# NixOS na telefonu

## Workshop: úvod do Mobile NixOS

Petr Hodina

OpenAlt 2026

# O mně



## Petr Hodina

Kontraktor

Embedded firmware

Přispěvatel do Mobile NixOS

# Pro koho je workshop

**Znáte NixOS z počítače**

- Moduly, `nix-build`, generace jsou stejné
- Nové: jádro, firmware a bootloader podle zařízení, **flashování** místo instalátoru, stage-1

**Zkusili jste Mobian a podobné**

- Znáte Linux na telefonu: obraz, instalace, apt
- Nové: systém sestavíte ze své konfigurace, aktualizace je nová generace

Telefony na workshopu: OnePlus 6 (`enchilada`) a Pixel 3 (`blueline`), oba SDM845

# Agenda

1. Co je Mobile NixOS
2. Jak to funguje
3. Repozitář a zařízení
4. Začínáme
5. Workshop
6. Buildy v praxi
7. Portování
8. Závěr

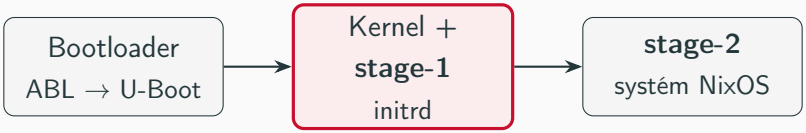
# Mobile NixOS

- Nadstavba nad **NixOS**, **Nixpkgs** a **Nixem**
- Cíl: sjednotit rozdíly mezi mobilními zařízeními
- Čtyřmi slovy: „NixOS na vašem telefonu“
- Stavební kameny, ne hotový produkt: uživatel si systém nastaví, jak chce
- Na Androidu s Qualcommem se bootuje přes **chainloaded U-Boot**: bootloader výrobce spustí U-Boot jako „jádro“ boot obrazu
- Stage-1 (initramfs) je napsaný v **Rustu**, projekt **daybreak**
- Funguje jen s větví **nixos-unstable**

# Proč Nix na telefonu

- **Deklarativní konfigurace**: celý systém je jeden zápis, žádné ruční klikání
- **Reprodukovatelnost**: totéž nastavení, tentýž výsledek, na více zařízeních
- **Generace**: každý systém je samostatná generace (viz boot proces)
- Všechny volby NixOS fungují i tady
- Podpora mnoha zařízení v jednom repozitáři: víc zařízení zvyšuje šanci, že se projekt uchytí

# Od zapnutí k systému




- **stage-1** (Rust initramfs *daybreak*) je první část po startu jádra; připojí kořenový systém a najde generaci
- **stage-2** je vlastní systém NixOS s vaší konfigurací
- Stage-2 je vhodné stavět nativně (aarch64 na aarch64), nástroje to zkusí i cross-kompilací

# DryDock: bootloader a nástroje

**DryDockFirmware**

- Opinionated **U-Boot** pro mobilní zařízení, s grafickým menu
- Vyrobí `boot.img`, který načte jádro a umožní první flash přes USB (fastboot, mass storage, DFU)
- Z RAM lze i jen vyzkoušet, bez opotřebení úložiště
- Některé Qualcomm cíle vystavují i **EUD**
- Nápad vychází z Tow-Boot

**DryDock (nástroj)**

- Instalace mainline Linuxu i návrat k Androidu
- Jeden nástroj místo fastboot, heimdall, adb
- Detekce zařízení, zálohy oddílů, varování před destruktivními kroky
- Popis zařízení v **YAML** (`drydockdevices`), viz další snímek

Rozdělení rolí: DryDock vlastní `boot.img` a U-Boot, Mobile NixOS dodává jádro jako FIT a rootfs.

