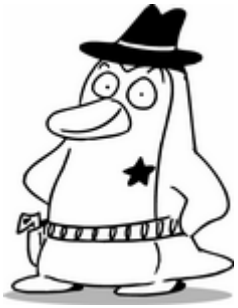
TOMOYO Linux

- The latest version of this presentation material will be available at
  - http://sourceforge.jp/projects/tomoyo/document/elc2008.pdf
- TOMOYO Linux Website
  - http://tomoyo.sourceforge.jp/ (English/Japanese)
  - http://elinux.org/TomoyoLinux (English)

How to analyze your Linux's behavior with TOMOYO Linux

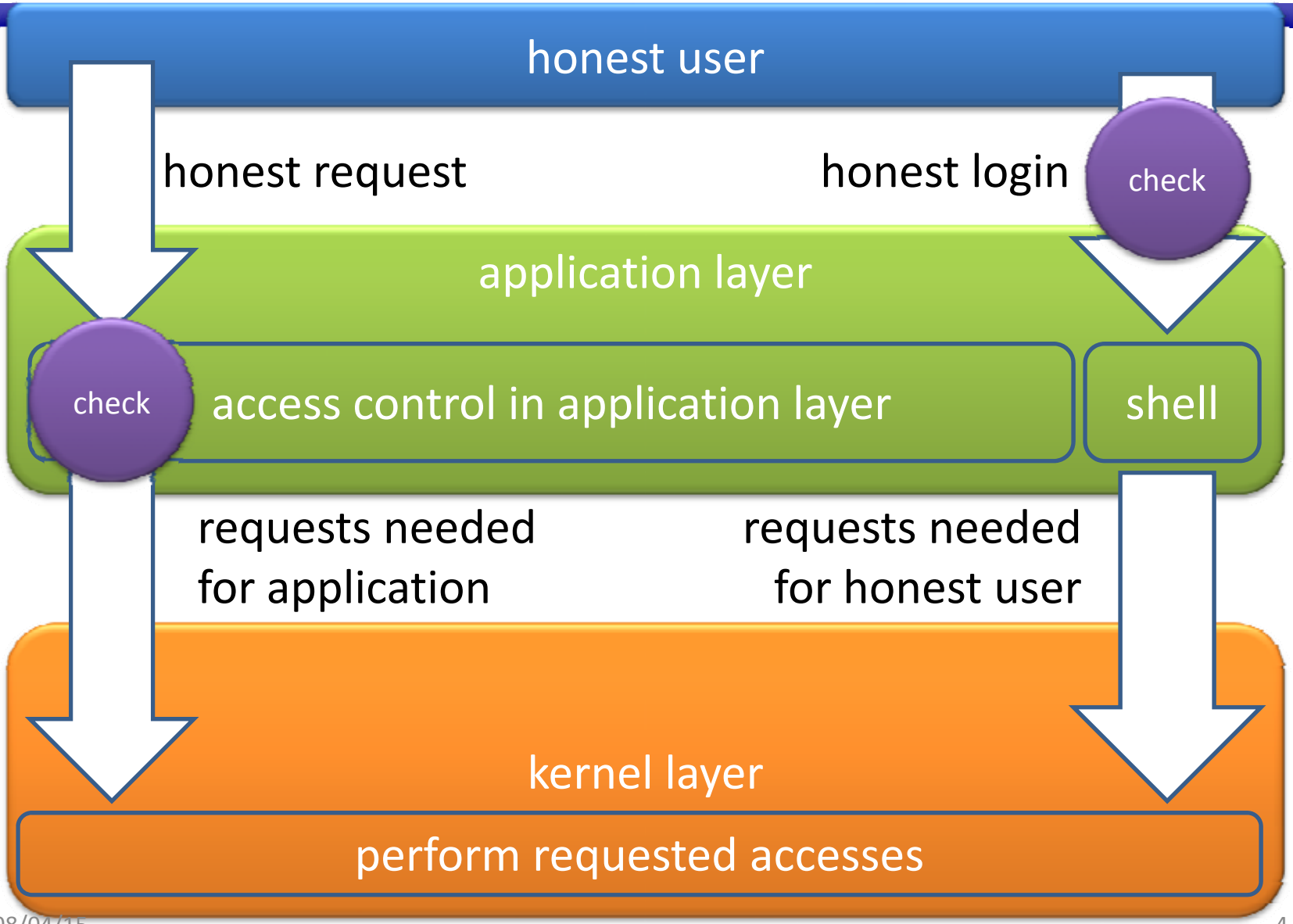


Kentaro Takeda
takedakn@nttdata.co.jp
NTT DATA CORPORATION
http://tomoyo.sourceforge.jp/

