Bhyve Management Daemon

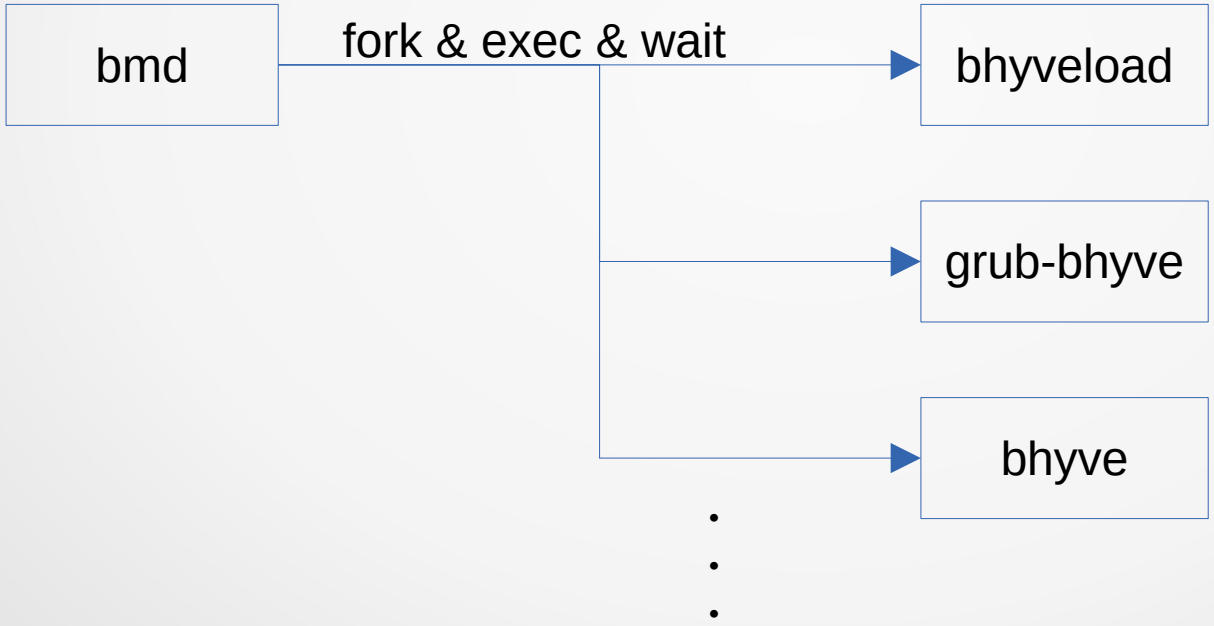
公開版

2023 年 9 月 30 日

（株）創夢　内藤　祐一郎

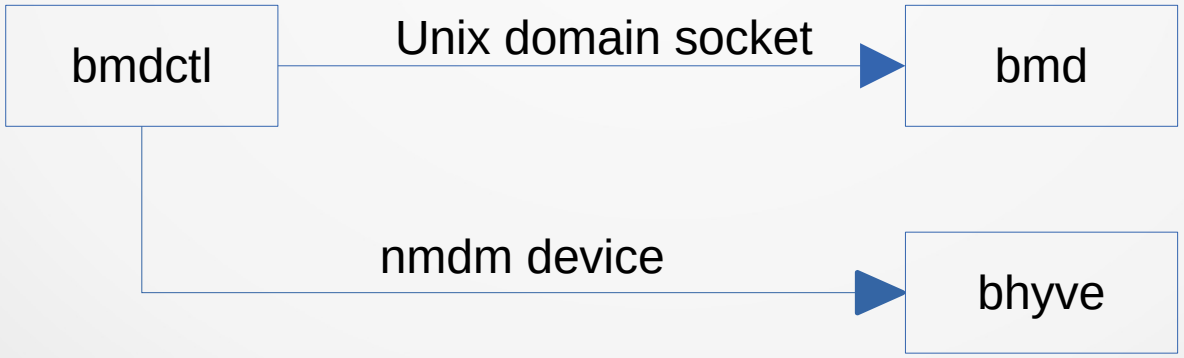
bmd とは

- bhyve やローダのプロセスを管理するデーモン
- 設定ファイルに応じてプロセスを起動します



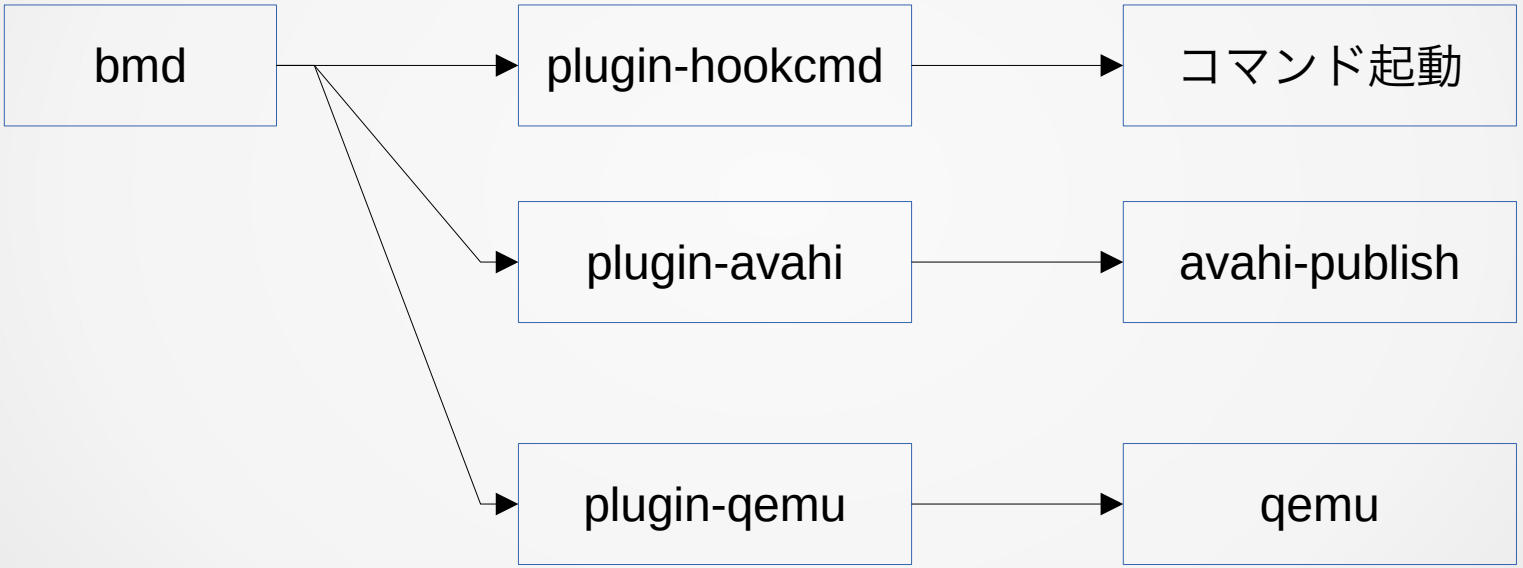
bmd とは

- bmdctl コマンドで仮想マシンを制御します
  - 仮想マシンの起動・停止・状態取得
  - コンソールへの接続



bmd とは

- プラグインにより機能を拡張できます



※plugin-qemu は現状バグっているので未公開

What’s new

- 設定ファイルを一つにまとめました
  - jail.conf(5) と似たようなフォーマットに
- 仮想マシン毎に所有者を設定可能にしました
  - bmd の内部管理です、 bhyve の uid ではありません
- bmdctl console コマンドを作成しました
- Ports に登録してもらいました

設定ファイル

- デフォルトでは /usr/local/etc/bmd.conf を読みます
- 書式は次のように jail.conf(5) に似せています

```
global {
  cmd_socket_mode=0660;
  ...
}

vm freebsd {
   ncpu = 4;
   memory = 4G;
   ...
}

.include “/usr/local/etc/bmd.conf.d/*”;
```

パラメータの書き方

