Copyright Notice

Presentation by: Alessandro Selli <alessandroselli@linux.com>

Presented at LinuxCon Europe 2014, Düsseldorf.
Synergia S.R.L. is not in any way associated with the Linux Foundation.

Version 2.1.3, 2014/10/10

Synergia

Linux Capabilites
Table of Contents

Linux Capabilites: what they are and accomplish 1-3

- Started as an implementation of POSIX:
  - 1003.1e (API), "Protection, Audit and Control Interfaces"
  - 1003.2c (Shell and Utilities), "Protection and Control Interfaces"
- Sometimes called Linux POSIX Capabilities
- 1003.1e and 1003.2c were last revised in 1997
- In 1999 they were set as withdrawn drafts
- Linux has thus gone it's own way

Linux Capabilites: what they are and accomplish 2-3

Advantages:

- They allow delegation of super-user rights to unprivileged processes, like suid bit

Linux Capabilites: what they are and accomplish 2-3

Advantages:

- They allow delegation of super-user rights to unprivileged processes, like suid bit
- They are recorded on the filesystem, like suid bit

Linux Capabilites: what they are and accomplish 2-3

Advantages:

- They allow delegation of super-user rights to unprivileged processes, like suid bit
- They are recorded on the filesystem, like suid bit
- They do not work on a "*everything or nothing*" basis, unlike suid

Linux Capabilites: what they are and accomplish 2-3

Advantages:

- They allow delegation of super-user rights to unprivileged processes, like suid bit
- They are recorded on the filesystem, like suid bit
- They do not work on a "*everything or nothing*" basis, unlike suid

In short:

- they allow a process to do *some* selected things with root rights, while not anything else

Linux Capabilites: what they are and accomplish 3-3

Disadvantages:

- Commands need code to be made LC-conscious

Linux Capabilites: what they are and accomplish 3-3

Disadvantages:

- Commands need code to be made LC-conscious
- Some FS do not support LC (NFS v.3, though squashfs, tmpfs and f2fs now do):

```
[root@debian ~]# getcap /mnt/nfs/nfsserver/dumpdates
```

Linux Capabilites: what they are and accomplish 3-3

Disadvantages:

- Commands need code to be made LC-conscious
- Some FS do not support LC (NFS v.3, though squashfs, tmpfs and f2fs now do):

```
[root@debian ~]# getcap /mnt/nfs/nfsserver/dumpdates
Failed to get capabilities of file `/mnt/nfs/nfsserver/dumpdates'
(Operation not supported)
[root@debian ~]#
```

Linux Capabilites: what they are and accomplish 3-3

Disadvantages:

- Commands need code to be made LC-conscious
- Some FS do not support LC (NFS v.3, though squashfs, tmpfs and f2fs now do):

```
[root@debian ~]# getcap /mnt/nfs/nfsserver/dumpdates
Failed to get capabilities of file `/mnt/nfs/nfsserver/dumpdates'
(Operation not supported)
[root@debian ~]#
```

- LCs are not dropped like SUID
  - The newer libcap-ng libs makes this easy

The `ping` case 1-4

- A classic example: `ping`

The `ping` case 1-4

- A classic example: `ping`
- It needs superuser rights to send echo-request ICMP packets

The `ping` case 1-4

- A classic example: `ping`
- It needs superuser rights to send echo-request ICMP packets
- You surely do *not* want to let anyone be able to produce arbitrary ICMP packets!

The `ping` case 1-4

- A classic example: `ping`
- It needs superuser rights to send echo-request ICMP packets
- You surely do *not* want to let anyone be able to produce arbitrary ICMP packets!
- This is the traditional approach:

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$ ll /bin/ping
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$ ll /bin/ping
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[alessandro@ubuntu ~]$
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$ ll /bin/ping
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[alessandro@ubuntu ~]$
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$ ll /bin/ping
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$ ll /bin/ping
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=55 time=46.8 ms
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$ ll /bin/ping
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=55 time=46.8 ms
64 bytes from route-add.net (195.182.210.166): icmp_seq=2 ttl=55 time=46.6 ms
```

The `ping` case 2-4

Starting situation:

```
[alessandro@ubuntu ~]$ lsb_release -drc
Description:    Ubuntu 14.04.1 LTS
Release:        14.04
Codename:       trusty
[alessandro@ubuntu ~]$ ll /bin/ping
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=55 time=46.8 ms
64 bytes from route-add.net (195.182.210.166): icmp_seq=2 ttl=55 time=46.6 ms
64 bytes from route-add.net (195.182.210.166): icmp_seq=3 ttl=55 time=45.5 ms

--- route-add.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 45.549/46.329/46.829/0.558 ms
[alessandro@ubuntu ~]$
```

The `ping` case 3-4

- According to standards, the process rights are those of the user who launched the exec

Not of the owner of the executable file!

The `ping` case 3-4

- According to standards, the process rights are those of the user who launched the exec

Not of the owner of the executable file!

- Unless the suid bit is set, that is.

```
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
```

The `ping` case 3-4

- According to standards, the process rights are those of the user who launched the exec

Not of the owner of the executable file!

- Unless the suid bit is set, that is.

```
-rwsr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
```

The `ping` case 4-4

- Let's “sabotage” `ping`:

```
[root@ubuntu ~]# chmod u-s /bin/ping
```

The `ping` case 4-4

- Let's “sabotage” `ping`:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]#
```

The `ping` case 4-4

- Let's “sabotage” `ping`:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
```

The `ping` case 4-4

- Let's “sabotage” `ping`:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[root@ubuntu ~]#
```

The aftermath:

```
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
```

The ping case 4-4

- Let's “sabotage” ping:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44178 mag  7 23:51 /bin/ping
[root@ubuntu ~]#
```

The aftermath:

```
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
ping: icmp open socket: Operation not permitted
[alessandro@ubuntu ~]$
```

What LC for `ping`? 1-3

- We want both the power and the safety

What LC for `ping`? 1-3

- We want both the power and the safety
- We want to be able to open ICMP sockets *without* assuming full root privileges!

What LC for `ping`? 1-3

- We want both the power and the safety
- We want to be able to open ICMP sockets *without* assuming full root privileges!
- How do Linux Capabilities help us out?

What LC for `ping`? 1-3

- We want both the power and the safety
- We want to be able to open ICMP sockets *without* assuming full root privileges!
- How do Linux Capabilities help us out?
- RTFM, of course!
  - man capabilities(7)
  - Most up-to-date list & info: `linux/include/uapi/linux/capability.h`

What LC for `ping`? 2-3

- We get many capabilities to choose from:

AUDIT_CONTROL	AUDIT_WRITE	BLOCK_SUSPEND	CHOWN	DAC_OVERRIDE	DAC_READ_SEARCHCAP_LEASE
FOWNER	FSETID	IPC_LOCK	IPC_OWNER	KILL	LEASE
LINUX_IMMUTABLE	MAC_ADMIN	MAC_OVERRIDE	MKNOD	NET_ADMIN	NET_BIND_SERVICE
NET_RAW	SETGID	SETFCAP	SETPCAP	SETUID	SYS_ADMIN
SYS_BOOT	SYS_CHROOT	SYS_MODULE	SYS_NICE	SYS_PACCT	SYS_PTRACE
SYS_RAWIO	SYS_RESOURCE	SYS_TIME	SYS_TTY_CONFIG	SYSLOG	WAKE_ALARM

What LC for `ping`? 2-3

- We get many capabilities to choose from:

AUDIT_CONTROL	AUDIT_WRITE	BLOCK_SUSPEND	CHOWN	DAC_OVERRIDE	DAC_READ_SEARCHCAP_LEASE
FOWNER	FSETID	IPC_LOCK	IPC_OWNER	KILL	LEASE
LINUX_IMMUTABLE	MAC_ADMIN	MAC_OVERRIDE	MKNOD	NET_ADMIN	NET_BIND_SERVICE
NET_RAW	SETGID	SETFCAP	SETPCAP	SETUID	SYS_ADMIN
SYS_BOOT	SYS_CHROOT	SYS_MODULE	SYS_NICE	SYS_PACCT	SYS_PTRACE
SYS_RAWIO	SYS_RESOURCE	SYS_TIME	SYS_TTY_CONFIG	SYSLOG	WAKE_ALARM

What LC for `ping`? 3-3

Man page says:

```
CAP_NET_RAW
   * use RAW and PACKET sockets;
   * bind to any address for transparent proxying.
```

Sounds like what we need.
How shall we apply it?

Managing LCs 1-4

Capabilities can be handled whith commands:

- `getcap`, display file capabilities
- `setcap`, set file capabilities
- `getpcaps`, display process capabilities
- `capsh`, capability shell wrapper

Managing LCs 1-4

Capabilities can be handled whith commands:

- `getcap`, display file capabilities
- `setcap`, set file capabilities
- `getpcaps`, display process capabilities
- `capsh`, capability shell wrapper

Each capability can have these sets associated:

- **`p`**, process is ***p***ermitted the capability
- **`e`**, capability is marked as ***e***ffective
- **`i`**, capability is ***i***nherited after an execve()

Managing LCs 1-4

Capabilities can be handled whith commands:

- `getcap`, display file capabilities
- `setcap`, set file capabilities
- `getpcaps`, display process capabilities
- `capsh`, capability shell wrapper

Each capability can have these sets associated:

- **p**, process is ***p***ermitted the capability
- **e**, capability is marked as ***e***ffective
- **i**, capability is ***i***nherited after an execve()

This has become a bit

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]#
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]#
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]#
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]# setcap CAP_NET_RAW=ep /bin/ping
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]# setcap CAP_NET_RAW=ep /bin/ping
```

Case-insensitive

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]# setcap CAP_NET_RAW=ep /bin/ping
```

Case-insensitive

Case sensitive!

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]# setcap CAP_NET_RAW=ep /bin/ping
[root@ubuntu ~]#
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]# setcap CAP_NET_RAW=ep /bin/ping
[root@ubuntu ~]# getcap /bin/ping
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]# setcap CAP_NET_RAW=ep /bin/ping
[root@ubuntu ~]# getcap /bin/ping
/bin/ping = cap_net_raw+ep
[root@ubuntu ~]#
```

Managing LCs 2-4

Let's start using them:

```
[root@ubuntu ~]# chmod u-s /bin/ping
[root@ubuntu ~]# ll /bin/ping
-rwxr-xr-x 1 root root 44168 mag  7 23:51 /bin/ping*
[root@ubuntu ~]# getcap /bin/ping
[root@ubuntu ~]# setcap CAP_NET_RAW=ep /bin/ping
[root@ubuntu ~]# getcap /bin/ping
/bin/ping = cap_net_raw+ep
[root@ubuntu ~]#
```

Can use this notation with setcap, too

Managing LCs 3-4

Does it work?

```
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
```

Managing LCs 3-4

Does it work?

```
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=55 time=30.7 ms
```

Managing LCs 3-4

Does it work?

```
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=55 time=30.7 ms
64 bytes from route-add.net (195.182.210.166): icmp_seq=2 ttl=55 time=30.9 ms
```

Managing LCs 3-4

Does it work?

```
[alessandro@ubuntu ~]$ ping -c 3 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=55 time=30.7 ms
64 bytes from route-add.net (195.182.210.166): icmp_seq=2 ttl=55 time=30.9 ms
64 bytes from route-add.net (195.182.210.166): icmp_seq=3 ttl=55 time=31.3 ms

--- route-add.net ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 30.709/30.988/31.350/0.304 ms
[alessandro@ubuntu ~]$
```

Managing LCs 4-4

Just a note: `ping` capabilities in Fedora 20:

```
[root@fedora ~]# getcap /bin/ping
```

Managing LCs 4-4

Just a note: `ping` capabilities in Fedora 20:

```
[root@fedora ~]# getcap /bin/ping
/bin/ping = cap_net_admin,cap_net_raw+ep
[root@fedora ~]#
```

Managing LCs 4-4

Just a note: `ping` capabilities in Fedora 20:

```
[root@fedora ~]# getcap /bin/ping
/bin/ping = cap_net_admin,cap_net_raw+ep
[root@fedora ~]#
```

CAP_NET_ADMIN is for packet tagging:

```
[alessandro@fedora ~]$ ping -c1 -m 123 route-add.net
```