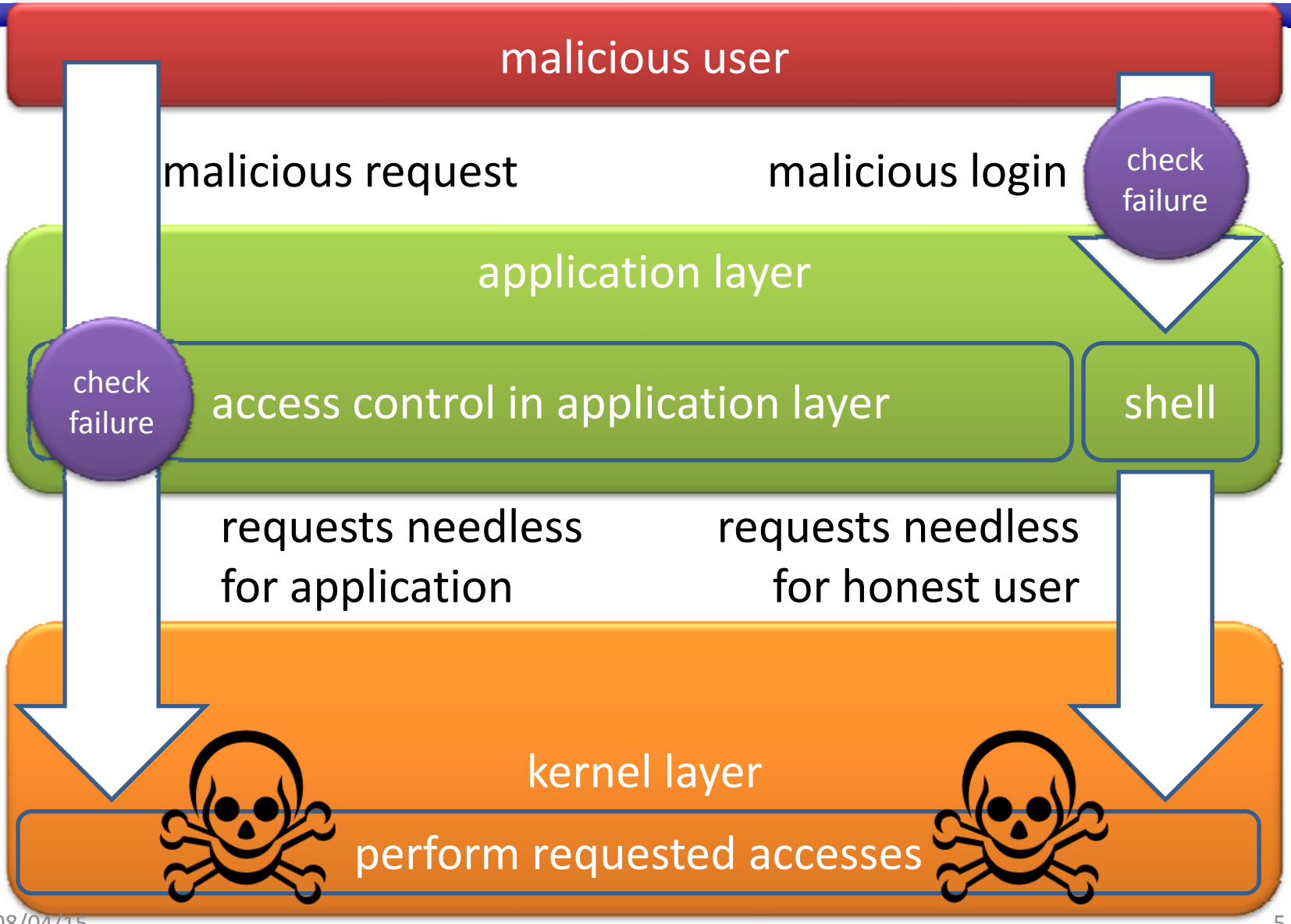
Hello, TOMOYO Linux!

