Linux Conference 2004

The need for setuid style functionality in SELinux environments

平成16年6月3日

フェルナンド・バスケス[¶] 保理江 高志[§] 原田 季栄[§]

[¶]Universidad de Vigo

[§](株)NTTデータ 技術開発本部

内容

- 従来のUnix系OSの問題
- SELinuxの概要
- Unix系のUIDの変更とSELinuxのSID変更
  - 従来の解決の問題点。
- 提案機能拡張
  - 拡張の実現
  - 実用例
- 今後の課題
- まとめ

従来のUnix系OSの問題

- アプリケーションレベルの対策だけではコンピュータシステムの守りは実現できない。
- 普通のOSは管理ユーザーがすべての権限を持つので、不正侵入の発生で重要なデータを破壊される恐れがある。
- root権限で動いているプロセスの防御が失敗すると,OSでの対策は不可能。

SELinuxの概要

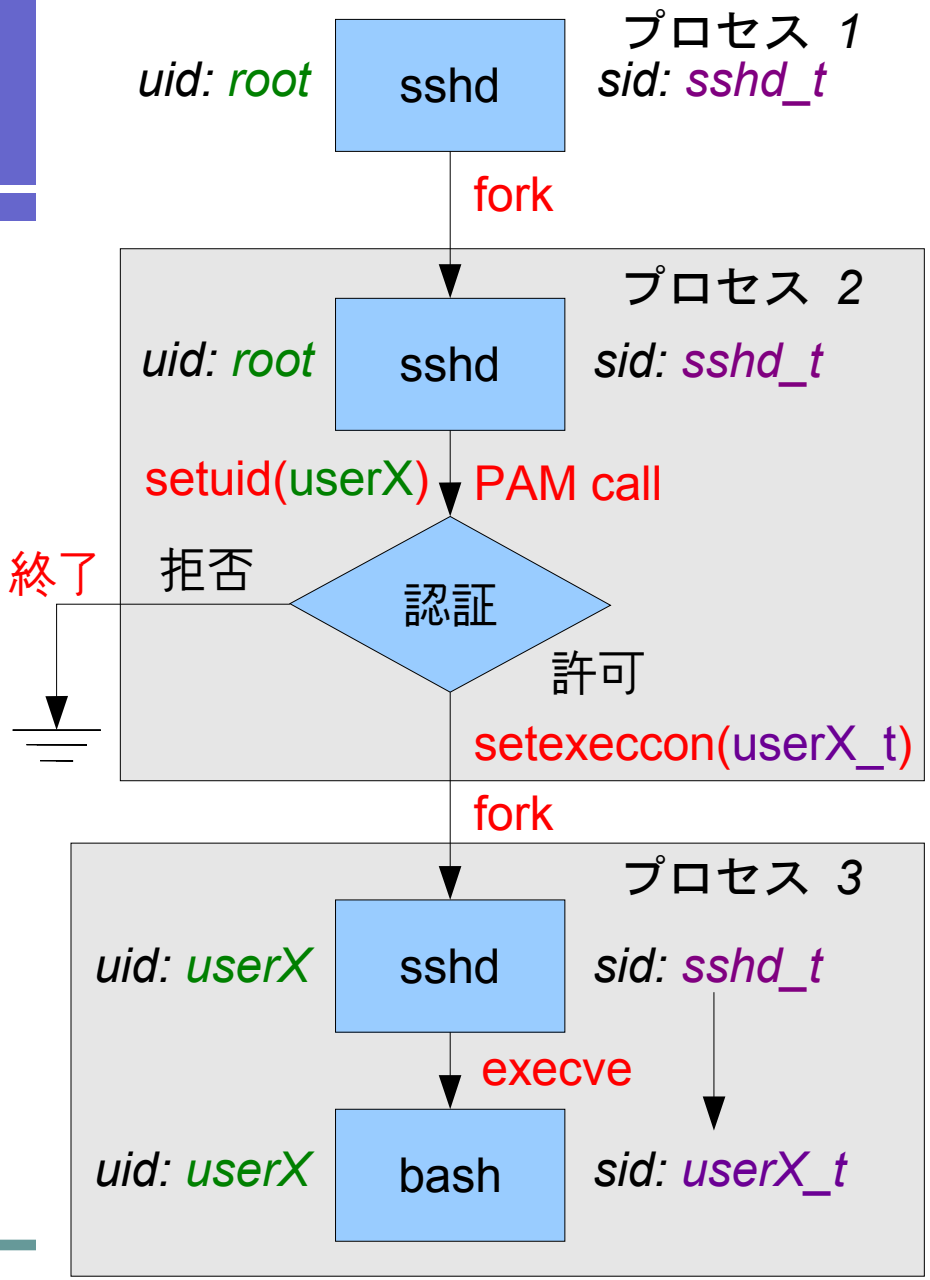
- SELinuxの強制アクセス制御を用いると、管理者でもセキュリティポリシーが禁止する行為ができなくなる。
- 各アプリケーションの振る舞いはsecurity contextというラベル情報により制御される。
- プロセスが実行されると、kernelが適切なSID(security ID)をプロセスに付ける。
- このSIDはプロセスsecurity contextに該当するラベルである。

