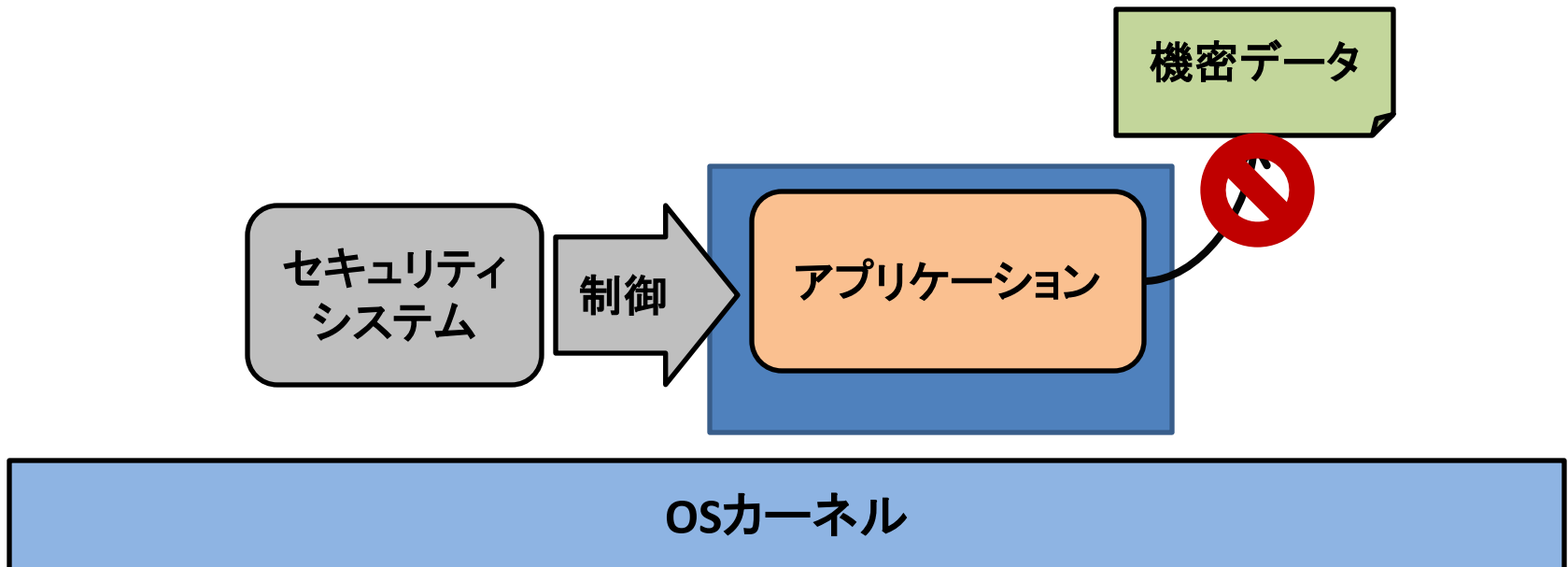


仮想マシンの外側から  
アプリケーションの安全性を  
向上させるセキュリティシステム

尾上 浩一  
東京大学

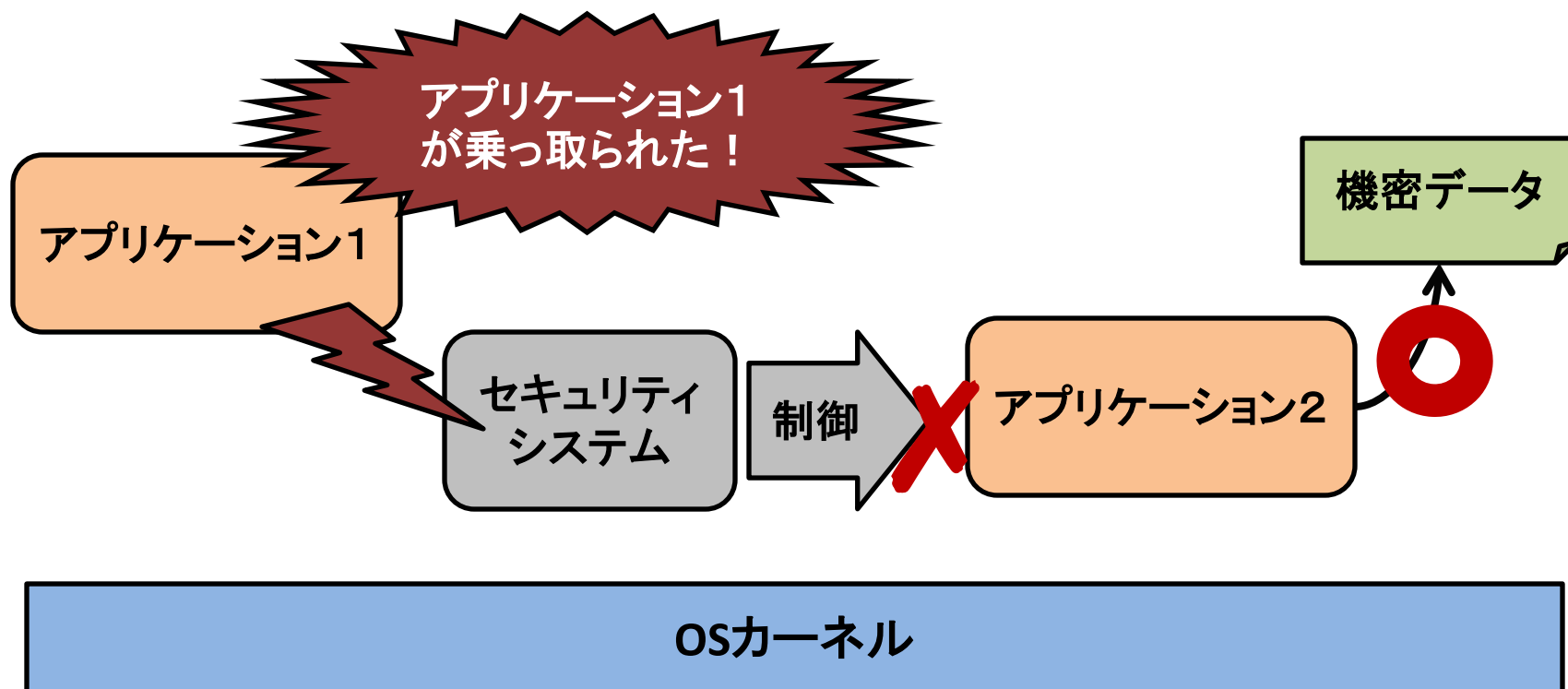
# アプリケーションの保護

- セキュリティシステムの適用が一般に普及
  - サンドボックスシステム
  - 侵入検知・防止システム (IDS・IPS)
  - アンチウイルスシステム



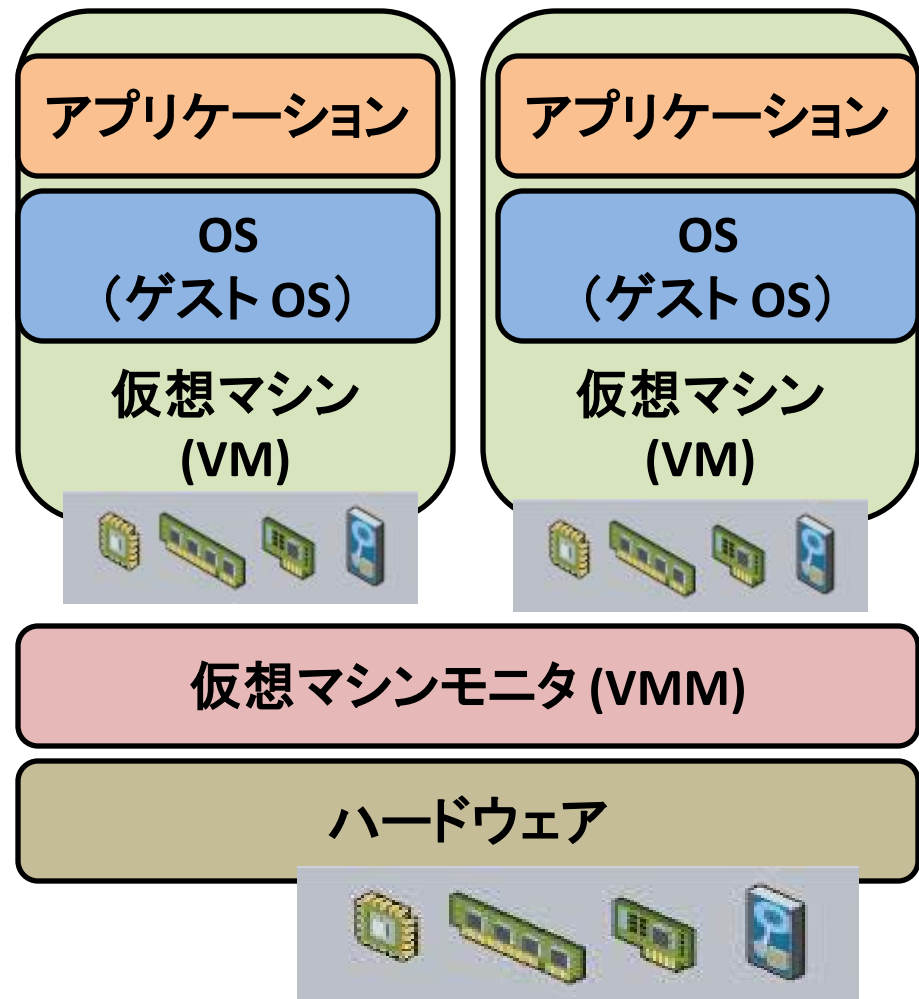
# セキュリティシステムも攻撃され得る！

- 他のアプリケーションと同じ実行空間で稼働



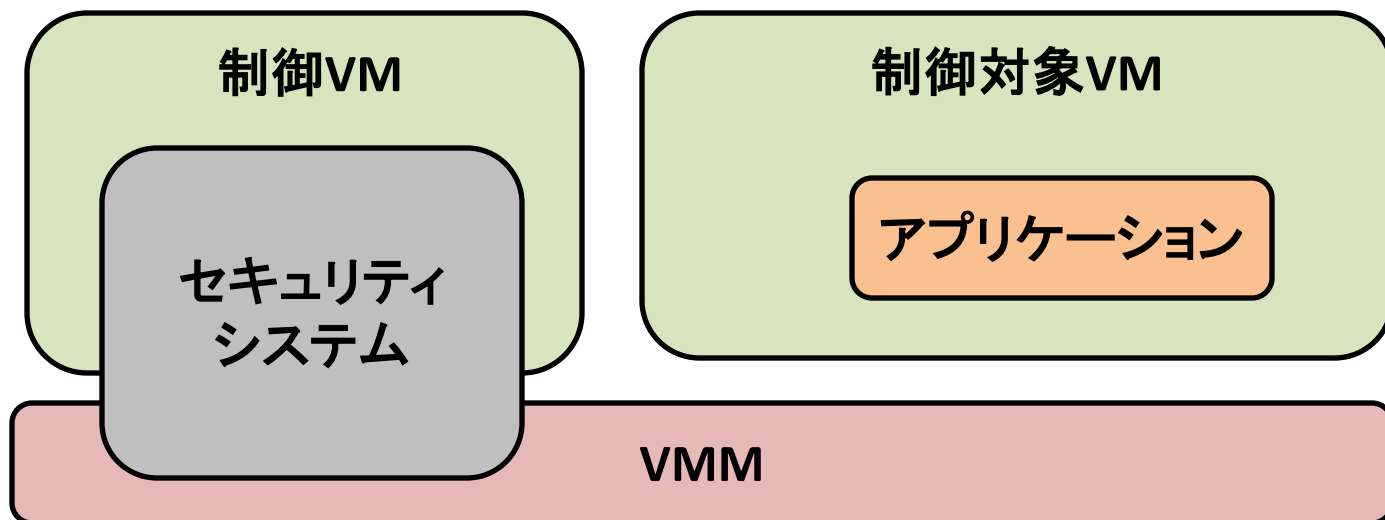
# 仮想マシンモニタ (VMM) を 利用することが効果的

- VM単位の隔離
  - たとえあるVMが奪取されても、VMMや他のVMを奪取することは困難
- VMの物理メモリやディスクなどの物理計算資源への操作を制御
  - VMMはVMよりも高い特権レベルで稼働



# 本研究の目標

- VMの外側からアプリケーションの安全性を向上させたい
  - VMMとセキュリティシステムを協調させ、アプリケーションの振る舞い制御とアプリケーションに関連するデータの保護を実現したい



# 本研究のアプローチ

- 制御対象VMの外側からアプリケーションが発行したシステムコールの実行を制御
    - アプリケーションプロセス単位で実行を制御
  - VMMと制御VMが、アプリケーションに関連するメモリ・ファイル操作を制御
    - メモリ上、仮想ディスク上のアプリケーションに関連するデータの漏洩・不正改竄の防止
- ✓ 利用者が指定したアプリケーションのみ実行を制御
- 準仮想化を利用したXenを用いて提案システムを構築

# 提案するシステムの実装

- VMM: Xen 3.0.3
  - 準仮想化技術 (Para-virtualization) の利用
