HITACHI
Inspire the Next

Linux Technology Center

仮想化環境における情報収集フレームワーク VESPERの設計と実装

Hitachi, Ltd., Systems Development Laboratory
Linux Technology Center

守屋　哲 <satoru.moriya.br@hitachi.com>
金　成昊 <sungho.kim.kd@hitachi.com>
大島　訓 <satoshi.oshima.fk@hitachi.com>

HITACHI
Inspire the Next

目次

1. 背景

HITACHI
Inspire the Next

サーバシステムへの要求

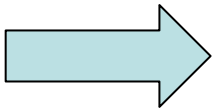
- 信頼性、可用性
  - 基幹系サーバ
  - クラウドコンピューティング
  - ➔ HA クラスタ
- 資源の有効活用
  - ハードウェア（CPU、メモリ等）の高機能化
  - 投資コストの削減
  - ➔ 仮想化

⇒ 仮想化環境におけるクラスタシステム

仮想化環境におけるクラスタシステム

- クラスタモデル
  - 仮想化環境におけるクラスタの構築方法
- VMの監視方法
  - クラスタモデルに依存

Linux-HA（Heartbeat）
- クラスタ管理ソフトウェア
- VMを管理



モデル、監視方法を検討

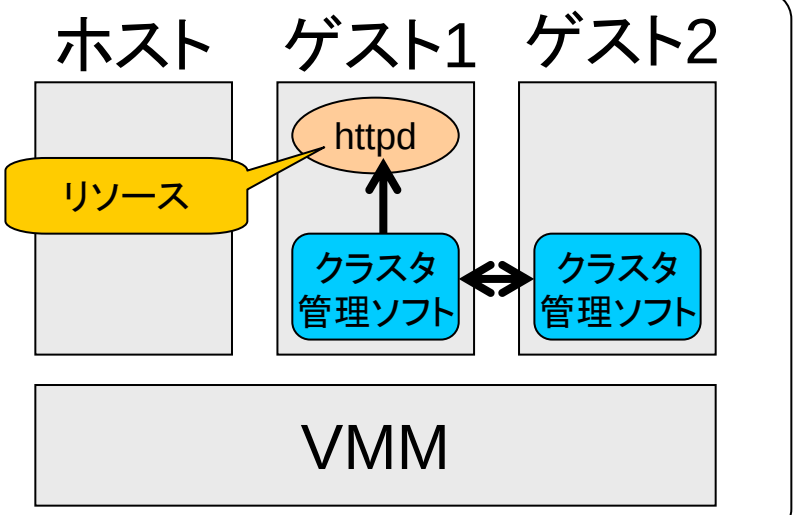
Heartbeat

- Heartbeatとは
  - クラスタ管理ソフトウェア
  - フェイルオーバ、ロードバランス等が可能
  - 監視対象をリソースとして管理
- リソースとは
  - ユーザにサービスを提供するために必要な全ての構成要素
  - IPアドレス、サーバプロセス（Web、DNS・・・）

Heartbeatのクラスタモデル

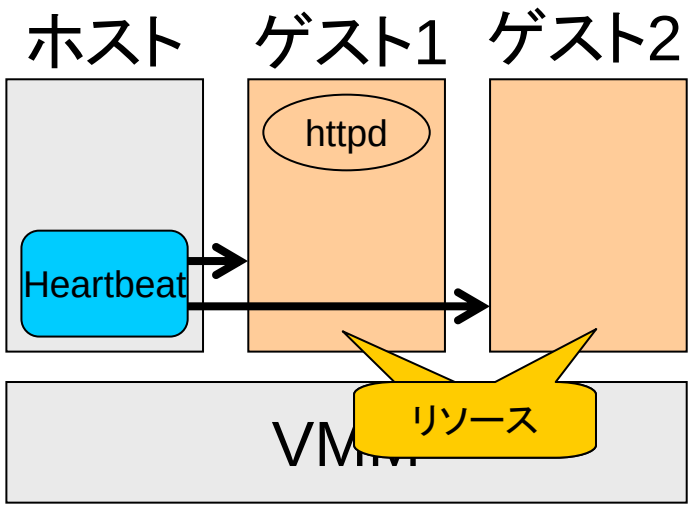
- 特徴：VMをリソースとして管理

通常のクラスタ



•クラスタ管理ソフトウェアはゲストOS上で動作
•httpdをリソースとして管理
•VMがどの物理サーバで動作しているか特定するのが困難

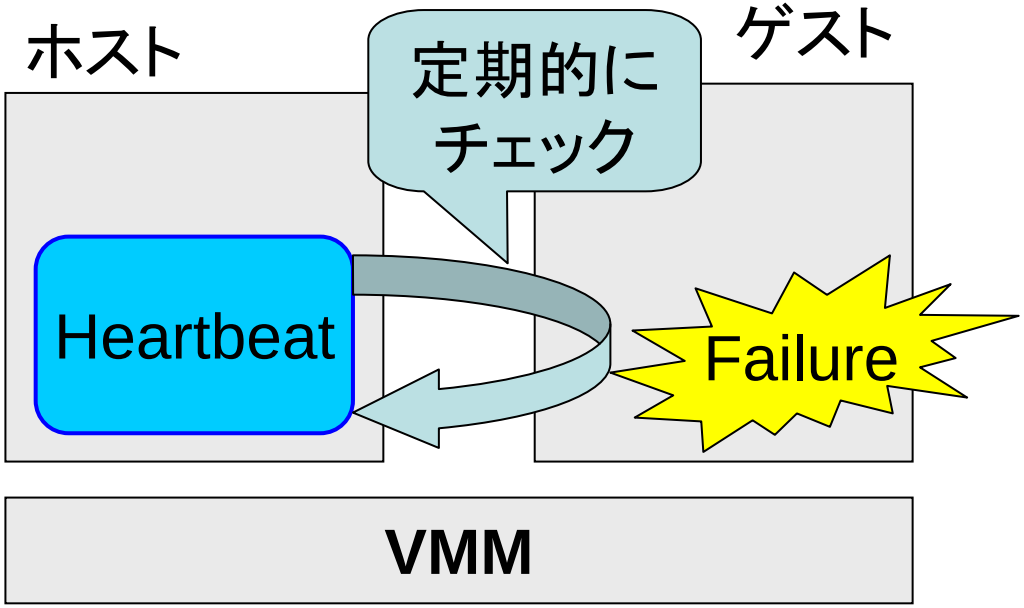
Heartbeatによるクラスタ



•クラスタ管理ソフトウェアはホストOS上で動作
•VMをリソースとして管理
•VMがどの物理サーバで動作しているか特定が容易

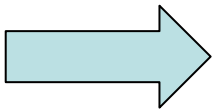
Heartbeatの生死監視

- ポーリング方式
  - 定期的に生死確認
- 障害検知のタイミング
  - ポーリング周期に依存



仮想マシンの監視方法

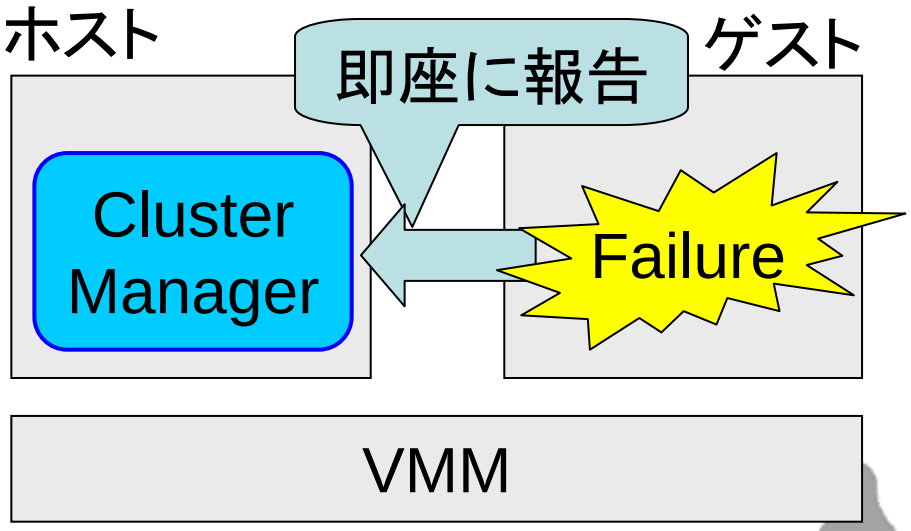
- Heartbeatの方法で十分？
  - これまでと同じ短所あり
- 障害検知の遅延
  - 定期的な生死確認
  - 時間切れによる障害検知
- 障害解析機能の不備
  - サービスのみの障害？
  - システム全体の障害？



他の生死監視情報を利用することで
障害検知の迅速化や障害解析を改善できるのでは？？

プローブによる生死監視

- 仮想マシンにプローブを挿入
  - 統計情報
  - 障害発生時に通るPath
- 障害検知遅延の改善
  - イベントドリブン方式
  - 障害の迅速な検出
- 障害解析機能の追加
  - メモリ障害？
  - ネットワーク障害？



