ハイパーバイザの作り方〜ちゃんと理解する仮想化技術〜 第6回 Intel VT-x を用いたハイパーバイザの実装 その1「仮想CPU の実行処理」

はじめに

前回までに、5 回にわたりハードウェアが提供している仮想化支援機能を解説しました。

今回からは、実際にどのようにして仮想化支援機構を使い、ハイパーバイザを実装するのかを解説していきます。解説には、実装の具体例として現在 FreeBSD への実装が進められているハイパーバイザ 「BHyVe (ビーハイブ)」を用います。

BHyVe とは

BHyVe は NetApp により開発された、FreeBSD で 動く新しいハイパーバイザです。2011 年に発表され、実装が進められています。現在、FreeBSD10 の新機能の 1 つとしてリリースすることを目指しています[*1]。設計は Linux KVM に似ており、FreeBSD 版の KVM とも言えるでしょう。ただし、BHyVe は現在開発の初期段階であるため、現在の KVM に比べ非常に機能は限られています。現状、実用に用いるのは難しい開発版の機能となります。しかし、ハイパーバイザの実装方法を学ぶための教材としては、以下の二点が優れています。

1 つは、最低限の機能のみ実装しているため KVM と比べるとコード量が非常にコンパクトであることです。もう 1 つは、デバイスエミュレーションなどを行うユーザランドプログラム (詳細は後述) がシンプルなものであることです。KVM ではユーザランドプログラムに既存のエミュレータである QEMU を流用しており、これにより既存 OS との高い互換性が得られていますがコードの見通しが悪くなっています。そこで、本連載では KVM ではなく BHyVe を用いて解説を行うことにします。

[*1] 2013 年 1 月に FreeBSD-CURRENT の開発ツリーにマージされました。

解説対象のバージョン

BHyVe は現在、リリース版が存在しません。また、開発の初期段階であるため、大きな変更が加えられることも予想されます。

そこで、本連載では現時点での FreeBSD-CURRENT でスナップショットが公開されている最新リビジョンである r245673 を用いて解説を行います。r245673 のインストールディスクは、次のアドレスからダウンロードできます。

ftp://ftp.freebsd.org/pub/FreeBSD/snapshots/amd64/amd64/ISO-IMAGES/10.0/FreeBSD-10.0-CURRENT-amd64-20

r245673 のソースコードは次のコマンドで取得できます。

```
svn co -r245673 svn://svn.freebsd.org/base/head src
```

また、BHyVe 用にコンフィグレーションされたゲストマシンのディスクイメージが配布されており、次のアドレスからダウンロードできます。

http://mirrors.nycbug.org/pub/BHyVe/r244024/diskdev.xz

BHyVe の動作環境

BHyVe を試すには、Intel VT-x と EPT(Extended Page Tables) をサポートした CPU が必要です。

今のところ AMD の CPU には対応していません。どのモデルの CPU が EPT をサポートしているかは ark.intel.com で調べられます。簡単な目安としては、 Core i3, i5, i7 ならばほぼすべての CPU が対応しています。Celeron や Pentium などの廉価版 CPU でも EPT 対応モデルが一部存在しています。実機にインストールするのが最も確実ですが、最近の VMware (VMware Player・VMware Workstation・VMware Fusion) ならば Nested VM[*2]に対応しているため仮想マシン上で試すこともできます。

BHyVe が提供する機能

現状、BHyVe ではゲスト OS として FreeBSD/ amd64 のみをサポートしています。

ハードディスクコントローラとしては、標準的な IDE や AHCI コントローラのエミュレーションはサポートせず、準仮想化ドライバである virtio-blk をサ ポートしています。

NIC コントローラとしても同様に、標準的な Intel e1000 などのデバイスのエミュレーションをサポートせず、準仮想化ドライバである virtio-net をサポートしています。virtio は多くの場合、標準的なデバイ

[*2] ハイパーバイザ上でハイパーバイザを実行可能にする機能。

スのエミュレーションと比較して高い性能が得られますが、ゲスト OS に virtio ドライバをインストールし、/boot/loader.conf の設定を行う必要があります。

システムコンソールとしては、標準的なビデオデバイスをサポートはサポートされず、PCI 接続の 16550 互換シリアルポートのみをサポートしています。PCI 接続のシリアルポートをシステムコンソールとして使うのは非標準的なハードウェア構成であるため、これについても/boot/loader.conf の設定を行う必要があります。また、ビデオデバイスをサポートしないため、X11 を起動して GUI 環境を表示することはできません。

BHyVe がエミュレート可能なデバイスは上述の 3 種類だけですが、Intel VT-d を用いて実機上の PCI・ PCI Express デバイスをゲストマシンへパススルー接続できます。この他、割り込みコントローラのエ ミュレーション (Local APIC、IO-APIC) や、タイマ デバイスのエミュレーション、ハードウェア構成をゲスト OS へ伝えるのに必要な APIC などをサポートしています。また、BIOS や UEFI などのファームウェアをサポートしていないため、ディスクイメージからブートローダをロードしてゲスト OS を起動することができません。このためにハイパーバイザの機能として FreeBSD カーネルをロードしゲストマシンを初期化する OS ローダが実装されています。

