

ネットワークを利用した ADvisor機能の拡張

2013年12月6日(金)

電気通信大学大学院

情報理工学研究科

山本 航洋 大山 恵弘

ADvisor

- VMMが画像を表示するシステム
 - 任意の位置に任意の画像
 - ゲストOSにほとんど依存しない
 - BitVisor1.3にその機能が実装されてる
- グラフィックハードウェアのVRAM領域にデータを書き込んで画像を表示
 - 1ピクセルごとの画像データ

Firefox ウェブブラウザ

kazuhiro@kazuhiro-OptiPlex-7010: ~
kazuhiro@kazuhiro-OptiPlex-7010:~\$

BitVisor: A Single-VM Lightweight Hypervisor - Mozilla Firefox

www.bitvisor.org

BitVisor

「BitVisor Summit 2」12月6日(金)開催!(ComSys 2013に併設)

保存 | 12pt | Japanese | 更新日:2013年8月5日(月)



純国産

eco.png

セキュア VM 機能
最新の研究開発
仮想化技術
最新の技術

新機能 [1]

なし

BitVisor: A Single-VM Lightweight Hypervisor - Mozilla Firefox

www.bitvisor.org

BitVisor

「BitVisor Summit 2」12月6日(金)開催!(ComSys 2013に併設)

ソフトウェア サポート リンク

概要 新機能 プレスリリース リリースノート

想マシンモニタ)の最新版

BitVisor
on 1.3

26日公開!

[ソースコード](#)

セキュア VM 機能の実運用システムとして (Window や Linux がゲスト OS として動作)、最新の研究開発のプラットフォームとして (BSD ライセンスでソースコード公開)、仮想化技術や OS カーネルなどの学習用教材として (Intel / AMD PC で動作)、最新の技術・研究成果を導入したハイパーバイザを存分にご活用ください。

新機能 [1] 新機能 [2]

ADvisorの利用例

- 組織内における情報表示
 - イベントの予定、ニュース、災害情報
- ネットカフェ端末などへの広告表示
 - 定期的な広告表示

ADvisorの問題点

- 現在ADvisor機能では表示する画像データをBitVisorのソースコードに埋め込んでいる
 - 表示する画像を変更できない
 - 変更したい場合再ビルド

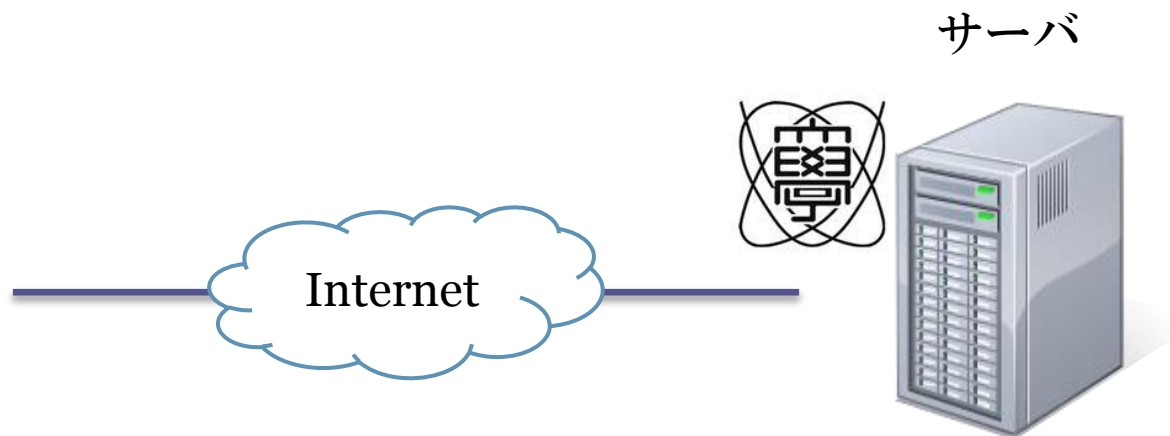
ADvisor機能で表示する画像を
再ビルドなしで変更できるようにしたい

方針

- ネットワーク通信によって画像を変更する
 - サーバから画像を配信
 - 受け取った画像をADvisor機能を用いて表示
- UDP/IPを用いて通信する
 - 特定ポートへのUDPパケットを画像データの通信としてフィルタリング
- ADvisorがサーバに通信できるようにする
 - 表示させたい画像をサーバに要求