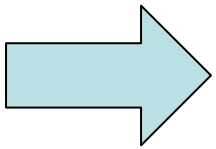
VESPERの提案

VMにプローブを挿入して監視するためには・・・

- 任意の部分へのプローブの挿入
- 動的にプローブを挿入

} kprobes

- ホストからのプローブの挿入/削除
- プローブしたデータへのホストからのアクセス



VESPER

（Virtual Embraced Space ProbER）

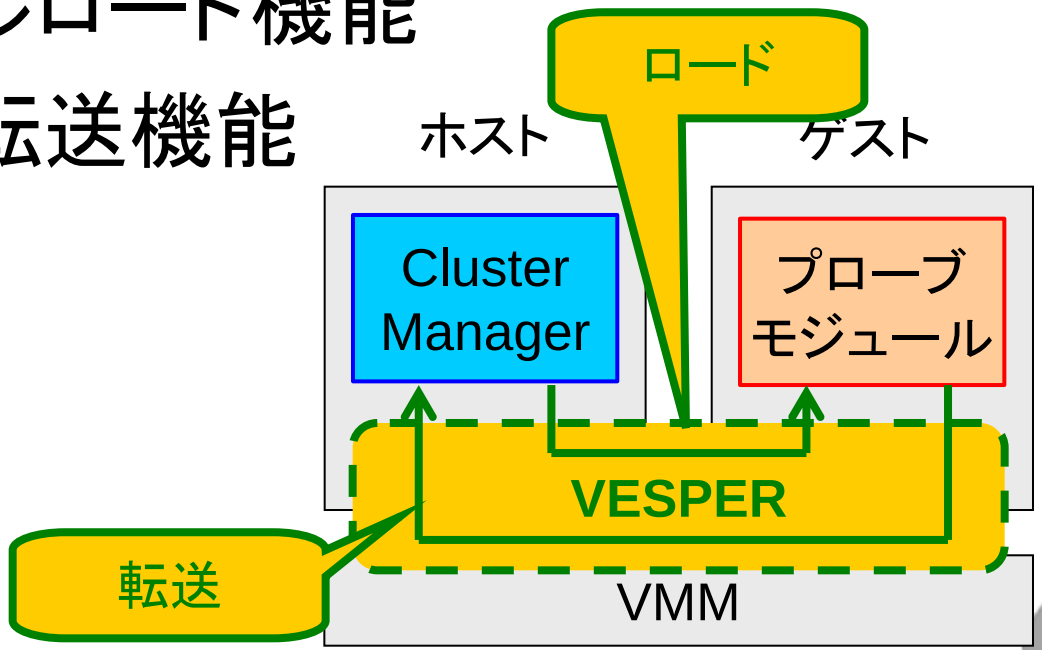
仮想化環境におけるKprobes活用フレームワーク

2. VESPERのコンセプトと構造

HITACHI
Inspire the Next

コンセプト

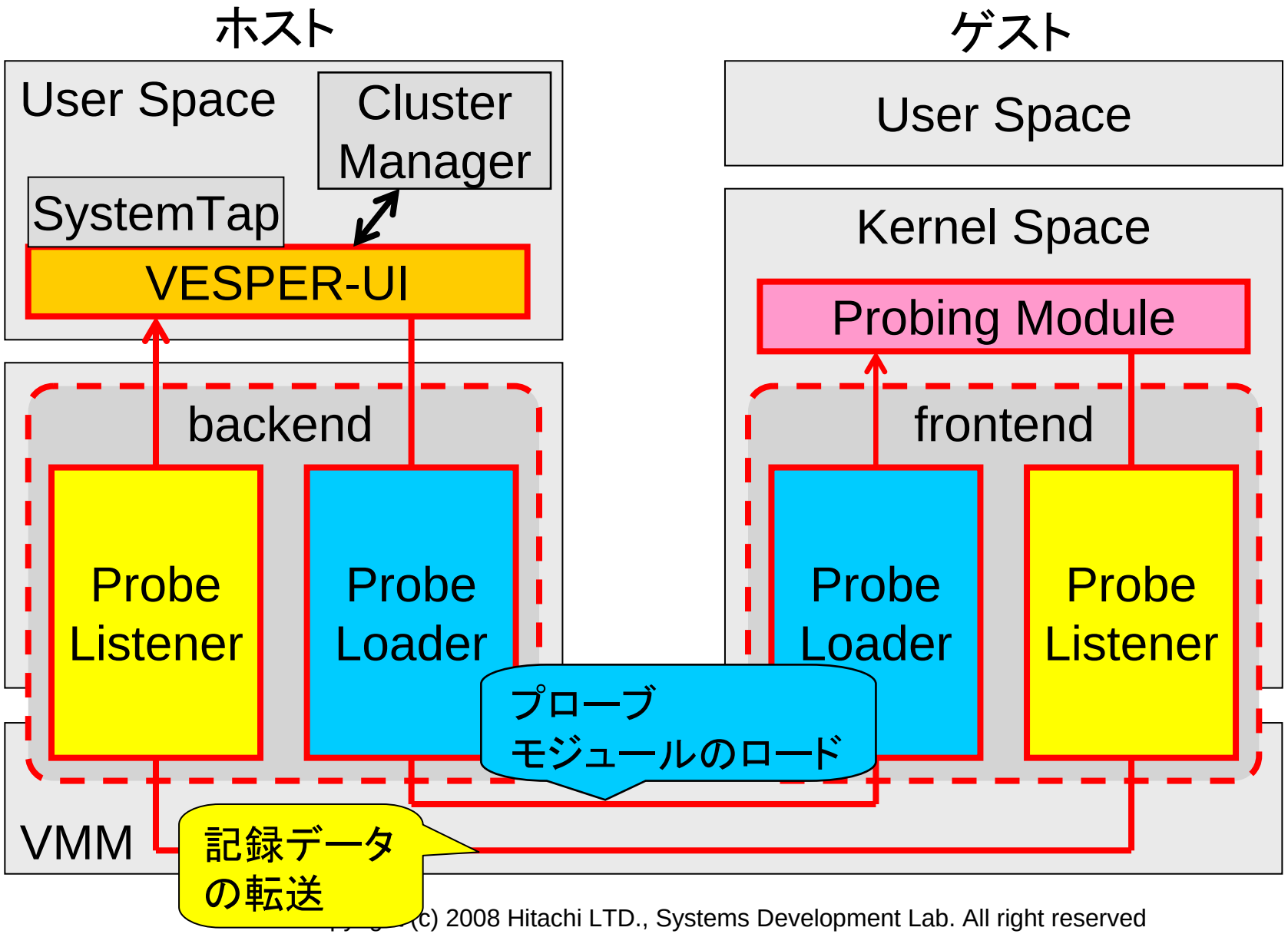
- VESPERはVMにプローブを挿入するための フレームワーク
- プローブの挿入はKprobesを利用
  - プローブモジュールをロードする
- プローブモジュールロード機能
- プローブデータの転送機能



設計方針

- 分割ドライバ方式
  - backend/frontendドライバ
- プローブモジュールのロード時にゲストのユーザ空間を利用しない
  - ユーザ空間に障害が生じた後でもプローブの挿入が可能
- SystemTap
  - 容易にプローブモジュール作成可能
  - kprobes + relayfs

