igel

Technology Consulting Company
Research, Development &
Global Standard

BitVisor 2017年の主な変更点

榮樂 英樹
株式会社イーゲル

2017-12-05 BitVisor Summit 6

BitVisor 2017年の主な変更点

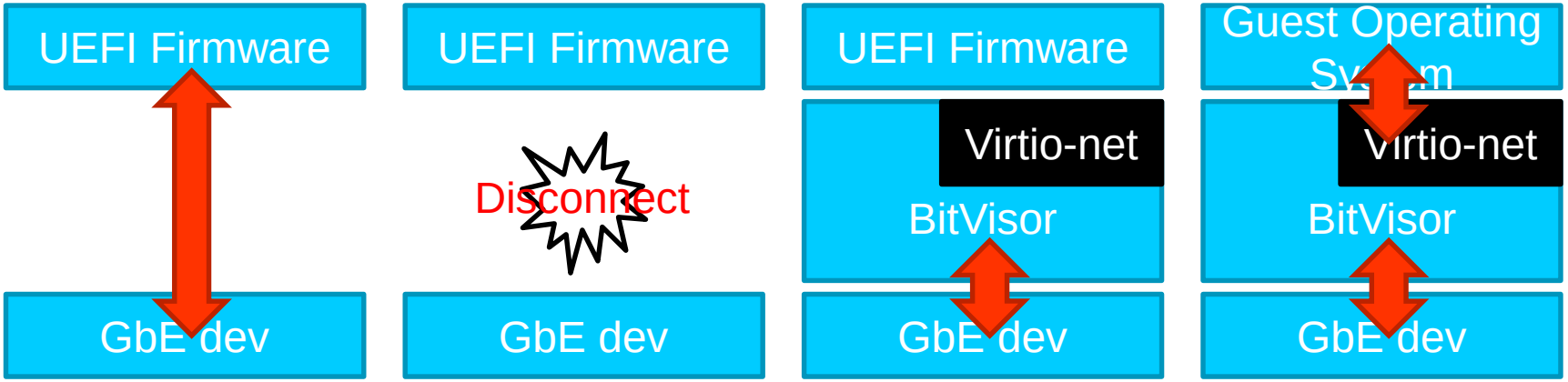
igel

- USB xHCI対応 - 昨年のBitVisor Summit 5で発表された機能の取り込み
- ファームウェアドライバー切断機能
- UEFI用ブートローダー改良 (パスワード認証機能対応)
- PCI configuration spaceアクセス改良
- Broadcom Thunderbolt Ethernet対応
- デバッグ機能改良
- 一部環境での問題に対応
- バグ修正
- コンパイル問題修正

ファームウェアドライバー切断機能

- BitVisorがデバイスにアクセスする前に、UEFIファームウェアのドライバーを切断する機能 (競合を避けるため)
- 下図のネットワークデバイスのVirtio-net化のようにデバイスそのものが変わる場合や、デバイス隠ぺい時に有用
- ストレージ暗号化にも使うべきだが未実装

起動の流れ→



UEFI用ブートローダー改良 (パスワード認証機能対応)

- boot/uefi-loader-login
  - BIOS環境のboot/login-simpleに相当するUEFI用ブートローダー
  - 起動ストレージ暗号化には未対応
    (ファームウェアにファイルシステムの再検出をさせる必要あり)

- UEFIブートの仕様変更
  - ブートローダーから任意のデータをVMMに渡したり、VMMから受け取れるような仕組みを追加
  - データはUUIDで識別されるので独自データの追加も簡単
  - ただし以前のバージョンとの互換性はなし

PCI configuration spaceアクセス改良: これまでの問題点

igel

- I/Oポート0xCF8 (アドレスポート) および0xCFC (データポート) の処理に競合状態がある
- MCFG (Memory mapped configuration) をドライバーから使うのが難しい (APIが全然違う、判定が必要、等)
- pci_handle_default_config_read/write()関数のオフセット指定がMCFGでしか参照されない
- pci_set_bridge_io()関数 (Broadcom NIC対応時に導入された) がMCFG使用不可能の場合panicする
- pci_msi_*()関数が初期化時にしかMCFG使用可能かどうかをチェックしておらず、バス番号変更により使用不可能になることがある