Unix系のUIDとSELinuxのSID

- UID (User identity)
  - プロセスのオーナーに関する情報である。
  - setuid system callでプロセスのrun timeに変更が可能。
- SID (Security Idenfier)
  - execveの実行のみで変更が可能。
  - このSID変更はドメイン遷移と言われる。
- SELinuxにおけるプロセスは同時にUIDとSIDを持つ。

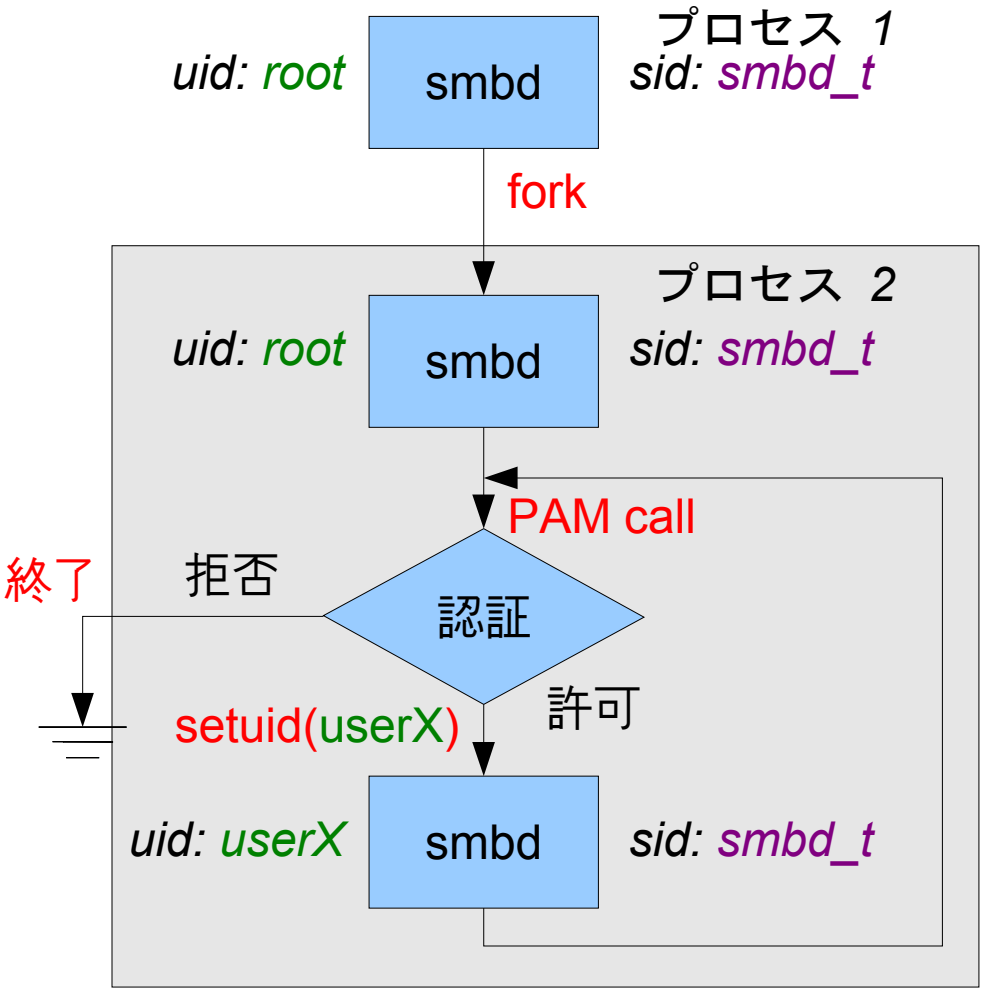
ドメイン遷移

- userXがssh接続すると、セッション処理のために新しいプロセスが起動される。
- ユーザ認証が成功したら、setexecconでドメインの遷移先が設定される。
- ユーザセッションのために、再び新しいプロセスが起動される。
- execveの実行では先に設定したドメインへの遷移が行われて、shellがuserX_tドメインに遷移。