基本構造



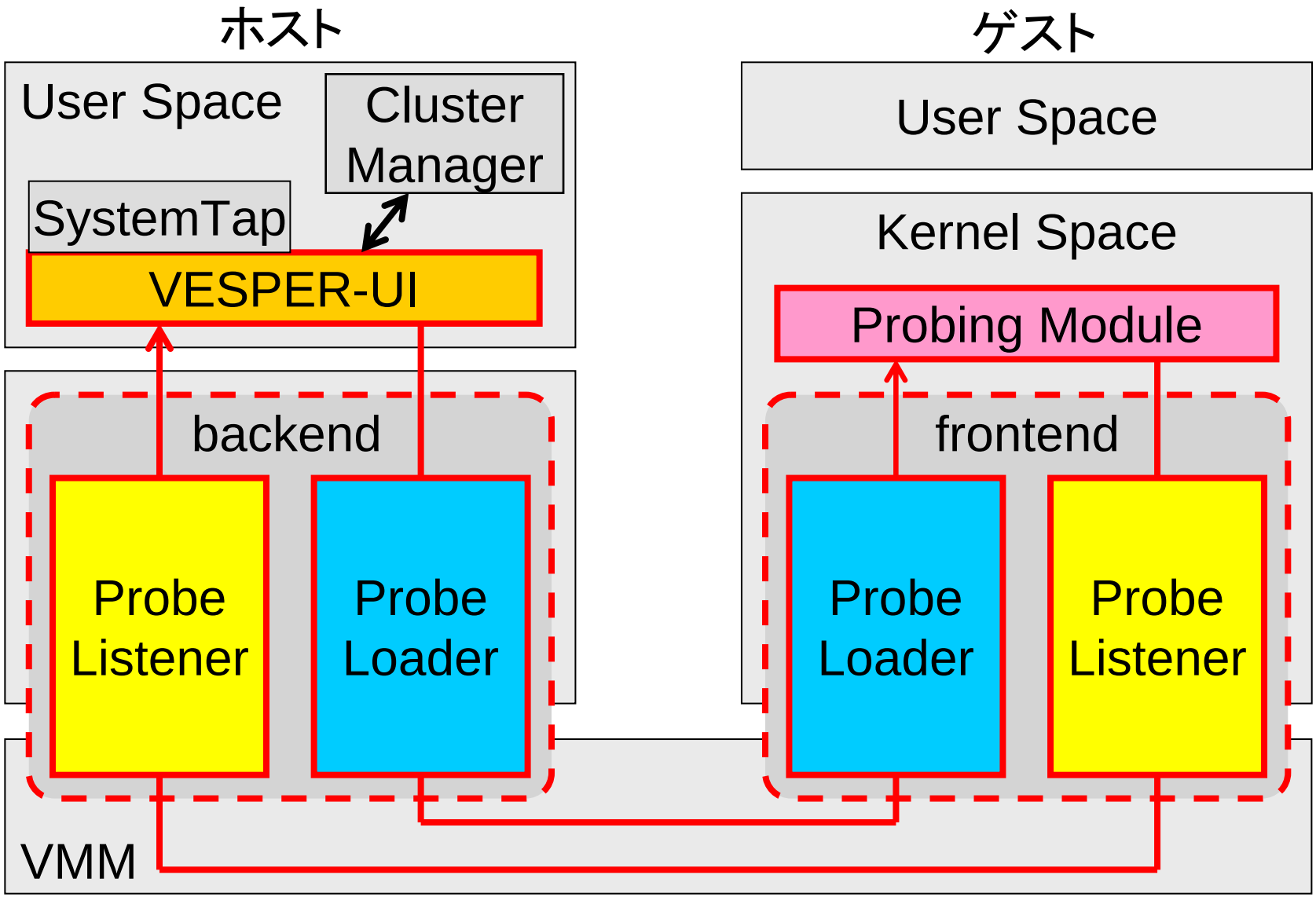
3. VESPERの実装

HITACHI
Inspire the Next

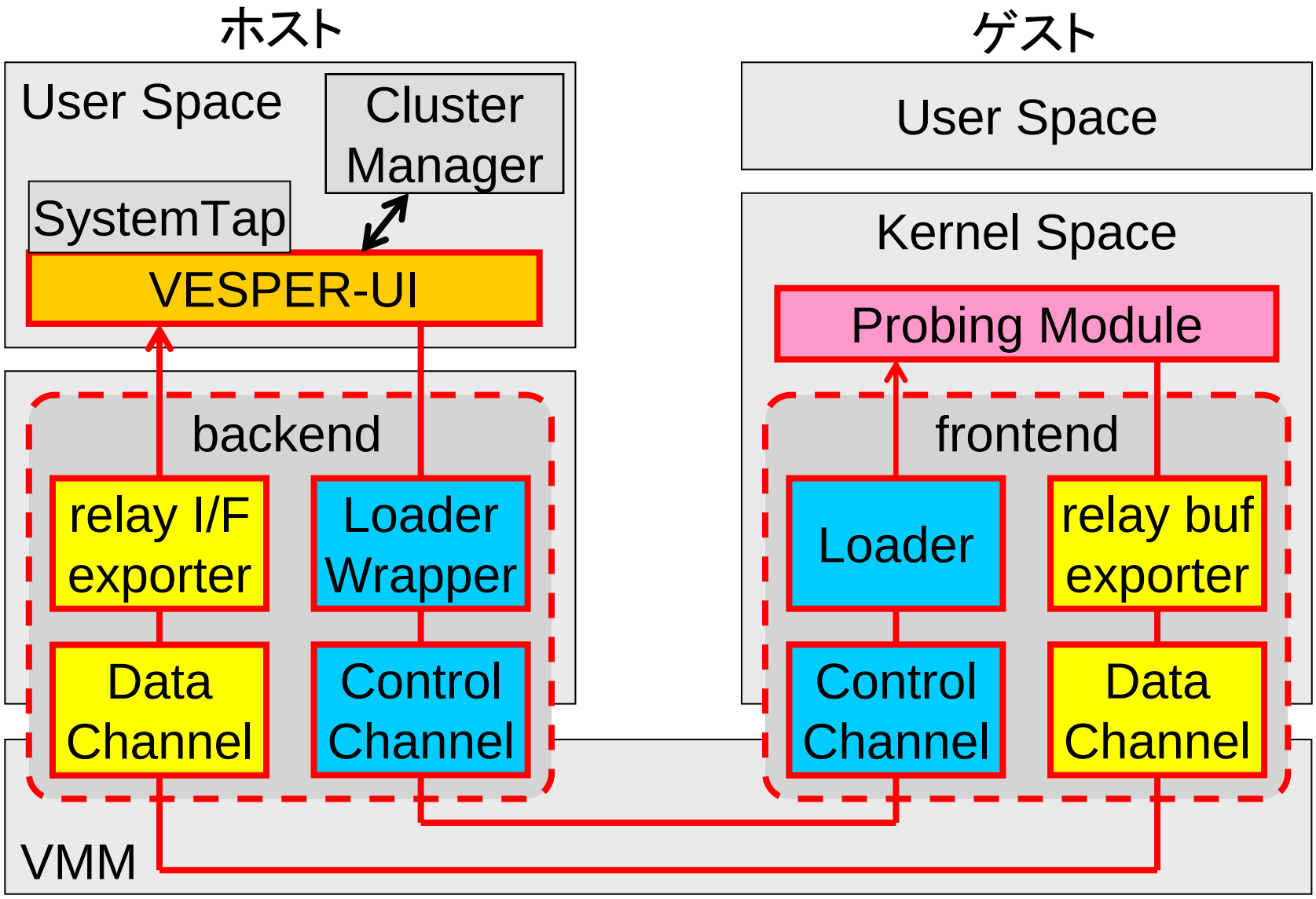
実装方針

- ターゲット：Xen（x86_32）
  - 充実したVM間通信インフラを提供
- 移植性：3階層構造
  - 階層に分けることで、VMM依存部分を局所化
  - 依存部分だけを変更することで他のVMMへ対応
- SystemTapのサポートなし
  - Kprobesとrelayfsを利用してプローブモジュールを実装

