互換層を信頼しない互換レイヤーのデモ —L4Re上のZpolineによる互換層とPachaOSでの実機動作—

大野 清隆[1,a)]

1. はじめに

マイクロカーネル上のLinux互換レイヤーは，L4Linux[1]のように，互換層が対象の求める権限とカーネル機能に必要な権限の和集合を持つことが多い．そのため，互換層が侵害されれば和集合のすべての権限が攻撃者の手に渡る．本発表では，Zpoline[3]によってシステムコール命令を書き換え，互換層（以下，shim）を対象と同じ非信頼のtaskに置き，権限の強制をtaskの外のカーネルとサーバーに任せる互換レイヤーを示す．あわせて，同じ書き換え方式を用いる著者のOSであるPachaOSでの実機動作を示す．

2. L4Re上の互換レイヤー

図1に構成を示す．対象のLinux ELFの`syscall`命令はロード時に`call *%rax`へ書き換えられ，0番地のトランポリンを経てshimへ制御が移る．shimはファイル記述子の表やアドレス空間の管理情報など，Linuxの意味論をtaskの中で再現し，task外の資源に触れる場合だけcapabilityを呼び出す．shimが壊されても，対象が使えるのはtaskに与えたcapabilityの範囲に限られる．

Fiasco.OC/L4Re[2]上のプロトタイプは，Moe，Ned，カーネルを変更せずに実装した．配布されているBusyBoxとSQLiteを無編集で実行でき，与えるcapabilityを変えると結果がそれに従って変わること，shimを迂回して直接IPCを発行した場合にも同じ上限が保たれることを確認した．カーネルを経由しない通常の経路は，1回のシステムコールを約75 nsで処理した．また，Linux上のプロトタイプを用いたDebianの上位100パッケージの調査では，90件でnativeと動作が一致した．書き換え方式に固有の非互換としては，red zoneの破壊，システムコールの実行位置への依存，0番地の使用によるNULL参照の検出の失敗が見つかった．

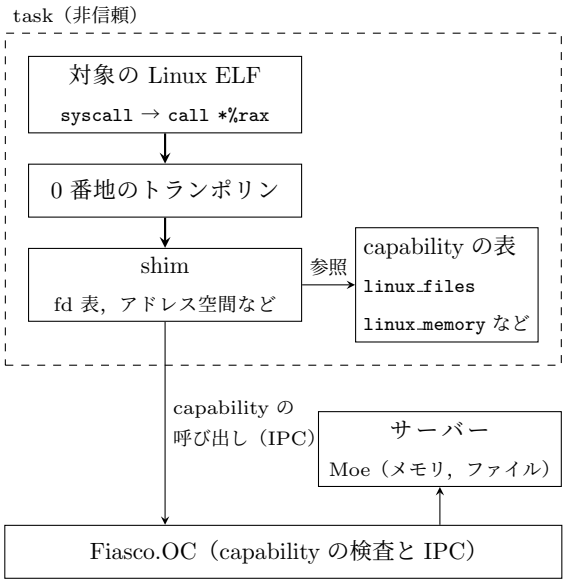


図 1 本手法の全体構成．破線の枠の外にあるカーネルとサーバーが，taskに与えたcapabilityの範囲で要求を検査する．

3. PachaOS

PachaOSでは，Linuxのアプリケーションもnativeのアプリケーションと同じ種類のプロセスとして動作する．Linuxのアプリケーションには，動的リンクしたLinux Personality Runtime（LPR）がZpolineによって`syscall`命令を関数呼び出しに変換し，nativeのシステムコールとユーザー空間のサービスを用いてLinuxのABIを実装する．PachaOSのカーネルは，ファイル記述子（FD）をcapabilityとして扱う．プロセスはOSのオブジェクトを，読み取り，書き込み，マップ，複製などの権利の付いたFD

[1] 開智所沢中等教育学校
[a)] kamezukidaga@gmail.com

© 1959 Information Processing Society of Japan

として受け取り，カーネルは操作ごとにその権利を検査する．FD の複製や移送では権利を増やせない．LPR が管理する Linux の fd 番号はこの FD とは別であり，LPR が壊されたり迂回されたりしても，プロセスが行使できるのは受け取った FD の権利の範囲に限られる．L4Re 上の構成と異なり，fork やメモリの管理など Linux の意味論の一部をカーネルの側でも実装し，プロセスの間の状態を協調用のサーバーで管理している．この構成で，xfce4 のデスクトップ上で GCC と Clang によるビルドや `go build` が動作する．

3.1 kobox2 による Linux ドライバの実行

PachaOS は，L4Linux と同様に Linux のカーネルのコードをユーザー空間で動かす kobox2 によって，Linux のドライバを利用する．kobox2 では，ドライバごとにプロセス単位の sandbox として動作するため，ドライバの再起動や並列の起動が容易である．

kobox2 は，ドライバを制御する Linux のユーザーランドのデーモンを，別のユーザー空間のプロセスとして動かす機能も持つ．このデーモンには静的リンクした Zpoline を組み込み，0 番地のトランポリンのページは，LPR と同じ実装を用いて kobox2 が作成し，読み取りと実行だけを許してデーモンのプロセスに配置する．フックしたシステムコールのレジスタと浮動小数点の状態は共有メモリに書き込まれ，カーネルのイベント通知を受けた kobox2 の中の Linux のコードが処理する．この経路はカーネルを経由し，別のプロセスで処理するため，前章の互換レイヤーの要件は満たさない．一方で，カーネルにシステムコールのトラップを実装せず，既存の基本的な機能だけでデーモンを動作させられる．この機能により，Intel AX210 のドライバと wpa_supplicant を動作させている．

4. デモ

4.1 L4Re（QEMU）

無編集の BusyBox と SQLite を実行し，capability を与えない場合や割り当て量を絞った場合に同じ操作が拒否される様子を示す．また，shim を迂回して IPC を直接発行しても，与えた capability の範囲を越えられないことを示す．

4.2 PachaOS（実機）

xfce4 上での Clang でのビルドと `go build` の実行，kobox2 で動かす Intel AX210 と wpa_supplicant による無線 LAN への接続を示す．また，Go で作成した，静的な HTML と API を提供する Web サーバーについても実装を進めており，その動作をデモで示す予定である．さらに，0 番地のトランポリンによって NULL 参照の検出が失敗する問題に対して，Intel PKU によって 0 番地のページへのデータアクセスを禁止する対策の実装を進めており，PKU の有無によって HotSpot の NullPointerException の挙動が変わることをデモで示す予定である．PKU による保護はデータアクセスだけを制限し，命令の取得は妨げないため，トランポリンを実行したまま 0 番地の読み書きを検出できる．

参考文献

[1] Härtig, H., Hohmuth, M., Liedtke, J., Schönberg, S. and Wolter, J.: The Performance of μ-Kernel-Based Systems, *Proc. 16th ACM Symposium on Operating Systems Principles (SOSP '97)*, pp. 66–77 (1997), DOI: 10.1145/268998.266660.

[2] Kernkonzept GmbH: The L4Re Operating System Framework (online), 入手先 ⟨`https://l4re.org/`⟩ (参照 2026-10-29).

[3] Yasukata, K., Tazaki, H., Aublin, P.-L. and Ishiguro, K.: zpoline: a system call hook mechanism based on binary rewriting, *Proc. 2023 USENIX Annual Technical Conference (USENIX ATC '23)*, pp. 293–300 (2023). 入手先 ⟨`https://www.usenix.org/system/files/atc23-yasukata.pdf`⟩ (参照 2026-10-29).

© 1959 Information Processing Society of Japan