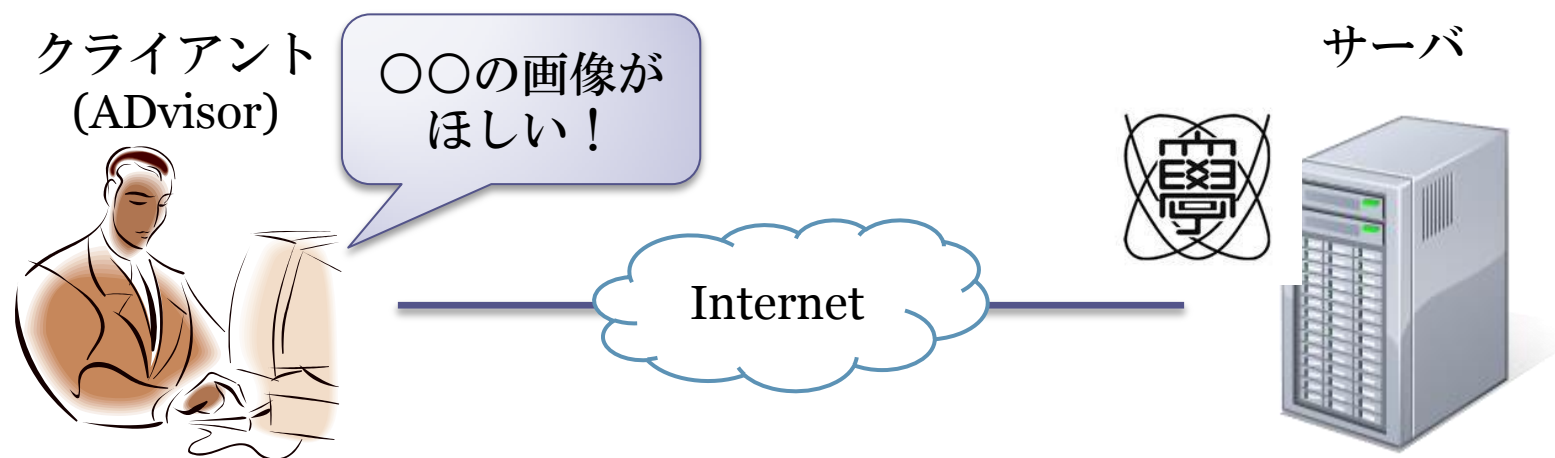
構成

- ADvisorはサーバからクライアントへの特定ポートの packets を捕捉する
- 捕捉したデータを元に画像データを構成、表示
- ADvisorからデータの要求も可能