- ゲストOS: Linux 2.6.16.19
- SMP環境に対応
  
- 設計方針
  - 制御対象VMに意識させない
  - 制御するアプリケーションを変更しない

# 制御対象VMの外側からの システムコールの実行制御

IA-32, AMD64に対応

# セキュリティシステムの運用比較

	VMを使わない 場合	VMの外側で 稼働させる場合
セキュリティシステム への攻撃	X 容易	○ 困難
セキュリティシステム の制御単位	○ OSレベル	X ハードウェアレベル

# システムコールの実行制御の目標

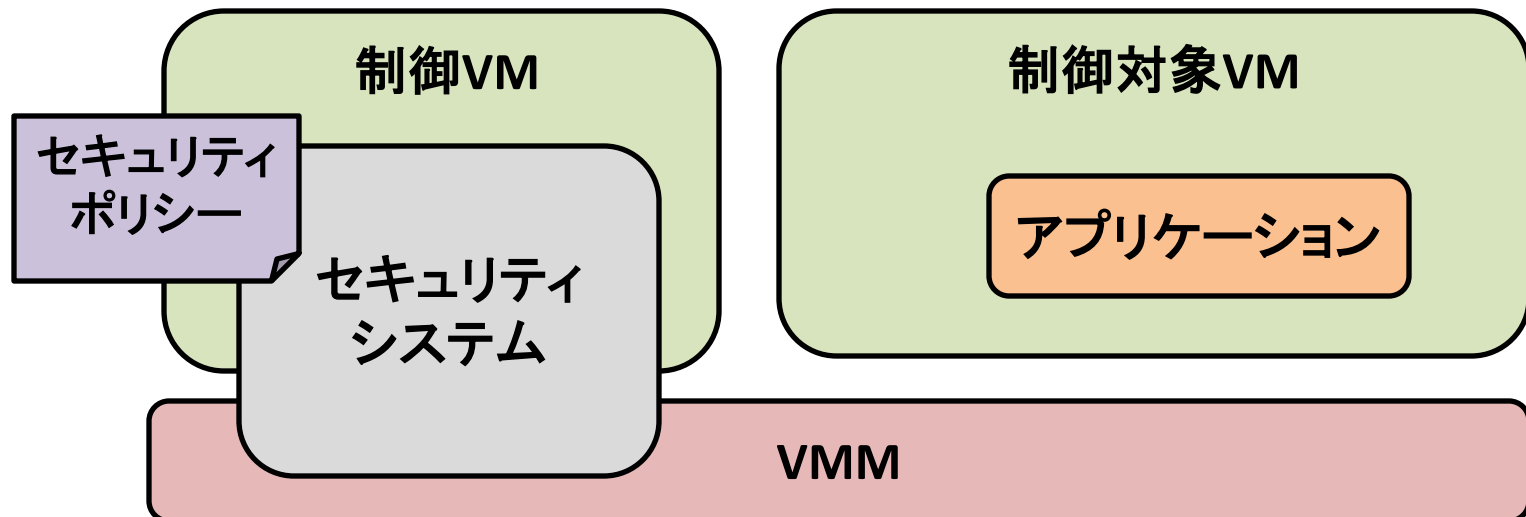
	VMを使わない 場合	VMの外側で 稼働させる場合
セキュリティシステム への攻撃	X 容易	○ 困難
セキュリティシステム の制御単位	○ OSレベル	X ハードウェアレベル

Semantic gap

本研究の目標

# 本研究のアプローチ

- VMの外側からシステムコールの実行を制御
  - ゲスト OS カーネルの情報を利用
- セキュリティポリシーに基づいた制御



# システムコールの実行を 制御するためには？

- セキュリティシステムが利用するアプリケーションの実行状態を取得することが必要
  - どのプロセスがイベントを発生させたか
  - 発行されたシステムコールによって何が実行されるのか
- システムコール実行の捕捉が必要

ゲストOSカーネルの情報を利用

# アプリケーションの実行状態の取得

- VMMが捕捉時に取得できるイベント・実行状態
  - イベント : 特権命令、割り込みなど
  - 実行状態 : レジスタ、メモリ上の値



- セキュリティシステムが必要とする情報
  - イベント : システムコール
  - 実行状態 : プロセス、システムコール番号など

# ゲストOSのプロセスに関する情報

- 「プロセス管理データのメモリアウト」があればよい
  - (例) タスク構造体中のプロセスIDのオフセットとそのサイズ
  - VMMからVM内のメモリを操作することは可能