Managing LCs 4-4

Just a note: `ping` capabilities in Fedora 20:

```
[root@fedora ~]# getcap /bin/ping
/bin/ping = cap_net_admin,cap_net_raw+ep
[root@fedora ~]#
```

CAP_NET_ADMIN is for packet tagging:

```
[alessandro@fedora ~]$ ping -c1 -m 123 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
Warning: Failed to set mark 123
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=53 time=45.8 ms

--- route-add.net ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 45.802/45.802/45.802/0.000 ms
[alessandro@fedora ~]$
```

Managing LCs 4-4

Just a note: `ping` capabilities in Fedora 20:

```
[root@fedora ~]# getcap /bin/ping
/bin/ping = cap_net_admin,cap_net_raw+ep
[root@fedora ~]#
```

CAP_NET_ADMIN is for packet tagging:

```
[alessandro@fedora ~]$ ping -c1 -m 123 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.
Warning: Failed to set mark 123
64 bytes from route-add.net (195.182.210.166): icmp_seq=1 ttl=53 time=45.8 ms

--- route-add.net ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 45.802/45.802/45.802/0.000 ms
[alessandro@fedora ~]$
```

More examples 1-10

Everyone knows of extended POSIX attributes, right?

More examples 1-10

Everyone knows of extended POSIX attributes, right?

Right?

More examples 1-10

Everyone knows of extended POSIX attributes, right?

Right?

If you don't, you ought to.

More examples 1-10

Everyone knows of extended POSIX attributes, right?

Right?

If you don't, you ought to.

- LC are useful, even necessary to make use of EPA.

More examples 2-10

EPAs let you, among the rest, set files that:

- `-a`: can only have data be appended to
- `-i`: are immutable
- `-S`: do synchronous writes on the media
- `-D`: synchronous writes on directories

Figuring out what filesystem supports which EPA and with not is a matter of guesswork...

More examples 3-10

There is one catch, tough:

- `chattr` does not work for ordinary users:

```
[alessandro@fedora ~]$ chattr +i .bashrc
```

More examples 3-10

There is one catch, tough:

- `chattr` does not work for ordinary users:

```
[alessandro@fedora ~]$ chattr +i .bashrc
chattr: Operation not permitted while setting flags on
.bashrc
[alessandro@fedora ~]$
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]#
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]# chmod u+s /usr/bin/chattr
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]# chmod u+s /usr/bin/chattr
[root@fedora ~]#
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]# chmod u+s /usr/bin/chattr
[root@fedora ~]# ll /usr/bin/chattr
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]# chmod u+s /usr/bin/chattr
[root@fedora ~]# ll /usr/bin/chattr
-rwsr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]#
```

It would work:

```
[alessandro@fedora ~]$ chattr +i .bashrc
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]# chmod u+s /usr/bin/chattr
[root@fedora ~]# ll /usr/bin/chattr
-rwsr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]#
```

It would work:

```
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]# chmod u+s /usr/bin/chattr
[root@fedora ~]# ll /usr/bin/chattr
-rwsr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]#
```

It would work:

```
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$ lsattr .bashrc
```

More examples 4-10

Classical UNIX solution is to make it SUID root:

```
[root@fedora ~]# ll /usr/bin/chattr
-rwxr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]# chmod u+s /usr/bin/chattr
[root@fedora ~]# ll /usr/bin/chattr
-rwsr-xr-x 1 root root 10528 giu 24 13:10 /usr/bin/chattr
[root@fedora ~]#
```

It would work:

```
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$ lsattr .bashrc
----i---------e-- .bashrc
[alessandro@fedora ~]$
```

More examples 5-10

But...

```
[alessandro@fedora ~]$ lsattr ~games/.profile
```

More examples 5-10

But...

```
[alessandro@fedora ~]$ lsattr ~games/.profile
--------------e-- /opt/games/.profile
[alessandro@fedora ~]$
```

More examples 5-10

But...

```
[alessandro@fedora ~]$ lsattr ~games/.profile
--------------e-- /opt/games/.profile
[alessandro@fedora ~]$ chattr +i ~games/.profile
```

More examples 5-10

But...

```
[alessandro@fedora ~]$ lsattr ~games/.profile
----------e-- /opt/games/.profile
[alessandro@fedora ~]$ chattr +i ~games/.profile
[alessandro@fedora ~]$
```

More examples 5-10

But...

```
[alessandro@fedora ~]$ lsattr ~games/.profile
--------------e-- /opt/games/.profile
[alessandro@fedora ~]$ chattr +i ~games/.profile
[alessandro@fedora ~]$ lsattr ~games/.profile
```

More examples 5-10

But...

```
[alessandro@fedora ~]$ lsattr ~games/.profile
--------------e-- /opt/games/.profile
[alessandro@fedora ~]$ chattr +i ~games/.profile
[alessandro@fedora ~]$ lsattr ~games/.profile
----i---------e-- /usr/games/.profile
[alessandro@fedora ~]$
```

More examples 5-10

But...

```
[alessandro@fedora ~]$ lsattr ~games/.profile
--------------e-- /opt/games/.profile
[alessandro@fedora ~]$ chattr +i ~games/.profile
[alessandro@fedora ~]$ lsattr ~games/.profile
----i---------e-- /usr/games/.profile
[alessandro@fedora ~]$
```

Which is normal for superuser rights.
We definitely do not want this to happen.

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
```

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
[root@fedora ~]#
```

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
[root@fedora ~]# ll /bin/chattr
```

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
[root@fedora ~]# ll /bin/chattr
-rwxr-xr-x 1 root root 10736 Aug  3 11:36 /bin/chattr
[root@fedora ~]#
```

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
[root@fedora ~]# ll /bin/chattr
-rwxr-xr-x 1 root root 10736 Aug  3 11:36 /bin/chattr
[root@fedora ~]# setcap CAP_LINUX_IMMUTABLE=ep /bin/chattr
```

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
[root@fedora ~]# ll /bin/chattr
-rwxr-xr-x 1 root root 10736 Aug  3 11:36 /bin/chattr
[root@fedora ~]# setcap CAP_LINUX_IMMUTABLE=ep /bin/chattr
[root@fedora ~]#
```