- This is the second presentation of TOMOYO Linux in ELC.
  - ELC2007 is the first international presentation for us.
- TOMOYO Linux is our work in security for Linux kernel.
  - Contrary to its name, TOMOYO Linux is NOT a distribution.
- TOMOYO Linux is...
  - “manageable and understandable” MAC (Mandatory Access Control).
  - very suitable for embedded systems.
  - usable for understanding your Linux’s behavior.

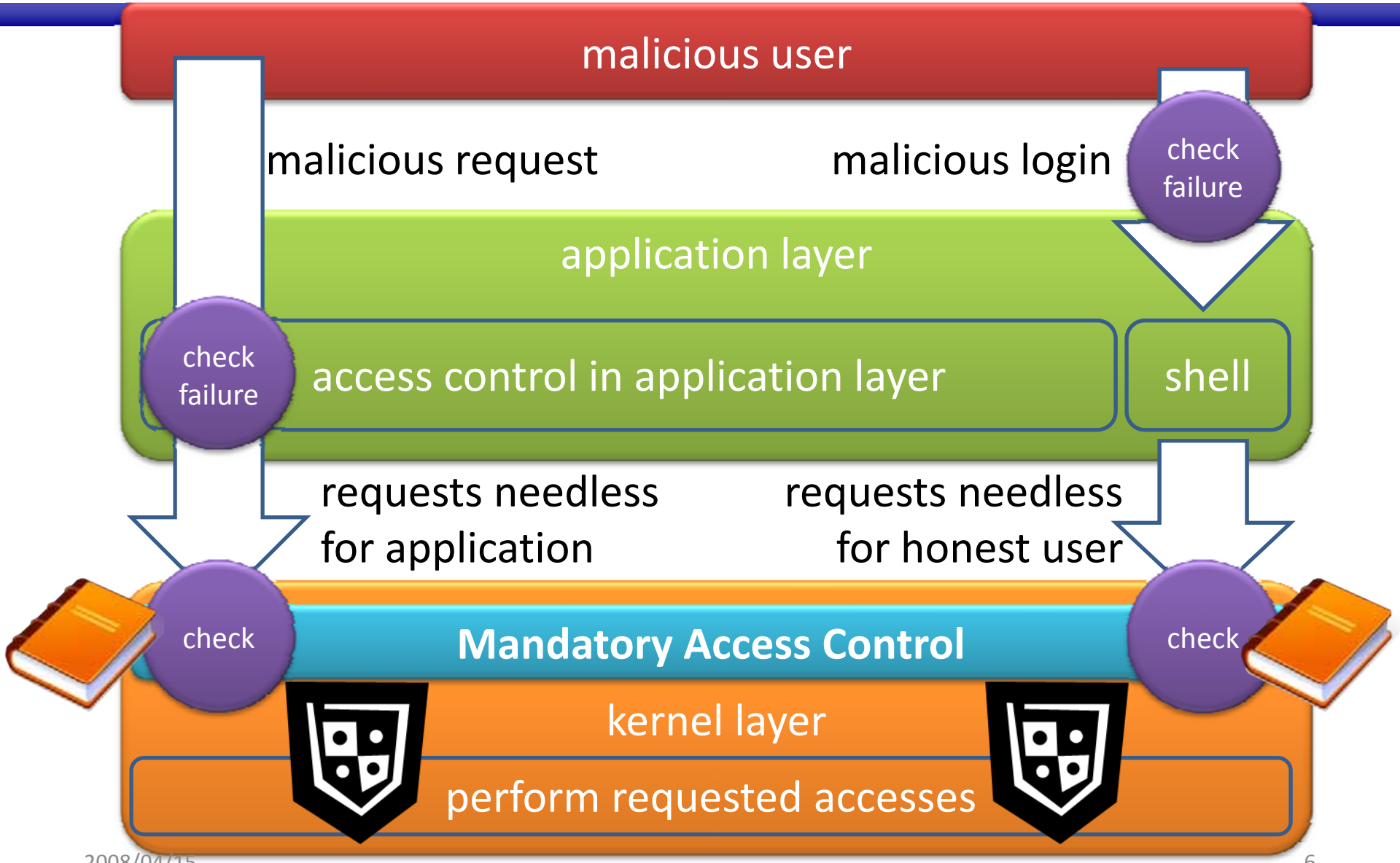
Normal Linux



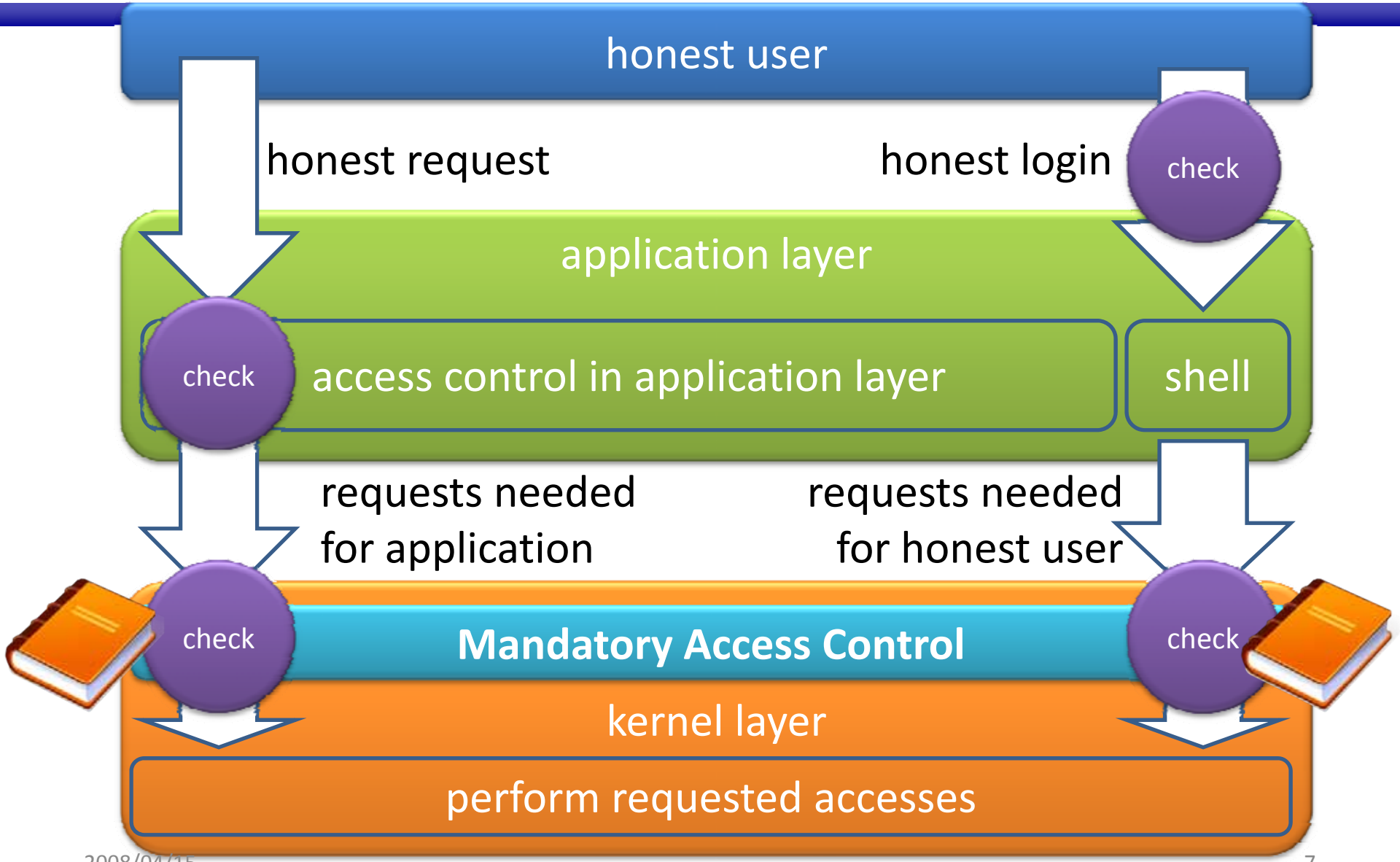
Normal Linux



Linux with MAC



Linux with MAC



Policy

- The most significant configuration of MAC.
  - Also in SELinux, AppArmor, Smack…
- You must write all accesses needed by honest user/application in policy.
  - Accesses not in policy are denied.