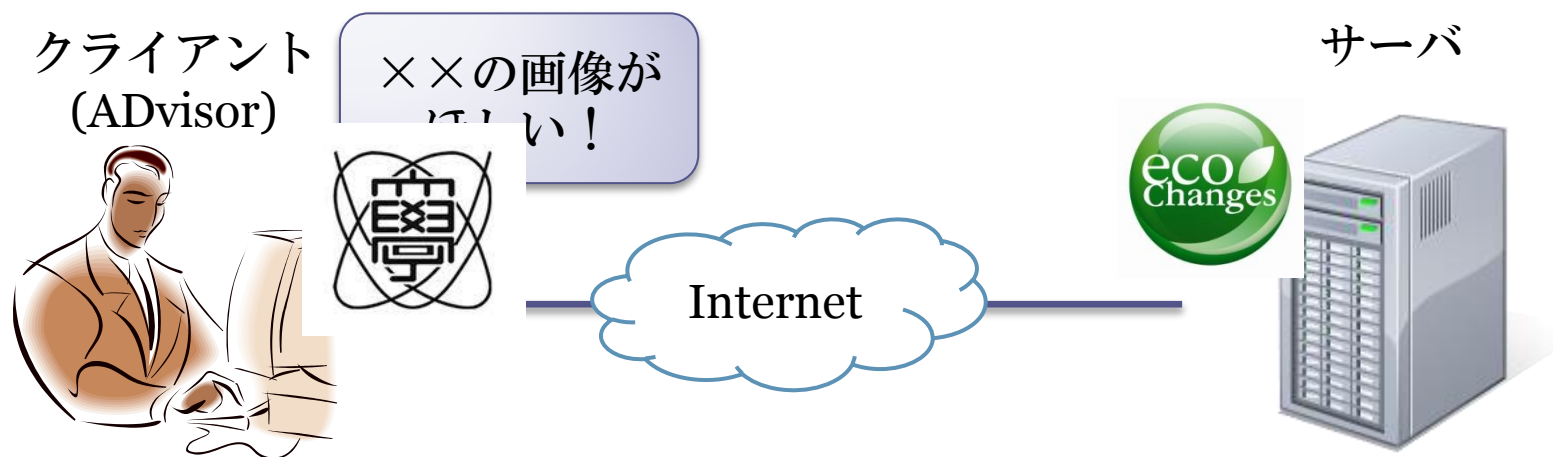
構成

- ADvisorはサーバからクライアントへの特定ポートの packets を捕捉する
- 捕捉したデータを元に画像データを構成、表示
- ADvisorからデータの要求も可能

