/usx2y/snd-usb-usx2y.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/virtio/virtio_snd.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/x86/snd-hdmi-lpe-audio.ko
  INSTALL /lib/modules/5.18.5/kernel/virt/lib/irqbypass.ko
  DEPMOD  /lib/modules/5.18.5
root@slackware15:/usr/src/linux-5.18.5#
Display all 120 possibilities? (y or n)
root@slackware15:/usr/src/linux-5.18.5# _
```

Następnie kopiuję pliki potrzebne do uruchomienia

```
root@slackware15:/usr/src/linux-5.18.5#
Display all 120 possibilities? (y or n)
root@slackware15:/usr/src/linux-5.18.5# cp arch/x86/boot/bzImage /boot/vmlinuz-nowaMetoda-5.18.5-smp
root@slackware15:/usr/src/linux-5.18.5# cp System.map /boot/System.map-nowaMetoda-5.18.5-smp
root@slackware15:/usr/src/linux-5.18.5# cp .config /boot/config-nowaMetoda-5.18.3-smp
root@slackware15:/usr/src/linux-5.18.5#
```

Po tym tworzę link symboliczny komendą *ln -s*

```
root@slackware15:/usr/src/linux-5.18.5# rm System.map
root@slackware15:/usr/src/linux-5.18.5# ln -s System.map-nowaMetoda-5.18.5-smp System.map
root@slackware15:/usr/src/linux-5.18.5#
```

Tworzę ramdisk i napotykam na problem:

```
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/line6/snd-usb-variax.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/misc/snd-ua101.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/snd-usb-audio.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/snd-usbmidi-lib.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/usx2y/snd-usb-us1221.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/usb/usx2y/snd-usb-usx2y.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/virtio/virtio_snd.ko
  INSTALL /lib/modules/5.18.5/kernel/sound/x86/snd-hdmi-lpe-audio.ko
  INSTALL /lib/modules/5.18.5/kernel/virt/lib/irqbypass.ko
  DEPMOD  /lib/modules/5.18.5
root@slackware15:/usr/src/linux-5.18.5#
Display all 120 possibilities? (y or n)
root@slackware15:/usr/src/linux-5.18.5# cp arch/x86/boot/bzImage /boot/vmlinuz-nowaMetoda-5.18.5-smp
root@slackware15:/usr/src/linux-5.18.5# cp System.map /boot/System.map-nowaMetoda-5.18.5-smp
root@slackware15:/usr/src/linux-5.18.5# cp .config /boot/config-nowaMetoda-5.18.3-smp
root@slackware15:/usr/src/linux-5.18.5# rm System.map
root@slackware15:/usr/src/linux-5.18.5# ln -s System.map-nowaMetoda-5.18.5-smp System.map
root@slackware15:/usr/src/linux-5.18.5# cd ..
root@slackware15:/usr/src# cd ..
root@slackware15:/usr# cd ..
root@slackware15:/# cd boot
root@slackware15:/boot# /usr/hare/mkinitrd/mkinitrd_command_generator.sh -k 5.18.5-smp
-bash: /usr/hare/mkinitrd/mkinitrd_command_generator.sh: No such file or directory
root@slackware15:/boot# /usr/share/mkinitrd/mkinitrd_command_generator.sh -k 5.18.5-smp
Modules for kernel 5.18.5-smp aren't installed.
root@slackware15:/boot# _
```

Poprawienie literówki z kopiowania nie naprawiło problemu. Po ponownym powtórzeniu wszystkich kroków dwukrotnie(dla jądra 5.18.5 powtórka i 5.18.2) zmieniłem komendę zabierając końcówkę *-smp i zadziałało. Następnie użyłem wygenerowanej komendy.*

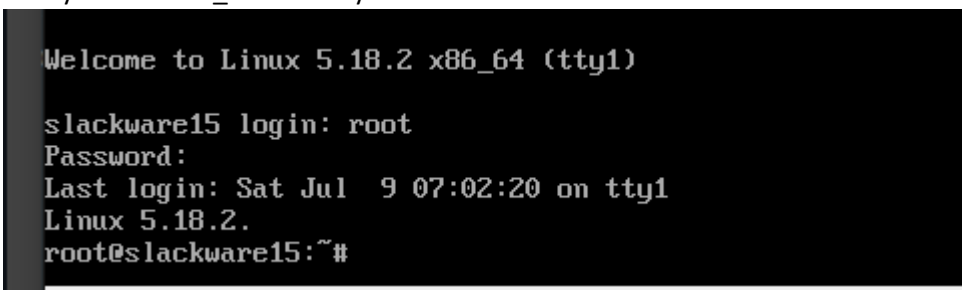
```
root@slackware15:/boot# /usr/share/mkinitrd/mkinitrd_command_generator.sh -k 5.18.2
#
# mkinitrd_command_generator.sh revision 1.45
#
# This script will now make a recommendation about the command to use
# in case you require an initrd image to boot a kernel that does not
# have support for your storage or root filesystem built in
# (such as the Slackware 'generic' kernels').
# A suitable 'mkinitrd' command will be:

mkinitrd -c -k 5.18.2 -f ext4 -r /dev/sda2 -m ext4 -u -o /boot/initrd.gz
root@slackware15:/boot# mkinitrd -c -k 5.18.2 -f ext4 -r /dev/sda2 -m ext4 -u -o /boot/initrd.gz

49981 blocks
/boot/initrd.gz created.
Be sure to run lilo again if you use it.
```

Dopisałem plik do lilo

```
image = /boot/vmlinuz-new-5.18.2-smp
  root = /dev/sda1
  initrd = /boot/initrd.gz
  label = "New kernel"
# Linux bootable partition config ends
```

Jeszcze użycie komendy lilo i restart.

Po wybraniu New_kernel wszystko działa



Podsumowanie

Podczas kompilacji jądra dwiema metodami nie miałem, żadnych problemów. Problemy zaczęły się podczas generowania kodu i bardzo możliwe, że to przez moją literówkę. Różnica polegała na czasie kompilacji jądra, modułów i ich instalacji. W przypadku nowej metody zajmowało to znacznie dłużej, bo gdy w przypadku starej metody czas był około 15 minut na obydwie te czynności, to sama instalacja modułów w nowej metodzie zajmowała ponad 2 godziny dla jądra 5.18.2, a dla 5.18.5 ponad 3. Co do samej trudności uważam, że nowa metoda jest łatwiejsza ponieważ w skrypcie jest opis jak wszystko zrobić. Druga sprawa to to, iż zauważyłem, że system odpala się znacząco szybciej po nowej metodzie. Reasumując, jeśli ktoś chce szybko zainstalować to metoda stara, jak ktoś chce wydajniejszy system to metoda nowa.