Understanding and Protecting

- If you understand your Linux's behavior deeply (in kernel level), you can protect it deeply
  - because you can permit only requests happening in ordinary behavior.
  - "Understanding is protecting."

- "Then, how can I understand my Linux's behavior?"
  - TOMOYO Linux is just for you!

TOMOYO Linux's Policy

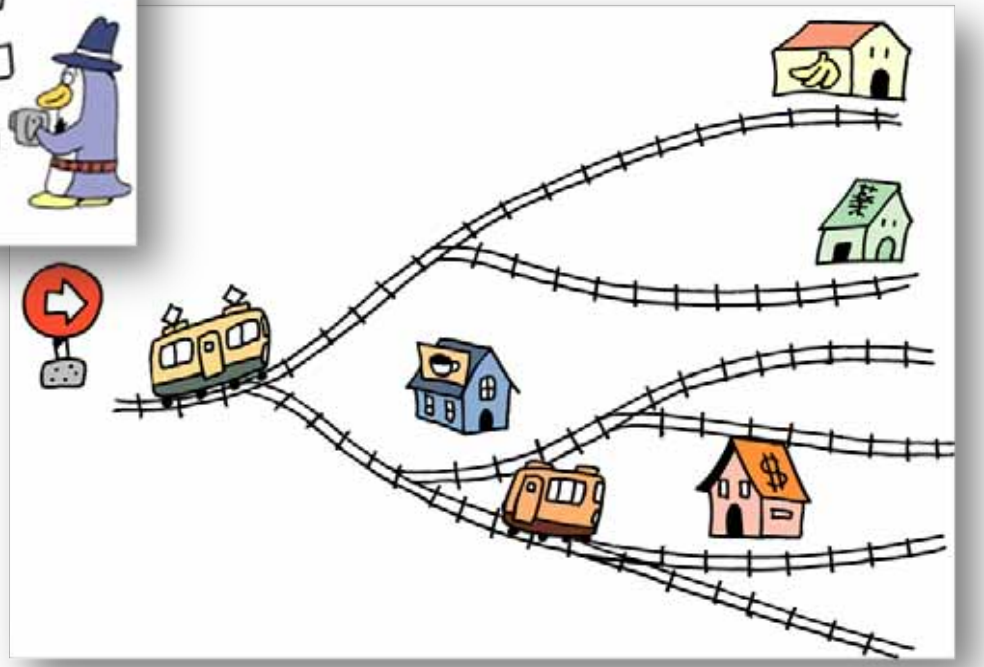
- TOMOYO Linux's policy consists of…
  - (domain + access permissions + mode) * N

- Domain:
  - Every process belongs to a domain.
  - Domain is expressed by process invocation history.
- Access Permissions:
  - Granted requests for the domain.
- Mode:
  - Enforcing, Permissive, Learning, Disabled
  - Each domain has a mode.
    - example) Apache is in Enforcing, Samba is in Permissive, Local login is in Disabled…

Automatic Policy Learning

Learning

Enforcing



Notice

- In this presentation, `black textbox` indicates text copied from TOMOYO Linux's policy editor.
- Almost all policies are generated by TOMOYO Linux's learning feature.
  - NOT manually written down.

Domain Transition

```
<kernel>
  /sbin/init
    /sbin/mingetty
      /bin/login
        /bin/bash
          /bin/ls
```

- Each line indicates a domain.
- Domain transition occurs whenever a program is executed.
- Domain transition tree reflects what programs are executed from what programs.

Policy for bash

```
<kernel> /usr/sbin/sshd /bin/bash

--x /bin/egrep
--x /bin/grep
--x /bin/hostname
-w- /dev/null
rw- /dev/tty
r-- /etc/bashrc
r-- /etc/inputrc
r-- /etc/nsswitch.conf
r-- /etc/passwd
r-- /etc/profile
r-- /etc/profile.d/colorls.sh
r-- /etc/profile.d/cvs.sh
r-- /etc/profile.d/glib2.sh
r-- /etc/profile.d/krb5-devel.sh
r-- /etc/profile.d/lang.sh
r-- /etc/profile.d/less.sh
r-- /etc/profile.d/which-2.sh
r-- /etc/sysconfig/i18n
r-- /etc/termcap
rw- /root/.bash_history
r-- /root/.bash_logout
r-- /root/.bash_profile
r-- /root/.bashrc
--x /sbin/consoletype
--x /usr/bin/clear
--x /usr/bin/dircolors
--x /usr/bin/id
--x /usr/sbin/ccs-editpolicy
allow_capability SYS_IOCTL
```

keyword “allow_read” is replaced “r--”,
keyword “allow_read/write” is replaced “rw-”,
keyword “allow_execute” is replaced “--x”
by TOMOYO Linux policy editor