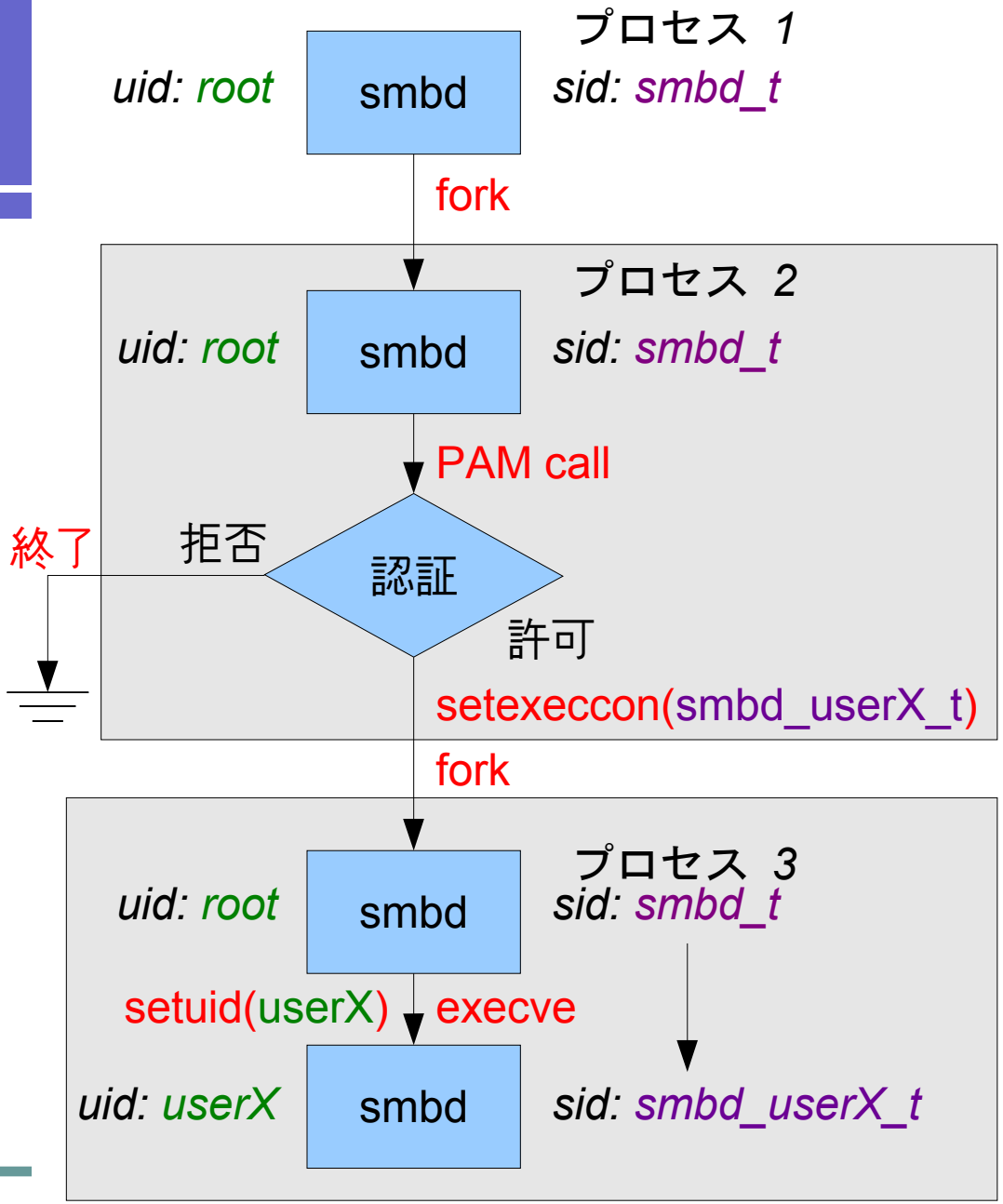
UIDの変更

- SIDと異なり、UIDはプロセスのrun timeに変更することが可能である。
- プロセスはUID変更により管理者権限を解放するが、再取得が可能である。
- Sambaはexecveを使わないので、ドメイン遷移ができない。接続ユーザによるSID変更はできない。
- child processがexploitで侵入されたら、sambaメインプロセスと同じ権限を得る。



execveの問題