# DryDock devices: popis zařízení v YAML

```
# qcom/sm8150/nabu.yaml
format: 2
devices:
  xiaomi/nabu:
    arch: arm64
    soc: sm8150
    description: Xiaomi Pad 5
    unlockable: true
    slot_suffix: true
    dynamic_partitions: true
    key_combos:
      fastboot:
        human: Power off, then hold
               Volume Up + Power
    storage:
      type: UFS
      variants_gb: [128]
```

- Jeden soubor na zařízení: `<vendor>/<soc>/<codename>.yaml`
- **Identita**: SoC, architektura, maintainer
- **Odemčení a flash**: A/B sloty, dynamické oddíly, klávesové zkratky pro fastboot a recovery
- **Nízká úroveň**: UART, U-Boot, EDL, geometrie `boot.img`
- **Úložiště a oddíly**, chráněné oddíly
- **ROM**: kroky instalace (`flash`, `erase`, `reboot`, ...)
- Licence CC-BY-SA-4.0

# EFI vs FIT boot

**EFI**

- Standardní cesta: U-Boot spustí `bootefi bootmgr`
- Systém a bootloader jsou na **ESP**
- Stejný princip jako na počítači (typ systému `uefi`)

**FIT**

- Jeden obraz `/mobile-nixos/boot.fit`: **jádro + initrd + DTB**
- Jádro je uložené jednou, konfigurací může být víc (dvojice initrd × DTB)
- Volitelně podepsaný: U-Boot ověří podpis a výsledek nahlásí do `/chosen`

**Pořadí v U-Bootu pro Qualcomm:** EFI (`bootmgr`) → FIT (`boot.fit`) → `extlinux.conf` → menu telefonu

# Boot menu

**Menu v U-Bootu (*phonemenu*)**

- Grafické menu na displeji telefonu
- Spustí se, když selžou předchozí způsoby bootu
- Položka **Previous generations** nabídne generace z `extlinux.conf`
- Nastavení (např. téma) se ukládá do `/uboot.env` na ESP
- Z menu jde i bootnout po síti: **TFTP** nebo **PXE**

**Starší recovery menu ve stage-1**

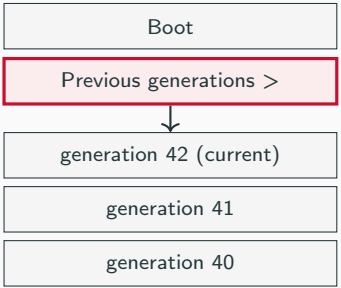
- Stage-1 se zastaví před spuštěním výchozí generace
- Vyvolá se podržením: Volume up / down, Shift, Ctrl, Esc
- U Androidu také boot do recovery nebo samostatný recovery obraz
- Na SDM845 ho nahrazuje phonemenu v U-Bootu a **DayBreak** (odemčení, připojení, `switch_root`)

# Generace

- **Generace** je jedna sestavená podoba systému: vlastní jádro, stage-1 a systém
- Každá změna konfigurace a nový build vytvoří novou generaci, předchozí zůstávají
- Když nová generace nenabootuje, **vyberete starší**: to je pojistka při experimentování
- Mobile NixOS je podle dokumentace „na půl cesty“: výběr generace jádra a stage-1 zatím nejde, nabootovat konkrétní generaci systému ano
- V U-Bootu: generace z `extlinux.conf` v menu *Previous generations*; FIT může nést víc konfigurací najednou

# Generace v menu U-Bootu

1. U-Boot se nejdřív pokusí o `bootmgr`, `boot.fit` a `extlinux.conf`; menu se zobrazí, až když nic nenabootuje, nebo při držení **Volume Down**
2. Hlavní menu: **Boot**, hned za ním **Previous generations >**
3. Položky jsou labely z `/extlinux/extlinux.conf` na ESP, výchozí je označená *(current)*
4. Výběr nastaví `pxe_label_override` a spustí `sysboot`; když boot selže, menu se vrátí




Bez `extlinux.conf` na ESP (nebo bez nalezeného ESP) zůstane menu beze změny. Pozor: `extlinux.conf` běžný `nixos-rebuild` nespravuje; generace je potřeba do souboru dostat jinak.

