社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－DPS－134 (20)
2008－CSEC－ 40 (20)
2008／3／6

Trusted Domain Enforcementの軽量実装による組み込みLinux 2.6系列のセキュリティ強化

安藤類央 門林雄基 篠田陽一

独立行政法人 情報通信研究機構 情報通信セキュリティ研究センター
〒184-8795 東京都小金井市貫井北町 4-2-1

あらまし

組み込み機器はオープンソースのソフトウェアを搭載することが多くなっており、脆弱性を利用した悪意のある入力を前提としたシステムの設計が重要になっている。組み込み機器は、特定の用途に特化されており、悪意のある入力に対しては、軽量な検証とサンドボックス化の実装が有効であると考えられる。本論文では、TCSEC(Trusted Computer System Evaluation Criteria) でのTDE(Trusted Domain Enforcement)を用いて、組み込みLinux 2.6系列に軽量に実装する方法を提案する。提案手法は、軽量実装のため、1つのアプリケーションのサンドボックス化を目的としたカーネルパッチを適用した。TDEの実装にあたっては、LIDS(Linux Intrusion Detection System) のlinux 2.4のバージョンを参考にし、組み込みWEBサーバに利用されることの多い、Tiny Httpdへの適用を行った。

A security enhancement of embedded Linux 2.6 series using lightweight trusted domain enforcement implementation

Ruo Ando, Youki Kadobayashi, Youichi Shinoda

National Institute of Information and Communication Technology, Tracable Network Group
4-2-1 Nukui-Kitamachi, Koganei,
Tokyo 184-8795 Japan
ruo@nict.go.jp

Abstract Deployment of open source software for embedded system is increasing. Sytem designer need to cope with malicious input using exposed vulnerability. Lightweight verification and sandbox approach is effective for embedded system. Because embedded system is specified, has less generic purpose than PC and servers. In this paper we apply TDE (Trusted Domain Enforcement) for embedded linux 2.6 series. TDE is a concept showed in TCSEC (Trusted Computer System Evaluation Criteria) for input validation and sandbox for protecting system. Our system is based on LIDS (Linux Intrustion System) for Linux 2.4 series. We implement our system as small kernel patch. We also discuss the modification of thttpd in deployment of our TDE system.

1 はじめに

組み込み機器に搭載されるOSに、オープンソースであるLinuxが採用されるケースが増えており、セキュリティが重要な課題の1つになっている。組み込み機器に限らず、悪意のある入力や、攻撃が成功した後を前提としたシステムの設計が重要になっており、セキュアOSが開発されている。現状のLSMをベースにしたシステムでは、パスの検証や、保護されていない入力を扱うシェルへの防御などの対策が手薄であるときがある。組み込み機器は、特定の用途に特化されており、悪意のある入力に対しては、軽量

なサンドボックス化の実装が有効であると考えられる。TCSEC(Trusted Computer System Evaluation Criteria) で示されている概念に、Trusted Domain En f orcement があり、Linux 2.4 系列では、LIDS などで実装されていたが、本論文では、Linux2.6 系列に TDE を軽量実装する手法を提案する。また、組み込み WEB サーバでの利用法について述べる。

2 関連技術

2.1 LIDS for Linux 2.4 series

LIDS (Linux Intrusion Detection System)[3] は、Xie Huagang 氏と Philippe Biondi 氏が中心となって開発されているセキュア OS の 1 つで、inode ベースのアクセス制御が特徴である。LIDS には、カーネル 2．4 と 2．6 について 2 つのバージョンがあり、2．4 系列では、TDE や NETMARK などの機能があるが、これらの機能は、2．6 系列版ではまだサポートされていないようである。

2.2 Linux Security Module

LSM[2] は、Linux2.6 から導入されたセキュリティのためのカーネルモジュールで、システムコールがリソースにアクセスする間に仲介し、アクセス制御を行う。2．4 でのセキュア OS は基本的にカーネルパッチの形で提供されていたが、2．6 からは LSM を用いた実装が増えたようである。