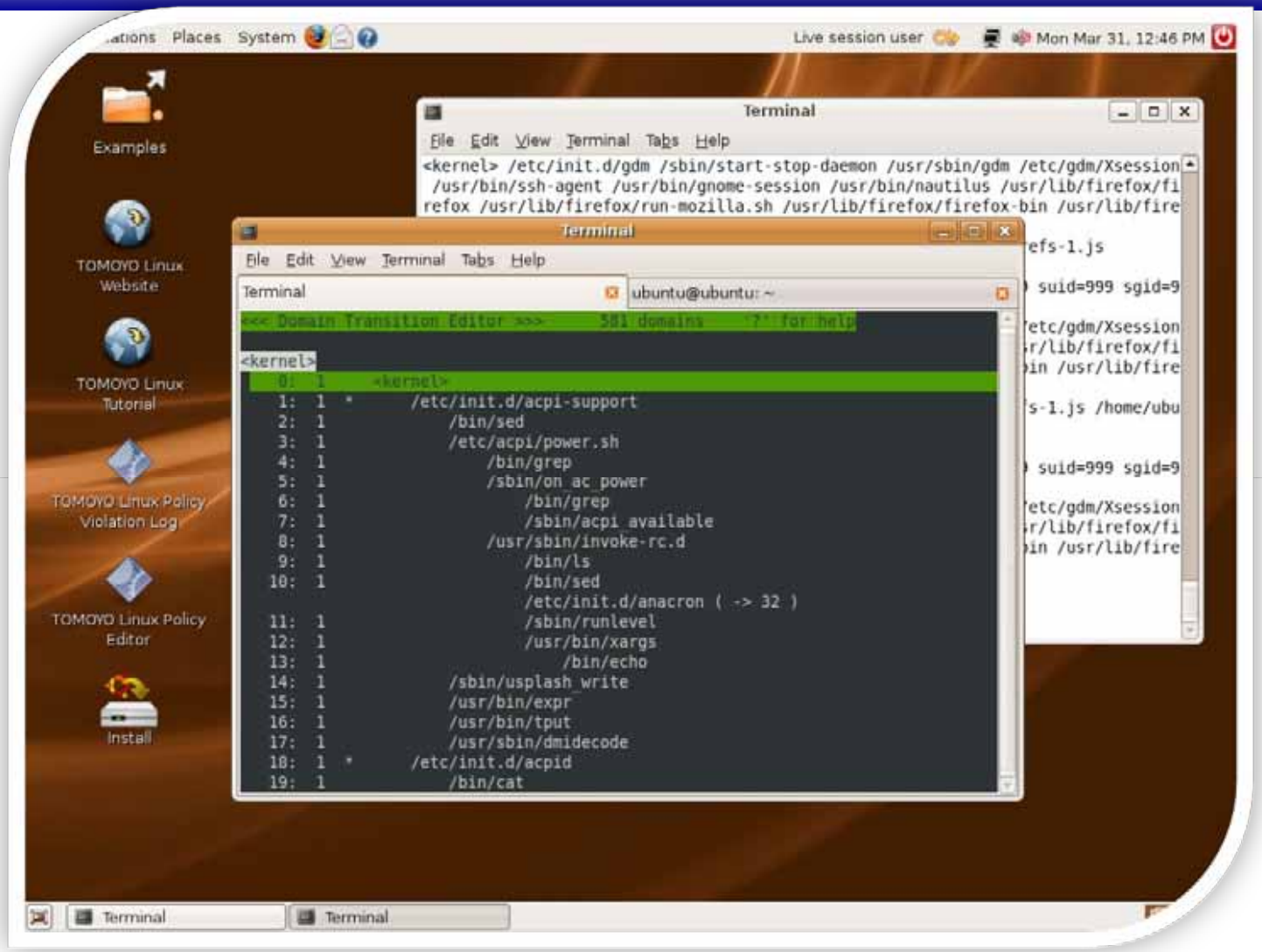
Policy for apache

```
<kernel> /usr/sbin/httpd

rw- /dev/null                                r-- /proc/sys/kernel/ngroups_max
r-- /dev/urandom                             r-- /usr/lib/httpd/modules/mod_¥*.so
r-- /etc/group                               -w- /var/log/httpd/access_log
r-- /etc/httpd/conf.d/proxy_ajp.conf        -w- /var/log/httpd/error_log
r-- /etc/httpd/conf.d/welcome.conf          -w- /var/run/httpd.pid
r-- /etc/httpd/conf/httpd.conf              r-- /var/www/html/¥*
r-- /etc/httpd/conf/magic                   allow_create /var/run/httpd.pid
r-- /etc/mime.types                         allow_network TCP bind 10.68.98.184 80
r-- /etc/nsswitch.conf                      allow_network TCP listen 10.68.98.184 80
r-- /etc/passwd                             allow_capability inet_tcp_create
r-- /etc/selinux/config                     allow_capability inet_tcp_listen
r-- /proc/filesystems                       allow_capability use_route
```