# ゲストOSのシステムコールに関する情報

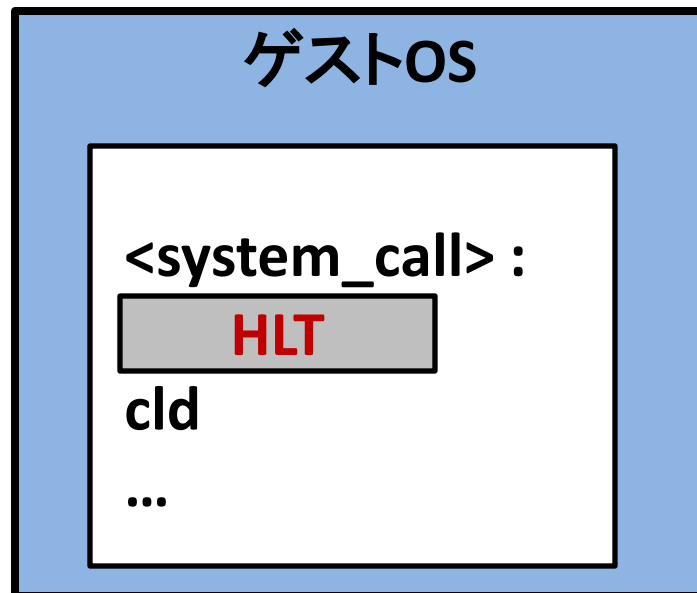
- システムコール名と番号の関係
  - (例) openシステムコールは5
- ポインタ引数に関する情報
  - (例) openの第1引数はファイル名
- システムコールの呼び出し規約
  - 引数、返り値の保存場所と保存方法
    - システムコール手続き中のスタック構造
      - レジスタの保存場所など

# システムコール実行の捕捉

- VMMが捕捉できないが、  
制御機構が必要とするアプリケーションの  
実行命令が存在
  - 最適化により、システムコール呼び出しが  
VMMを経由しない実行命令
  - システムコール終了時の実行命令
- ゲストOSカーネルのバイナリ書き換えを導入し、  
システムコール実行の開始・終了時を捕捉する

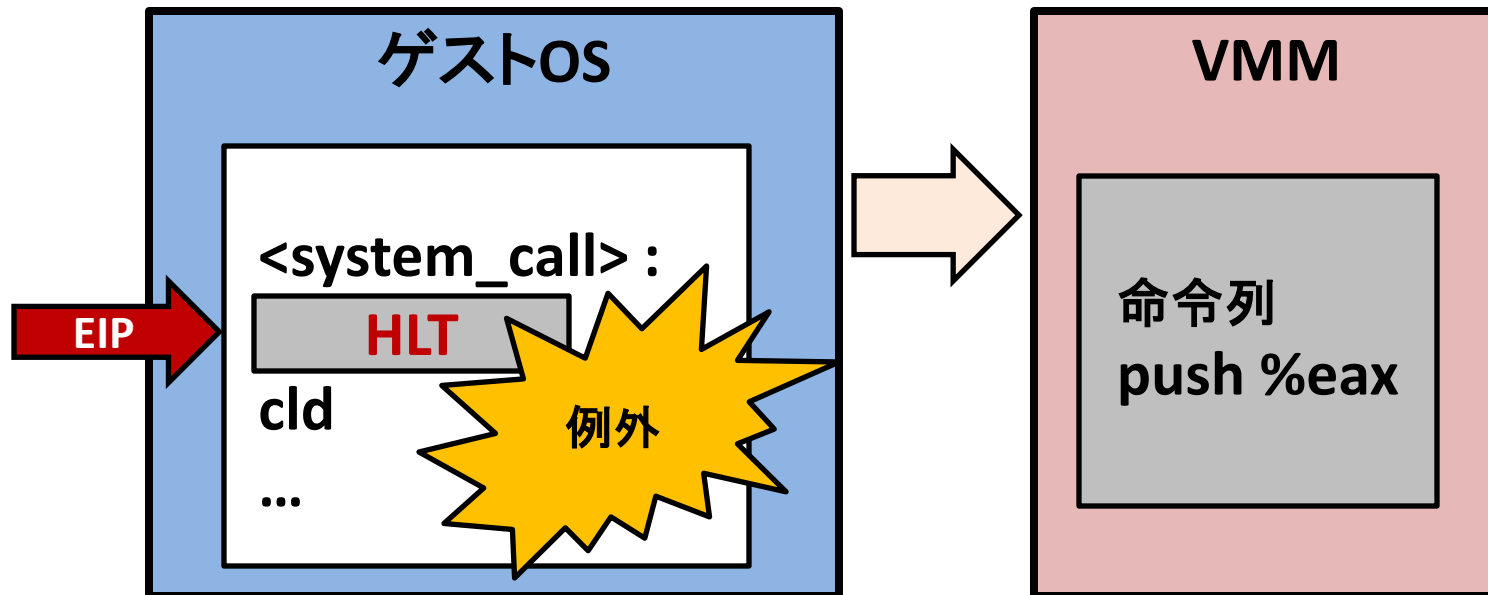
# ゲストOSカーネルのバイナリ書き換え (設定時)

- VMM が捕捉可能な特権命令で書換
- 利用者から捕捉する実行命令の  
仮想アドレスを提供してもらうことが必要
  - OSカーネルのシンボル情報を利用



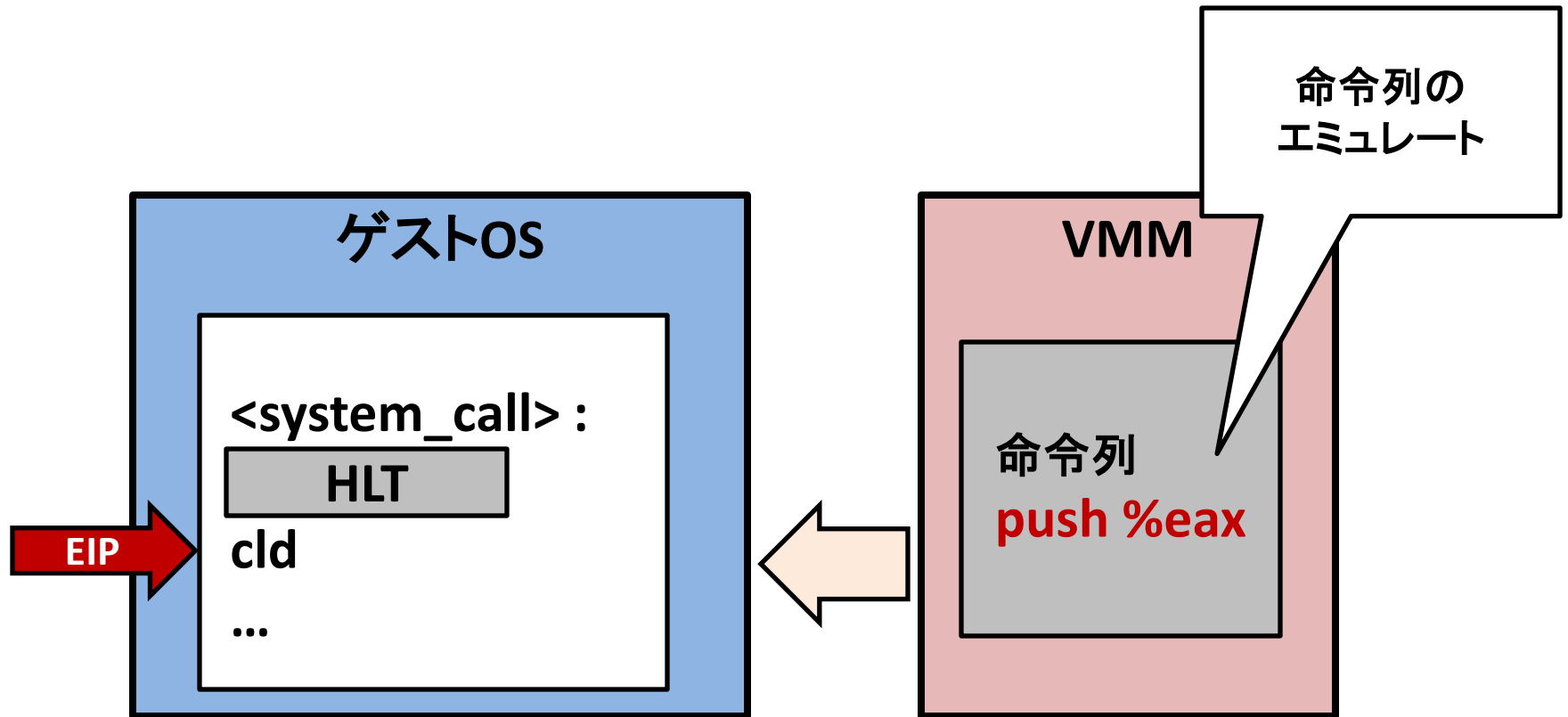
# ゲストOSカーネルのバイナリ書き換え (実行時)(1/2)