2.3 Trusted Domain Enforcement

悪意のある実行ファイルへのセキュリティ確保の手法として、TPE (Trusted Path Execution) と TDE (Trusted Domain Execution) がある。TPE は、外部から持ち込まれ生成された実行ファイルに対して有効であるが、シェルやその他の実行処理系が信頼された場合、悪意のある入力による動作を検査することができない。TDE は、実装処理系をサンドボックス化する、あるいは保護されてない入力があったときに実行処理系のケーパビリティを変更するというものである。LIDS 1 系列では、TDE が有効化すると、write protect されたいない入力を受け付けたプロセスのケーパビリティが剥奪される。この際に、フィルタの制限方法が通常と逆になり、コマンドを実行するときは、明示的に関連するライブラリなどにも、PERMIT 権限を与える必要がある。図 1 は、通常の TDE と軽量 TDE を比較したものである。提案システムでは、TDE が有効になったときに、明示的に、DENY フィルタを適用することで、プロセスのアクセスを制御する方法をとった。

3 実装方法

3.1 プロセスのケーパビリティの設定

TDE は、プロセスベースのアクセス制御であり、プロセスは、task_struct 構造体で管理されるため、本論文では、task_struct 構造体に、TDE のフラグを付加し、do_fork や do_execve の実行時に、このフラグの設定を行う。Trusted Domain Enforcement には、ファイルシステムのどの位置で、どのプロセスから、どのアプリケーションを実行するかという 3 つの条件を検査することが必要になる。そのため、do_exec 関数内で、inode 番号の取得、プロセス番号の取得を行い、どのプロセスがどのアプリケーションを実行しているか把握し、サンドボックス化したプロセスの制御を行う。

3.2 入力の検査

Trusted Domain Enforcement 適用の際には、保護されているファイルとは、READONLY 属性のあるファイルのことを指すようである。

```
Subject    ACCESS   inherit Object
-----------------------------------
Any file   READONLY:   0  /sbin
Any file   READONLY:   0  /bin
Any file   READONLY:   0  /boot
Any file   READONLY:   0  /lib
Any file   READONLY:   0  /usr
Any file   READONLY:   0  /etc
Any file       DENY:       0  /etc/lids
Any file       DENY:       0  /etc/shadow
Any file     APPEND:     0  /var/log
Any file      WRITE:       0  /var/log/wtmp
/bin/login  READONLY: 0 /etc/shadow
/bin/su  READONLY:     0 /etc/shadow
/bin/login GRANT:        0 CAP_SETUID
```

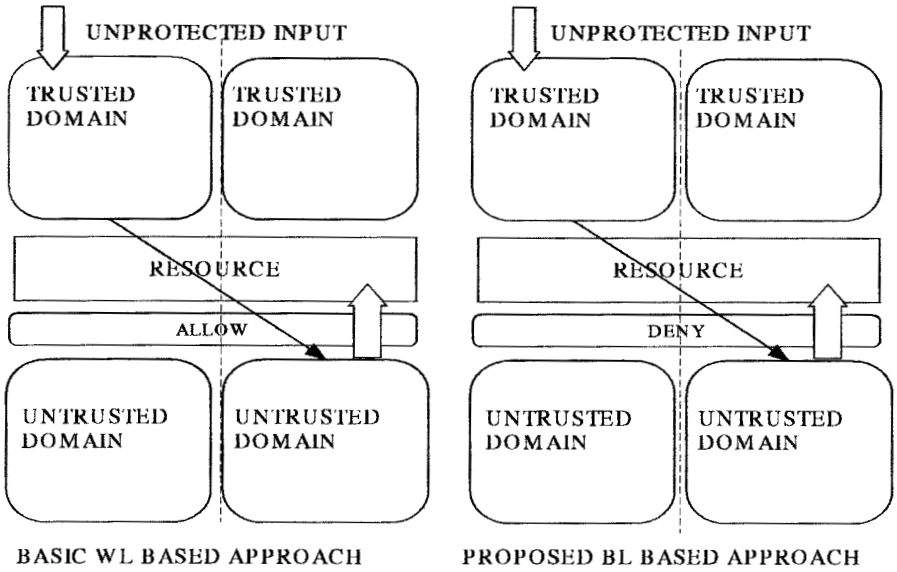

図 1: 通常の TDE と提案システムの TDE の比較。左側は、ケーパビリティを設定し、それを変更する。PERMIT を明示的に設定する。提案では、TDE が実行されたとき、アクセス権 DENY を明示的に設定する。