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
[root@fedora ~]# ll /bin/chattr
-rwxr-xr-x 1 root root 10736 Aug  3 11:36 /bin/chattr
[root@fedora ~]# setcap CAP_LINUX_IMMUTABLE=ep /bin/chattr
[root@fedora ~]# getcap /bin/chattr
```

More examples 6-10

This is the right way to do it:

```
[root@fedora ~]# chmod u-s /bin/chattr
[root@fedora ~]# ll /bin/chattr
-rwxr-xr-x 1 root root 10736 Aug  3 11:36 /bin/chattr
[root@fedora ~]# setcap CAP_LINUX_IMMUTABLE=ep /bin/chattr
[root@fedora ~]# getcap /bin/chattr
/bin/chattr = cap_linux_immutable+ep
[root@fedora ~]#
```

More examples 7-10

Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
```

More examples 7-10

Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
----------e-- .bashrc
[alessandro@fedora ~]$
```

More examples 7-10

Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
----------e-- .bashrc
[alessandro@fedora ~]$ chattr +i .bashrc
```

More examples 7-10

Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
----------e-- .bashrc
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$
```

More examples 7-10

Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
----------e-- .bashrc
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$ lsattr .bashrc
```

More examples 7-10

Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
-------------e-- .bashrc
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$ lsattr .bashrc
----i--------e-- .bashrc
[alessandro@fedora ~]$
```

More examples 7-10

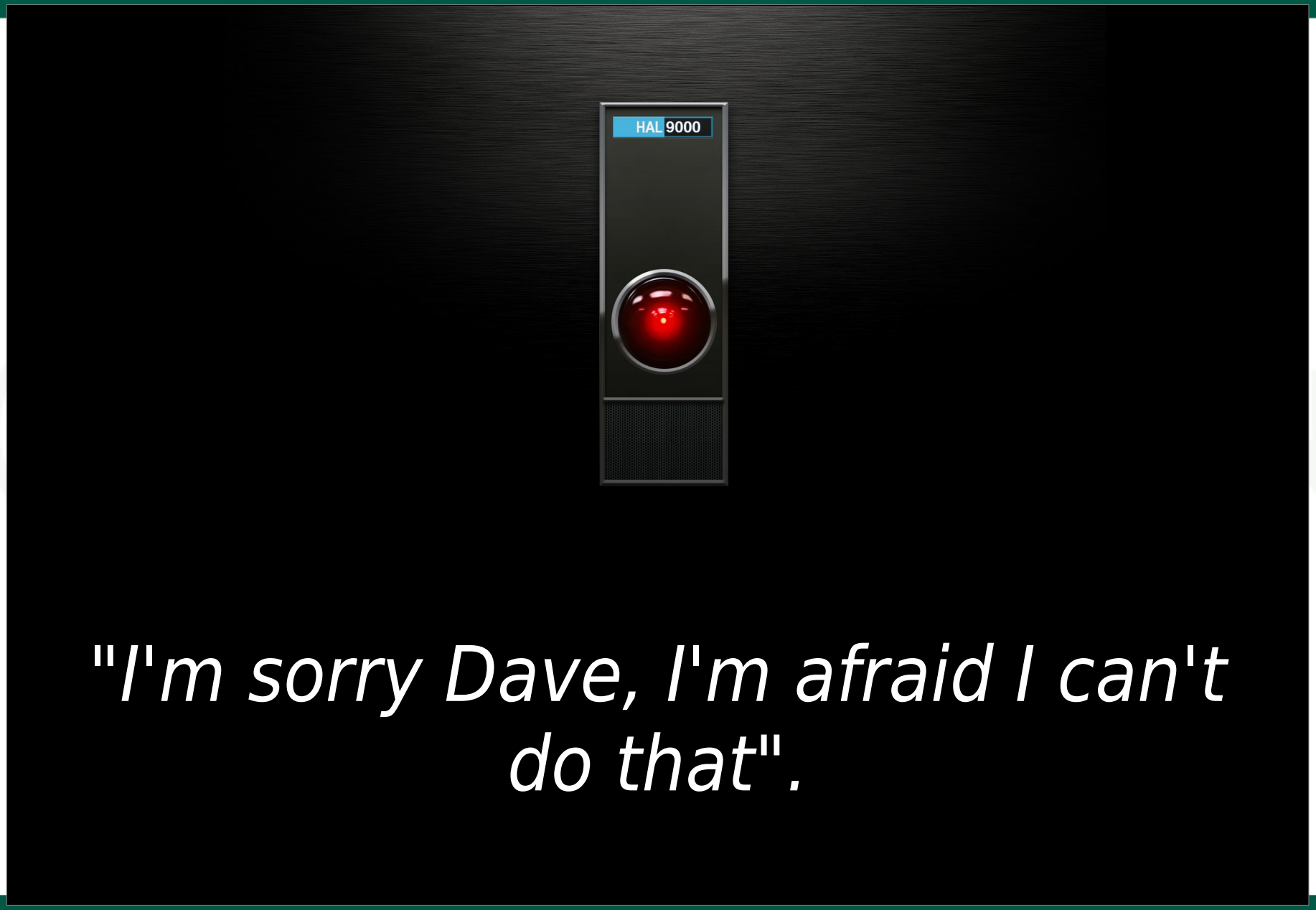
Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
-------------e-- .bashrc
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$ lsattr .bashrc
----i--------e-- .bashrc
[alessandro@fedora ~]$ chattr +i ~games/.profile
```

More examples 7-10

Which causes the right things to happen:

```
[alessandro@fedora ~]$ lsattr .bashrc
-------------e-- .bashrc
[alessandro@fedora ~]$ chattr +i .bashrc
[alessandro@fedora ~]$ lsattr .bashrc
----i--------e-- .bashrc
[alessandro@fedora ~]$ chattr +i ~games/.profile
chattr: Permission denied while setting flags on
/usr/games/.profile
[alessandro@fedora ~]$
```



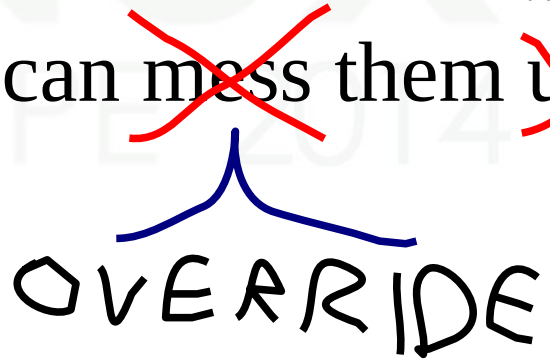
"*I'm sorry Dave, I'm afraid I can't do that*".