BHyVe の構成

BHyVe は、カーネルモジュールとユーザランドプラグラムの 2 つから構成されます。

カーネルモジュール vmm.ko は、CPU に対して VT-x 命令を発効するなど、ハードウェアに近い処理を行います。

ユーザランドプログラム/usr/sbin/bhyve は、ユーザインタフェースを提供し、ハードウェアエミュレーションを行います。

また、BHyVe には/usr/sbin/bhyveload というツールがあります。前章で述べた通り、BHyVe ではディスクイメージ上のブートローダを実行できません。このため、ゲストカーネルをロードして起動可能な状態に初期化するゲスト OS ローダ (/usr/sbin/bhyveload) が付属します。

/usr/sbin/bhyveload は FreeBSD ブートローダを FreeBSD 上で実行可能なプログラムに改変し、ゲストマシンのディスクイメージからカーネルを読み込んでゲストメモリ空間へ展開するようにしたものです。/usr/sbin/bhyveload を実行すると、FreeBSD のブート時に表示されるのと同じメニューが表示されます。このため、一見するとゲストマシンの実行が開始されたように見えます。しかし、これはホスト OS で/usr/sbin/bhyveload が出力している画面で、ゲストマシンの起動は開始されていません。

このほか、ゲストマシンの実行とは直接関係しませんが、VM インスタンスの削除などを行うための VM インスタンス管理ツールとして/usr/sbin/bhyvectl が提供されています。

これらのユーザランドのプログラム群は、VM 管理用のライブラリ (libvmmapi) を通してゲストマシンの初期化や実行をします。libvmmapi は mmap や ioctl を発行し、vmm.ko が提供するデバイスを操作します。

全体図を図 1 に示します。

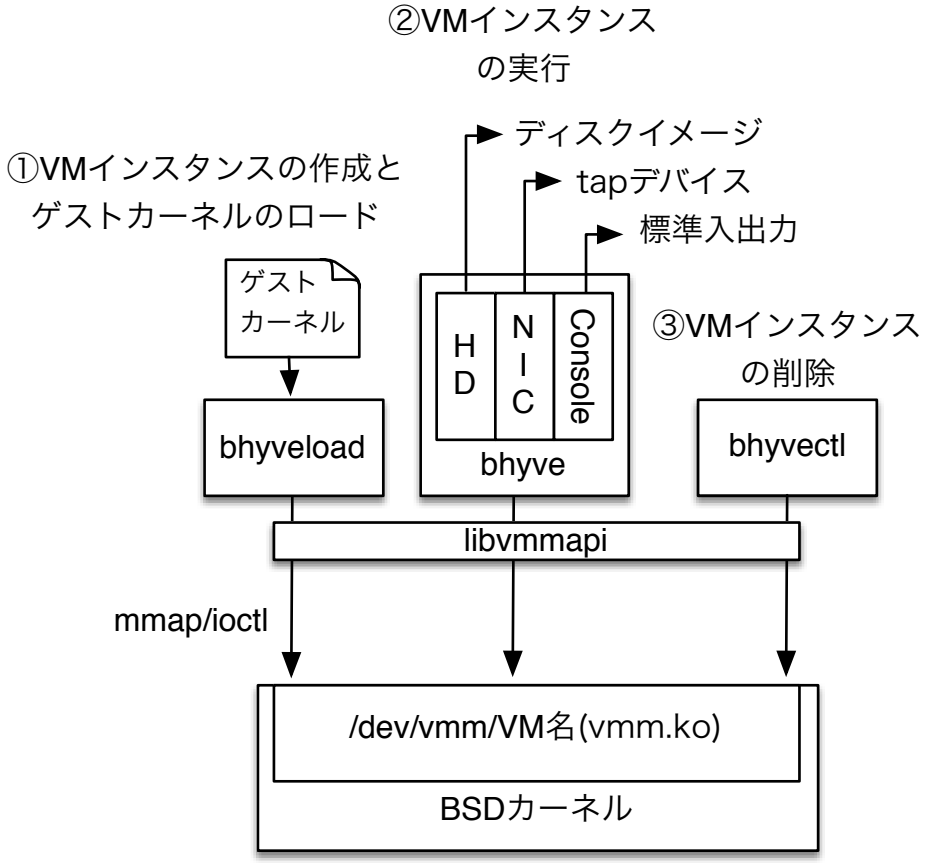

図 1　BHyVe の構成

BHyVe の使い方

BHyVe を試すには、ホストマシンのセットアップとゲストマシンのディスクイメージの準備が必要です。前述のアドレスを参照してください。リスト 1 にシェルから BHyVe を用いてゲストマシンを実行する方法を示します。