keyword “allow_read” is replaced “r--”,
keyword “allow_read/write” is replaced “rw-”,
keyword “allow_execute” is replaced “--x”
by TOMOYO Linux policy editor

Demo


Places System
Live session user
Mon Mar 31, 12:46 PM
Examples
TOMOYO Linux Website
TOMOYO Linux Tutorial
TOMOYO Linux Policy Violation Log
TOMOYO Linux Policy Editor
Install
Terminal
File Edit View Terminal Tabs Help
<kernel> /etc/init.d/gdm /sbin/start-stop-daemon /usr/sbin/gdm /etc/gdm/Xsession
/usr/bin/ssh-agent /usr/bin/gnome-session /usr/bin/nautilus /usr/lib/firefox/fi
refox /usr/lib/firefox/run-mozilla.sh /usr/lib/firefox/firefox-bin /usr/lib/fire
Terminal
File Edit View Terminal Tabs Help
Terminal
ubuntu@ubuntu: ~
<<< Domain Transition Editor >>> 581 domains '?' for help
<kernel>
0: 1 <kernel>
1: 1 * /etc/init.d/acpi-support
2: 1 /bin/sed
3: 1 /etc/acpi/power.sh
4: 1 /bin/grep
5: 1 /sbin/on_ac_power
6: 1 /bin/grep
7: 1 /sbin/acpi available
8: 1 /usr/sbin/invoke-rc.d
9: 1 /bin/ls
10: 1 /bin/sed
/etc/init.d/anacron (-> 32)
11: 1 /sbin/runlevel
12: 1 /usr/bin/xargs
13: 1 /bin/echo
14: 1 /sbin/usplash_write
15: 1 /usr/bin/expr
16: 1 /usr/bin/tput
17: 1 /usr/sbin/dmidecode
18: 1 * /etc/init.d/acpid
19: 1 /bin/cat
Terminal
Terminal

Automatic Policy Learning

- “Learning system behavior” means…
  - Monitoring
  - Recording
    - accesses from process (subject) to resource (object)

- A sort of access analysis.
- The result of access analysis = Policy
- You can understand your Linux’s behavior by browsing policy.

readahead

- Caching mechanism and utility program for faster boot.
- /usr/sbin/readahead is executed from /etc/rc.d/init.d/readahead_early (one of start up scripts) and it caches files listed in /etc/readahead.d/readahead_early.

```
<kernel>
  /sbin/init
    /etc/rc.d/rc
      /etc/rc.d/init.d/readahead_early
        /usr/sbin/readahead
```

readahead configuration

- should include…
  - files read in system boot sequence
- shouldn't include…
  - files NOT read in system boot sequence

- Default configuration is just a general example.
- "How can I create proper configuration for my Linux box?"
  - Now, TOMOYO Linux is for you!