More examples 8-10

- DAC = Discretionary Access Control
- It's the classic Unix “who gets to do what” decision mechanism
- In the filesystem it shows as the `rwx` file rights
- Linux Capabilities can mess them up as well! :-)

More examples 8-10

- DAC = Discretionary Access Control
- It's the classic Unix “who gets to do what” decision mechanism
- In the filesystem it shows as the `rwx` file rights
- Linux Capabilities can ~~mess~~ them ~~up~~ as well! :-)



More examples 9-10

Let's beep the beeper!

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
Could not open /dev/tty0 or /dev/vc/0 for writing
open: No such file or directory
[alessandro@fedora ~]$
```

Man page says:

By default beep is not installed with the suid bit set, because that would just be zany. On the other hand, if you do make it suid root, all your problems with beep bailing on ioctl calls will magically vanish, which is pleasant, and the only reason not to is that any suid program is a potential security hole. Conveniently, beep is very short, so auditing it is pretty straightforward.

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
```

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
[root@fedora ~]#
```

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
```

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
ioctl: Operation not permitted
ioctl: Operation not permitted
[alessandro@fedora ~]$
```

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
ioctl: Operation not permitted
ioctl: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap \
> CAP_DAC_OVERRIDE,CAP_SYS_TTY_CONFIG=pe /bin/beep
```

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
ioctl: Operation not permitted
ioctl: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap \
> CAP_DAC_OVERRIDE,CAP_SYS_TTY_CONFIG=pe /bin/beep
[root@fedora ~]#
```

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
ioctl: Operation not permitted
ioctl: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap \
> CAP_DAC_OVERRIDE,CAP_SYS_TTY_CONFIG=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
```

More examples 10-10

Let's try using LC instead:

```
[root@fedora ~]# setcap CAP_DAC_OVERRIDE=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
ioctl: Operation not permitted
ioctl: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap \
> CAP_DAC_OVERRIDE,CAP_SYS_TTY_CONFIG=pe /bin/beep
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ beep -f 2000 -l 100
[alessandro@fedora ~]$
```

"*All right Dave, I think I figured out a way to do that*"

Synergia

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]#
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]#
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

```
[alessandro@ubuntu ~]$ /tmp/bash -l
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

```
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

```
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
```

Synergia

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

```
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `15400': = cap_dac_override,cap_kill+ep
[alessandro@ubuntu ~]$
```

Synergia

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

```
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `15400': = cap_dac_override,cap_kill+ep
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
```

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

```
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `15400': = cap_dac_override,cap_kill+ep
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Synergia

Extracting and Decoding LC info 1-4

What LCs are active now?

```
[root@ubuntu ~]# cp /bin/bash /tmp/
[root@ubuntu ~]# setcap CAP_KILL,CAP_DAC_OVERRIDE+epi /tmp/bash
[root@ubuntu ~]# getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[root@ubuntu ~]#
```

Inheritance is set

```
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `15400': = cap_dac_override,cap_kill+ep
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Inheritance is missing!

Extracting and Decoding LC info 2-4

Decoding Hex LC bitmap:

```
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
```

Extracting and Decoding LC info 2-4

Decoding Hex LC bitmap:

```
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Extracting and Decoding LC info 2-4

Decoding Hex LC bitmap:

```
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --decode=0000000000000000
```

Extracting and Decoding LC info 2-4

Decoding Hex LC bitmap:

```
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --decode=0000000000000000
0x0000000000000000=
[alessandro@ubuntu ~]$
```

Extracting and Decoding LC info 2-4

Decoding Hex LC bitmap:

```
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --decode=0000000000000000
0x0000000000000000=
[alessandro@ubuntu ~]$ capsh --decode=0000000000000022
```

Extracting and Decoding LC info 2-4

Decoding Hex LC bitmap:

```
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --decode=0000000000000000
0x0000000000000000=
[alessandro@ubuntu ~]$ capsh --decode=0000000000000022
0x0000000000000022=cap_dac_override,cap_kill
[alessandro@ubuntu ~]$
```

Extracting and Decoding LC info 2-4

Decoding Hex LC bitmap:

```
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000022
CapEff: 0000000000000022
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --decode=0000000000000000
0x0000000000000000=
[alessandro@ubuntu ~]$ capsh --decode=0000000000000022
0x0000000000000022=cap_dac_override,cap_kill
[alessandro@ubuntu ~]$
```

Still represented as a set

Extracting and Decoding LC info 3-4

`0000001fffffffff` = Capability Bounding Set

1 = allowable cap (kept if present)

0 = masked cap (or no LC associated to bit)

Bits are mapped one-to-one to a LC

`0000001fffffffff` is a 64-bit mask

- It's been a 64 bit mask since libcap vers. 2.03
- Before it was 32-bit

Extracting and Decoding LC info 4-4

Number of bit-mapped capabilities is consistent with zero-based value stored in `/proc/sys/kernel/cap_last_cap`

Kernel 3.14.19: `1fffffffff` → 1+9*4=37
`/proc/sys/kernel/cap_last_cap` = 36
Kernel 3.16.3: `3fffffffff` → 2+9*4=38
`/proc/sys/kernel/cap_last_cap` = 37

System Configuration 1-2

`capability.conf`[1]: users who inherit LCs

```
[root@ubuntu ~]# cat /etc/security/capability.conf
```

System Configuration 1-2

`capability.conf`[1]: users who inherit LCs

```
[root@ubuntu ~]# cat /etc/security/capability.conf
cap_kill        alessandro
none   *
[root@ubuntu ~]#
```

System Configuration 1-2

`capability.conf`[1]: users who inherit LCs

```
[root@ubuntu ~]# cat /etc/security/capability.conf
cap_kill        alessandro
none   *
[root@ubuntu ~]#
```

If missing all users get all available capabilities!

System Configuration 1-2

`capability.conf`[1]: users who inherit LCs

```
[root@ubuntu ~]# cat /etc/security/capability.conf
cap_kill        alessandro
none   *
[root@ubuntu ~]#
```

Changes are effective after a new login:

```
Ubuntu 14.04.1 LTS ubuntu tty1

ubuntu login: alessandro
Password:
[alessandro@ubuntu ~]$
```

System Configuration 1-2

`capability.conf`[1]: users who inherit LCs

```
[root@ubuntu ~]# cat /etc/security/capability.conf
cap_kill        alessandro
none   *
[root@ubuntu ~]#
```

Changes are effective after a new login:

```
Ubuntu 14.04.1 LTS ubuntu tty1

ubuntu login: alessandro
Password:
[alessandro@ubuntu ~]$ getpcaps $$
```

System Configuration 1-2

`capability.conf`[1]: users who inherit LCs

```
[root@ubuntu ~]# cat /etc/security/capability.conf
cap_kill        alessandro
none   *
[root@ubuntu ~]#
```

Changes are effective after a new login:

```
Ubuntu 14.04.1 LTS ubuntu tty1

ubuntu login: alessandro
Password:
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `8586': = cap_kill+i
[alessandro@ubuntu ~]$
```

System Configuration 1-2

`capability.conf`[1]: users who inherit LCs

```
[root@ubuntu ~]# cat /etc/security/capability.conf
cap_kill        alessandro
none  *
[root@ubuntu ~]#
```

Changes are effective after a new login:

```
Ubuntu 14.04.1 LTS ubuntu tty1

ubuntu login: alessandro
Password:
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `8586': = cap_kill+i
[alessandro@ubuntu ~]$
```

CAP_KILL is now inherited

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$ getcap /tmp/bash
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$ getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[alessandro@ubuntu ~]$
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$ getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[alessandro@ubuntu ~]$ /tmp/bash -l
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$ getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$ getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$ getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16849': = cap_kill+eip cap_dac_override+ep
[alessandro@ubuntu ~]$
```

System Configuration 2-2

Child process of new shell now does inherit LCs:

```
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16332': = cap_kill+i
[alessandro@ubuntu ~]$ getcap /tmp/bash
/tmp/bash = cap_dac_override,cap_kill+eip
[alessandro@ubuntu ~]$ /tmp/bash -l
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `16849': = cap_kill+eip cap_dac_override+ep
[alessandro@ubuntu ~]$
```

Cap is in config file

Cap is not in config file

1) Ubuntu package libpam-cap is required. Fedora has package libcap, but this setup does not work (`PAM failure: System error`)

Capability assignment 1-10

System-wide bounding-set Per-thread bounding-set

- Before 2.6.25 Bounding Set was system-wide
- Beginning 2.6.25 it's per-thread
- Can be dropped by process with CAP_SETPCAP via prctl(2) PR_CAPBSET_DROP operation
- Inherited at fork(2), preserved on execve(2)
- Optional until 2.6.33, then built-in

Capability assignment 2-10

N	**N**ew	P	**P**revious	F	**C**apab. set to the **F**ile
e	**E**ffective capability	p	**P**ermitted capability	i	**I**nherited capability
CBS	**C**apability **B**ounding **S**et[1]				

New capabilities ***used*** to be computed this way:

N^p	$(P^i \cup F^i) \cap (F^p \cup CBS)$
N^e	$F^e \cup N^p$
N^i	P^i

1) Mask of permitted capabilities retained after an execve(2). No effect on inherited caps.
From kernel 2.6.25, limits inherited capabilities that can be added to self even if they are permitted.

Capability assignment 3-10

N	**N**ew	P	**P**revious	F	**C**apab. set to the **F**ile
e	**E**ffective capability	p	**P**ermitted capability	i	**I**nherited capability
CBS	**C**apability **B**ounding **S**et[1]				

Now they are computed this way:

N^p	$(P^i \cup F^i) \cap (F^p \cup CBS)$
N^e	$F^e \ ? \ N^p : 0$[2]
N^i	P^i

2) Not and AND any more, because Effective set is a bit, no longer a set (one bit per capability) as it used to be.

Capability assignment 4-10

What does the Capability Bounding Set do?

1) On an execve(2), it ANDs out thread's permitted caps from file's permitted caps.
2) Limits thread's inheritable caps that can be added with a capset(2) (kernel >= 2.6.25).
   a) Filtered capability can still be permitted
   b) Inheritable cap can still be set if it's in file's inheritable set.

Capability assignment 5-10

Possibile security risk: undesired permitted caps.

Capability assignment 5-10

Possibile security risk: undesired permitted caps.
Consider the situation:

1) Parent process has CAP_X Inh, not in CBS

Capability assignment 5-10

Possibile security risk: undesired permitted caps.
Consider the situation:

1) Parent process has CAP_X Inh, not in CBS
2) CAP_X cannot be in Perm set

Capability assignment 5-10

Possibile security risk: undesired permitted caps.
Consider the situation:

1) Parent process has CAP_X Inh, not in CBS
2) CAP_X cannot be in Perm set
    a) But it is retained in Inh set by children

Capability assignment 5-10

Possibile security risk: undesired permitted caps.
Consider the situation:

1) Parent process has CAP_X Inh, not in CBS
2) CAP_X cannot be in Perm set
   a) But it is retained in Inh set by children
3) Process can still exec child with CAP_X in both Perm and Inh sets!

Capability assignment 5-10

Possibile security risk: undesired permitted caps.
Consider the situation:

1) Parent process has CAP_X Inh, not in CBS
2) CAP_X cannot be in Perm set
   a) But it is retained in Inh set by children
3) Process can still exec child with CAP_X in both Perm and Inh sets!

Dropping CBS can instill a false sense of security!
Alone it's not sufficient!