リスト 1

```
# kldload vmm.ko                                    (1)
# ls /dev/vmm                                       (2)
# /usr/sbin/bhyveload -d diskdev -m 1024 guest0     (3)

Consoles: userboot
```

```
FreeBSD/amd64 User boot, Revision 1.1
(root@bhyve, Sat Dec  8 17:35:59 PST 2012)
Loading /boot/defaults/loader.conf
/boot/kernel/kernel text=0xd01c58 data=0x15e9a0+0x2bb490 syms=[0x8+0x14bbd8+0x8+0x1a7a89]
/boot/kernel/virtio.ko size 0x5a90 at 0x1810000
/boot/kernel/if_vtnet.ko size 0xd910 at 0x1816000
/boot/kernel/virtio_pci.ko size 0x6cc0 at 0x1824000
/boot/kernel/virtio_blk.ko size 0x6b08 at 0x182b000
|


  ______               ____   _____ _____
 |  ____|             |  _ \ / ____|  __ \
 | |___ _ __ ___  ___ | |_) | (___ | |  | |
 |  ___| '__/ _ \/ _ \|  _ < \___ \| |  | |
 | |   | | |  __/  __/| |_) |____) | |__| |
 | |   | | |    |    ||     |      |      |
 |_|   |_|  \___|\___||____/|_____/|_____/    ```                        `
                                             s` `.....---.......--```   -/
 ?????????????Welcome to FreeBSD???????????? +o   .--`         /y:`     +.
 ?                                          ?  yo`:.            :o     `+-
 ?  1. Boot Multi User [Enter]              ?   y/               -/`  -o/
 ?  2. Boot [S]ingle User                   ?  .-                  ::/sy+:.
 ?  3. [Esc]ape to loader prompt            ?  /                     `--  /
 ?  4. Reboot                               ? `:                          :`
 ?                                          ? `:                          :`
 ?  Options:                                ?  /                          /
 ?  5. Configure Boot [O]ptions...          ?  .-                         -.
 ?                                          ?   --                       -.
 ?                                          ?    `:`                   `:`
 ?                                          ?      .--             `--.
 ?                                          ?         .---.....----.
 ??????????????????????????????????????????


Booting...
# ls /dev/vmm                                       (4)

guest0
# /usr/sbin/bhyve -A -I -m 1024 -s 0:0,virtio-blk,diskdev10 \
-S 31,uart,stdio guest0                             (5)
```

```
GDB: no debug ports present
KDB: debugger backends: ddb
KDB: current backend: ddb
Copyright (c) 1992-2012 The FreeBSD Project.
Copyright (c) 1979, 1980, 1983, 1986, 1988, 1991, 1992, 1993, 1994
  The Regents of the University of California. All rights reserved.
FreeBSD is a registerd trademark of The FreeBSD Foundation.
FreeBSD 10.0-CURRENT #0 r243948: Thu Dec  6 14:03:56 UTC 2012
    root@kaos.glenbarber.us:/usr/obj/usr/src/sys/GENERIC amd64
......(省略)......
# ls /dev/vmm                                          (6)
guest0
# /usr/sbin/bhyvectl --vm=guest0 --destroy             (7)
# ls /dev/vmm                                          (8)
```

1. カーネルモジュールをロードし、BHyVe を使用可能にする
2. vmm.ko がロードされると/dev/vmm ディレクトリが作成される。この状態では VM インスタンスが存在しないので、ディレクトリは空
3. /usr/sbin/bhyveload は VM インスタンスを作成し、BSD カーネルをディスクイメージから読み込みます。そして VM インスタンスのメモリ領域へロードして起動可能な状態を作る。カーネルをロードするプログラムは FreeBSD のブートローダのコードをそのまま使っているため、ブート時に表示されるのと同じメニューが表示される。しかし、これはホスト側で実行されているプログラムでゲストマシンの起動は始まってない。引数-d はディスクイメージ、-m はメモリサイズ、最後の引数は VM 名を指定している
4. /usr/sbin/bhyveload が作成した VM インスタンスは、/dev/vmm 以下にデバイスファイルとして表示される
5. /usr/sbin/bhyve は/usr/sbin/bhyveload が初期化した VM インスタンスを実行する。引数-m はメモリサイズ、-s と-S はゲストマシンの PCI 上に接続される仮想デバイス、最後の引数は VM 名を指定している
6. shutdown を行なって bhyve プロセスを終了した後も、カーネル側に VM インスタンスの状態が残る
7. VM インスタンスを削除しゲストメモリを開放するには、これを削除する必要がある。/usr/sbin/bhyvectl の–destroy オプションを使い、VM インスタンスを削除する
8. VM インスタンスを削除するとデバイスファイルも消去される

使い方は KVM に似ていますが、VM インスタンスの状態が VM 実行プログラムと分離されており、/dev/vmm/VM 名に保持される点が KVM と異なっています。このため、OS ローダが VM 実行プログラムと別プログラムになっていたり、VM 実行プログラムを終了とは別に VM インスタンスの削除コマンドを実行する必要が出てきます。

まとめ

今回は、BHyVe の概要と使い方について説明しました。次回からいよいよソースコードの解説に移りたいと思います。

ライセンス