PCI configuration spaceアクセス改良: 問題点の修正方法

igel

- APIの整理
  - MCFG対応かどうかを区別してもしなくても使えて、MCFGでなくても競合の問題が起こらないAPIを導入する
- 排他制御の見直し
  - MCFGでなくても競合の問題が起こらないようにする
  - ゲストオペレーティングシステムからのPCI configuration spaceのアクセスをドライバーが処理中に呼び出されても、デッドロックにならないようにする

PCI configuration spaceアクセス改良: APIの整理

igel

- 以下の関数の削除: pci_config_address_t型ではMCFGの4096バイトアドレス空間に対応できないため
  - pci_{read,write}_config_data_port
  - pci_{read,write}_config_data{8,16,32}
- 以下の関数の追加
  - pci_{readwrite,read,write}_config_pmio : 既存のmmio用関数に対応するもの
  - pci_config_{read,write} : デバイスドライバーからの使用を主に想定し、struct pci_device *型でデバイスを指定し、MCFGかどうかを自動判別してアクセスするもの

PCI configuration spaceアクセス改良: 新旧API比較 (1/3)

igel

- MCFGによるアクセスを行う関数: 変更なし

```
void pci_readwrite_config_mmio (struct pci_config_mmio_data *p, bool wr,
                                uint bus_no, uint device_no, uint func_no,
                                uint offset, uint iosize, void *data);
void pci_read_config_mmio (struct pci_config_mmio_data *p, uint bus_no,
                           uint device_no, uint func_no, uint offset,
                           uint iosize, void *data);
void pci_write_config_mmio (struct pci_config_mmio_data *p, uint bus_no,
                            uint device_no, uint func_no, uint offset,
                            uint iosize, void *data);
```

- MCFGでないアクセスを行う新関数: 追加

```
void pci_readwrite_config_pmio (bool wr, uint bus_no, uint device_no,
                                uint func_no, uint offset, uint iosize,
                                void *data);
void pci_read_config_pmio (uint bus_no, uint device_no, uint func_no,
                           uint offset, uint iosize, void *data);
void pci_write_config_pmio (uint bus_no, uint device_no, uint func_no,
                            uint offset, uint iosize, void *data);
```

PCI configuration spaceアクセス改良: 新旧API比較 (2/3)

- MCFGでないアクセスを行う旧関数: 削除

```
extern u32  pci_read_config_data_port();
extern void pci_write_config_data_port(u32 data);

extern u8 pci_read_config_data8(pci_config_address_t addr, int offset);
extern u16 pci_read_config_data16(pci_config_address_t addr, int offset);
extern u32 pci_read_config_data32(pci_config_address_t addr, int offset);
extern void pci_write_config_data8(pci_config_address_t addr, int offset, u8
data);
extern void pci_write_config_data16(pci_config_address_t addr, int offset, u16
data);
extern void pci_write_config_data32(pci_config_address_t addr, int offset, u32
data);
```

  - pci_write_config_data_port()関数の中身がreadになっていたことが判明😅

PCI configuration spaceアクセス改良: 新旧API比較 (3/3)

igel

- MCFGかどうかを意識せず使える新関数: 追加

```
void pci_config_read (struct pci_device *pci_device, void *data, uint iosize,
                      uint offset);
void pci_config_write (struct pci_device *pci_device, void *data, uint iosize,
                       uint offset);
```

- MCFGかどうかを意識せず使える関数: 引数そのまま
  - 新版ではMCFGでなくてもoffset指定が有効になる

```
extern void pci_handle_default_config_read (struct pci_device *pci_device,
                                            u8 iosize, u16 offset,
                                            union mem *data);
extern void pci_handle_default_config_write (struct pci_device *pci_device,
                                             u8 iosize, u16 offset,
                                             union mem *data);
```