Capability assignment 6-10

Like SUID, shell scripts' LC are not honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test_cap.sh
```

Capability assignment 6-10

Like SUID, shell scripts' LC are not honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test_cap.sh
#!/bin/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$
```

Capability assignment 6-10

Like SUID, shell scripts' LC are not honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test_cap.sh
#!/bin/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test_cap.sh
```

Capability assignment 6-10

Like SUID, shell scripts' LC are not honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test_cap.sh
#!/bin/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test_cap.sh
/home/alessandro/bin/test_cap.sh = cap_linux_immutable+ep
[alessandro@ubuntu ~]$
```

Capability assignment 6-10

Like SUID, shell scripts' LC are not honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test_cap.sh
#!/bin/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test_cap.sh
/home/alessandro/bin/test_cap.sh = cap_linux_immutable+ep
[alessandro@ubuntu ~]$ test_cap.sh
```

Capability assignment 6-10

Like SUID, shell scripts' LC are not honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test_cap.sh
#!/bin/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test_cap.sh
/home/alessandro/bin/test_cap.sh = cap_linux_immutable+ep
[alessandro@ubuntu ~]$ test_cap.sh
Capabilities for `9093': =
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Capability assignment 6-10

Like SUID, shell scripts' LC are not honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test_cap.sh
#!/bin/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test_cap.sh
/home/alessandro/bin/test_cap.sh = cap_linux_immutable+ep
[alessandro@ubuntu ~]$ test_cap.sh
Capabilities for `9093': =
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Not honored

Not honored

Capability assignment 7-10

Only binary executables' LC are honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test2_cap.sh
```

Capability assignment 7-10

Only binary executables' LC are honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test2_cap.sh
#!/tmp/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$
```

Capability assignment 7-10

Only binary executables' LC are honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test2_cap.sh
#!/tmp/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test2_cap.sh /tmp/dash
```

Capability assignment 7-10

Only binary executables' LC are honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test2_cap.sh
#!/tmp/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test2_cap.sh /tmp/dash
/tmp/dash = cap_linux_immutable+ep
[alessandro@ubuntu ~]$
```

Capability assignment 7-10

Only binary executables' LC are honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test2_cap.sh
#!/tmp/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test2_cap.sh /tmp/dash
/tmp/dash = cap_linux_immutable+ep
[alessandro@ubuntu ~]$ test2_cap.sh
```

Capability assignment 7-10

Only binary executables' LC are honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test2_cap.sh
#!/tmp/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test2_cap.sh /tmp/dash
/tmp/dash = cap_linux_immutable+ep
[alessandro@ubuntu ~]$ test2_cap.sh
Capabilities for `9218': = cap_linux_immutable+ep
CapInh: 0000000000000000
CapPrm: 0000000000000200
CapEff: 0000000000000200
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Capability assignment 7-10

Only binary executables' LC are honored:

```
[alessandro@ubuntu ~]$ cat ~/bin/test2_cap.sh
#!/tmp/dash
getpcaps $$
grep ^Cap /proc/$$/status
[alessandro@ubuntu ~]$ getcap ~/bin/test2_cap.sh /tmp/dash
/tmp/dash = cap_linux_immutable+ep
[alessandro@ubuntu ~]$ test2_cap.sh
Capabilities for `9218': = cap_linux_immutable+ep
CapInh: 0000000000000000
CapPrm: 0000000000000200
CapEff: 0000000000000200
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Now honored

Now honored

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
```

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$
```

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
```

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$
```

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
```

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$
```

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$
```

New shell

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
```

Synergia

Capability assignment 8-10

Command `capsh` can reduce shell CBS:

```
[alessandro@ubuntu ~]$ getcap $(which capsh)
/sbin/capsh = cap_setpcap+p
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffffff
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffdfff
[alessandro@ubuntu ~]$
```

Capability assignment 9-10

CBS reduction with `capsh`. Initial state:

```
[alessandro@ubuntu ~]$ getcap $(which capsh ping)
```

Capability assignment 9-10

CBS reduction with `capsh`. Initial state:

```
[alessandro@ubuntu ~]$ getcap $(which capsh ping)
/sbin/capsh = cap_setpcap+p
/bin/ping = cap_net_raw+p
[alessandro@ubuntu ~]$
```

Capability assignment 9-10

CBS reduction with `capsh`. Initial state:

```
[alessandro@ubuntu ~]$ getcap $(which capsh ping)
/sbin/capsh = cap_setpcap+p
/bin/ping = cap_net_raw+p
[alessandro@ubuntu ~]$ getpcaps $$
```

Capability assignment 9-10

CBS reduction with `capsh`. Initial state:

```
[alessandro@ubuntu ~]$ getcap $(which capsh ping)
/sbin/capsh = cap_setpcap+p
/bin/ping = cap_net_raw+p
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `7842': =
[alessandro@ubuntu ~]$
```

Capability assignment 9-10

CBS reduction with `capsh`. Initial state:

```
[alessandro@ubuntu ~]$ getcap $(which capsh ping)
/sbin/capsh = cap_setpcap+p
/bin/ping = cap_net_raw+p
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `7842': =
[alessandro@ubuntu ~]$ ping -qc1 route-add.net
```

Capability assignment 9-10

CBS reduction with `capsh`. Initial state:

```
[alessandro@ubuntu ~]$ getcap $(which capsh ping)
/sbin/capsh = cap_setpcap+p
/bin/ping = cap_net_raw+p
[alessandro@ubuntu ~]$ getpcaps $$
Capabilities for `7842': =
[alessandro@ubuntu ~]$ ping -qc1 route-add.net
PING route-add.net (195.182.210.166) 56(84) bytes of data.

--- route-add.net ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 27.718/27.718/27.718/0.000 ms
[alessandro@ubuntu ~]$
```

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
```

Synergia

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$
```

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
```

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffdfff
[alessandro@ubuntu ~]$
```

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffdfff
[alessandro@ubuntu ~]$ capsh --decode=2000
```

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffdfff
[alessandro@ubuntu ~]$ capsh --decode=2000
0x0000000000002000=cap_net_raw
[alessandro@ubuntu ~]$
```

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffdfff
[alessandro@ubuntu ~]$ capsh --decode=2000
0x0000000000002000=cap_net_raw
[alessandro@ubuntu ~]$ ping -qc1 route-add.net
```

Capability assignment 10-10

Running new shell with lowered CBS via `capsh`:

```
[alessandro@ubuntu ~]$ capsh --drop=cap_net_raw --
[alessandro@ubuntu ~]$ grep ^Cap /proc/$$/status
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000001fffffdfff
[alessandro@ubuntu ~]$ capsh --decode=2000
0x0000000000002000=cap_net_raw
[alessandro@ubuntu ~]$ ping -qc1 route-add.net
ping: icmp open socket: Operation not permitted
[alessandro@ubuntu ~]$
```

