# Begin Gentoo Install Battle with systemd-nspawn and btrfs

Mitsuru Takigahira

プログラミング生放送勉強会 第 51 回 @IIJ

# わたし

| | |
|---|---|
| Name | Mitsuru Takigahira |
| Twitter | @MysticDoll |
| Belongs to | Tokyo University of Science. B4 |
| PGP | 7D69EF8CB2EFAD913F497591F1FB32960137ED43 |

# TL;DR

UEFI+btrfs でインストールされた Linux 上から
systemd-nspawn を使って Gentoo Linux の環境を構築した

# systemd-nspawn

- コンテナ仮想化技術
- Systemd の中で動く chroot の強化版
- Dockerfile の様な構成管理ファイルを持たない
  - 昔は Dockerfile からマシンを作成するサブコマンドがあったが音楽性の違いで爆発四散したらしい
- ゲストの環境は Systemd を含んでいる必要がある

# Gentoo Linux

- Portage[1]でパッケージをインストールするディストリビューション
- ebuild[2]を元に全てのアプリケーションを自分でビルドする
  - distcc などを使えば手元のマシンでビルドしなくても良い
  - gcc とか webkit とかはクソ遅いのでデカいインスタンスで distcc するほうがいいらしい
- 自分で CFLAGS や USE フラグを設定することでコンパイルオプションや使用する/しないパッケージを選択できる

[1]パッケージマネージャ

[2]アプリケーションのビルドスクリプト

# btrfs

- B-Tree FS バター FS などと呼ばれるファイルシステム
- Copy on Write のファイルシステム
- lzo や gzip アルゴリズムでの透過圧縮に対応している
- サブボリュームという機能がある
  - 今回はサブボリュームの上に Gentoo Linux の環境を構築する

# GentooInstallBattle

## ベースシステムの準備

1. gentoo.org の Downloads から自分の環境に対応するアーキテクチャの systemd を含む stage3 archive の url を確認
   - 今回の場合は x86_64 なのでそのようにする
2. btrfs のサブボリュームの作成
   - # btrfs subvolume create /path/to/mountpoint
   - 私の場合は/gentoo にサブボリュームを作成した
3. ここに stage3 archive を展開
   - # cd /gentoo && wget -O - $URL | tar jxvf -
4. # mount /dev/sdaX /gentoo/boot を実行し /boot をマウント
5. # systemd-nspawn -bD /gentoo を実行し
   Gentoo コンテナを起動する

# GentooInstallBattle

## ベースシステムの構築

1. eselect コマンドで systemd を含むプロファイルを使用する設定をする
   - # eselect profile set default/linux/amd64/17.0/desktop/gnome/systemd
2. /etc/portage/make.conf を編集し、使ってる CPU 向けに CFLAGS を変更
   - CFLAGS="-mtune=native -pipe -O2" などを設定
   - distcc を使って外部でビルドする場合は native ではなく CPU のアーキテクチャを書く (haswell とか)
3. /etc/portage/make.conf を編集し、適宜 USE をフラグを設定
   - qt4 いらねえとかなら USE 変数の中に -qt4 とかいれる
   - いるばあいは qt4 とか入れておく
4. world の更新
   - # emerge -auDN @world

# GentooInstallBattle

## ベースシステムの構築

locale とかタイムゾーンとか設定する。
Arch とかと変わらないのでいろいろ見てください
いい感じに出来たらカーネルをやっていきます

# GentooInstallBattle

## カーネルのビルド

1. # emerge -a sys-kernel/gentoo-sources
2. # cd /usr/src/linux && make defconfig && make menuconfig
   - UEFI 関連のコンフィグと btrfs のサポート設定が必須
     見つけ次第 * マークをつけていく[3][4]
   - UEFI 関連は Processor Type and Features とか Firmware Drivers のあたりにあるはず
   - btrfs は File systems あたり
   - うまく動かないデバイスがあればコンフィグとビルドし直す。根気
3. # make -j4 && make modules_install && make install

[3] 心配なら.config を編集して Arch Linux の Wiki の UEFI のページに書いてあるカーネル設定オプションを全部 y にしておく

[4] M にするとカーネルに組み込まれないので initramfs が必要になり面倒

# GentooInstallBattle

## UEFI のブートエントリを作成

私の場合は systemd-boot を使っているのでこんな感じに /boot/loader/entries/gentoo.conf を書く

```
title      Gentoo Linux
linux      vmlinuz-4.9.76-gentoo-r1
options    root=PARTUUID=<partuuid> rw rootflags=subvol=/gentoo init=/usr/lib/systemd/systemd
```

- 最低限起動に必要なものは全部カーネルの中にいれるため、initramfs とかはいらない
- しかし init には OpenRC が刺さってるのでカーネル起動オプションで systemd のパスを指定してあげる
- rootfs の指定は/dev/sdX とかでもいいが、パーティション UUID を使ったほうが安全
- サブボリュームを rootfs にするのでマウントオプションとして rootflags=subvol=/gentoo を渡してあげる

# まとめ

- GentooInstallBattle はクソ長いので辛い
- webkit や blink は 1,2 時間じゃ終わらないので覚悟が必要
  - 何度も Ctrl+C して暇な時に emerge -r した
- そのうち distcc を使って爆速インスタンスの上からやりたい
- みなさんも GentooInstallBattle しましょう