- 基本的には key = value; です
- 複数の値を書くには key1 = value1, value2, value3; とカンマ区切りで書きます（一部パラメータのみ）
- または複数に分けて key1 = value1; key1 += value2; とします
- スペースなどの特殊文字を入れたい場合は
  key = “value sample”; と ’ または “ でくくります

設定ファイル

- 設定ファイルには 3 つのセクションがあります

1. global セクション（ bmd 全体の設定）
2. template セクション（仮想マシン設定の一部）
3. vm セクション（仮想マシン設定）

global セクション

- bmd の全体の設定を行います
- bmdctl コマンドとの通信用 UNIX ドメインソケットを指定できますので、 bmdctl を実行するユーザも同じ設定ファイルを読み込める必要があります
- 通信用 UNIX ドメインソケットのパーミッションも指定できます
- また、グローバル変数の定義を行います

変数

- 変数は $name = value; の形で定義します
- キー名が $ で始まれば、変数定義になります
- 値の部分で ${name} と書くと value に置換されます
- キー名に変数を使用することはできません
- グローバル変数はどこからでも参照できます
- 仮想マシン変数はその仮想マシン設定のみで有効です

定義済み変数

スコープ	変数名	内容
グローバル	LOCALBASE	コンパイル時の LOCALBASE と同じ値 デフォルトは /usr/local
仮想マシン	NAME	仮想マシンの名前
仮想マシン	ID	仮想マシン毎に一意な０から始まる整数値 デーモンが再起動されるまで変わらない