Distro status 1-7

The plan is to replace as many SUID/SGID executables with capabilities as possible:

- Fedora: Last update: 2011-04-05, completion: 100% (many, but not all, SUID taken off)
- Ubuntu: WIP (Last update: 2011-09-27)

Not everything SUID could be ported to Linux Capabilities *(yet)*

Distro status 2-7

Fedora 20 (Heisenbug): a couple dozen suid files (in a non-standard LXDE install)

```
# find / -xdev -type f -perm /111 -perm /4000 -user root | wc -l
23
#
```

/usr/sbin/mount.nfs	/usr/bin/mount	/usr/bin/chsh	/usr/bin/write
/usr/bin/fusermount	/usr/bin/umount	/usr/bin/chage	/usr/bin/passwd
/usr/bin/newgrp	/usr/bin/locate	/usr/bin/su	/usr/bin/sudo
/usr/bin/gpasswd	/usr/bin/cgexec	/usr/bin/chfn	/usr/bin/Xorg
/usr/bin/ssh-agent	/usr/bin/crontab	/usr/bin/at	/usr/bin/ksu
/usr/sbin/postqueue	/usr/sbin/postdrop	/usr/sbin/ssmtp	/usr/bin/pkexec

Distro status 3-7

Ubuntu 14.04.1 (Trusty Tahr): a few more (also in a non-standard, XFCE install)

```
# find / /usr -xdev -type f -perm /111 -perm /4000 -user root | wc -l
29
#
```

They used to be 34 in 13.10 (Saucy Salamander)

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Let's strip off the SUID bit and any cap:

```
[root@fedora ~]# chmod u-s /usr/bin/at
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Let's strip off the SUID bit and any cap:

```
[root@fedora ~]# chmod u-s /usr/bin/at
[root@fedora ~]#
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Let's strip off the SUID bit and any cap:

```
[root@fedora ~]# chmod u-s /usr/bin/at
[root@fedora ~]# setcap -r /usr/bin/at
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Let's strip off the SUID bit and any cap:

```
[root@fedora ~]# chmod u-s /usr/bin/at
[root@fedora ~]# setcap -r /usr/bin/at
[root@fedora ~]#
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Let's strip off the SUID bit and any cap:

```
[root@fedora ~]# chmod u-s /usr/bin/at
[root@fedora ~]# setcap -r /usr/bin/at
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Let's strip off the SUID bit and any cap:

```
[root@fedora ~]# chmod u-s /usr/bin/at
[root@fedora ~]# setcap -r /usr/bin/at
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set egid: Operation not permitted
[alessandro@fedora ~]$
```

Distro status 4-7

Let's see one case where LC do not work:

```
[alessandro@fedora ~]$ ll /usr/bin/at
-rwsr-xr-x 1 root root 53208  5 dic 12.34 /usr/bin/at
[alessandro@fedora ~]$ getcap /usr/bin/at
[alessandro@fedora ~]$
```

Let's strip off the SUID bit and any cap:

```
[root@fedora ~]# chmod u-s /usr/bin/at
[root@fedora ~]# setcap -r /usr/bin/at
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set egid: Operation not permitted
[alessandro@fedora ~]$
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set euid: Operation not permitted
[alessandro@fedora ~]$
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set euid: Operation not permitted
[alessandro@fedora ~]$
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set euid: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap CAP_SETUID,CAP_SETGID+ep /usr/bin/at
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set euid: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap CAP_SETUID,CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set euid: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap CAP_SETUID,CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set euid: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap CAP_SETUID,CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
PAM failure: System error
[alessandro@fedora ~]$
```

Distro status 5-7

Let's try to make it happy:

```
[root@fedora ~]# setcap CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

No joy:

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
cannot set euid: Operation not permitted
[alessandro@fedora ~]$
```

```
[root@fedora ~]# setcap CAP_SETUID,CAP_SETGID+ep /usr/bin/at
[root@fedora ~]#
```

```
[alessandro@fedora ~]$ at -f ~/bin/at.sh now+5min
PAM failure: System error
[alessandro@fedora ~]$ :-(
```

Distro status 6-7

Log says:

```
[alessandro@fedora ~]$ tail -1 /var/log/secure
```

Distro status 6-7

Log says:

```
[alessandro@fedora ~]$ tail -1 /var/log/secure
Feb 23 10:08:43 fedora at: PAM audit_log_acct_message()
failed: Operation not permitted
[alessandro@fedora ~]$
```

Distro status 6-7

Log says:

```
[alessandro@fedora ~]$ tail -1 /var/log/secure
Feb 23 10:08:43 fedora at: PAM audit_log_acct_message()
failed: Operation not permitted
[alessandro@fedora ~]$
```

Game over, for now.

Distro status 7-7

Ubuntu has taken a different approach:

- it uses the same `at`/`atd` implementation (by Thomas Koenig, ig25@rz.uni-karlsruhe.de)
- it runs the daemon as user daemon
- the client command is SUID and SGUI daemon:daemon, allowing it to:
  - write into /var/spool/cron/atjobs/.SEQ
  - send atd signals

Old school, simple, effective, safe.

Issues? 1-2

«Why are not more people aware of and using capabilities? I believe that poor documentation is the primary reason. For example, Fedora 10 is missing the man pages for `getpcaps`, `capsh` *and* `pam_cap`, *and the Fedora Security Guide does not even mention capabilities»*

(Finnbarr P. Murphy, May 28th, 2009)

Every bit as true today!

Issues? 2-2

- Online distro docs stopped years ago
- Fedora Security Guide in same condition in 19.1, and is absent in 20 documentation
  - At least man pages are there
  - SELinux only future?
- Behaviour of LC changing over time
  - Same commands, capabilities and sets yield different result from 2.6.24 with no errors
- libcap-ng should make programming easier

Aknowledgments 1-1

- The Linux man-pages Project people (https://www.kernel.org/doc/man-pages/)
- Finnbarr P. Murphy (blog post and references, useful though outdated)
- And kernel developers, of course!
  - Andrew G. Morgan <morgan@kernel.org>
  - Alexander Kjeldaas <astor@guardian.no> with help from Aleph1, Roland Buresund and Andrew Main.