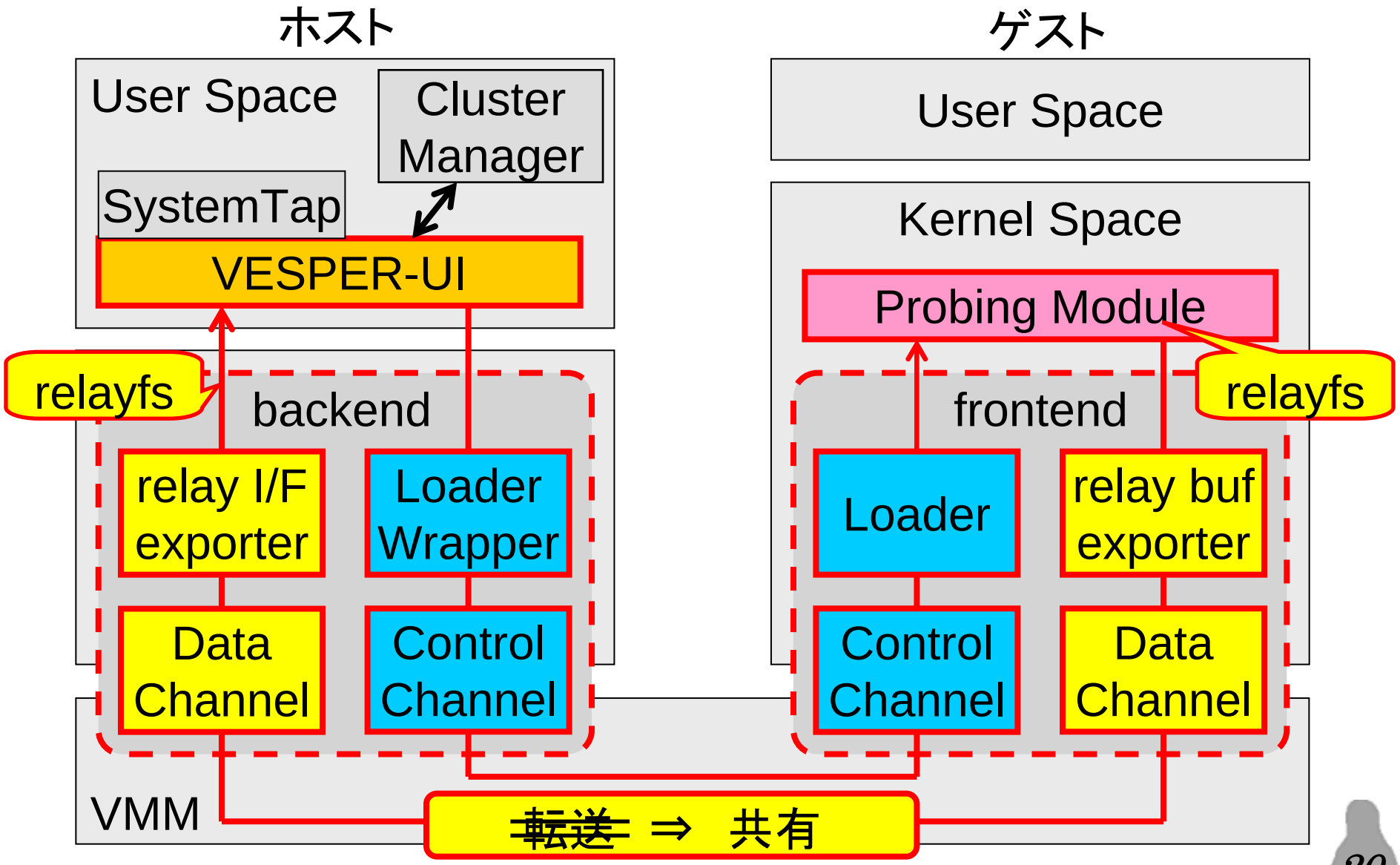
基本構造



詳細構造



詳細構造



VESPERの動作

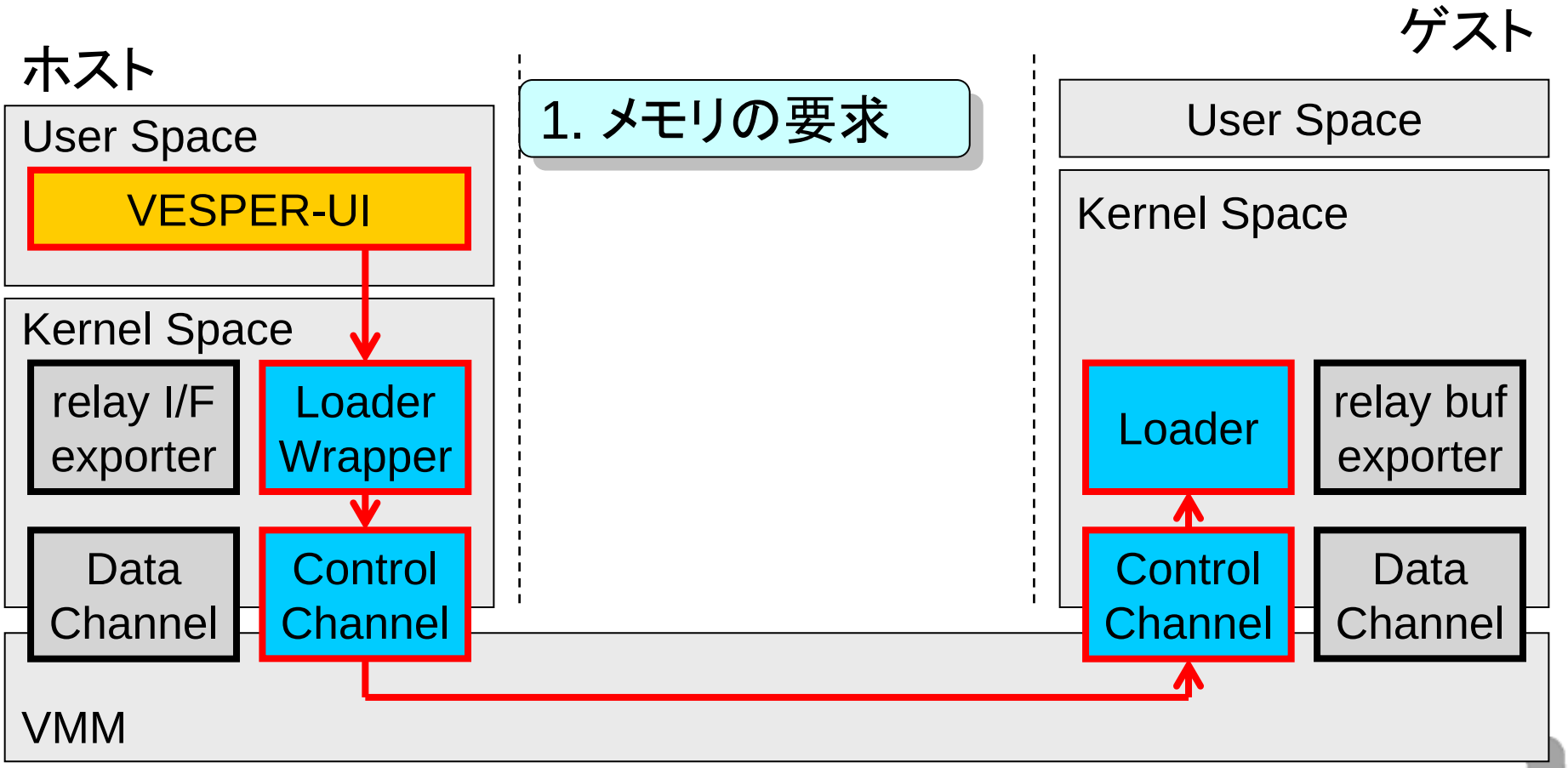
1. VESPERのロード
2. Activate
3. プローブモジュールのロード
4. ゲスト情報の収集
5. プローブモジュールのアンロード
6. Deactivate
7. VESPERのアンロード