上は、LIDS のデフォルトのアクセス設定を示したものである。TDE のもうひとつの適用法として、/var/log/wtmp や/tmp などに WRITE 権限のあるディレクトリから入力のあったプロセスに対しては、TDE を適用する。これにより、保護されていない入力により、対象プロセスの挙動が意図しない方向へ誘導されるのを防ぐことができる。実装法としては、ファイルシステム系のシステムコール関数を修正し、プロセスがファイルを入力として扱う前に対象ファイルが READONLY であるかどうかチェックする。TDE のケーパビリティが付与されたプロセスが、READONLY 以外の属性のファイルにアクセスする場合、TDE が施行され、プロセスの実行が防止される。具体的には、read_write.c の sys_read 関数内で、現在の（呼び出し元の）プロセス構造体を取得し、TDE のフラグが立って検査する。一方で、dentry 構造体から、inode 構造体をたどり、inode 番号を取得し、PROTECT されていない (WRITE 属性のある）ディレクトリ内のリソースアクセスであるか検査する。この 2 条件がそろったとき、TDE を実行し、/bin や、/sbin 以下のファイルが実行されているようなら、その動作を禁止する。

3.3 組み込み Web サーバへの適用

本論文では、TDE の拡張として、オーバーフロー攻撃を対象として、WEB デーモンのサンドボックス化の実装を行った。Linux では、current マクロを用いて実行中のプロセスの構造体を取得することができる。図 2 は、提案システムの WEB サーバへ保護されていないあるいは悪意のある（オーバーフローを意図した）入力のあった場合の TDE フラグの変更を示したものである。不正アクセスがあった場合、TDE を適用し、カーネル空間でフィルタを適用することで、HTTPD のサンドボックス化を行う。

オーバーフローの検出は、ユーザモードとカーネル空間の 2 つで行うことが可能である。カーネル空間であ、fault handler 内でプロセス構造体の TDE フラグを変更する。この場合、攻撃でない fault なども補足し、untrusted domain に移行してしまうので、ユーザモードで補足した方が適切であると考えられる。ユーザモードの signal hander などでオーバーフローを検出する場合、提案システムでは、ソフトウェアブレークポイントを用いて、カーネル空間のハンドラに制御を移し、トラップハンドラを修正してプロセス構造体の TDE フラグを変更する。

そして、ファイル系のシステムコール実行時にアクセスフィルタを適用し、これにより、HTTPD あるいは、そこから生成されるプロセスをサンドボックス化する。

修正方法の概要は以下のようになる。

```
---- user space ----
# include <signal.h>
int sigaltstack
(const stack_t *ss, stack_t *oss);
int main()
if(sigsetjmp(return_point,1)==0
{
- main code of thttpd -
}
else
{
- reaction against buffer overflow -
software breakpoint;
}
}
void handler()
{
if(sig=SIGSEGV)
if(stack_overflow)
siglongjmp(return_point,1);
}

---- kernel space ----
do_trap
{
  task_struct = current;
- tde enabled -
}
```

その他、ユーザ空間から、カーネルの構造体を変更する方法があればそれを適用することができる。

3.4 Busybox

TDE はプロセスベースのアクセス制御のため、どのプロセスが、どの実行ファイルにアクセスしているか把握する必要があるが、組み込みシステムの場合、実行ファイルが busybox[8] にまとまられている場合が多い。そのため、ファイル実行時のディレクトリ位置と、バス名の取得などにより、実行ファイル名の特定を行う必要がある。あるいは、busybox 内のソースコードを修正し、実行コマンドごとのアクセス制御を行うことが可能であると想定される。

