SELinuxユーザ会2007年夏の勉強会

～小ネタ～
BusyBoxのWrapperについて

日本SELinuxユーザ会
(セキュアOSユーザ会)
宍道　洋

BusyBoxとは？

- 沢山のコマンド(アプレット)を一つのバイナリにまとめ、コンパクトにしたもの
  - スイス・アーミー・ナイフのようなツール
- 通常リンクを張って使用する
  - **Ex.)**

ln –s /bin/busybox /bin/ps
ln –s /bin/busybox /bin/vi

- もしくは引数で直接指定
  - **Ex.)**

/bin/busybox ls -l
/bin/busybox date

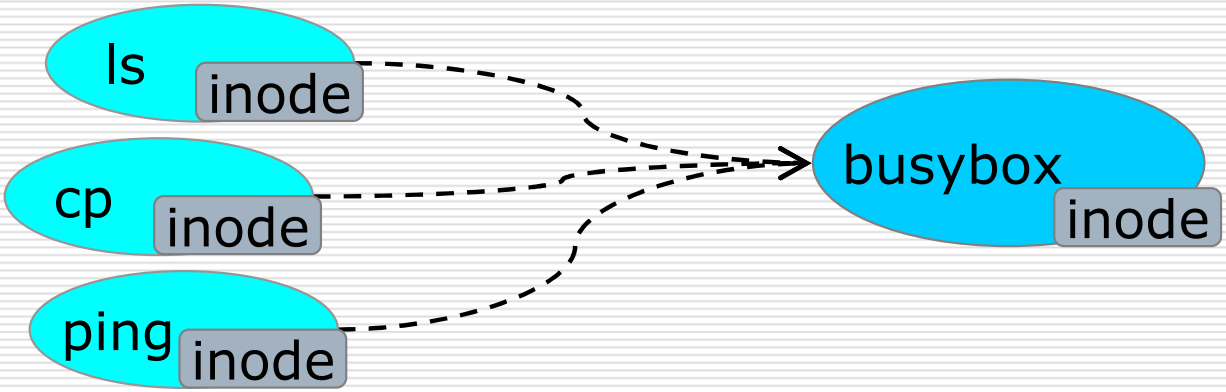
- 共通コードをまとめて、フットプリントを小さくできる
  → 組込み向け

```
~ # busybox
BusyBox v1.2.2 (2006.10.31-00:55+0000) multi-call binary

Usage: busybox [function] [arguments]...
   or: [function] [arguments]...

        BusyBox is a multi-call binary that combines many common Unix
        utilities into a single executable.  Most people will create a
        link to busybox for each function they wish to use and BusyBox
        will act like whatever it was invoked as!

Currently defined functions:
        [, [[, addgroup, adduser, ash, basename, busybox, cat, chgrp,
        chmod, chown, chroot, clear, cmp, cp, cut, date, dc, dd, delgroup,
        deluser, df, dirname, dmesg, du, dumpleases, e2fsck, echo, egrep,
        env, expr, false, fdisk, fgrep, find, free, fsck, fsck.ext2, fsck.ext3,
        getopt, getty, grep, gunzip, gzip, halt, head, hexdump, hostid,
        hostname, httpd, hwclock, id, ifconfig, ifdown, ifup, inetd, init,
        insmod, kill, killall, klogd, less, ln, logger, login, logname,
        ls, lsmod, md5sum, mkdir, mke2fs, mkfs.ext2, mkfs.ext3, mknod,
        mkswap, modprobe, more, mount, mv, netstat, nslookup, passwd,
        pidof, ping, pivot_root, poweroff, ps, pwd, rdate, reboot, reset,
        rm, rmdir, rmmod, route, run-parts, sed, sh, sha1sum, sleep, sort,
        strings, stty, su, swapoff, swapon, sync, syslogd, tail, tee,
        telnet, telnetd, test, time, top, touch, true, tty, udhcpc, udhcpd,
        umount, uname, uniq, uptime, usleep, vi, wc, wget, which, whoami,
        xargs, yes, zcat

~ #
```

BusyBoxとセキュアOS

- BusyBoxをリンクで呼び出すと、リンク元の実体に定義されたアクセス制御しか働かない。
  - SELinux、LIDS → 実体のinode内の情報を利用
  - AppArmor → リンク解決後のパス名を利用

リンクではなく実ファイルからの呼び出し（Wrapperの利用）



バイナリラッパ

- /bin/busyboxを呼び出すバイナリを作成する
  - Cで書くとこんな感じ

```
void main(int argc, char* argv[], char* envp[]) {
    argv[0] = "cat";
    execve("/bin/busybox", argv, envp);
}
```

  - これをアセンブラで記述（サイズは1k未満）

```
/bin # ls -l
-rwxr-xr-x    1 root     root           696 Aug 17 12:01 addgroup
-rwxr-xr-x    1 root     root           692 Aug 17 12:01 adduser
-rwxr-xr-x    1 root     root           688 Aug 17 12:01 ash
-rwxr-xr-x    1 root     root           688 Aug 17 12:01 cat
-rwxr-xr-x    1 root     root           692 Aug 17 12:01 chgrp
...
```