# Když boot selže: „smutný telefon“

Při fatální chybě stage-1 zobrazí barvu pozadí:

| Barva | Význam |
|---|---|
| Žlutá `0xFFFF00` | nelze připojit kořenový systém |
| Fuchsiová `0xFF00FF` | připojeno, ale žádná použitelná generace |
| Černá `0x000000` | boot zablokovala závislost |
| Červená `0xFF0000` | selhal přechod (`exec`) do init generace |
| Hnědá `0x95681C` | nekontrolovaný pád |

# Struktura repozitáře

| Složka | Obsah |
|---|---|
| `devices/` | konfigurace jednotlivých zařízení (`xiaomi-nabu`, `oneplus-enchilada`, `pine64-pinephone`, ...) |
| `devices/families/` | společné části pro rodiny SoC (`sm8150-mainline`, `sdm845-mainline`, ...) |
| `modules/` | moduly NixOS (boot, hardware, `quirks`, ...) |
| `examples/` | ukázkové konfigurace, např. `phosh` |
| `doc/` | dokumentace, včetně průvodce portováním |
| `bin/` | pomocné nástroje (`menuconfig`, ...) |

Nic dalšího není potřeba: vše je v repozitáři samo obsažené.

# Podporovaná zařízení

**Telefony a tablety**

- Xiaomi Pad 5 (`xiaomi-nabu`), `surya`
- OnePlus 6 (`enchilada`), 6T (`fajita`)
- Pixel 3 (přes `sdm845-generic`)
- Fairphone 5, Motorola `potter`
- PinePhone, PinePhone Pro, PineTab, PineNote

**Rodiny a obecné cíle**

- `sdm845-`, `sm8150-`, `sm7150-mainline`
- `qcs6490-generic`
- Chromebooky (Acer, ASUS, Lenovo)
- `uefi-x86_64`: pro vyzkoušení bez telefonu

Každé zařízení má `mobile.device.supportLevel`, např. `experimental`. Workshop se soustředí na OnePlus 6 a Pixel 3.

# Jak vypadá zařízení: `oneplus-enchilada`

```
{ config, lib, pkgs, ... }:
{
  imports = [ ../families/sdm845-mainline ];

  mobile.device.name = "oneplus-enchilada";
  mobile.device.identity = { name = "OnePlus 6"; manufacturer = "OnePlus"; };
  mobile.device.supportLevel = "supported";

  mobile.hardware = { ram = 1024 * 8; screen = { width = 1080; height = 2280; }; };

  mobile.device.firmware = pkgs.callPackage ./firmware {};
  mobile.system.android.device_name = "OnePlus6";
}
```

- Jádro, FIT, DayBreak a zvuk jsou **společné** v rodině `sdm845-mainline`; zařízení přidává jen identitu a firmware
- **Pixel 3** nemá vlastní složku: používá `sdm845-generic` (firmware přes droid-juicer)

# Zdroje a první build

Předpoklad: zařízení má odemčený bootloader.

```
$ git clone https://github.com/mobile-nixos/mobile-nixos.git
$ cd mobile-nixos
```

- Identifikátory zařízení: `oneplus-enchilada` (OnePlus 6), `sdm845-generic` (Pixel 3); sestavení viz Krok 1
- Postup instalace závisí na zařízení: najděte jeho stránku v dokumentaci

# Vlastní nastavení: `local.nix`

Soubor `local.nix` v kořeni repozitáře, mimo git (*untracked*), se použije při každém `nix-build`:

```
{ lib, ... }:

{
  # vlastni parametr jadra
  boot.kernelParams = [ "quiet" ];
}
```

- Je to konfigurace NixOS: všechny jeho volby, i ty z Mobile NixOS
- Většina voleb NixOS ovlivní jen stage-2 (rootfs, `system.img`)

# Do vlastní konfigurace systému

```
# configuration.nix
{
  imports = [
    (import <mobile-nixos/lib/configuration.nix> { device = "xxx-yyy"; })
    # ...
  ];
}
```

- `xxx-yyy` je identifikátor zařízení, např. `xiaomi-nabu`
- `NIX_PATH` musí obsahovat `mobile-nixos`
- Z `examples/` se nedoporučuje importovat přímo: mění se; zkopírujte si je

# Příklad: Phosh

- `examples/phosh`: hotová konfigurace s grafickým rozhraním **Phosh**
- Mobilní prostředí nad GTK4 a libadwaita
- Obsahuje i patche pro telefony (např. ovládací centrum, `libgtop`, `vte`)
- Dobrý výchozí bod: zkopírujte a upravte

