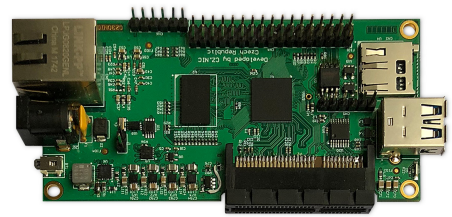
# Automatické aktualizace
# A proč Turris Updater není APT

Karel Kočí
13.4.2019

# Turris router ve zkratce



- Již tři routery (1.x, Omnia, Mox)
- Výkonný hardware
- Založené na OpenWRT
- Sběr dat, dynamický firewall, rozhraní Foris
- **Automatické aktualizace**

# Proč neaktualizovat?

Protože se to rozbije!

# Proč aktualizovat? A proč automaticky?

- Záplaty a bezpečnostní opravy
- Oprava chyb
- Nové funkce

- Včasná aplikace bezpečnostních oprav
- Nižší nároky na administraci

# Různé implementace aktualizací

- System reflash (OpenWRT, většina embedded světa)
- Offline update (MS Windows)
- Runtime update (Většina Linux distribucí)
- Automatic runtime update

# Výzvy automatických aktualizací

- Není admin kterého by jste oslovili
- Update nesmí neprojít na problému který nezanesl uživatel
- Update nesmí svým konáním poškodit systém

# Naše řešení

- Používáme OPKG balíčky
- Máme předpis toho co má být v systému na našem serveru
- Uživatel může do předpisu přidávat pravidla
- V průběhu updatu je nutné občas provést reboot
- Update se může provádět na několikrát (update updateru a zbytku systému)
- Je možné označit balíčky které mění ABI a způsobit reinstalaci dalších balíčků
- Nutná podpora obnovy update procesu po rebootu nebo pádu (journal)

# Balíčky a jejich meta informace

- Soubory k nainstalování
- Meta-informace
  - Verze
  - Závislosti (depends, conflicts, provides)
  - {post,pre}-{install,rm} skripty
  - a další..

# OPKG pro automatické aktualizace

- Nekontroluje kolize mezi soubory mezi balíčky
- Má problém s případy jako náhrada složky souborem a obráceně
- Sebevražedný (odstraní si svoji závislost a spadne, není transakční)
- Nechává po sobě systém v nekonzistentním stavu po pádu

Prostě naprosto nevyhovující

# APT pro automatické aktualizace

- Ask first – do later
- Pracuje výhradně lokálně
- Neumí řešit stavy kdy je nutný reboot
- Není jednoduše možné oddělit preference distribuce od preferencí uživatele

Dalo by se, ale možná stejně práce jako napsat vše od znovu

# Introducing: Updater(-ng)

- Lua předpis/konfigurace z více zdrojů
- update ve formě transakce (instalace souborů naráz a pak až spouštění skriptů)
- Konfigurace může určovat další závislosti, reboot, replan
- Schopnost přeinstalovat rekurzivně balíčky
- Dynamická konfigurace rozhodující se podle stavu systému
- Silná kontrola konfliktů
- Až otravná kontrola minimálních závislostí

# Updater-ng konfigurace

```
Install('updater-ng', 'updater-supervisor',
        { critical = true })
Package('updater-ng', { replan = 'finished' })
Package('l10n_supported', { replan = 'finished' })
```

```
if board == "mox" then
        Install("mox-otp")
elseif board == "omnia" then
        Install("rainbow-omnia")
        Install("libatsha204")
elseif board == "turris1x" then
        Install("rainbow")
        Install("libatsha204", "update_mac")
end
```

# Jak updater-ng funguje?

1. Spustí konfigurační skripty (Lua program)
2. Stáhne indexy repositářů

```
Repository("turris-core", "../packages/omnia/core")
```

3. Z požadavků z konfigurace a balíků z repositářů sestaví požadovaný stav systému

```
Install("base-files", "busybox")
Unistall("htop")
```

4. Sestaví seznam změn pro dosažení cílového stavu
5. Stáhne potřebné balíky
6. Provede změny v systému

# Je tedy vše růžové?

Ne, uživatel si vždy najde cestu..

- Uživatelé mají plný přístup
- Ne všichni "konfiguruní"jak se má

# Kam dál?

- Vyšší kontrola uživatele a kam sahá (healthcheck)
- Updater může být dokonalý, ale pokud selže programátor softwaru tak nic nenadělá (testování a testování)
- Plná náhrada OPKG?

CSNOG

CSNOG 2019: 28.–29. května 2019, Brno

# Děkuji za pozornost. Otázky?

`git.cynerd.cz/presentations/about/`