- システムコール操作時に例外が発生



# ゲストOSカーネルのバイナリ書き換え (実行時)(2/2)

- 書換前の命令をエミュレート

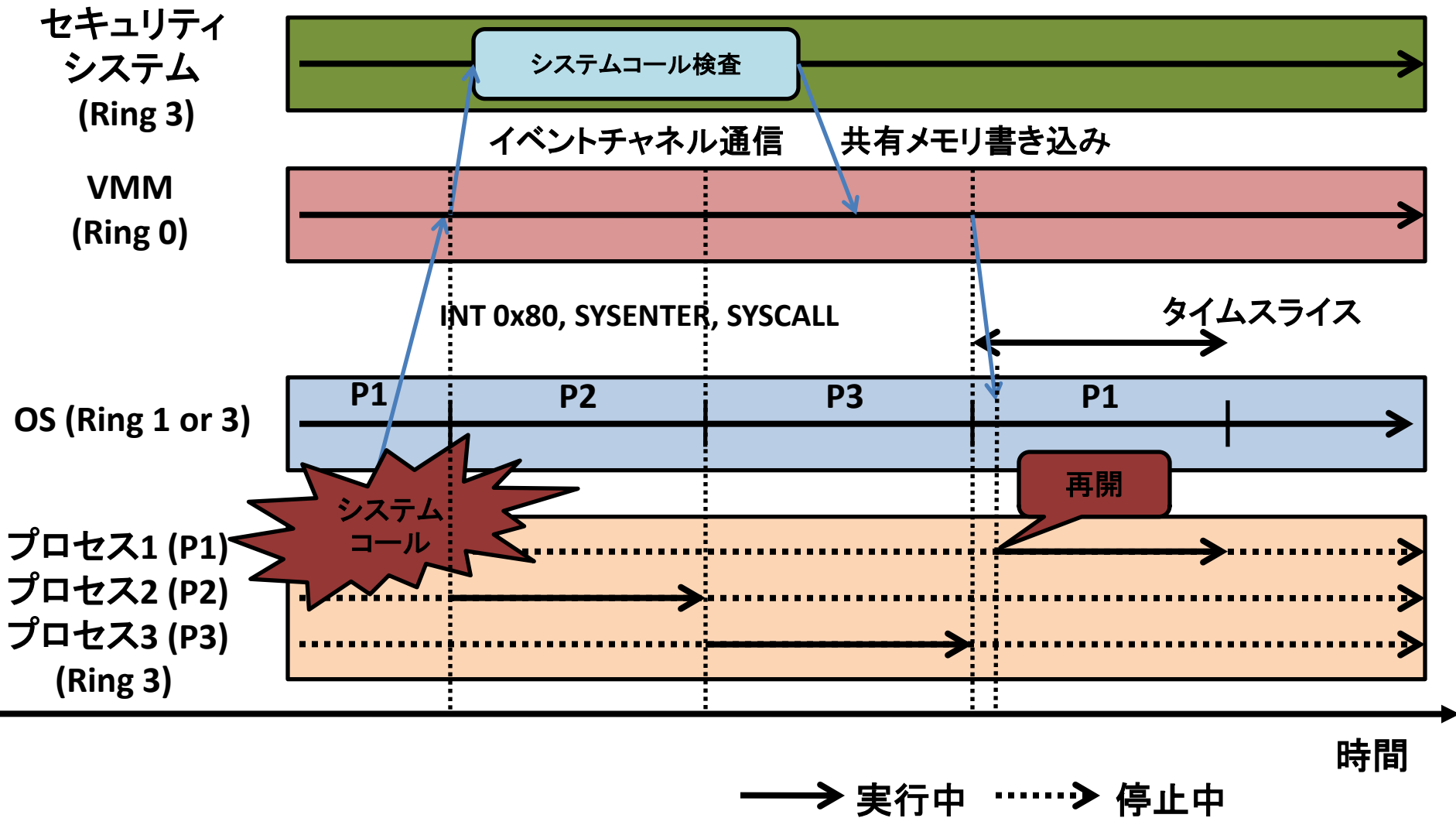


# 利用者から提供してもらう情報

- プロセス関連のデータ構造
  - プロセスのメモリレイアウトなど
- システムコールに関連した情報
  - システムコール番号と名前の対応など
- システムコール実行と開始・終了に関連した仮想アドレス

※プロセスのメモリレイアウトやシステムコールのポインタ引数のサイズに関する情報はカーネルコンパイル時に自動生成する仕組みを提供

# システムコール手続きの 捕捉から再開までの流れ



# システムコールの実行制御のための セキュリティポリシー

- システムコール名と引数情報を用いたパターンマッチ

...

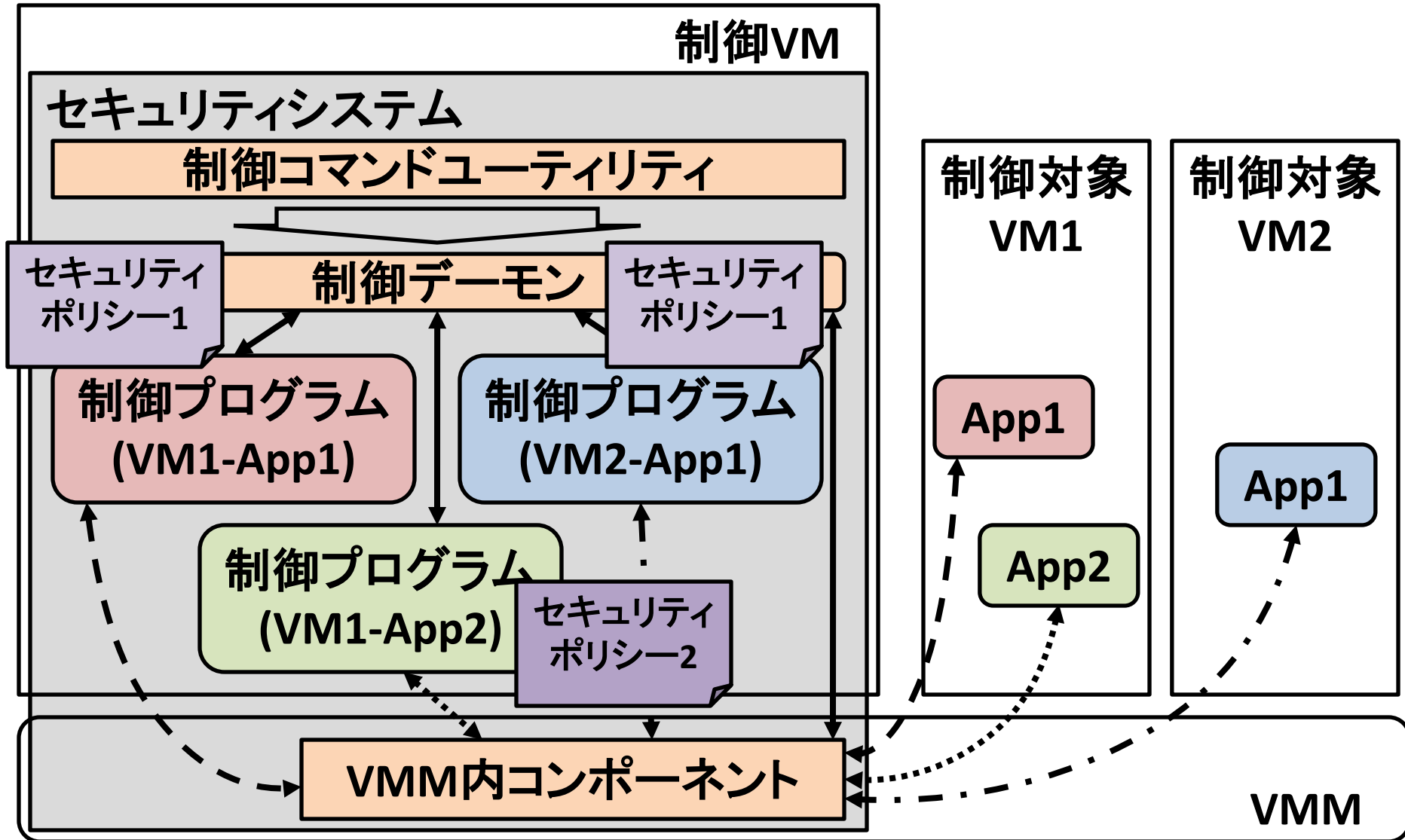
```
open default: allow  
  fileEq("/etc/passwd")  
  or filePrefixEq("/etc/cron.d")  
  deny(EPERM)
```

...

# 提案するシステムの特長

- セキュリティシステムへの攻撃が困難
- プロセス単位の実行制御が可能
- 複数VM内の同一アプリケーションに対して  
統一的な制御が可能
- セキュリティシステムが  
各VMの安全性向上を支援

# 提案するシステムの構成



# 関連研究

- Livewire[Garfinkel et al., 2003]
  - OS情報を用いて、VMの外側でIDSを構築するためのアーキテクチャを提案
  - VMMが捕捉できないイベント制御を行うことができない
  - 捕捉後の制御方法がVM単位である
- Vmwatcher[Jiang et al., 2007]
  - OS情報を用いて、周期的にVMの実行状態を検査するアンチマルウェアシステム
  - 本研究はイベント(システムコール実行)に同期した制御
  - 本研究ではセキュリティポリシーに基づいて実行を制御
- Honeypot として利用する VM 内部の情報収集 [Asrigo et al., 2006]
  - ファイルやソケット操作を VMM が捕捉
  - VMM 内ですべての監視処理を行う
- Licosid[Jones et al., 2008]
  - VMMが取得可能なハードウェアの実行状態のみを利用して、隠蔽されたプロセスの検出・識別を行うシステム
  - プロセス、ファイルなどのOSレベルの実行状態を正確に取得できない