  - 新関数との違い: ゲストオペレーティングシステムによるアクセスに対するデフォルト処理で、以下の処理が追加で実行される
    - fake_command_mask/fixed/virtualの処理 (ブリッジのバスマスター強制有効化等に使用される)
    - 書き込み後のpci_device->config_spaceの更新

PCI configuration spaceアクセス改良: 排他制御の見直し

igel

- 以前のロック
  - config_data_lock: ホットプラグデバイス登録用
  - pci_config_lock: PCI configuration spaceアクセス用
  - ただし、ドライバー検索時には使用されるが仮想マシン内のアクセスには使用されないなど、中途半端
- 新しいロック
  - I/Oロック: 0xCF8・0xCFCポートのアクセス用
  - アドレスロック: 0xCF8ポートに書き込まれた値を保持する変数の排他制御と、VMMと仮想マシンのアクセスの排他制御で、ネストOK、最後のアンロック時に0xCF8ポートの内容を元に戻す(0xCF8ポートの書き換え回数を減らす目的)
- 排他制御のため0xCF8・0xCFCポートのアクセスに対してCORE_IO_RET_DEFAULTは使用しない

Broadcom Thunderbolt Ethernet 対応: 状況ときっかけ

- これまでの状況
  - VMM起動開始直後は通信OK
  - macOS: 通信不可能
  - Windows: virtio-netドライバーを入れるとハングアップorクラッシュ
- 問題発生のきっかけとなるオペレーティングシステムの挙動
  - macOS: PCIeブリッジのバス番号やメモリアドレスの変更
    (内蔵Ethernetではバス番号の変更はなかった)
  - Windows: PCIeブリッジのBus Master EnableおよびMemory Space Enableの一時変更
    (内蔵Ethernetでも同様の事象があり対策済みだった)

Broadcom Thunderbolt Ethernet 対応: 原因と対処

- 原因
  - バス番号やメモリアドレスがブリッジの範囲外の時にアクセスしようとすると通信できなくなる
  - Thunderbolt Ethernetアダプターのデバイス側に見えるPCIeブリッジの、他では見られない変な挙動
    - Memory Space Enableが0の間にバスマスター転送しようとすると、Bus Master Enableが1でもそれ以降通信できなくなる
    - Bus Master Enableを一時0にした場合、PCIe configuration space経由でデバイスのBARを書き直さないとアクセスできない
- 対処
  - BARの処理の修正、ブリッジの範囲等が合わない場合の処理スキップ、Memory Space Enableの強制有効化

デバッグ機能改良

- tools/dbgsh-uefi: UEFI Shellから使えるdbgsh
  - 例外ハンドリングができないため、プロセッサごとに別コマンドを用意
    - dbgsh-a: AMDプロセッサ用
    - dbgsh-i: Intelプロセッサ用
- tools/ieee1394log: リトライ機能 (-rオプションに回数を指定)
  - Resource temporarily unavailableエラーが出て終了してしまう問題の対策

一部環境での問題に対応: ハードウェア・ファームウェア

igel

- ThinkPad L570
  - ACPI spec 4.0a→6.1 (Errata A): AMLの定義に誤りが散見されるが推測して対応
  - ACPI DSDTに含まれる謎のOneOpを無視
- Mac
  - 最新ファームウェアがVT-dを有効にするようになったため無効化機能を追加 (CONFIG_DISABLE_VTD)
- iMac (Retina 5K, 27-inch, Late 2015) および MacBook (Retina, 12-inch, Early 2016)
  - PCIeに見えるID 0xFFFFのデバイスを隠ぺいするとEFI variablesのアクセスでエラーが発生するため隠ぺいしないようにした (これまではMCFGの場合だけ隠ぺいする変な状態だった)