算術演算

- $((式)) で整数の算術演算が行えます
- 整数値は 10 進数、 8 進数（ 0 始まり）、 16 進数（ 0x 始まり) です
- 整数値の取りうる範囲は LONG_MIN ～ LONG_MAX です
- 実行可能な算術演算は + - * / % ( ) です
- 変数は値が整数値に限り使用できます

template セクション

- template は仮想マシン設定の一部を定義します
- 例：

```
template default_disk {
  disk = ${image_path}/${NAME};
}

 template grub_inspect {
   loader = grub;
   loadcmd = auto;
   installcmd = auto;
 }
```

template セクション

- 同じ設定をまとめて書いておくのに使用します
- vm セクションから .apply マクロで呼び出します
- template で定義した変数は仮想マシン変数となり .apply 後に参照可能です

vm セクション

- 仮想マシンの設定を定義します
- 例：

```
vm openbsd {
    .apply default_disk, grub_inspect;
    boot=yes;
    ncpu=4;
    memory=4G;
    comport=auto;
    iso=/zpool/iso/OpenBSD-7.3-amd64.iso;
    network=bridge0;
}
```

vm セクション

- どのようなパラメータが使えるかは bmd.conf(5) を参照してください
- ここで定義した変数は仮想マシン変数になります
- .apply の前に定義した変数は適用した template でも参照できます

.include マクロ

- .include マクロで別ファイルを読み込むことができます
- .include マクロだけは例外でセクションの外側に書きます
- ファイルパスには * と？、 [ ] を用いたパターンマッチングが可能です
- / で始まらないパスは bmd.conf からの相対パスとして扱われます

循環参照違反

- template の中から .apply で他 template を読み込むことができます
- 同一仮想マシン設定内で同じ template を 2 回以上 .apply することはできません
- .include でも同様に同じファイルを 2 回以上読み込むことはできません
- それぞれエラーになります

.include の制限

- .include マクロにより一般ユーザの所有するファイルを読み込むことができます
- 一般ユーザの所有するファイルに記載された .include マクロは実行されません
- 一般ユーザの所有するファイルに記載された global セクションは無視されます

仮想マシンの所有権

- 仮想マシン設定には owner （所有者）が設定されます
- owner の初期値は vm セクションが書かれているファイルの owner です
- bmdctl を実行したユーザと同じ owner の仮想マシンが閲覧・制御可能です

仮想マシンの所有権の変更

- owner の初期値が root に限り所有権を変更できます
- owner パラメータにユーザ名を記載します
  - 例： owner = yuichiro;
- 初期値が一般ユーザの場合は自分自身のみ設定可能です

仮想マシンの所有権の変更

- グループの変更はユーザ名の後に：グループ名を追記します
  - 例： owner = yuichiro:staff;
- グループに属するユーザも該当仮想マシンを閲覧・制御できるようになります

仮想マシンの所有権の変更

- グループが指定されない場合はグループによる所有権のチェックが行われません
- owner の初期値が一般ユーザの場合、自身が所属するグループにのみ変更可能です

設定例

- GitHub にある設定例をみてください
- https://github.com/yuichiro-naito/bmd#example-configurations

hookcmd プラグイン

- 任意のコマンドを仮想マシンのロード・起動・停止時に実行するプラグイン
- 仮想マシン設定に hookcmd パラメータが追加され、その値に起動するコマンドを指定します

hookcmd プラグイン

- 起動されるコマンドの uid は仮想マシンの所有者と同じに変更しました
  - さもないと一般ユーザが root 権限で任意のコマンドを起動できてしまう
- gid はグループ指定があればそのグループに、なければ getpwuid(3) を引いた結果のグループを設定します

err_logfile パラメータ

- 仮想マシンの標準出力と標準エラー出力を指定されたファイルに書き出します
- bhyve のエラーメッセージを確認するために使用します
- 仮想マシンの owner が一般ユーザの場合、そのユーザの権限でファイルを開きます
  - setuid(2) 後に open(2) するプロセスを fork(2) して、ファイル記述子を UNIX ドメインソケット経由で返す

bmdctl console コマンド

- nmdm デバイスにアクセスするには cu を使いますが
- cu -l は /var/spool/lock にロックファイルを生成します
- root ユーザか dialer グループに所属しているユーザでないとロックファイルの生成に失敗します
- 事実上 cu -l は一般ユーザには使えません

bmdctl console コマンド

- そこで bmdctl 内部に cu の内部動作を真似て nmdm デバイスに接続するコードを作成しました
- 8bit 透過環境のみのサポートと割り切り、各文字のビット長チェックを行わないので若干高速です
- bmd が管理、または割り当てた nmdm デバイス名を取得し、そこに接続します

bmdctl console コマンド

- bmd は仮想マシンの所有者に合わせて nmdm デバイスの owner を変更します
- 排他制御は nmdm デバイスを flock(2) します
- これにより一般ユーザでも自身が所有する仮想マシンの console にアクセスできます

まとめ

- 設定ファイルを大きく変更しました
- 一般ユーザが仮想マシンを設定可能になるため、権限周りの考慮が大変でした
- 所有権の概念は権限の問題を解決するのに役立ちました

最後に

- マニュアルを一式作成しました
  - bmd(8), bmdctl(8), bmd.conf(5)
- ports に入りましたので、パッケージも利用可能です