# Plán workshopu

1. **Příprava**: Nix, nástroje `fastboot`, zařízení s odemčeným bootloaderem
2. **Build**: sestavit FIT a systém pro OnePlus 6 nebo Pixel 3
3. **Změna konfigurace**: vlastní `local.nix`, nový build
4. **Flash**: U-Boot přes fastboot, pak FIT a systém na disk telefonu
5. **První boot**: co čekat, recovery menu, když něco selže

# Nix obecně: nativní a cross build

Příklad: GNU `hello` pro aarch64

```
# nativni: aarch64 na aarch64 (nebo binfmt / aarch64 builder)
$ nix build nixpkgs#legacyPackages.aarch64-linux.hello
/nix/store/0xy6jl...-hello-2.12.3

# cross: x86_64 builduje pro aarch64
$ nix build nixpkgs#legacyPackages.x86_64-linux.pkgsCross.aarch64-multiplatform.hello
/nix/store/q597z1...-hello-aarch64-unknown-linux-gnu-2.12.3
```

- **Nativní** build odpovídá `cache.nixos.org`: většina se jen stáhne
- **Cross** build je jiná derivace: jiná store cesta, jiný celý *closure* (i libc, kompilátor)
- Cross-sestavený systém proto zařízení nepoužije: na zařízení se znovu staví
- Proto je lepší stavět nativně (aarch64 stroj, builder nebo binfmt); u Mobile NixOS se stage-2 doporučuje nativně

Hashe se liší podle revize nixpkgs.

# Krok 1: build

```
$ git clone https://github.com/mobile-nixos/mobile-nixos.git && cd mobile-nixos

# systemovy obraz (system.img)
$ nix-build --argstr device oneplus-enchilada -A outputs.default   # OnePlus 6
$ nix-build --argstr device sdm845-generic    -A outputs.default   # Pixel 3

# FIT: jadro + DayBreak initramfs + DTB
$ nix-build --argstr device oneplus-enchilada -A outputs.fit
$ nix-build --argstr device sdm845-generic    -A outputs.fit
```

- `boot.img` s U-Bootem nestavíme: dodá DryDock (další snímky)
- FIT obsahuje DTB všech SDM845 desek, U-Boot vybere ten svůj
- Prázdný, nenastavený obraz může vypadat, že při bootu visí: proto změníme konfiguraci

# Krok 2: změna konfigurace

```
# local.nix
{ lib, pkgs, ... }:
{
  networking.hostName = "workshop-phone";
  environment.systemPackages = [ pkgs.htop pkgs.git ];
  boot.kernelParams = [ "quiet" ];
}
```

- Obyčejná konfigurace NixOS: změníte, znovu spustíte `nix-build`
- Pro grafické rozhraní vyjděte z `examples/phosh`

# Krok 3: flash

1. Záloha dat: flashování smaže obsah zařízení
2. Fastboot: **OnePlus 6** Volume Up + Power, **Pixel 3** Volume Down + Power
3. U-Boot z DryDocku (`blueline` nebo `enchilada`):

```
$ fastboot flash boot_a boot-<kodove-jmeno>.img
$ fastboot erase dtbo_a
$ fastboot flash vbmeta_a vbmeta-<kodove-jmeno>-disabled.img
```

4. **Disk telefonu připojit k PC** (U-Boot ho vystaví jako USB mass storage): FIT na ESP jako `/mobile-nixos/boot.fit`, pak systém
5. Pro novější generaci stačí nakopírovat closure, např. `nix copy -to ssh://mobile@<adresa>`

# Příprava podle zařízení: OnePlus 6 a Pixel 3

| | OnePlus 6 (`enchilada`) | Pixel 3 (`blueline`) |
|---|---|---|
| Fastboot | Volume Up + Power | Volume Down + Power |
| Firmware | aktualizovat na OxygenOS 11, sjednotit firmwarové oddíly (`copy-partitions`) | firmware se získá z telefonu (droid-juicer) |
| Oddíly | `fastboot erase dtbo_a` a `dtbo_b` | `fastboot erase dtbo_a` |
| `vbmeta` | s vypnutým ověřením | s vypnutým ověřením |
| UART | testpointy | přes USB-C (SBU), 115 200 baud |

Duální boot s Androidem se u OnePlus 6 nepodporuje.