# アプリケーションに関連する メモリ・ファイル操作の制御

AMD64に対応

# アプリケーションに関連する データの保護(1/2)

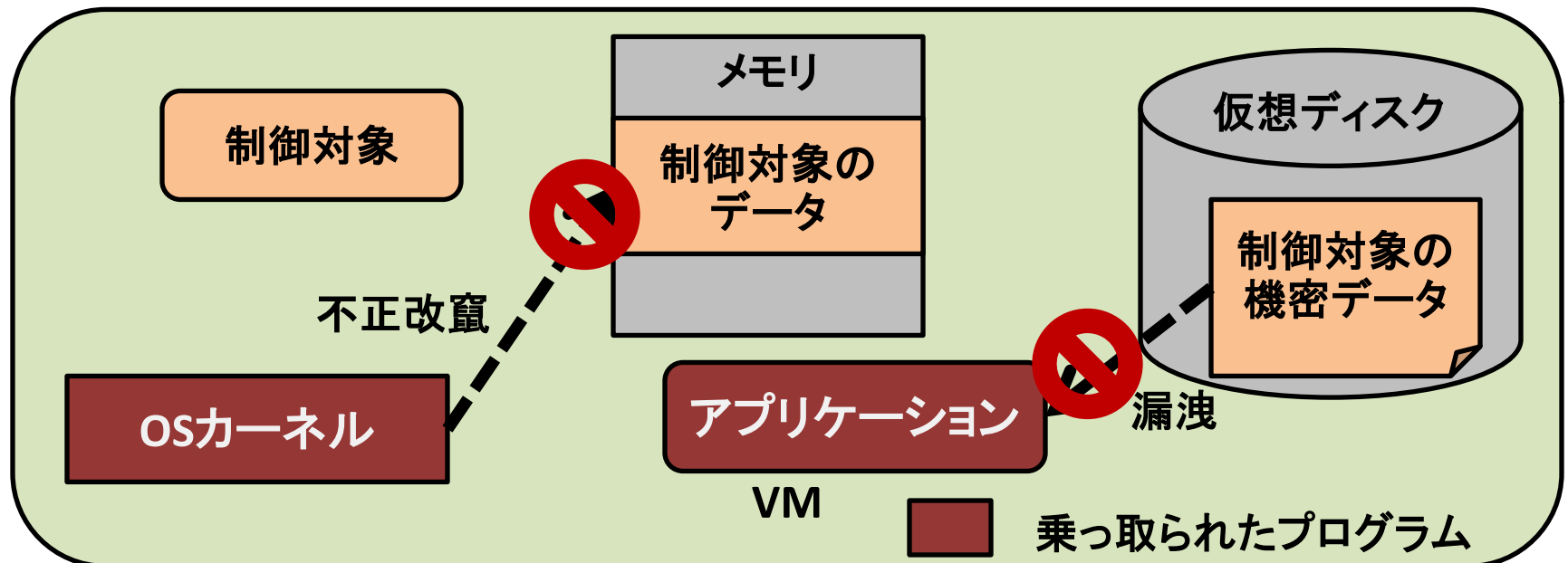
- **制御対象VM内でアプリケーションを稼働させながら、データを保護したい**
  - アプリケーションが依存する共有ライブラリなどのファイルは制御対象VMごとに固有であることを想定
- **攻撃者の観点から、アプリケーションが正常に稼働しているように制御対象VMの利用者に見えることも重要**
  - アプリケーションが乗っ取られていることがわかってしまえば、利用者に対処されてしまう
  - Mimicry attack [Wagner et al., 2002]

# アプリケーションに関連する データの保護(2/2)

- 機密データの漏洩やアプリケーションのコード領域を含むデータの不正改竄だけではなく、  
コード領域を含むデータ漏洩を防ぐことも  
有用な場合がある
  - アプリケーションの振る舞いを解析することで  
攻撃が容易になる可能性がある
  - 政府や企業などで固有のアプリケーション開発へ  
応用

# アプリケーションのデータ保護の目標

- 制御対象のアプリケーション(制御対象)のデータの漏洩・不正改竄の防止
  - ptraceやカーネルモジュールなどを利用した攻撃
  - シンボリックリンクを利用した攻撃
  - レースコンディションを利用した攻撃



# 本研究のアプローチ

- 制御対象のメモリ・ディスク上の実体を制御対象外のプログラム（制御対象外）から隠蔽
  - OSカーネルも制御対象外に含まれる
- メモリ上のデータ
  - コード、データ、スタック領域など
  - 制御対象の物理メモリ領域を多重化
    - Overshadow[Chen et al., 2008]
    - [Rosenblum et al., 2008]
- ディスク上のアプリケーション固有のデータ
  - 実行ファイル、設定ファイルなど
  - 異なるVMで管理
- メモリ・ディスク上のデータ保護はセキュリティポリシーで指定