- 従来のSELinuxではsambaユーザ毎のSID変更を行いたい場合、アプリケーションはexecveを使用するように修正が必要。
- ソースコードの修正量が多すぎるので、この解決は良くない。
- 各アプリケーションに個別の修正が必要。
  ex)apache,fam ..

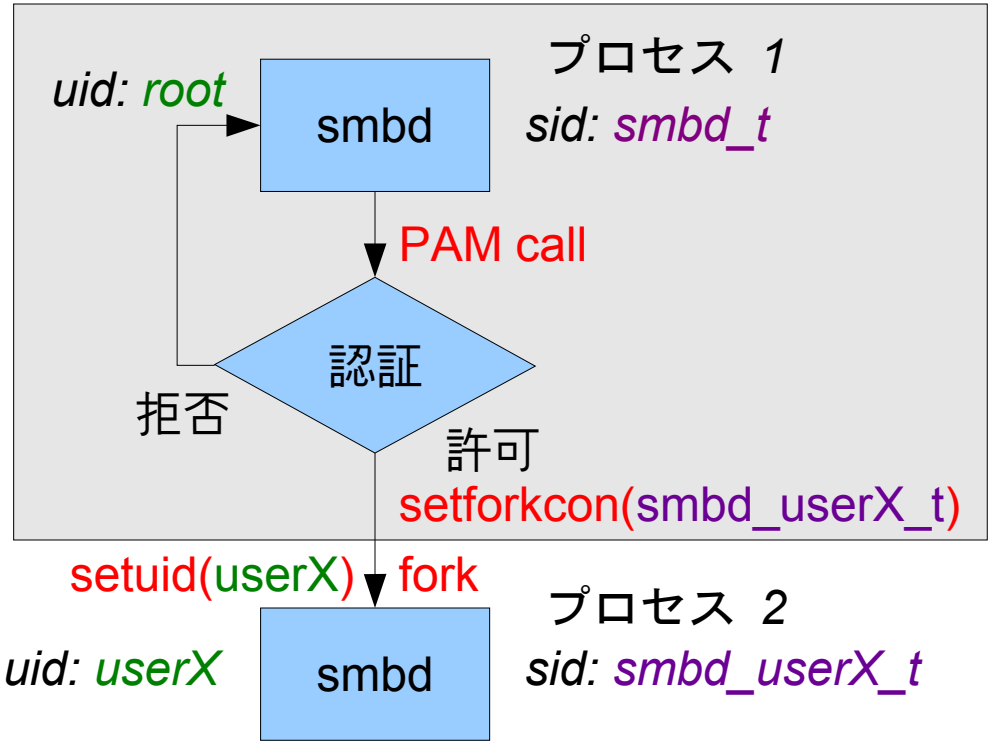


ドメイン遷移を実現する機能

- 前記の問題の解決として、SELinuxのSecurity Context変更について、３つの案を検討し、拡張を行った
  - setforkcon
  - setcon
  - setfssid