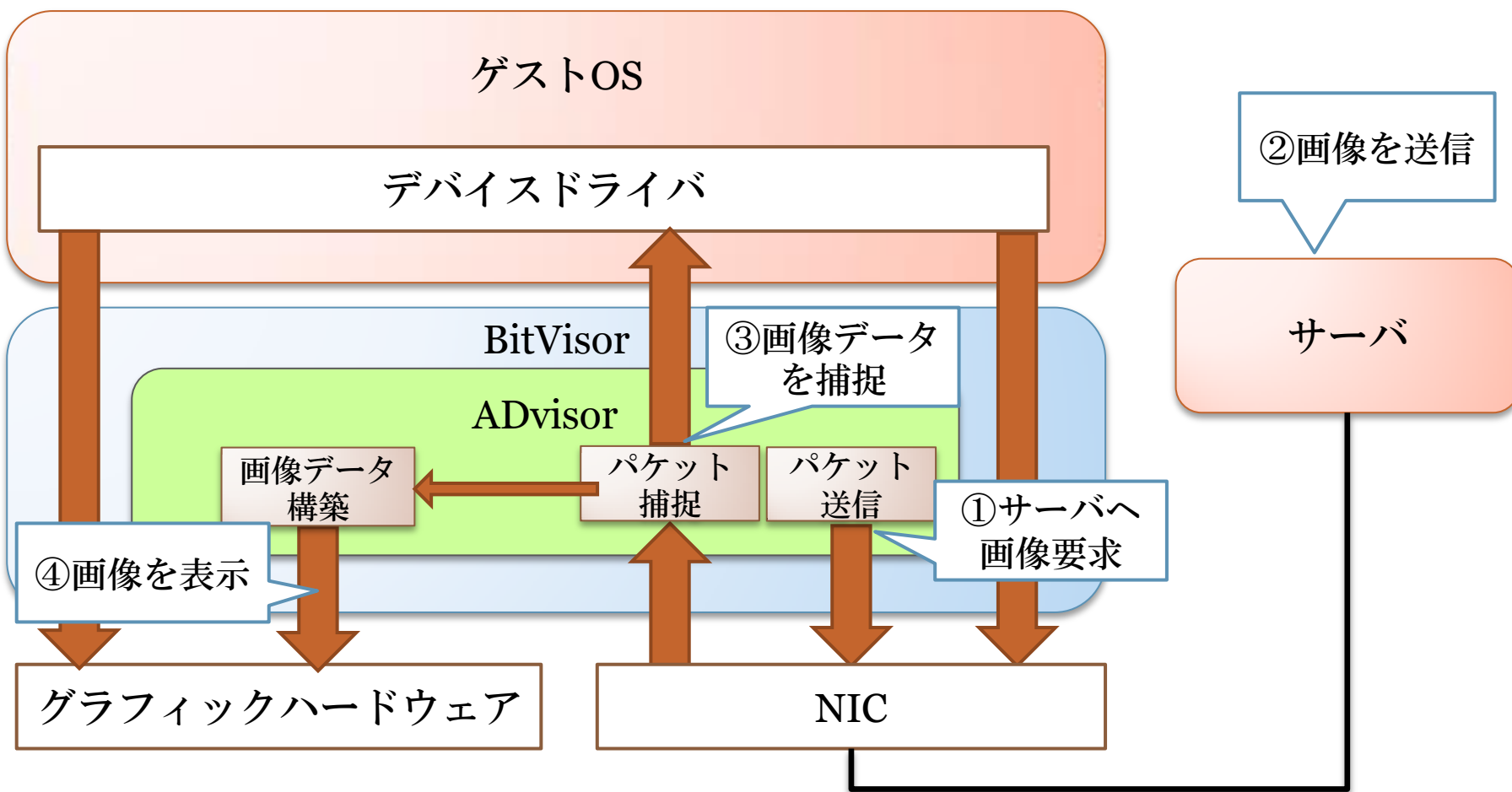


構成

- ADvisorはサーバからクライアントへの特定ポートの packets を捕捉する
- 捕捉したデータを元に画像データを構成、表示
- ADvisorからデータの要求も可能



概要



画像データの送信・構築

- サーバは送信する画像のサイズを最初に送信
その後画像データをパケットに分けて送信
 - 指定したポートにパケットを送信
 - 指定ポートのUDPパケットは画像用
- ADvisorはサイズ分のメモリを確保
その後受信した画像データをメモリに格納

パケットの制御

- 順序制御

- パケット内に画像データの番号を入れる
- ADvisorはその番号をもとにデータを格納する
- ただし最初の画像サイズが送れていないと後のデータは無視される

パケットの捕捉

- vpn/Se/SeVpn.c内でパケットを捕捉
 - UDP/IPパケットを見てIPアドレス、ポート番号を確認
- 画像用のパケットであるなら表示のために処理を行う
 - ゲストOSへパケットは流さない
 - 他のパケットはすべてゲストOSへ流す
- 100行程度

パケットの送信

- core/tty.c内の関数を参考に実装
 - tty_udp_putchar
 - ログをEthernetに流すために利用
 - 一文字ずつログの中身を流している
 - tty_udp_sendを実装
 - 任意の長さのUDP/IPパケットを送信
- 20行程度

IPフラグメンテーション

- 通信回線により転送可能な最大データ長
 - Maximum Transmission Unit (MTU)
- MTUより大きいデータを送る場合パケットを分割する必要がある
- 実装環境ではイーサネットによる接続を想定しているため1500byte以下にする必要がある

フラグメンテーションの回避

- ADvisorで表示できる画像
 - 1ピクセルの各RGBデータが1バイトで表現
 - ・ 1ピクセルあたり3バイト
 - 400*300ピクセルの画像で大体350KBくらい
- IPフラグメンテーションを起こさないために480ピクセル(1440バイト)ずつデータを送信
 - 想定する環境に合わせてサーバ側でプログラムを変更する必要がある