# メモリ・ファイル操作制御のための セキュリティポリシー(1/2)

**Executable: *testAOutVM***

**MemoryProt : *all***

**FileProtection:**

***configInVM -> configOutVM***

***DBInVM -> DBOutVM***

**...**

# メモリ・ファイル操作制御のための セキュリティポリシー(2/2)

**Executable: *testAOutVM***

**MemoryProtection : *partially***

**Segments:**

***text, data, brk, stack(Down,1M),  
env, arg***

***MmappedRegion: yes***

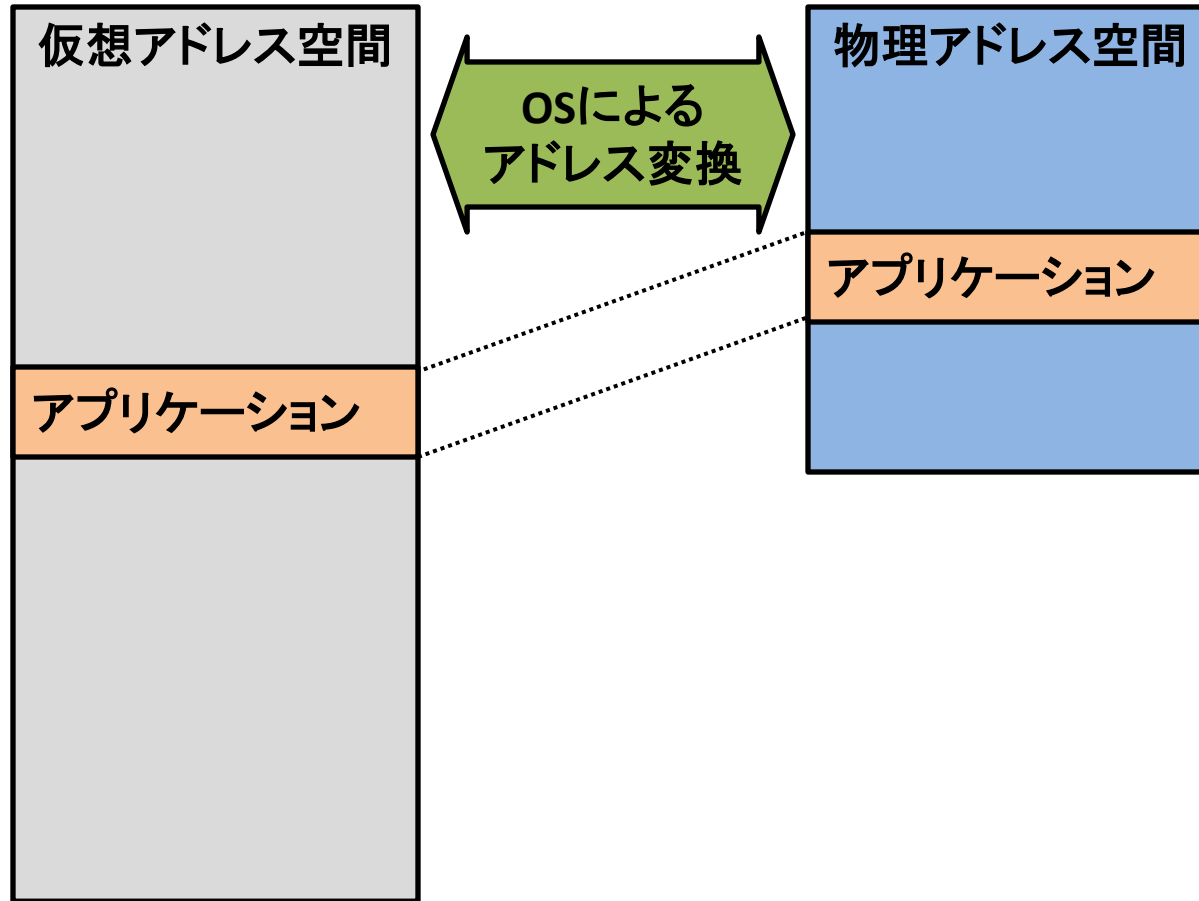
**FileProtection:**

***configInVM -> configOutVM***

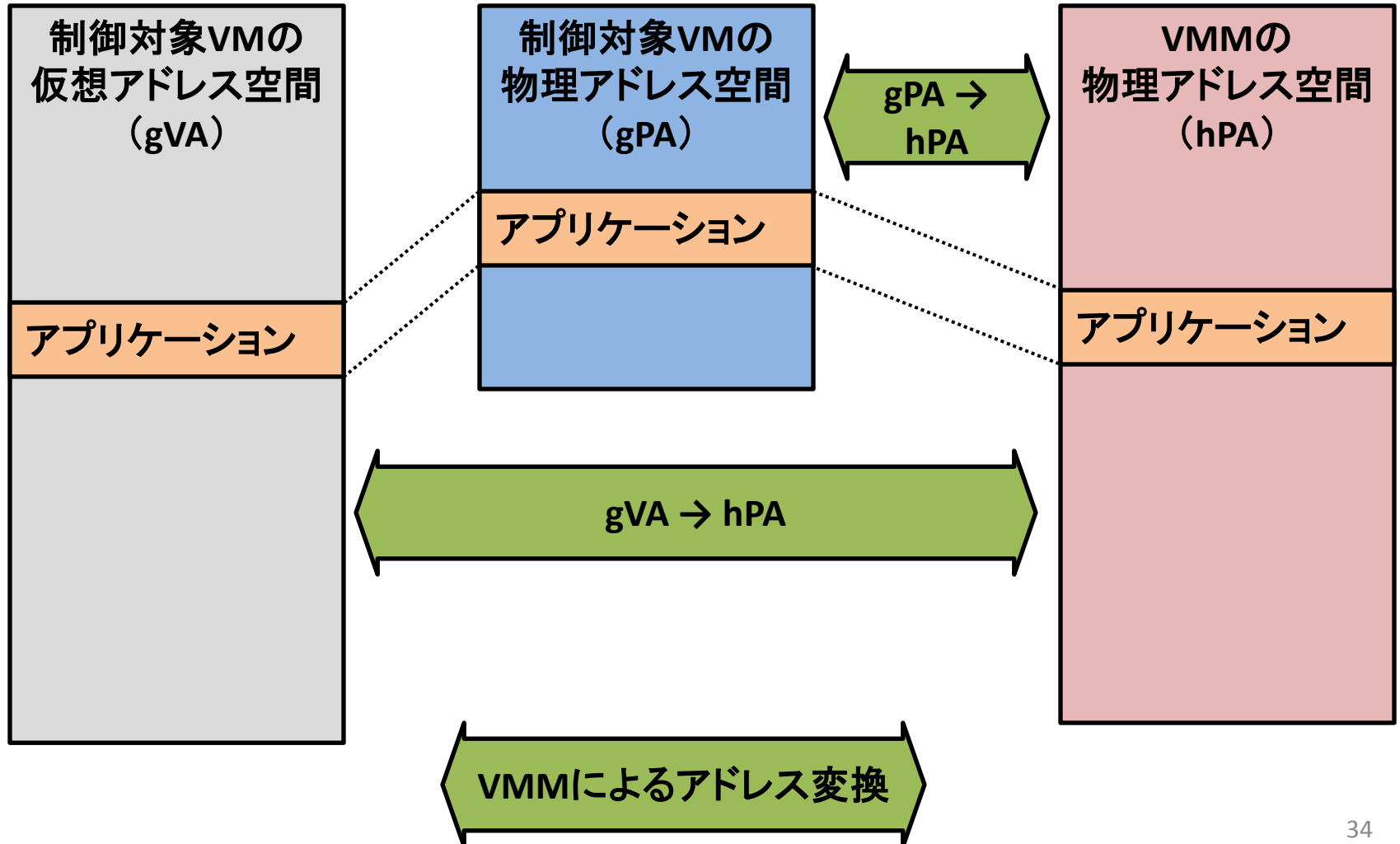
***DBInVM -> DBOutVM***

**...**

# OSによるメモリ管理

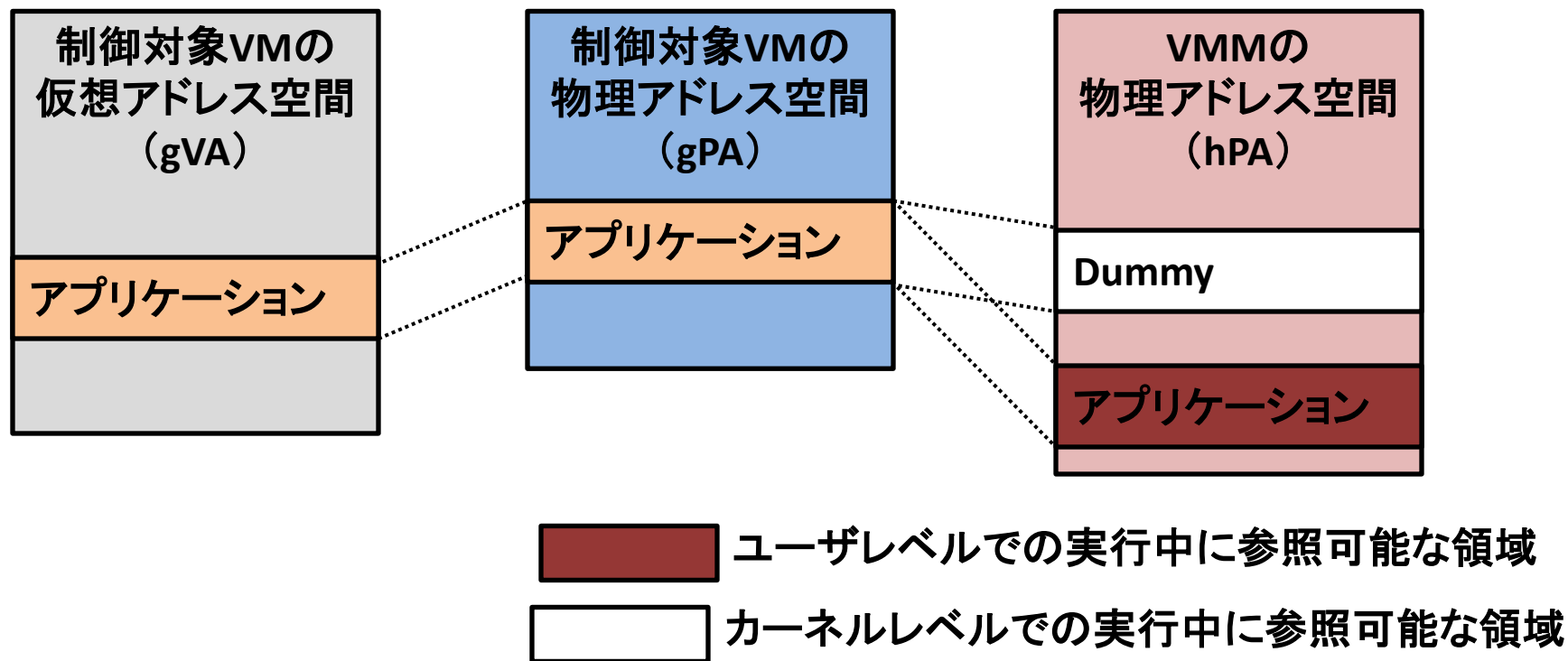


# VMMによるメモリ管理



# 本研究におけるメモリ上のデータ保護 (1/3)

- カーネルレベル・ユーザレベルで異なる物理アドレス領域を見せる



# 本研究におけるメモリ上のデータ保護 (2/3)

- 制御対象アプリケーションに関する  
カーネルレベル・ユーザレベル間の切り換え  
が発生したとき、ページテーブルを切り換える
  - 例外・割り込み処理
  - システムコール処理

# 本研究におけるメモリ上のデータ保護 (3/3)

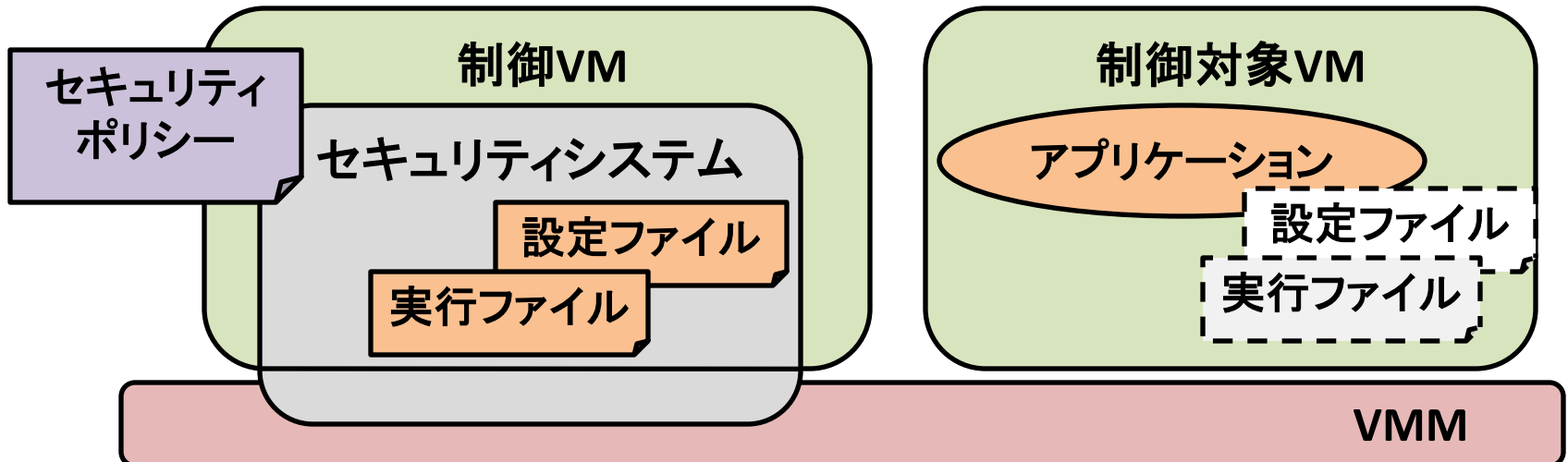
- 保護すべきメモリ領域をVMMが管理する
  - 保護すべき仮想アドレス領域
    - プロセスごと
  - ページテーブル・物理ページ
    - 制御対象VMごと
  - ページフォルト、プロセス生成などのイベント発生時に適宜これらの情報を更新