4 結論と今後の課題

組み込み機器はオープンソースのソフトウェアを搭載することが多くなっており、脆弱性を利用した悪意のある入力を前提としたシステムの設計が重要になっている。組み込み機器は、特定の用途に特化されており、悪意のある入力に対しては、軽量なサンドボックス化の実装が有効であると考えられる。本論文では、TCSEC(Trusted Computer System Evaluation Criteria) での、Trusted Domain En f orcement を組み込み Linux 2.6 系列に軽量に実装する方法を提案した。提案手法は、軽量実装のため、1 つのアプリケーションのサンドボックス化を目的としたカーネルパッチを適用した。TDE の実装にあたっては、LIDS(Linux Intrusion Detection System) の linux 2.4 のバージョンをベースし、組み込み WEB サーバに利用されることの多い、Tiny Httpd への適用を行った。

Appendix: LSM での実装について

本文で述べた Trusted domain enforcement は、LSM ベースのシステムにも適用することができる。1 つの方法としては、ファイルにアクセスするとき、inode permmision ルーチンは親ディレクトリの inode 番号も参照するため、このルーチンから現在のプロセス構造体を取得し、TDE を実施することが可能である。この場合、LSM をカーネルに組み込むか、もしくは LKM(Loadable Kernel Module) の形で提供できる資源環境が必要である。

参考文献

[1] Trusted Computer Systems Evaluation Criteria http://www.auditmypc.com/acronym/TCSEC.asp

[2] Chris M Wright, Linux Security Module Framework, Linux Symposium 2002.

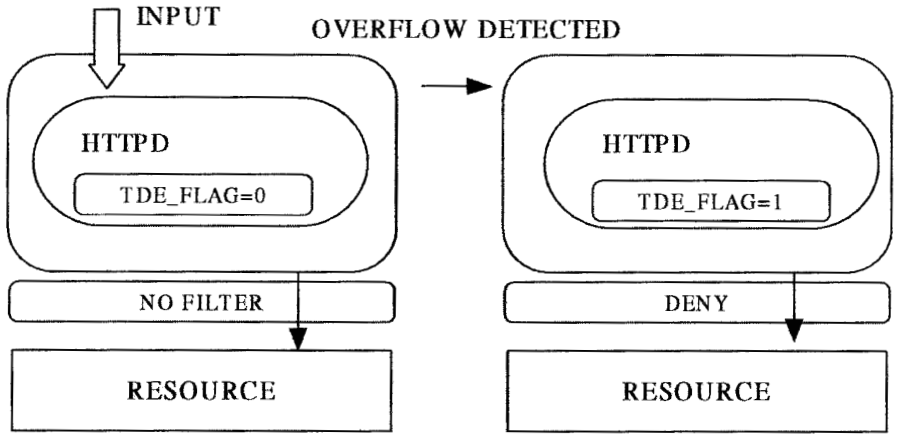

図 2: （悪意のある入力により）オーバーフローが起きたときに、拡張したプロセス構造体の TDE のフラグを立てて、リソースのアクセス禁止を明示的に設定する。

[3] LIDS Secure Linux System available at:
http://www.lids.org/

[4] Yusuf Wilajati Purna, "LIDS Trusted Domain Enforcement", 2004
available at: http://www.lids.org/

[5] Niki A. Rahimi, "Trusted path execution for the linux 2.6 kernel as a linux security module", In Proceedings of the USENIX Annual Technical Conference 2004.

[6] Ruo Ando, Youki Kadobayashi, "Improving VMM based IPS for real-time snapshot and nullification of buffer overflow exploitation" The 1st Joint Workshop on Information Security September, 20-21, Sookmyung Women's University, Korea

[7] thttpd - tiny/turbo/throttling HTTP server:
http://www.acme.com/software/thttpd/

[8] busybox availabe at:
http://www.busybox.net/

[9] Trusted OS Research and Development:
http://itslab.csce.kyushu-u.ac.jp/ssr/kim_report.htm