VESPERの動作

1. VESPERのロード
2. Activate
3. プローブモジュールのロード
4. ゲスト情報の収集
5. プローブモジュールのアンロード
6. Deactivate
7. VESPERのアンロード

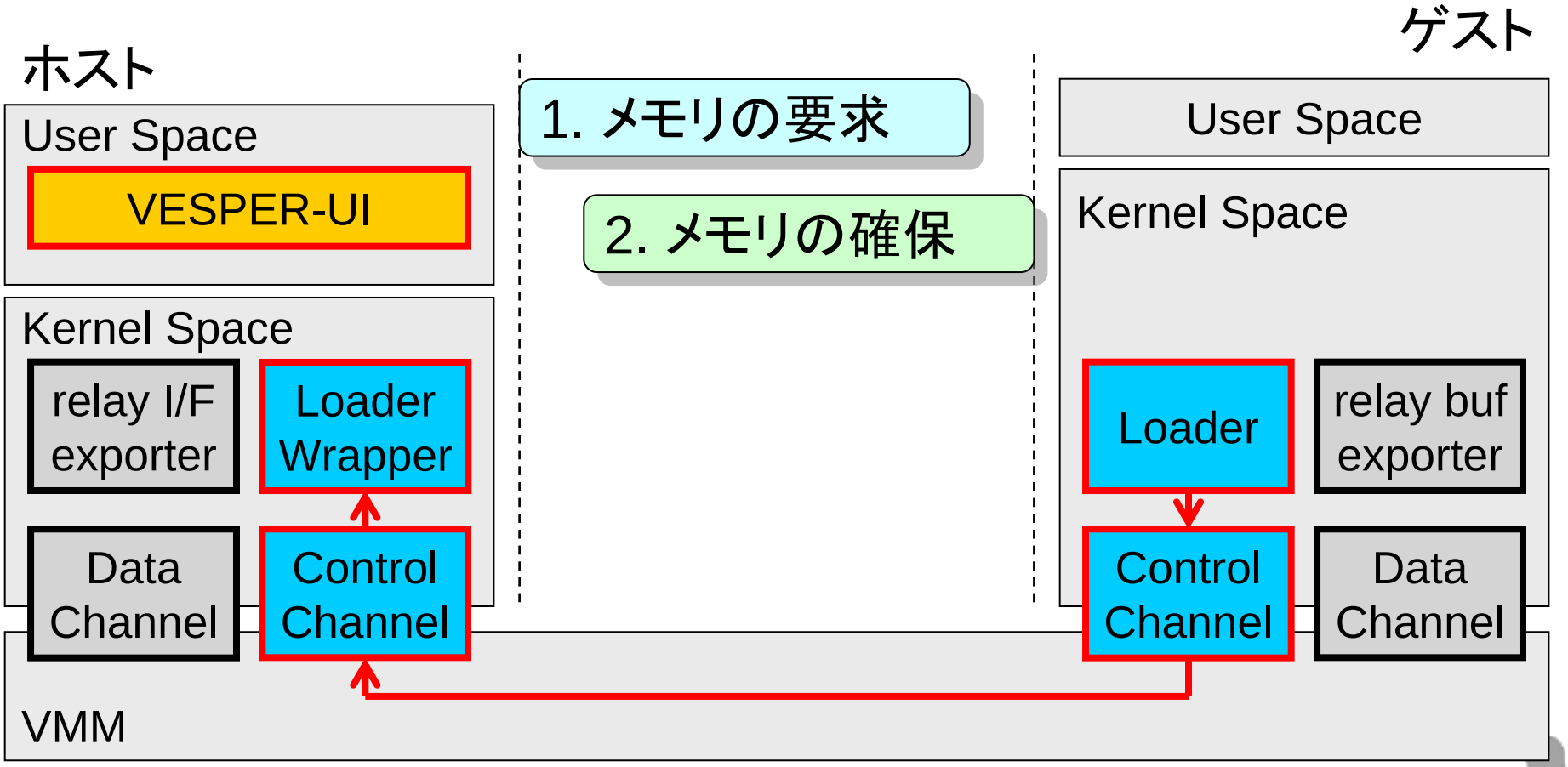
モジュールのロード - Loader

- モジュールイメージの転送
  - メモリの要求



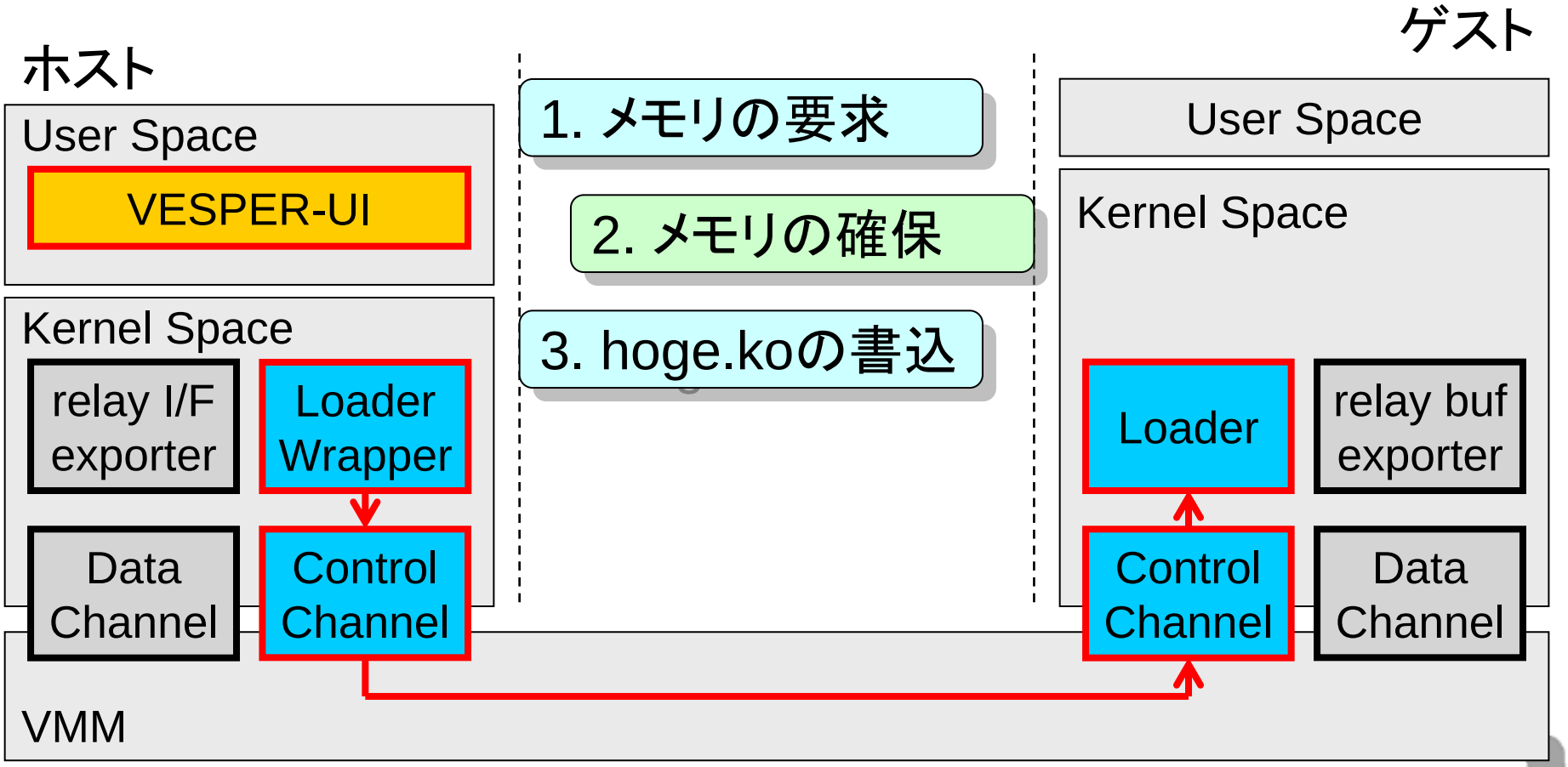
モジュールのロード - Loader

- モジュールイメージの転送
  - メモリの確保



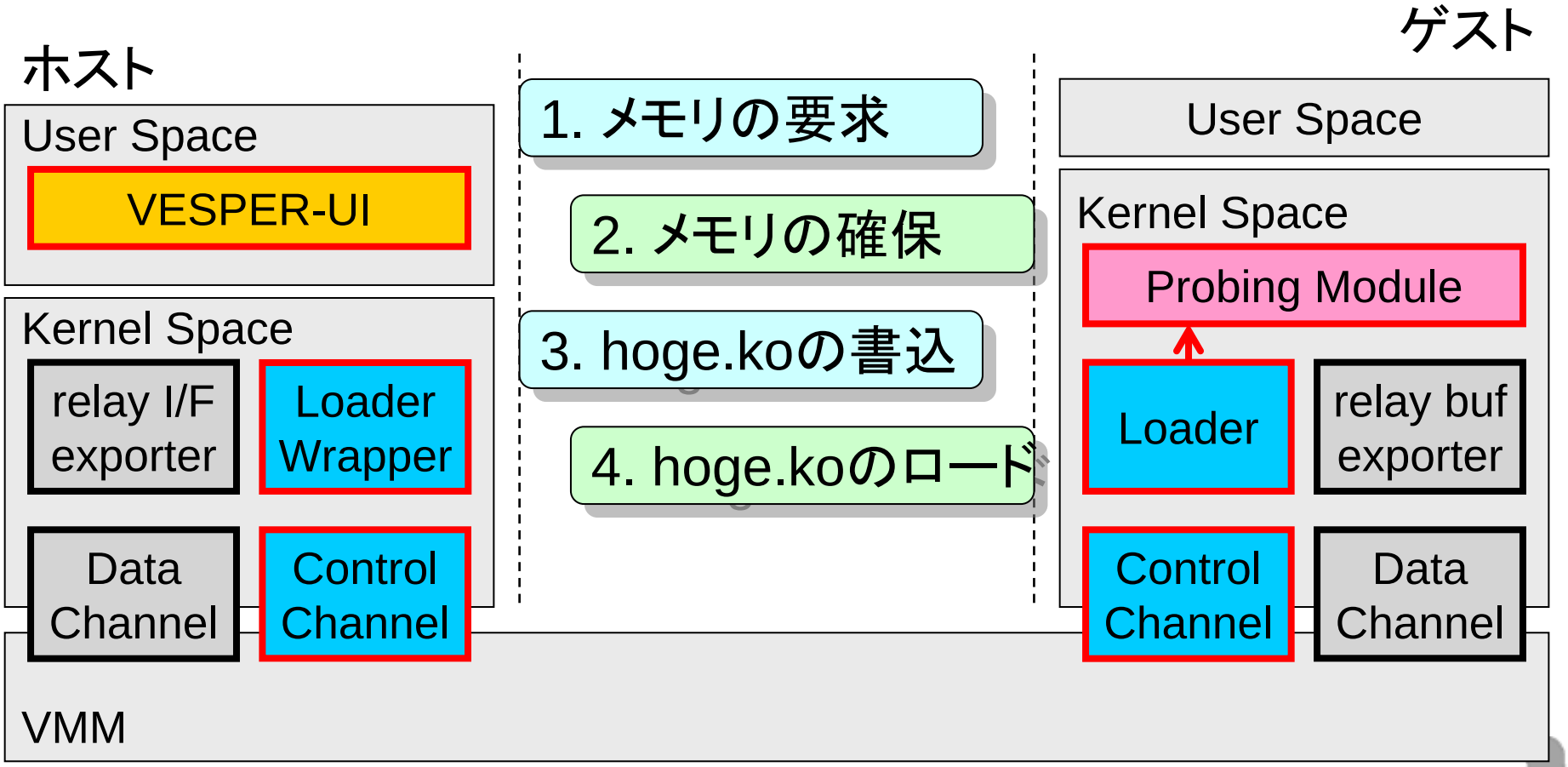
モジュールのロード - Loader

- モジュールイメージの転送
  - モジュールイメージの書き込み