# ユーザ空間へのポインタ

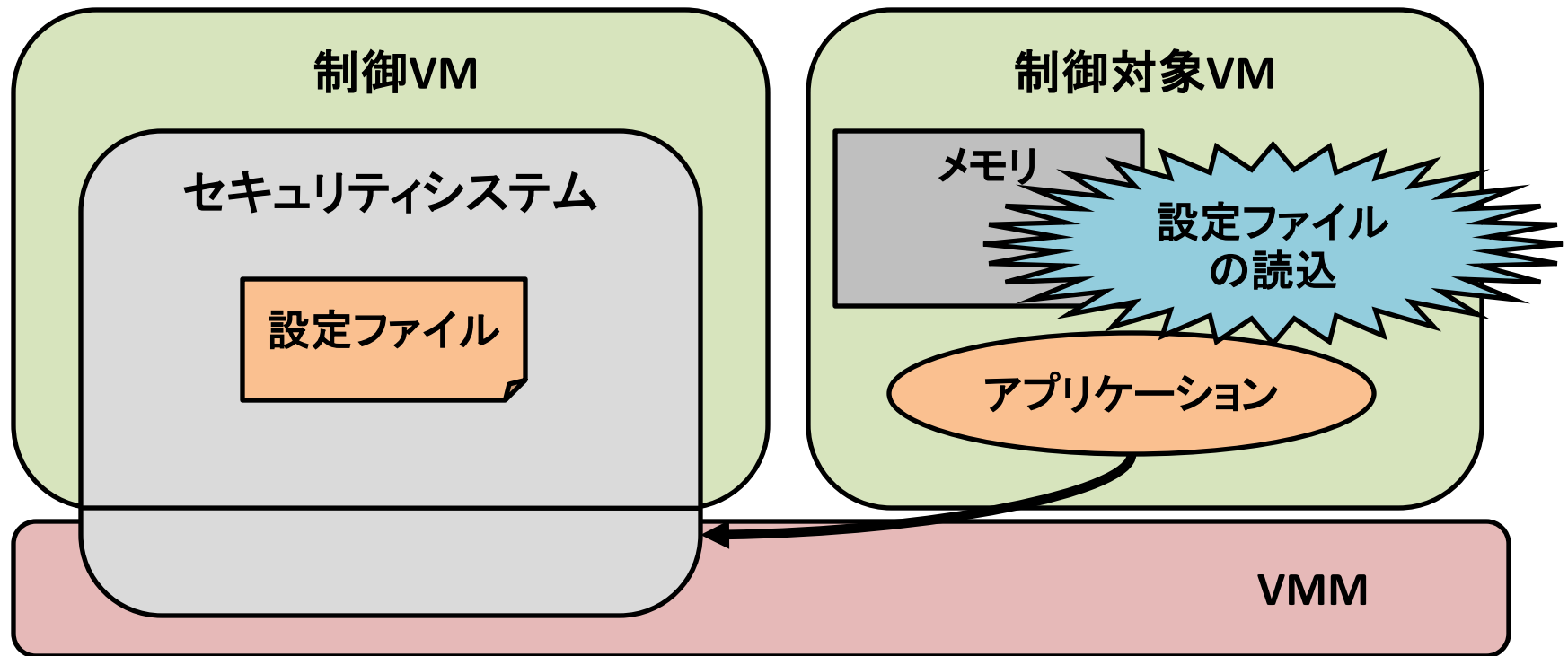
- OSカーネルがユーザ空間の操作  
(参照・書込)を行う
  - システムコール引数
  - シグナル処理
- カーネルが操作するときだけ、一時的に元のユーザ空間に存在するようにする

# 本研究におけるディスク上の データ保護(1/5)

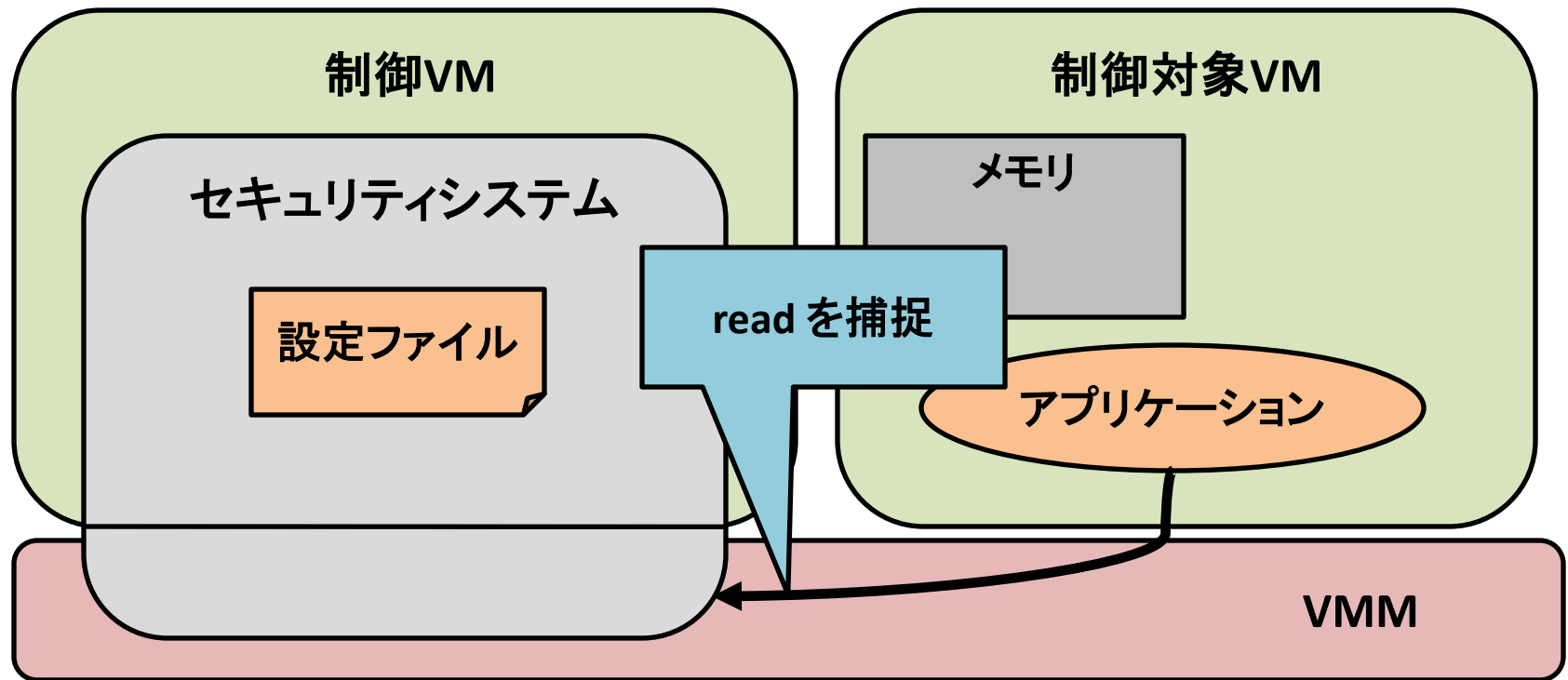
- 異なるVMで制御対象ファイルの実体を管理
  - 実行ファイル、設定ファイル、データベースファイルなど
  - 制御対象はセキュリティポリシーで指定



