

# TwinOS におけるヘルスチェック 機能の実現と評価

岡山大学 工学部 情報工学科


長田 一帆

# 研究背景

インターネットの普及に伴い、計算機に対する攻撃が増加

## <既存手法の問題点>

セキュリティシステムの多くは攻撃対象となるOS上に実装

 OS自体が乗っ取られた場合、信頼性が低下

## <対策>

攻撃の対象となるOSの外部から監視

 TwinOSを用いたヘルスチェック機能を提案

## <既存のヘルスチェック機能>

- TwinOSを用いたヘルスチェック機能の設計
- ヘルスチェックを支援するTwinOSの実現



TwinOSにおけるヘルスチェック機能の実現と性能評価

# TwinOS

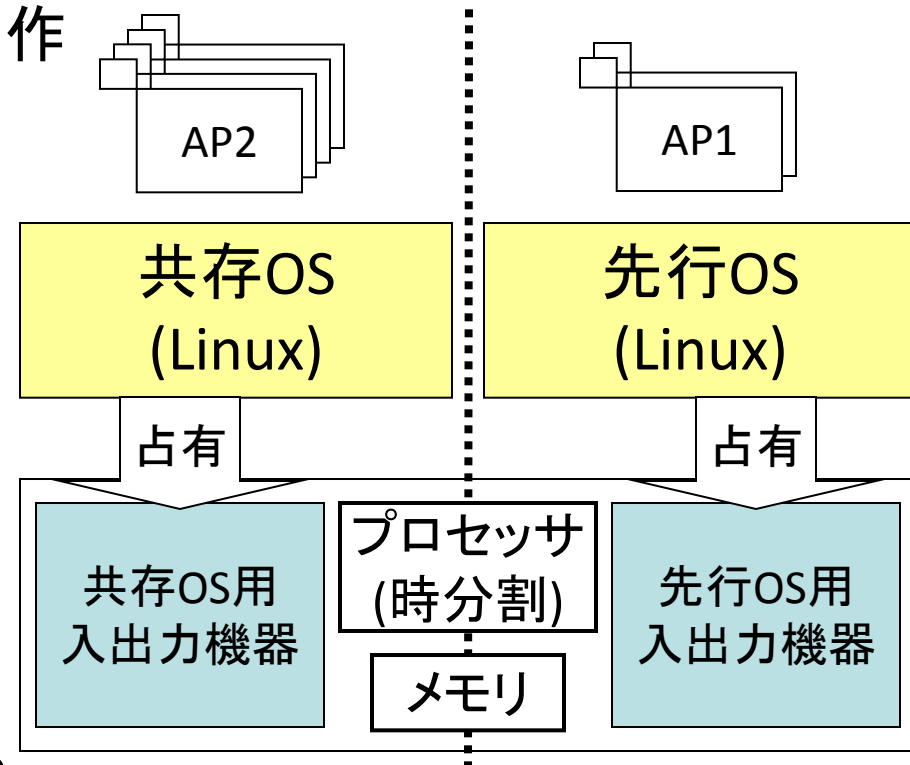
2つのLinuxを1台の計算機上で動作させる方式

## <構成>

- (1) 1台の計算機上に2つのOSが動作
- (2) ハードウェアの分割方法
  - (A) メモリ: 空間分割
  - (B) CPU: 時分割(タイマ割込み)
  - (C) 入出力機器: OSごとに占有

## <特徴>

- (1) 相互に処理負荷の影響を受けない
- (2) 入出力性能を十分に利用できる



先行OSにヘルスチェック機能を実装し共存OSを監視

# ヘルスチェック機能

OS を検査し異常を検出するための機能

大きく分けて二つの処理を実行

## <正常データ作成処理>

- (1) カーネルテキスト部のハッシュ値を算出
- (2) 算出したハッシュ値を**正常データ**として保存

## <検査処理>

- (1) 正常データと同様に算出したハッシュ値を保存
- (2) 正常データと(1)(**検査データ**)を比較する
- (3) 比較した結果、値が異なれば**OSの異常**とする

# 課題

## <従来の検査範囲>

- (1) カーネルテキスト部
- (2) システムコールテーブル

 **LKM**(Loadable Kernel Module) の対応が不十分

## <課題>


(課題1) 不正なLKMのロードへの対応

不正なLKMのロードによる**OSの改ざん**が発生

 LKMに対応した検査範囲の追加が必要

(課題2) LKMのロードによるカーネルテキスト部の変更

LKMをロードする際、新しくページテーブルが  
作成されるとカーネルテキスト部が**変化**

 OSの異常と検知されないように対応が必要

# 課題への対処

## <対処>

(対処1) LKMを検査する機能(正常LKM検査機能)を追加

(対処2) LKMのロード完了後にカーネルテキスト部を再検査

## <正常LKM検査機能の設計>

(1) 共存OSのLKMロードの完了を先行OSが認識できる

LKMをロードすると特定のページテーブルが変化

このページテーブルを取得し、監視することで対処

(2) 検査対象となるLKMのテキスト部を取得できる

ページテーブルからLKM展開先アドレスを算出

物理メモリから取得したLKMのデータのテキスト部を検査

これらを実装することでLKMに対処した

# 評価

## <環境>

TwinOS(Linux kernel 2.4)

CPU : Pentium4 3.0GHz

## <目的>

(1) LKMに対応しているかを調査

ロードされたLKMごとに検査を実行できることを確認

(2) 検査方法の違いによるオーバヘッドの調査

(A) バイナリ検査とハッシュ検査のオーバヘッド

検査データにバイナリデータとハッシュ値のどちらを用いるか

(B) 検査処理のオーバヘッドの定量化

カーネルとLKMの検査処理にかかる時間を明確に表現

# 結果と考察

## <評価結果>

- (1) 検査データのサイズが360バイト以上の場合ハッシュ検査の方がオーバーヘッドが小さい
- (2) ハッシュ検査でカーネルを検査すると約3 $\mu$ 秒のオーバーヘッド発生

## <考察>

- (1) 検査データが360バイト未満であることはほとんどない
  - ➡ バイナリ検査よりハッシュ検査の方が有用
- (2) 検査処理を頻繁に実行するとシステムへの影響が大きくなる
  - (A) 1回の検査処理でカーネルの検査は1回のみ実行
  - (B) LKMのサイズはカーネルと比較するとかなり小さい
  - ➡ 検査処理のオーバーヘッドは十分に小さい



# おわりに

## <ヘルスチェック機能の実装>

### (1) 正常データ作成処理

カーネルテキスト部とLKMの正常データを作成

### (2) 検査処理

カーネルテキスト部とLKMを検査

### (3) 共存OS停止, 再開処理

(1)と(2)において, 共存OS停止・再開処理を適宜実行

## <ヘルスチェック機能の評価>

ヘルスチェック機能の検査処理のオーバヘッドは十分に小さい

## <残された課題>

### (1) 実際のネットワークを介した攻撃に対しての有効性の証明