→ アーキテクチャ毎にコードを作成・メンテナンスする必要

参考：「BusyBoxを使った組み込み機器でLIDSを動かすには」（佐藤 祐介氏）
http://www.selinux.gr.jp/LIDS-JP/document/general/web_lids_busybox/main.html

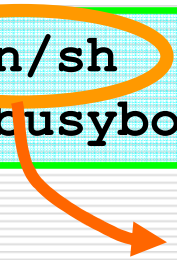
じゃ、シェルスクリプトでやってみる？

- 例えば“cat”

```
#!/bin/sh
/bin/busybox cat $*
```

- じゃあ、“sh”は？

```
# /bin/sh
/bin/busybox sh $*
```



これもbusyboxだったりする・・・

- 例えば“cp”が

```
#!/bin/sh
/bin/busybox rm $*
```

と置き換えられたら・・・

execve()を見てみた

execve(2)のmanpage

書式

```
#include <unistd.h>

int execve(const char *filename, char *const argv[],
           char *const envp[]);
```

説明

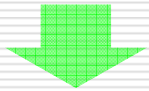
execve() は、filename によって指定されたプログラムを実行する。Filename は、バイナリ実行形式か、**“#! interpreter [arg]”という形式の行で始まるスクリプトでなければならない。**後者の場合、interpreter は適切な実行ファイルのパス名でなければならず、それ自身がスクリプトであってはならない。**そしてそれは interpreter [arg] filename の形で呼び出される。**

では、“#!/bin/busybox”と書いたファイルだけでOK?

スクリプトラッパ

- /bin/lsに“#!/bin/busybox”と書いたファイルを置いて実行

```
# ls
busybox: applet not found
#
```



lsのフルパスがbusyboxの引数として入るため
“/bin/ls” ≠ “ls”

BusyBoxの修正

- アプレット名を拾う部分を修正

```
-           applet_name = argv[0];
-           run_applet_and_exit(argv[0], argv);
+           applet_name = bb_get_last_path_component(argv[0]);
+           run_applet_and_exit(applet_name, argv);
```

フルパスからファイル名のみ引っ張ってくる

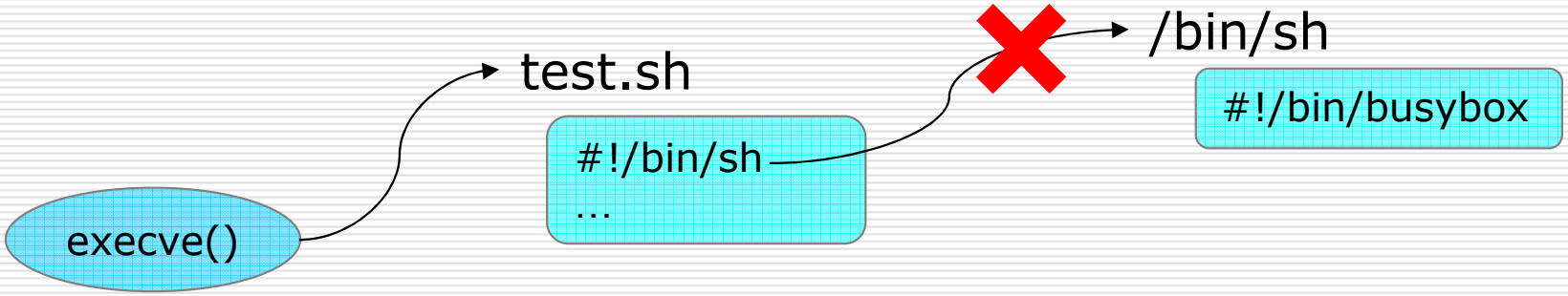
```
# cat /bin/ls
#!/bin/busybox
# ls -l /bin/l*
-rwxr-xr-x    1 root     root         18128 Jan 11 00:06 /bin/link
-rwxr-xr-x    1 root     root         29840 Jan 11 00:06 /bin/ln
-rwxr-xr-x    1 root     root         79412 Sep 29 20:27 /bin/loadkeys
-rwxr-xr-x    1 root     root         25756 Feb  9 21:43 /bin/login
-rwxr-xr-x    1 root     root            15 Feb 21 04:47 /bin/ls
-rwxr-xr-x    1 root     root         93560 Jan 11 00:06 /bin/ls.orig
#
```

execve()の制限

説明

execve() は、filename によって指定されたプログラムを実行する。Filename は、バイナリ実行形式か、“#! interpreter [arg]”という形式の行で始まるスクリプトでなければならない。**後者の場合、interpreter は適切な実行ファイルのパス名でなければならず、それ自身がスクリプトであってはならない。**そしてそれは interpreter [arg] filename の形で呼び出される。

- /bin/shの中身も“#!/bin/busybox”にすると、execve()でシェルスクリプトを呼び出した時に上の条件により、実行できない。(initからのスクリプト実行など)



→ /bin/shのみはリンクにする。

まとめ

- スクリプトラッパの利点
  - 名が体を表わす。
  - すべて中身は“#!/bin/busybox”15byteのみ
- SELinux、及びAppArmorで、ドメイン遷移、アクセス制御の動作を確認（中村さん）
- ただし/bin/shのみはリンク

- 今後
  - BusyBoxへのスクリプトラッパのパッチ（インストーラ含む）を本家へ投稿