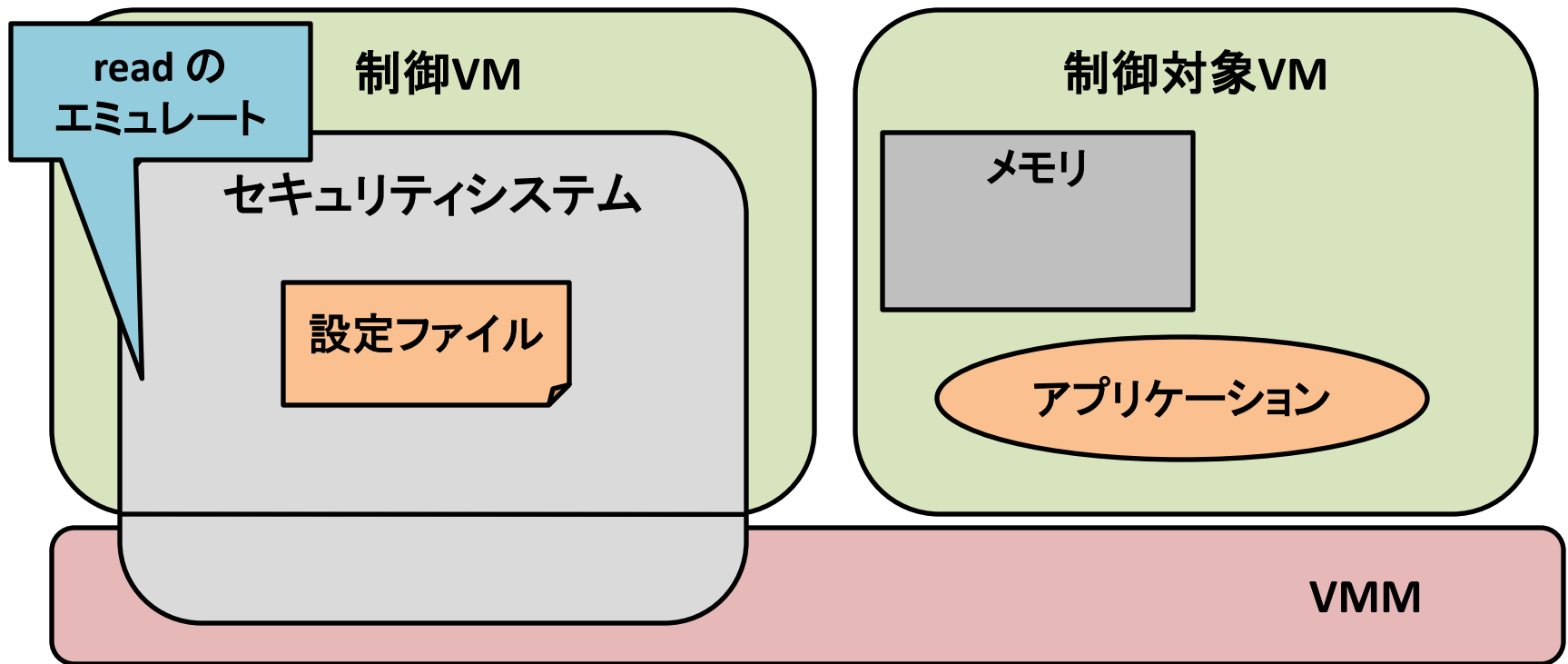
# 本研究におけるディスク上の データ保護(2/5)



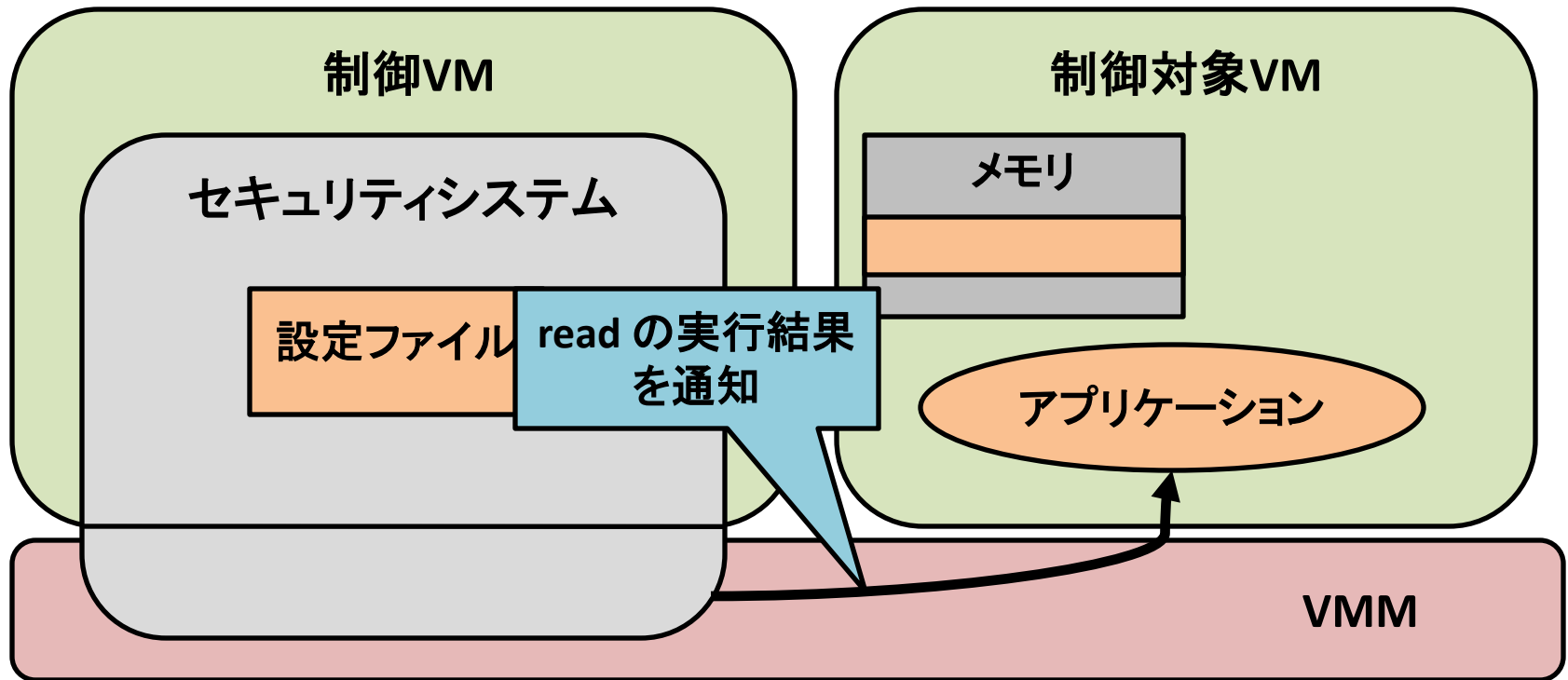
# 本研究におけるディスク上の データ保護(3/5)



# 本研究におけるディスク上の データ保護(4/5)



# 本研究におけるディスク上の データ保護(5/5)



# 制御対象VM内に存在する 実行ファイル

- 一部の実行ファイルに関する情報が必要
  - ELF Header
  - Program Header
  - “.interp” section
  - 他の部分はzeroで埋める
- 提案システムでは以上の操作を行うプログラムを提供

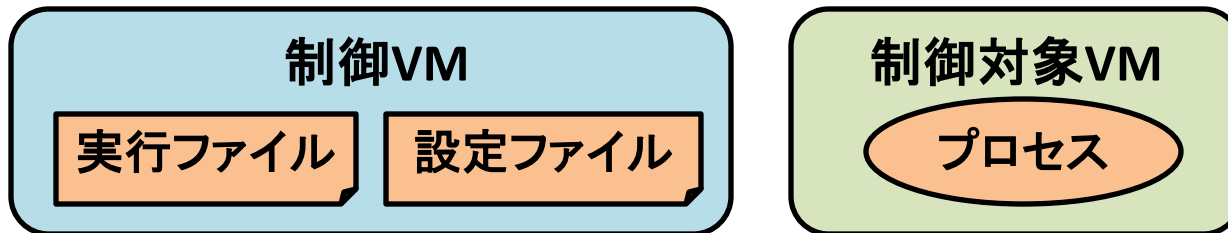
# 関連研究(1/2)

- アプリケーションに関連したデータ保護
- Overshadow[Chen et al., 2008]
  - 物理アドレスの複製と暗号化により、メモリ・ディスク上のデータを保護
  - SMPやマルチコアには未対応
  - メモリ・ディスク上のファイルは改竄可能
  - 専用のユーザレベルプログラムが必要
- Proxos[Ta-Min et al., 2006]
  - 保護対象とするアプリケーションを異なるVMで稼働
  - アプリケーションを修正する必要がある

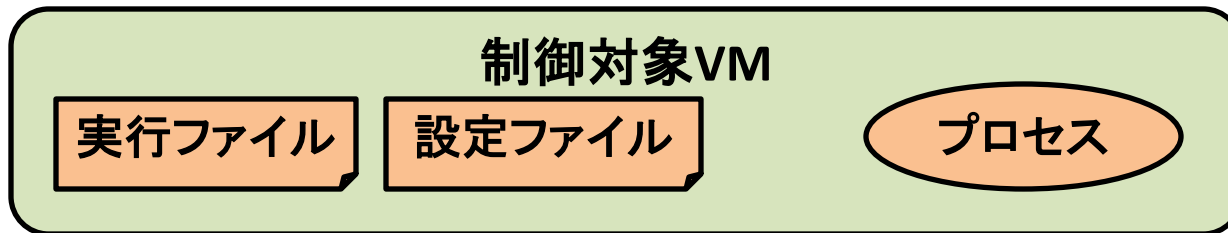
# 関連研究(2/2)

アプリケーション

- 提案するシステム



- Overshadow



- Proxos



# まとめ

- VMの外側からアプリケーションを保護するセキュリティシステムの提案
  - アプリケーションの振る舞いの制御
    - システムコールの実行制御
  - メモリ・ディスク上のデータの保護
    - メモリ上のデータ: VMMによる物理メモリ領域の多重化
    - ディスク上のデータ: 異なるVMで管理

終