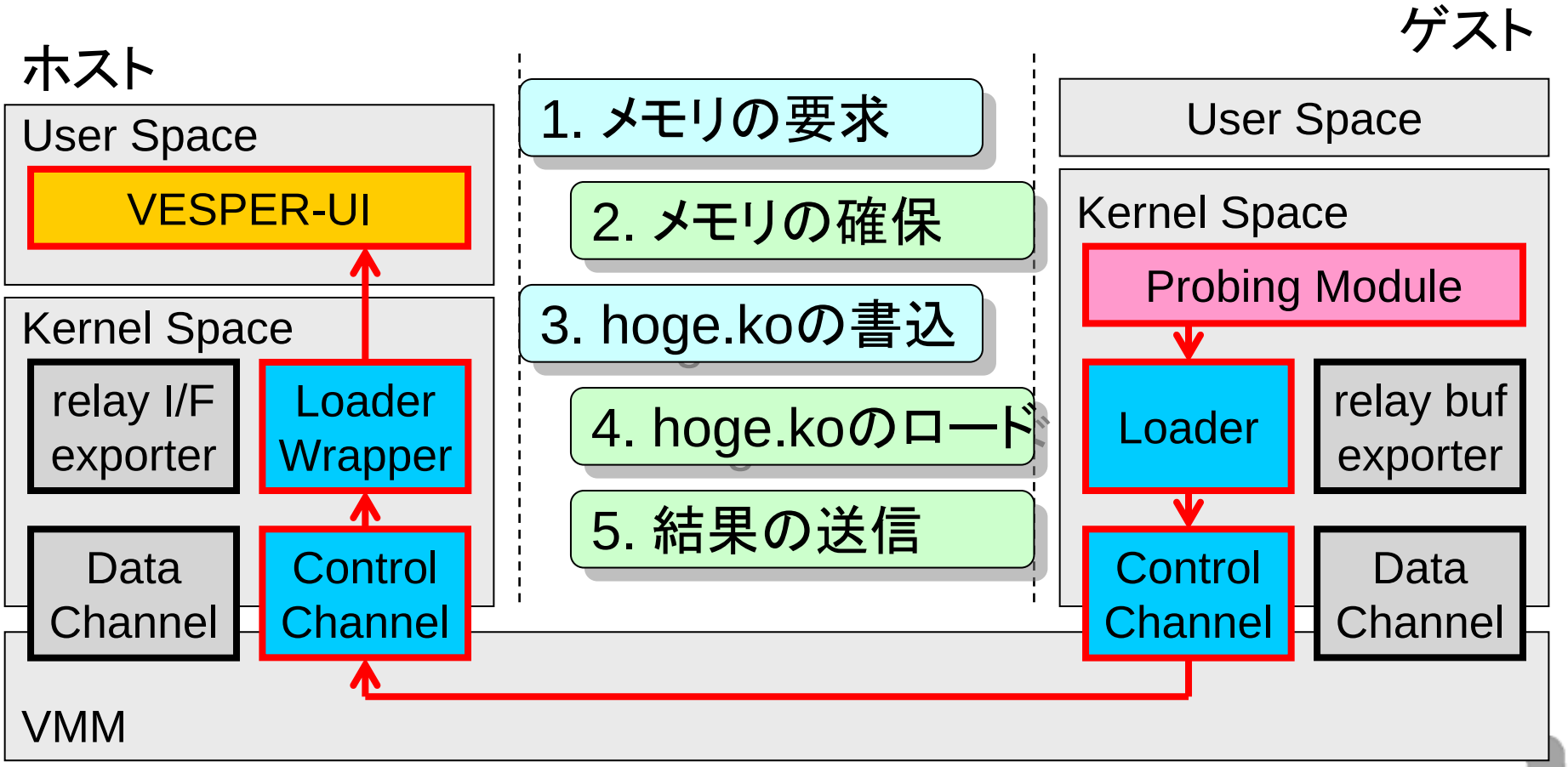
モジュールのロード - Loader

- モジュールのロード
  - sys_init_module()の呼び出し



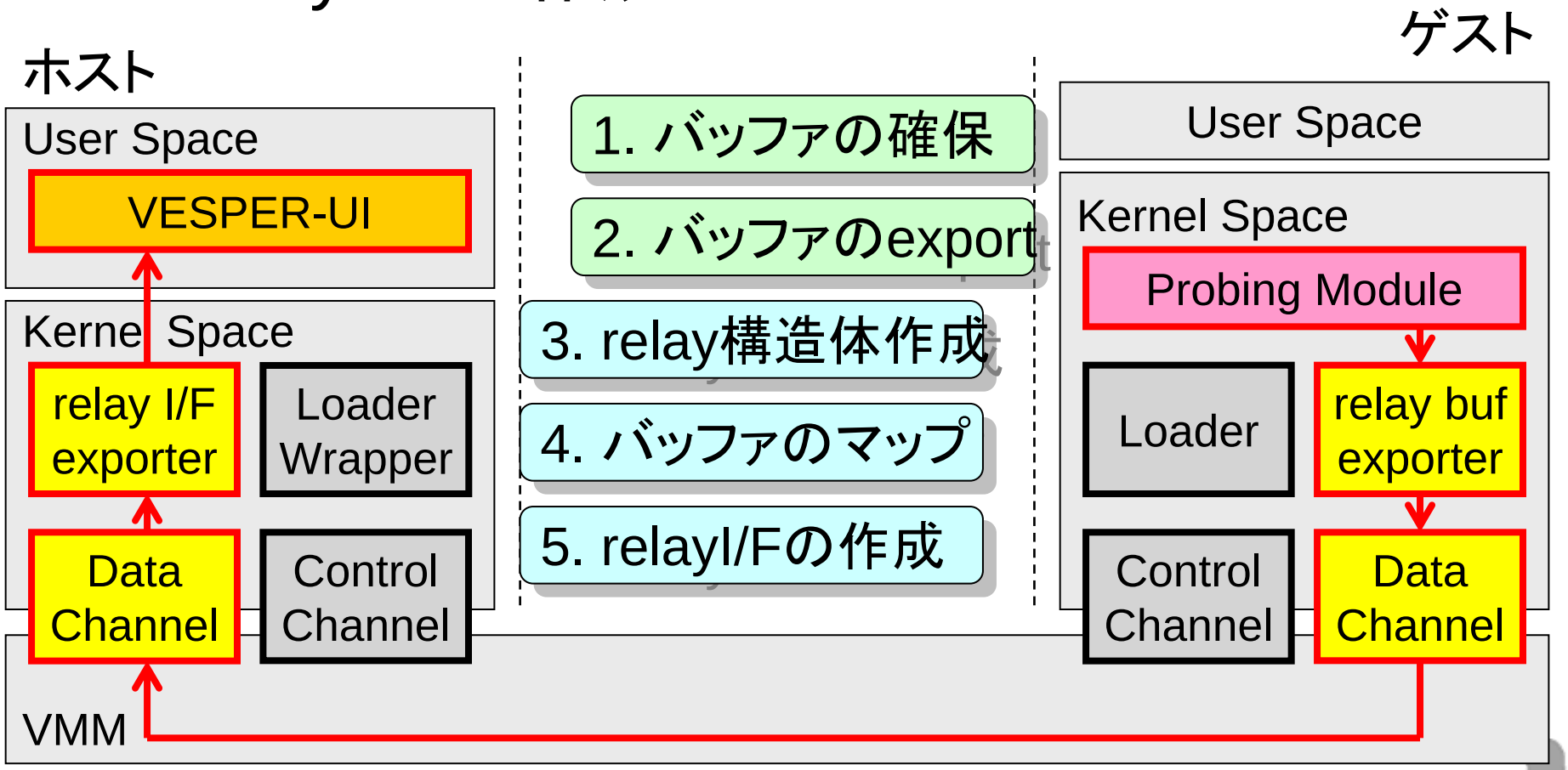
モジュールのロード - Loader

- モジュールのロード
  - 結果の送信



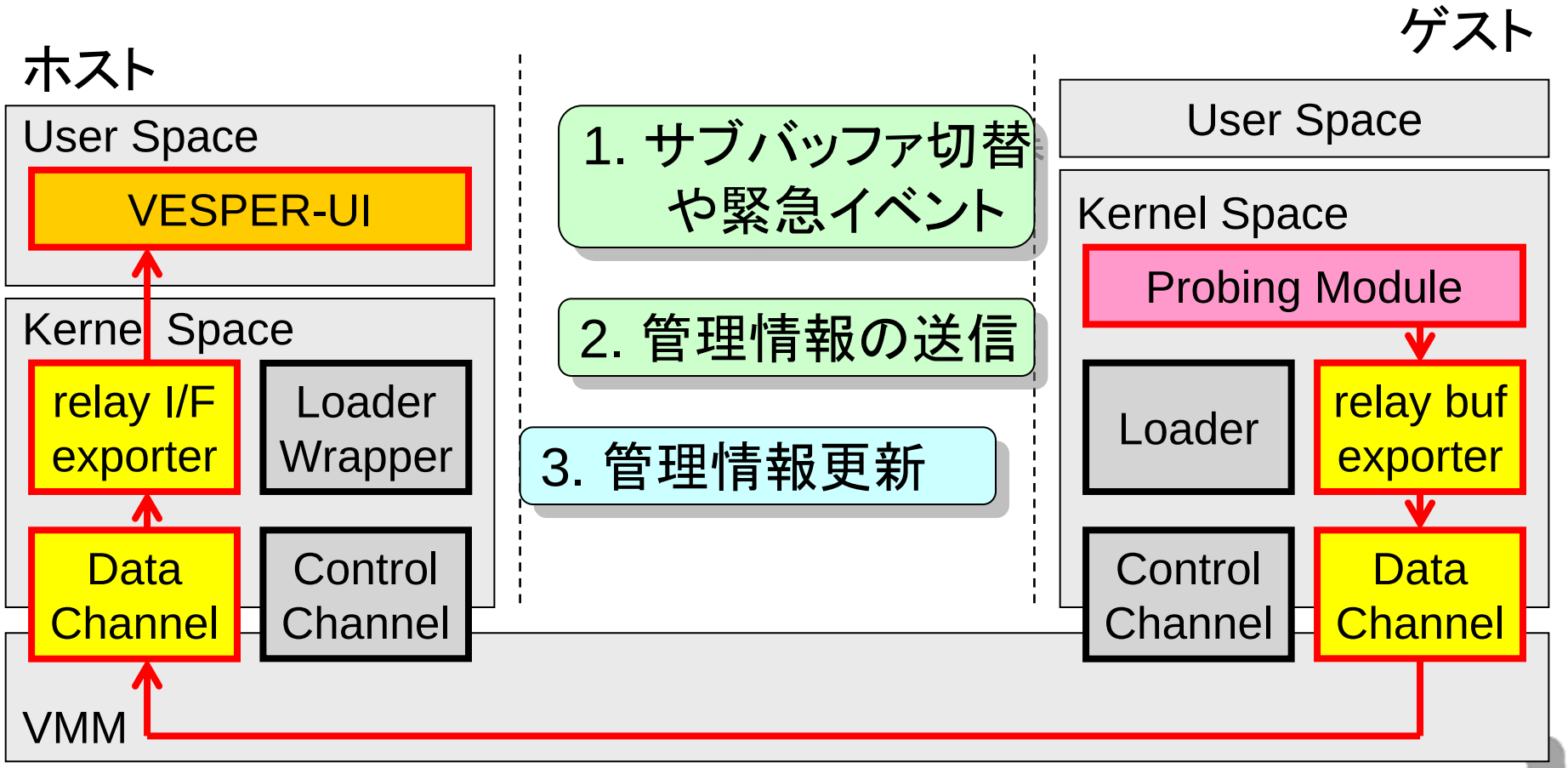
モジュールのロード - Listener

- Relayバッファの共有
- Relay I/Fの作成



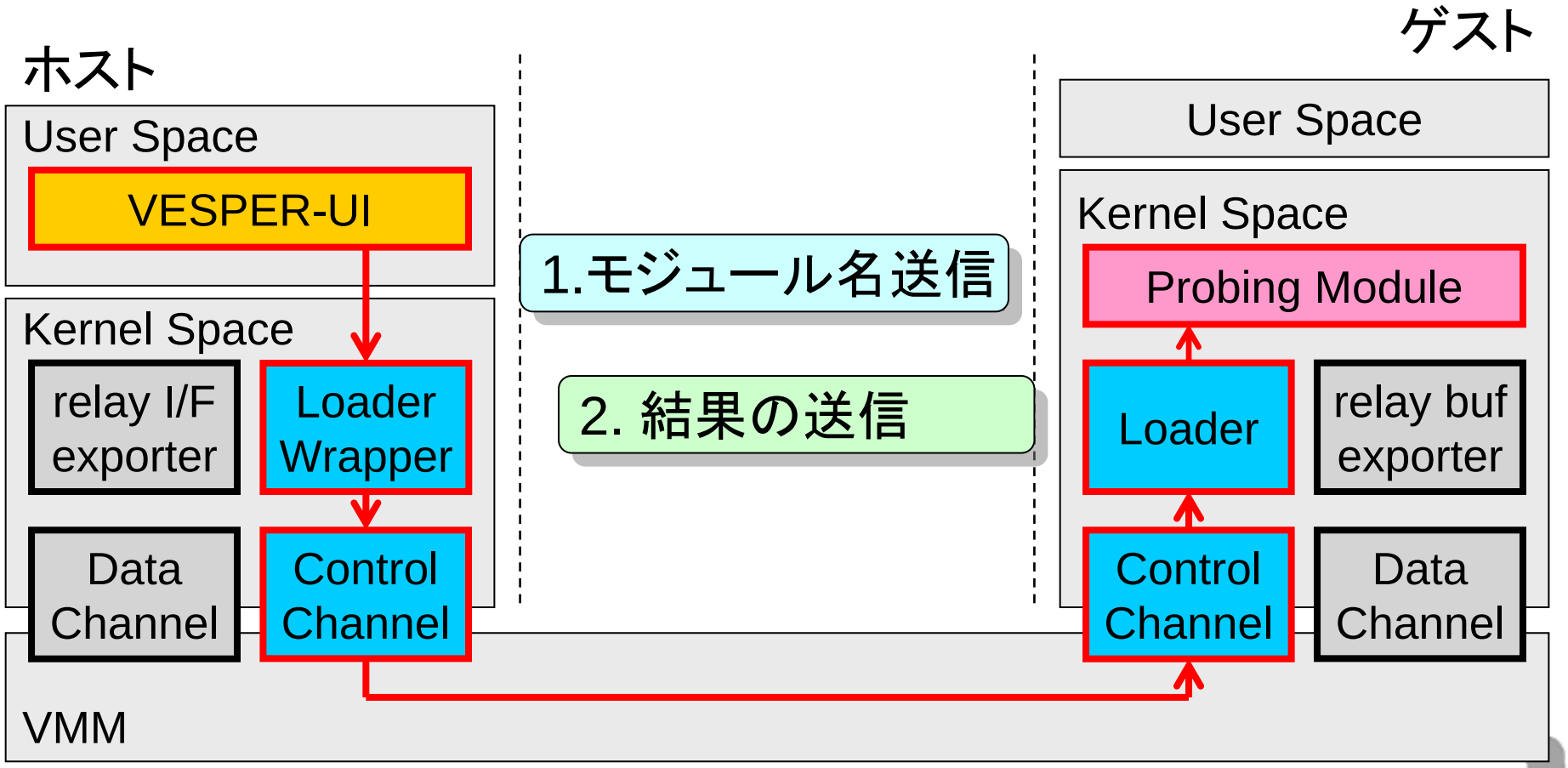
ゲスト情報の収集 - Listener

- バッファ管理情報の同期
  - 管理情報の転送



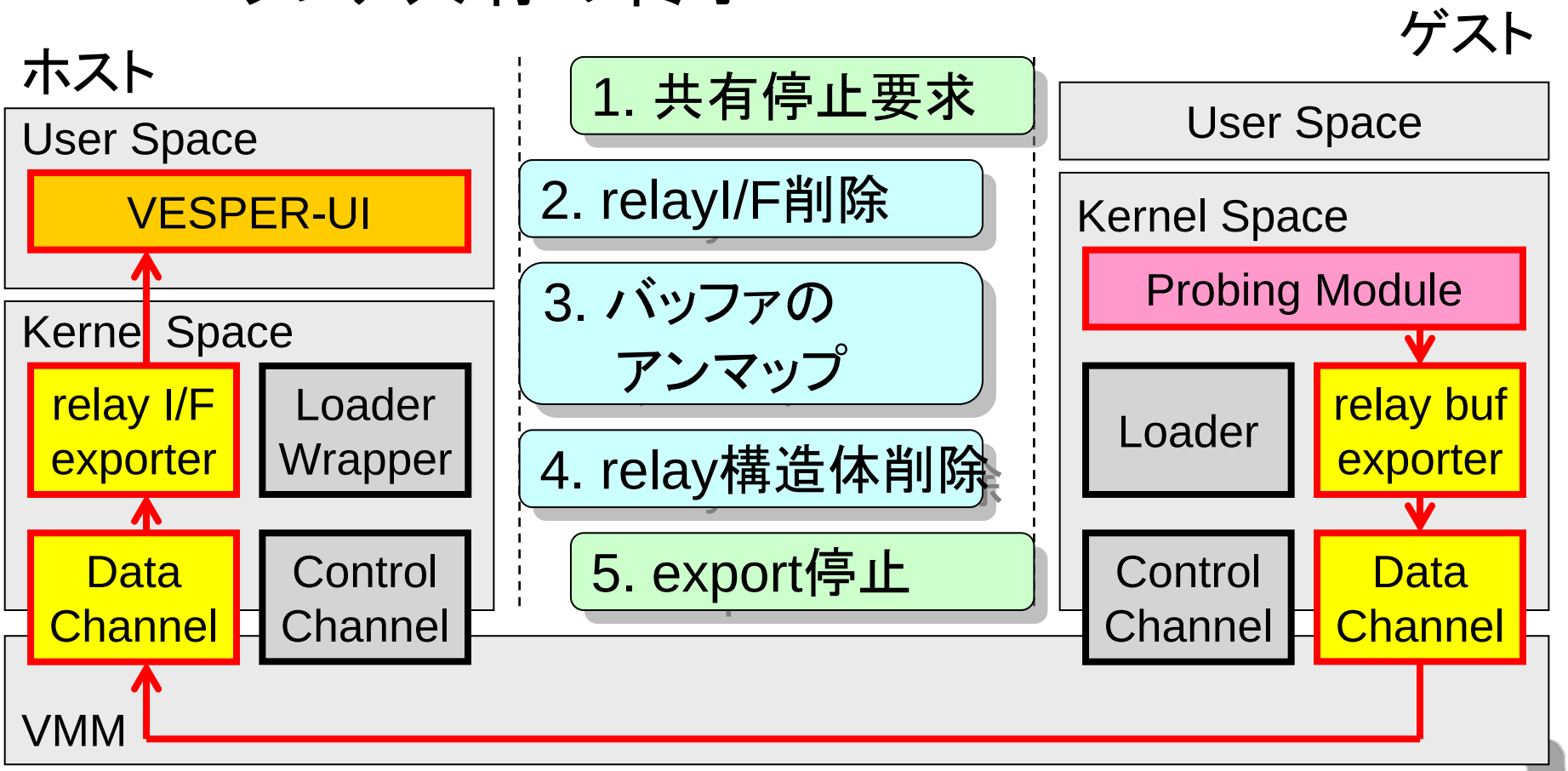
モジュールのアンロード - Loader

- モジュール名の送信
- 結果の送信



モジュールのアンロード - Listener

- Relay I/Fの削除
- バッファ共有の終了



プローブモジュールの作成方法

- Kprobesとrelayfsを利用
- VESPERが提供するI/Fを利用
  - バッファのexport
    - int vsp_relay_export_start(struct rchan *rchan, const char *modname)
  - exportの停止
    - void vsp_relay_export_end(const char *modname)
  - 管理情報の同期
    - void vsp_update_host_index(struct rchan_buf *buf)