# droid-juicer: firmware z vašeho telefonu

- Firmware (modem, Wi-Fi, Bluetooth, GPU, DSP) je proprietární a nelze ho distribuovat, v obrazu proto není
- **droid-juicer** (Rust) ho **při prvním startu vytáhne** z původních oddílů Androidu a poté se zařízení restartuje
- Cíl: `/var/lib/droid-juicer/firmware`; běží jen jednou (`status.json`)
- **Pixel 3** (`sdm845-generic`): firmware z telefonu
- **OnePlus 6**: firmware z balíčku zařízení (`firmware-oneplus-sdm845`)

```
# google,blueline (Pixel 3)
dynpart = "system"
firmware = [
 { partition = "vendor",
   origin = "firmware",
   destination =
     "qcom/sdm845/Google/blueline",
   files = [ { name = "a630_zap.mdt" },
             { name = "adsp.mdt" },
             { name = "wlanmdsp.mbn" },
             ... ] },
]

mobile.device.droidJuicer.enable = true;
```

Pixel 3: `vendor` musí zůstat (metadata jsou na `system_a`); repartition z DryDock maže `system` a `vendor`, extrakce musí proběhnout dřív.

# První boot a co dělat, když něco selže

- První start trvá déle: **kořenový systém se roztahuje** na dostupné místo
- Displej může na chvíli zčernat nebo ukazovat artefakty
- Selhání ukáže „smutný telefon“ barvou (viz tabulka dříve)
- **Recovery menu**: držet Volume nebo Shift, Ctrl, Esc při bootu, vybrat generaci
- Další změny: znovu sestavit a nahrát. Na zařízení lze systém sestavit znovu přes `nix-build` (modul `etc-nixos`), ale aktivace nového buildu na něm zatím není vyzkoušená

# Build U-Bootu: OnePlus 6 a Pixel 3

Zdroje U-Bootu (větev `drydock-2027.01`) a `DryDockFirmware`, nástroje v jeho dev shellu:

```
$ cd DryDockFirmware && nix develop
$ UBOOT_SRC=~/u-boot ./build.sh enchilada   # OnePlus 6 -> out/enchilada/boot-enchilada.img
$ UBOOT_SRC=~/u-boot ./build.sh blueline    # Pixel 3   -> out/blueline/boot-blueline.img

# univerzalni obraz s vice DTB (vyber za behu), i cisty Nix
$ UBOOT_SRC=~/u-boot ./build.sh sdm845-universal
$ nix build .#bootimg-sdm845-universal
```

- Pro oba telefony: `make qcom_defconfig google-pixel.config qcom-phone.config` a další fragmenty
- `google-pixel.config` nastavuje `TEXT_BASE=0x80080000`, jaké potřebuje ABL Pixelu 3
- Vedle `boot-*.img` vznikne i `vbmeta-*-disabled.img`

# U-Boot a FIT v Mobile NixOS (SDM845)

```
# devices/families/sdm845-mainline/default.nix
mobile.system.android.u-boot.fitDtbs = map (n: "dtbs/qcom/sdm845-${n}.dtb") [
  "google-blueline" "oneplus-enchilada" "oneplus-fajita" "xiaomi-beryllium-tianma" # ...
];

# devices/sdm845-generic/default.nix
mobile.system.android.u-boot = {
  enable = false;     # boot.img (U-Boot) dodava DryDock
  fitOnEsp = true;    # FIT zvlast: mobile.outputs.android.fit
};
```

- Do FIT jdou DTB **všech** SDM845 desek, po jedné konfiguraci; U-Boot (`CONFIG_FIT_BEST_MATCH`) vybere tu, která sedí na telefon
- Díky tomu obsluhuje jeden `boot.fit` OnePlus 6 i Pixel 3
- Vlastní U-Boot lze stavět i přes `pkgs.u-boot-qcom.override`