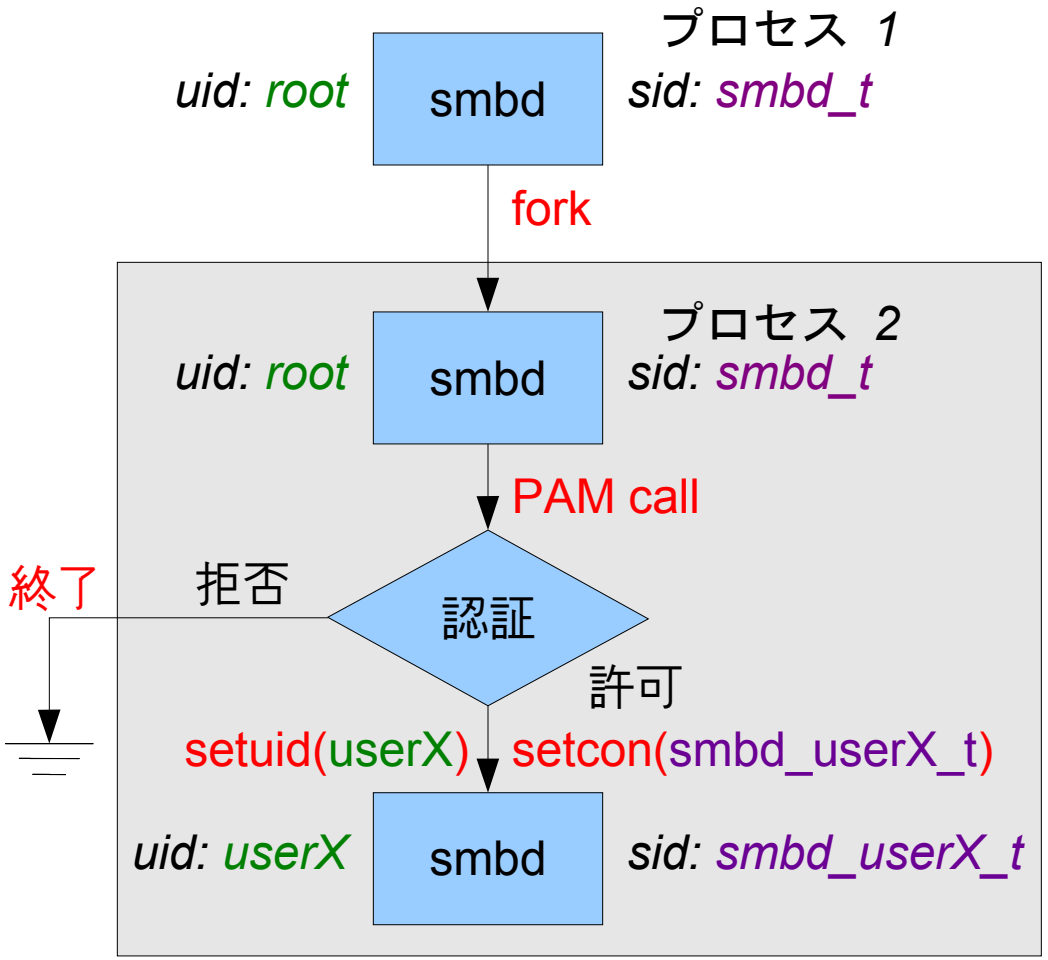
setforkcon

- forkの実行だけでドメイン遷移ができる。
- setexecconと同様に実現した機能。使い方もsetexecconと同じである。
- execveがいらない。