一部環境での問題に対応: ソフトウェア

igel

- BitVisor on Linux KVM
  - SYSCFG MSRが読み込めるにも関わらず書き込めないという特殊な環境でも動作するようにした
- Linux 4.6 on BitVisor
  - INVPCID命令でBitVisorが落ちる: 2016-10にBitVisor開発者メーリングリストにおいて米司伊織氏より報告があった件
  - INVPCID命令を隠ぺいした
- Linux 4.10 on BitVisor
  - LinuxのIntel microcode updateの処理が変更になり、アップデートが適用されないと永遠に繰り返すようになってしまった
  - AMDにあわせIntelでもパススルーとした
  - MSRの単純なパススルーは仕様上不可能 - サイズがわからないので、ページフォールトを利用して必要なページをマップする

バグ修正 (1/2)

igel

- Broadcom NetXtreme GbE
  - 受信せず送信を続けるとバッファーがいっぱいになり送信できなくなる問題を修正した
  - VLANタグ付きパケットの受信に対応した (ハードウェアが分離してしまったものを内部で付け直す)
- Intel GbE
  - virtio=1の時に実デバイスのI/O空間をオフにするようにした (I/Oアドレスが重なってしまいフックが機能しなくなった問題の対応)
  - virtio=1の時に割り込み生成が行われなかった問題を修正した (ケーブルを抜いているとWindowsのシャットダウン時にBSoDになっていた問題の対応)
  - 受信処理でレジスターの値が範囲外でもpanicしないようにした
  - バスマスター/メモリ空間有効化 (FF:FF:FF:FF:FF:FF問題対策)

バグ修正 (2/2)

igel

- USB
  - Endpoint descriptorのNULLポインターチェック漏れ修正
  - usb_mscd: URB_STATUS_ERRORSの対応
- AHCI
  - 初期化前にPxCMDレジスターに書き込まれたときに、Command List Base Addressレジスターを壊してしまい動かなくなる問題の修正
- PCI
  - pci_driver_option_get_int()関数の範囲チェックに誤りがあり正常に動作しなかった問題を修正
    - 原因: int型が32ビットのCでは、-0x80000000はunsignedだが、signedだと思って比較を行っていた

コンパイル問題修正

igel

- tools/common/call_vmm.[ch]
  - glibcバージョン2.20以上に対応した (_BSD_SOURCEがdeprecatedになったことへの対応)
  - Position-independent executable (pie) に対応した
- tools/dbgsh
  - MinGWがなくてもWindows用バイナリ以外はビルド可能にした
- tools/log
  - GNU Make 4.0に対応した
- core/uefi.c
  - 32ビットコンパイル時の警告抑制
- tools/iccard
  - WinMain関数の型の修正

```
# Makefile中の記述
load : # GNU Make 4.0ではエラー
load:  # GNU Make 4.0でもOK
```

その他見つかった問題 (未解決)

igel

- ACPI SSDT解釈中ハングアップする環境がある (ASUSマザーボードのAMDプロセッサ環境で確認)
  - DSDT/SSDT解釈処理で名前の管理をしていないために、メソッドの引数の数 (Nameなら0) を確定できず、以下のような式では候補が発散してしまい、数分程度では絞り込めなくなってしまう

```
        Name (A001, One)
        OperationRegion (A002, SystemMemory, Add (A001,
Add (DerefOf (Index (A001, Add (A001, One
            ))), DerefOf (Index (A001, Add (A001,
0x03))))
            ), 0x1000)
        Field (A002, ByteAcc, NoLock, Preserve)
        {
        }
```

まとめ

igel

BitVisor 2017年の主な変更点

- USB xHCI対応
- ファームウェアドライバー切断機能
- UEFI用ブートローダー改良 (パスワード認証機能対応)
- PCI configuration spaceアクセス改良
- Broadcom Thunderbolt Ethernet対応
- デバッグ機能改良
- 一部環境での問題に対応
- バグ修正
- コンパイル問題修正