# Jak upravit kernel

Jádro SDM845 (`devices/families/sdm845-mainline/kernel/`) se staví z **zdrojů, patchů a** `defconfig`:

```
# 1) konfigurace: pridat radky do defconfig (nad ‘make defconfig‘ + sdm845.config)
$ echo ’CONFIG_WIREGUARD=m’ >> devices/families/sdm845-mainline/kernel/defconfig

# 2) patch: soubor do kernel/patches/ a radek do seznamu patchFiles v default.nix
$ cp muj.patch devices/families/sdm845-mainline/kernel/patches/0099-muj.patch

# 3) build jadra, pak cely obraz
$ nix-build --argstr device sdm845-generic -A config.mobile.boot.stage-1.kernel.package
$ nix-build --argstr device sdm845-generic -A outputs.default
```

- `olddefconfig` dořeší závislosti; patchované Kconfig se použije, takže jde zapnout i volby z vlastních patchů
- `bin/menuconfig <zařízení>` funguje jen u zařízení s `kernel/config.*`; pro SDM845 upravujte `defconfig`
- QCS6490 má vlastní jádro: `devices/qcs6490-generic/kernel/`

# Konfigurace Daybreaku (Rust initramfs)

```
# local.nix
{ ... }: {
  mobile.boot.stage-1 = {
    compression = "zstd";                     # kernel musi mit CONFIG_RD_ZSTD
    gui.enable = true;                        # splash na displeji
    modules = [ "gpi" "algif_skcipher" ];     # moduly nacitane v initramfs
    luks = { enable = true; device = "partlabel=userdata"; };   # heslo se pta pri bootu
    disks = [
      { device = "label=NIXOS_SYSTEM"; fstype = "ext4"; mountpoint = "/"; mode = "rw"; }
    ];
    swissknife = {                            # vyvojove nastroje
      enable = true;                          # USB ethernet + dropbear (key-only ssh)
      authorizedKeys = [ "ssh-ed25519 AAAA... uzivatel@host" ];
    };
  };
}
```

- `daygen initramfs` vygeneruje a zvaliduje seznam modulů proti konfiguraci jádra
- Swissknife: gadget `usb0` s adresou `169.254.1.1`, ssh jako root po klíči

# Build FIT a nasazení

```
$ nix-build --argstr device oneplus-enchilada -A outputs.fit   # nebo sdm845-generic (Pixel 3)
# vysledek je soubor boot.fit = jadro + Daybreak initramfs + DTB

# na ESP telefonu (cesta, kterou U-Boot hleda):
/mobile-nixos/boot.fit
```

- Obsah FIT: `fitDtbs` (po jedné konfiguraci na DTB), jádro komprimované `kernelCompression` (výchozí `zstd`)
- `boot.img` s U-Bootem je zvlášť (DryDock): `fitOnEsp = true`
- **Podpis**: `fitKeyName` přidá uzly pro podpis, hodnoty se doplní mimo Nix:

```
$ mkimage -F -k <adresar-klicu> -r boot.fit      # nebo DryDockFirmware: scripts/fit-sign.sh
```

U-Boot s veřejným klíčem pak odmítne nepodepsaný nebo změněný FIT.

# Jak přidat nové zařízení

1. Připravit zařízení pro vývoj (**odemknout bootloader**)
2. Najít zdroje, hlavně **jádro**
3. Napsat výrazy, které je sestaví

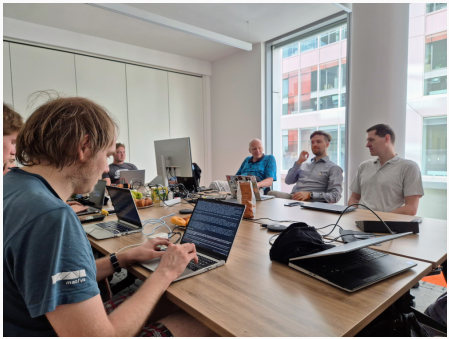
```
$ bin/menuconfig $device_name
$ bin/kernel-normalize-config $device_name
```

Po úpravě konfigurace nebo verze jádra vždy spustit normalizaci.

# Kam dál

- Web a dokumentace: `mobile-nixos.github.io`
- Zdrojový kód: `github.com/mobile-nixos/mobile-nixos`
- Průvodce portováním: `doc/porting-guide.adoc`
- Přispívání: `CONTRIBUTING.adoc`, pull requesty na GitHubu
- Podpora NixOS: Matrix `#nix:nixos.org`, Discourse

# Děkuji!





Připojte se na Matrixu `#sdm845:ixit.cz`