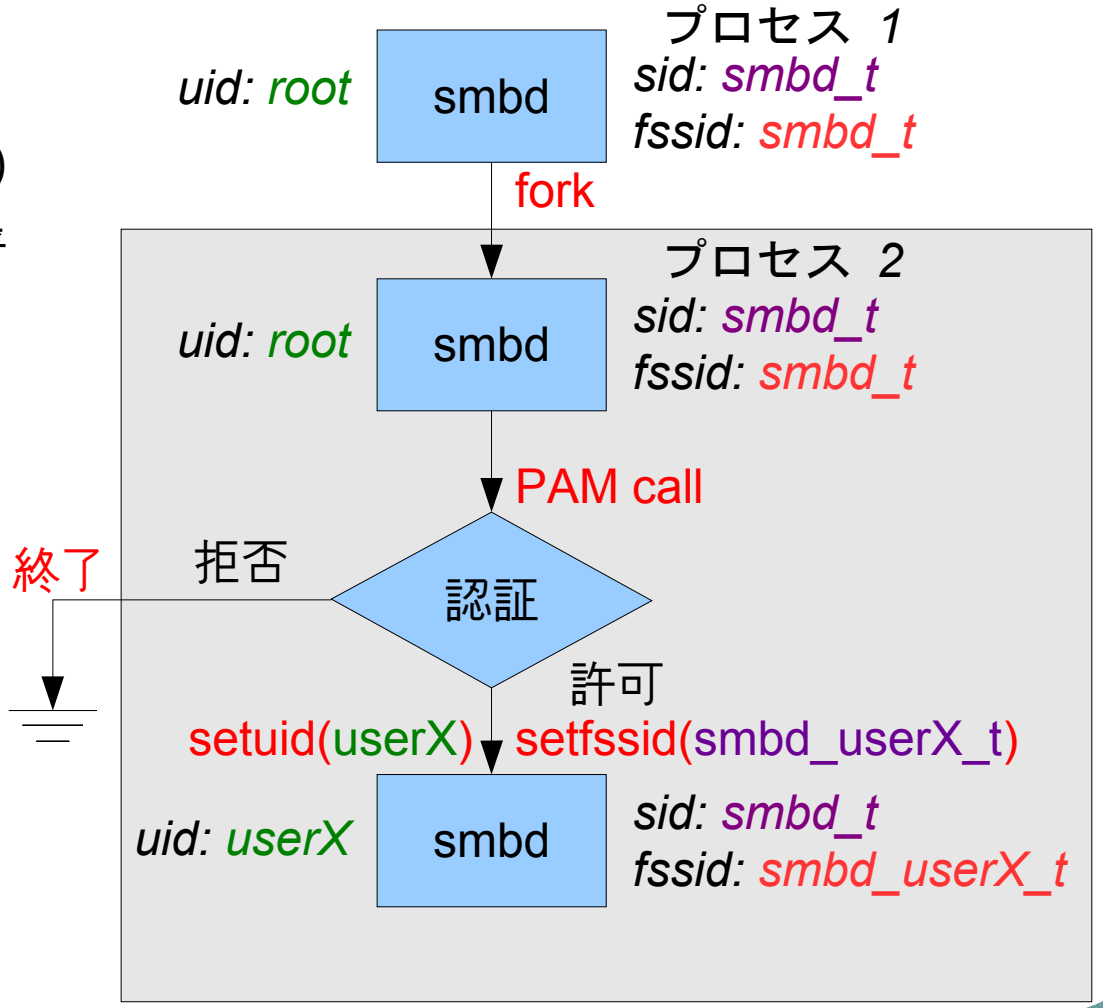
setcon

- UIDを変えるsetuidと同様に、setconはrun-timeにドメインの遷移が行える。
- 新しいドメインはほとんどの元のドメインのステート(open sockets,fd,data空間)などを受け継ぐ。
- sambaメインプロセスの機密情報が子プロセスへのexploitにより奪われる危険がある。



setfssid

- setfssidはSIDの代わりにFSSIDを変える。(setfsuidと同様)
- setfssidの使い方は普通のsetfsuidと似ている。
- SIDは全てのプロセスのアクセス制御に関係しているが、プロセスのFSSIDはfile systemのアクセス制御のための識別情報である。



SELinux環境に行った変更

- Kernelパッチ
  - proc fs
  - LSMフック
  - proc fsとhooksの通信
- SELinuxセキュリティポリシー
- libselinux
- Sambaパッチ

kernelの変更の概要

- SELinuxでは、SIDの切り替えについて、execveのみを契機とするように設計されている。
- execveなしでドメイン遷移を行うために、kernelを修正する必要がある。
- ユーザスペースアプリケーションのために、機能拡張とのインターフェースを提供しなければならない。

kernelの変更の概要 (続き)