4. 評価

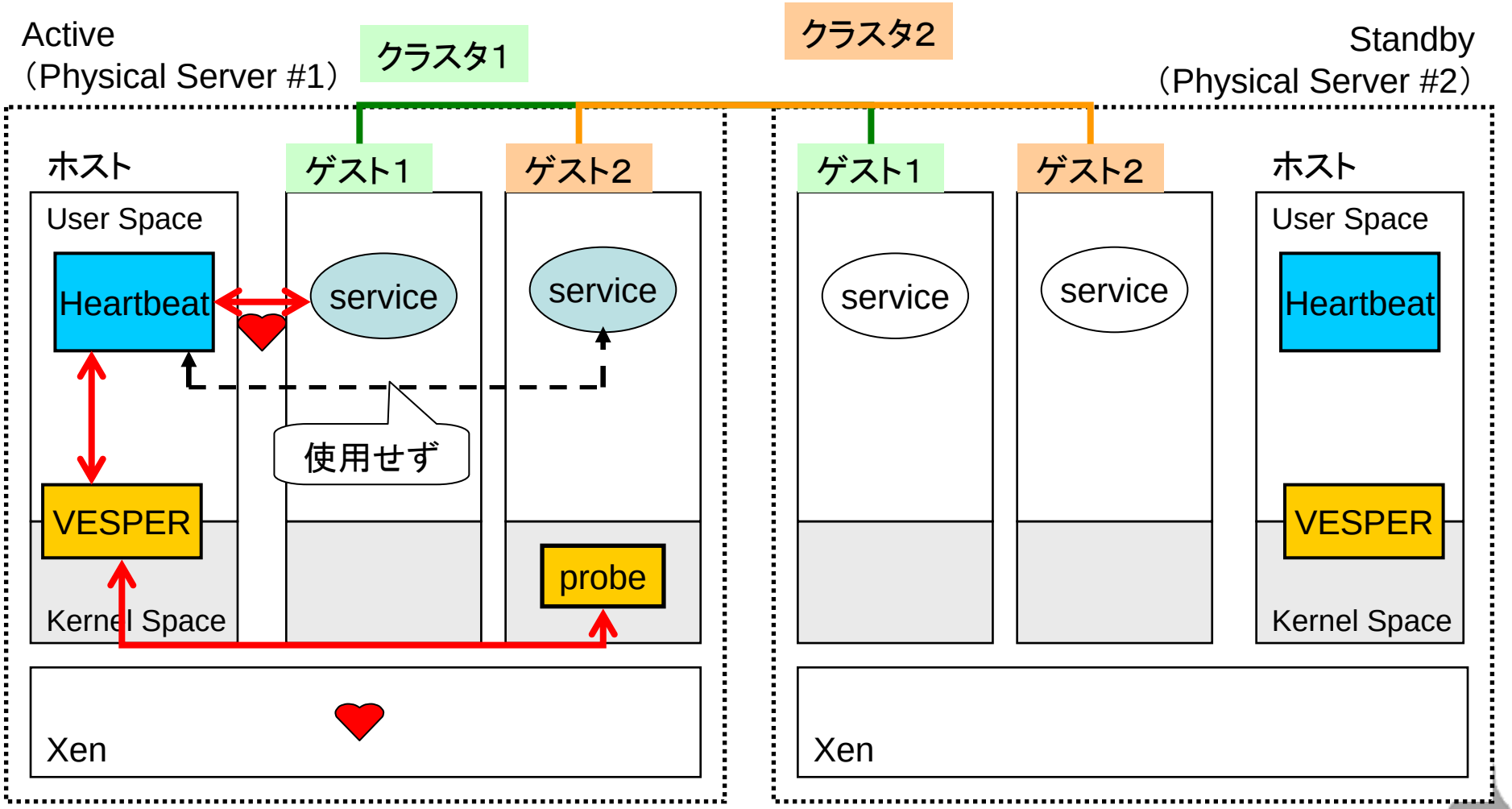
HITACHI
Inspire the Next

テストシステム – Webサービス

•物理サーバ:2台
-Active
-Standby

•Fedora 7
- xen-3.1.0.rc7.1.fc7
- linux-2.6.21-fc7xen

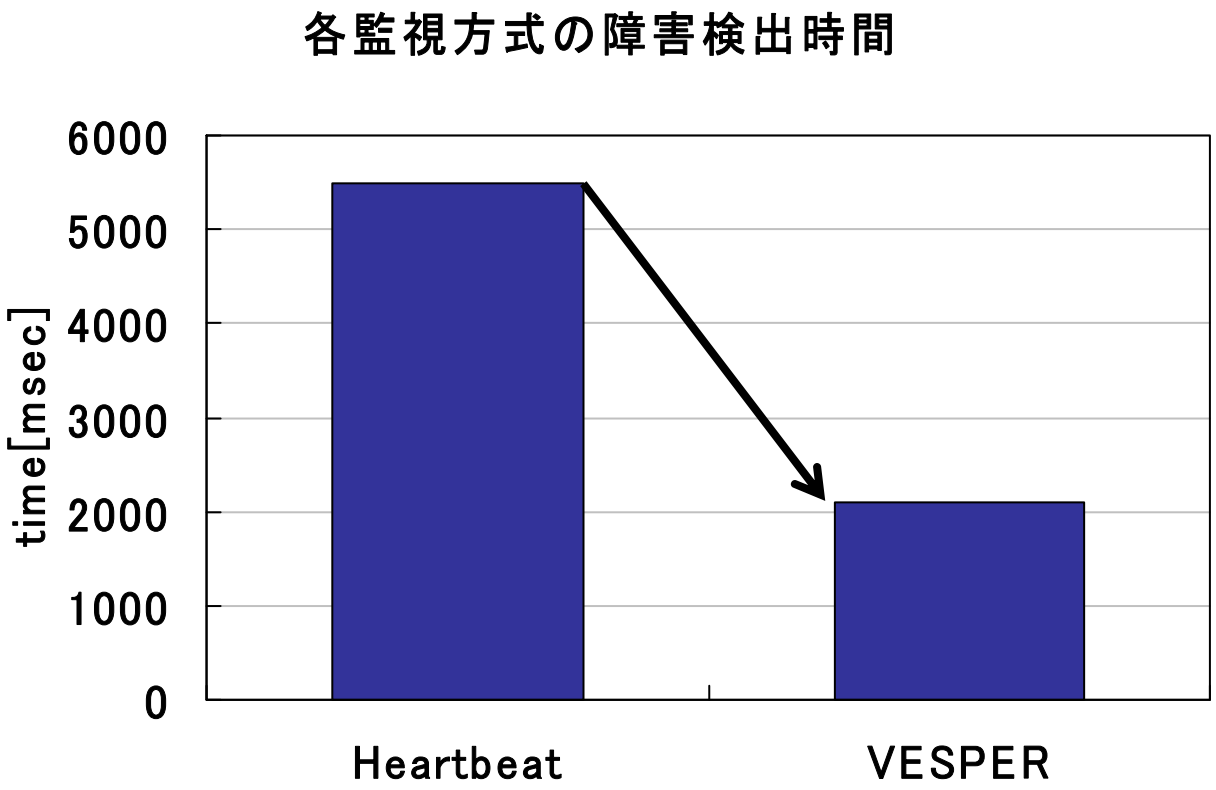
•準仮想化



テストケース1

- httpdに障害が発生し、SIGSEGVが発生
  - クラスタ1
    - Heartbeatにより監視
    - 生死監視間隔は10秒
  - クラスタ2
    - VESPERにより監視
    - send_signal()を監視
    - signal番号
    - 送信元プロセス

障害検知時間の比較

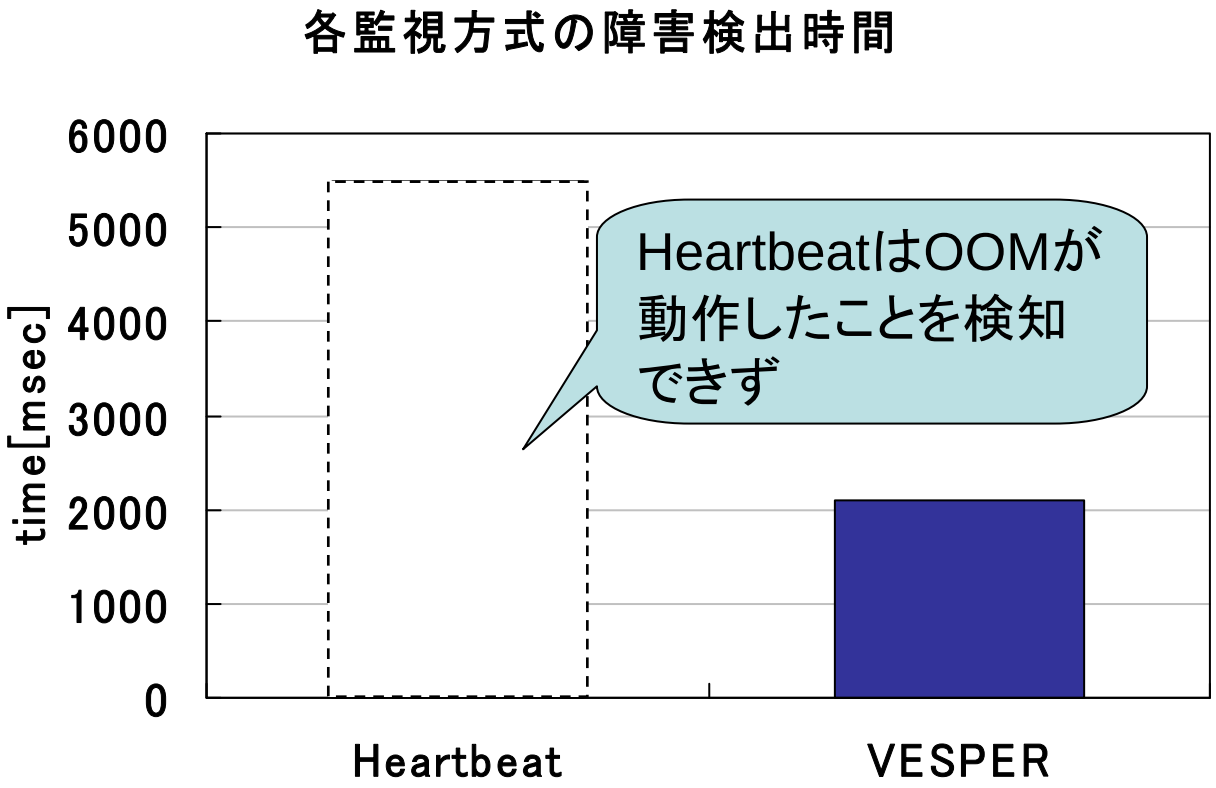


- 障害検知時間を約50%低減
- VESPERはシグナルの種類、送信元を取得

テストケース2

- メモリが不足しOOMが動作
  - クラスタ1
    - Heartbeatにより監視
    - 生死監視間隔は10秒
  - クラスタ2
    - VESPERにより監視
    - out_of_memory()を監視
    - OOMが発生したら障害と判断

障害検知時間の比較



- 場合によってはHeartbeatも検知可能
- VESPERは常に検知可能であり検出時間もHeartbeatの半分程度

5.

結論

HITACHI
Inspire the Next

結論

- VESPER
  - 仮想化環境におけるゲスト情報収集フレームワーク
  - プローブモジュール（kprobes+relayfs）を利用
  - プローブモジュールをホストからゲストへロード可能
  - ゲスト内で取得した情報にホストからアクセス可能
  - クラスタ管理ソフトウェアに対し、生死監視情報を提供
  - 障害検知時間の改善
  - 障害解析機能の改善

今後の予定

- KVMのサポート
- SystemTapのサポート
- Knowledgeエンジンの開発
  - プローブポイント選択時に補助
- クラスタ管理ソフトウェアとの連携
  - CIMHA（Cluster Installer Monitor & Health Analyser）*

*) http://sourceforge.net/projects/cimha/

Legal Statement

- Linuxは、日本およびその他の国における Linus Torvalds氏の登録商標です。
- その他の会社や製品名称に言及したものは、それぞれの所有者の商標です。

ご清聴ありがとうございました