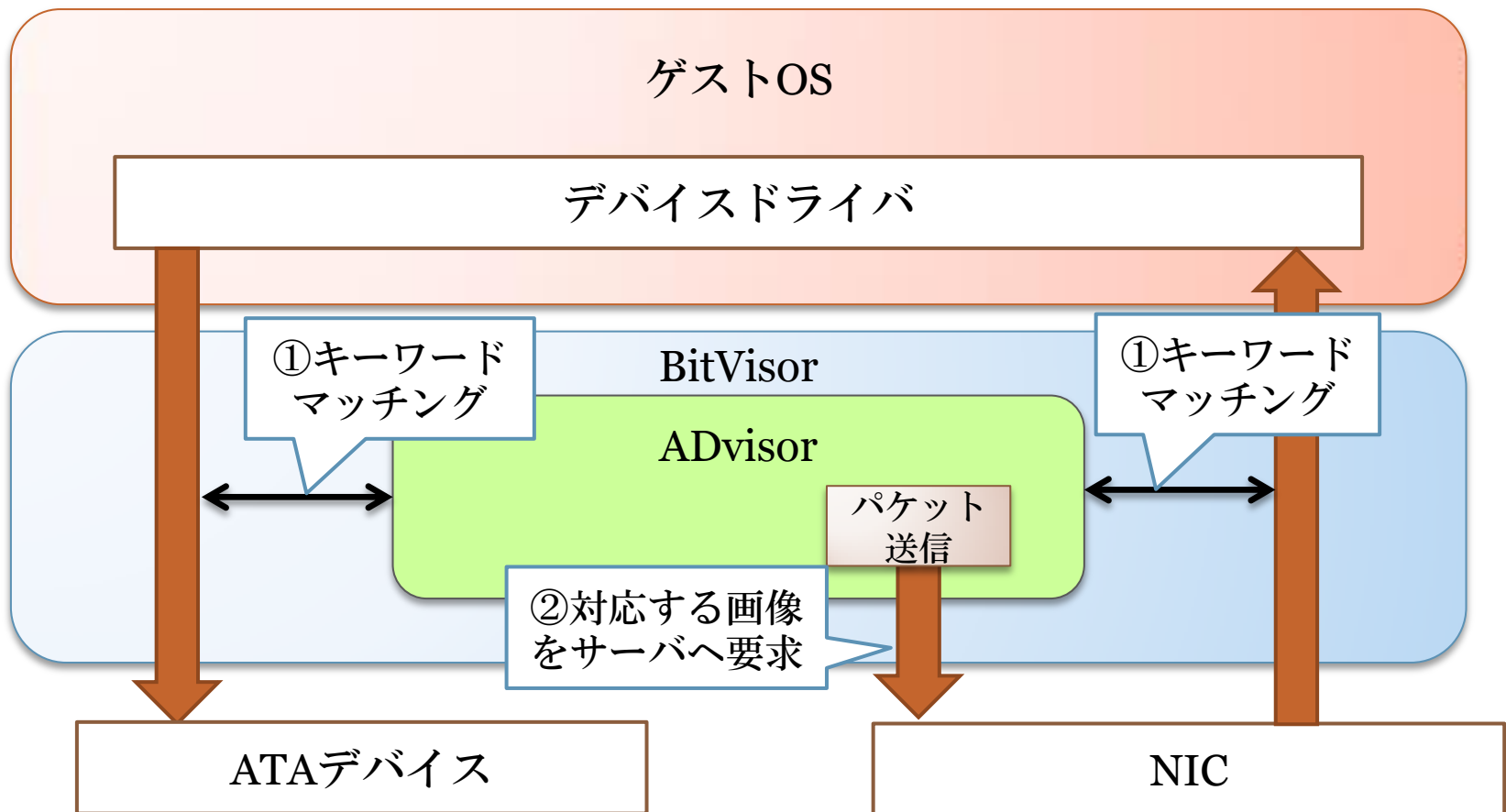
画像の表示タイミング

- 画像データがそろっているか定期的なチェック
 - CR3レジスタの更新時
 - 一定回数更新が行われたらチェック
 - 一定時間の経過
 - 一定周期にチェックを行うタスクの追加
 - 今回の実装ではCR3レジスタの更新時

ゲストOSとの連動

- 文字列のマッチングによる画像表示
 - ディスクへの書き込み
 - ゲストOSが受信したHTTPパケット
 - キーワードはあらかじめADvisorが保持
- キーワードがマッチしたら表示させたい画像をサーバへ要求
- サーバは要求された画像を送信
 - サーバ側で画像を変えることで表示する画像が変更できる

ゲストOSとの連動



テキストメッセージの表示

- ADvisorで任意の文字列を表示させる
 - サーバで文字列を画像化して送信
 - Pythonのライブラリを使用
 - 日本語などのマルチバイト文字も表示可能

まとめと今後の課題

- まとめ
 - 通信により任意の画像をADvisorで表示
 - ゲストOSのディスクI/Oやネットワーク通信に関連した画像を表示可能
- 今後の課題
 - 画像を裏で更新する
 - ・ 現在の実装は受け取ったデータをすぐに表示
 - ・ 受け取った画像を保存
 - パケット損失への対応
 - 画像データの圧縮