readahead configuration by TOMOYO Linux

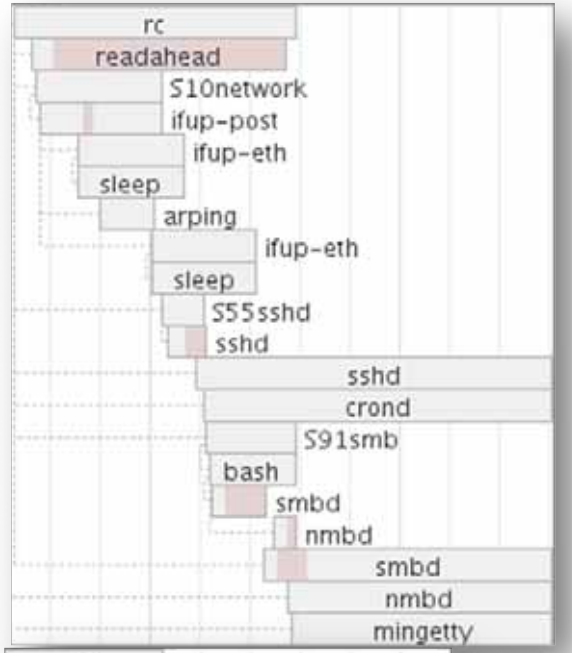
- Cleanup exception_policy.
  - exception_policy defines globally readable files.
- Start your system under learning mode.
  - All opened files are accumulated into domain_policy.
- Pick up files opened for reading.

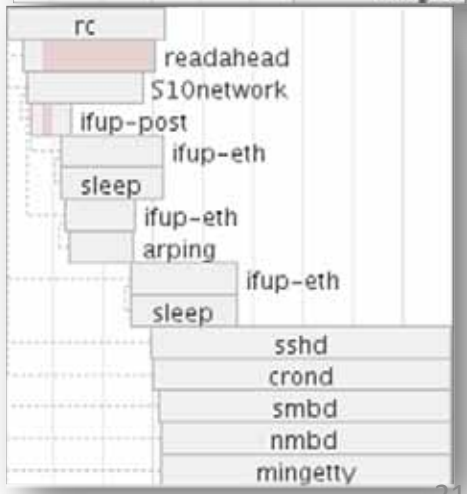
```
egrep '^allow_read|^allow_execute' /proc/ccs/domain_policy | ¥
  awk '{ print $2; }' | ¥
  sort | uniq | ¥
  egrep '^/bin/|^/etc/|^/lib/|^/sbin/|^/usr/|^/var/' ¥
    > /etc/readahead.d/default.early
```

readahead – results

- Using default configuration…
  - 827 files are read.
  - 23 seconds to boot.
- Using configuration by Learning…
  - only 236 files are read.
  - 21 seconds to boot.

- In the same way, you can pick up files actually used in system whole level.
  - You can determine minimum set of files to use.





Suitabilities for Embedded Linux

- 2.4/2.6 kernel support.
- Filesystem independent.
  - Extended attribute (xattr) support is unneeded.
- Built-in BusyBox support.
  - Multi call binary can be distinguished as its argv[0].
- Small memory footprint.
  - About 100KB for code, a few hundreds KB for policy.
- Legacy application support.
  - No modification needed for userland application.
- Remote administration.
  - No GUI needed for administration.

Not only MAC for files

- Gorgeous features of TOMOYO Linux
  - network/capability/mount/signal controls.
  - execution handler (added in 1.6.0).
  - built-in execution parameter check (added in 1.6.0).

Execution Handler

- Execution handler executes a program specified by policy instead of a program requested by the process.
  - The program specified by policy checks parameters and executes the requested program only if the parameters are appropriate.
- Example:
  - On demand honey pot
  - Virus scan
  - Web application firewall

Built-in execution parameter check

```
<kernel> /usr/sbin/httpd

--x /bin/sh if exec.argc=3
               exec.argv[1]="-c"
               exec.argv[2]="/usr/sbin/sendmail"
```

- /usr/sbin/httpd can execute /bin/sh only if
  - # of arguments is 3 and
  - its first argument is "-c" and
  - its second argument is "/usr/sbin/sendmail".
- This /bin/sh is dedicated for sending a mail.

- You can use execute handler for supporting more complicated conditions.

Plan

- We will have a BoF session in Ottawa Linux Symposium.
  - MAC for Linux, Time to Glean