- 主要な操作(ファイルのwrite,socketのconnect...)の制御はLSMフックという構造体で行われる。
- 機能拡張が新しい操作として実現されたので、フックを修正する必要がある。
- 機能拡張に対するアクセスポリシーを定義するために、各機能に個別のパーミッションを作らなければならない。

setfork関数の実行に対するポリシー定義
allow smbd_t self:process { setfork };

setforkcon機能拡張

- setforkcon(forkでのchildプロセスのSID変更)のために、forkの拡張を行った。
- setforkconを使わない場合は従来のforkと同じ動作。
- 従来のアプリケーションでもSELinuxによって修正したアプリケーションでも、同じように新しいforkが使える。

setcon機能拡張

- execveなしでドメイン遷移が可能
- 普通のSELinuxの環境で、Samba運用に適する機能がない。☞
- この機能拡張の導入で、Sambaなどのアプリケーションの権限をユーザ毎に分けることは可能である。☞

setfssid機能拡張

- setfssidでは、SIDを切り替える代わりに、FSSIDだけを変える。
- SELinuxにおけるsetfssidはsetfsuidのような機能を提供する。
- 管理ユーザ権限で動く必要があるアプリケーションのために、実装した。
- この機能はfile systemフックの修正で実現される。

Kernelインターフェース

- 追加した機能はユーザスペースの部分とkernelの部分に分かれている。
- 使い易さのために、libselinuxというlibraryはこのインターフェースをラップする。
- 機能拡張によって下記の関数をlibselinuxに追加した。
  - setforkcon, setcon, setfssid

*proc file system*のファイル

- 実際にkernel・ユーザスペースのインターフェースは*/proc/pid#/attr/*にあるファイルである。
- 機能拡張のために下記のファイルを追加した。
  - *fork, fscreate, fssid*
- libselinuxに追加した関数は別の関数と同様に上記のファイルを介してkernelと通信する。

proc file systemのファイル(続き)

- ファイルに書き込んだデータは右のkernelの構造体で保存される。
- ファイルから読み込んだデータもこの構造体から来る。
- 機能拡張のために、ボールドしたメンバーを追加した。

```
struct task_security_struct {
  unsigned long magic;
  struct task_struct *task;
  u32 osid;/*execveの前のSID*/
  u32 sid;/*現在のSID*/
  u32 fssid;/*現在のFSSID*/
  u32 exec_sid;/*execのSID先*/
  u32 create_sid;
  u32 fs_sid;/*setfssidの
               FSSID先*/
  u32 fork_sid;/*setforkconのSID
先*/
  struct avc_entry_ref avcr;
};
```

新たなポリシーの制御

- 各機能の制御のために、該当するパーミッションを作らなければならない。
- パーミッションを追加するために、ユーザスペースもkernelも変更する必要がある。
- 初めにSELinuxポリシーディレクトリにあるファイル(access_vectors)で新たなパーミッションが定義される。

新たなポリシーの制御(続き)

- kernelのために、ポリシーからヘッダーファイルを生成してkernelを再コンパイルする。
- パーミッションの定義が変わったら、kernelを再コンパイルする必要がある。

sambaのメインプロセス		ユーザセッションを処理するsambaプロセス	
UID:	root	UID:	foousr
SID:	smbd_t	SID:	smbd_foousr_t ☞
		FSSID:	smbd_foousr_t ☞

setfork関数の実行に対するポリシー定義

```
allow smbd_t self:process { setfork };
```

今後の課題

- Sambaのためのポリシー変更の自動化。
  - useradd/userdelコマンドラッパー。
- ファイルアクセスについて、VFS (Virtual File System) Pluginの実現を検討している。
- ユーザ認証のためにネットワークワイドSIDの使用も検討中である。

まとめ

- execveのみのドメイン遷移は厳しい制御ができる。しかし、いくつかのアプリケーション運用(Samba, etc..)には使いにくい。
  - セキュリティ・使い勝手のトレードオフで、セキュリティがより重要
- 今回の３つの拡張によってSambaなどのアプリケーションは、より正確な制御が可能。
- SELinuxのポリシーの管理については、macroを作る必要がある。

fin/終り

- Muchas gracias por su paciencia y atención.
- ご静聴ありがとうございました。

フック(Hooks)

- 機能拡張を実装するために、いつもLSM (Linux Security Modules)における適切なフック(hooks)関数を修正しなければならない。
- フックは関数呼び出しにより重要な各kernel機能が実現される。
- フックでもっと正確なアクセス制御は可能です。SELinuxのフックによって強制アクセス制御は実装されている。