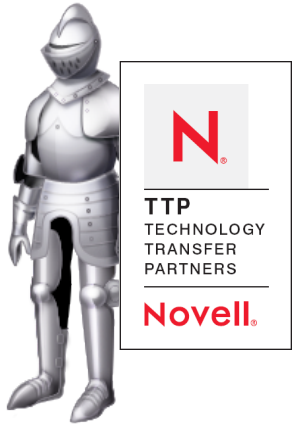
MindWorks UK



Apparmor: Linux internal security

Joe Doupnik

Mindworks and Oxford University

jrd@netlab1.oucs.ox.ac.uk

What does it do?

Apparmor applies access rules to processes which

- **Touch files**
- **Use communications ports**
- **Do system calls**

A monitored process is identified by its pathname

Apparmor ignores which user ID runs the process

Rules are applied in the kernel, difficult to bypass

What does it do?

The kernel portion is a native part of Linux

Red Hat & friends use Security Enhance Linux user-mode interface, which has a horrid reputation

Novell uses the Apparmor user-mode interface, far easier and nicer

Apparmor protects internals from unauthorised ops. It is an internal fence for each monitored app.

What does it do?

A monitored program can be watched two ways:

COMPLAIN: let things happen, but record violations in a log

ENFORCE: thou shalt not; the rules are enforced

Complain mode is for discovery of needed rules

Apparmor has a wizard to cull logs and recommend rule changes to us

There are many pre-written patterns to help us with rules

An experiment to add new rules

The plan is to monitor Apache with Apparmor

Use the existing Apache profile, see what happens when we add a new location URL to visit

Look at the *complain* versus *enforce* modes

Run the update wizard to suggest a rule for us

Active rules in /etc/apparmor.d

```
# cd /etc
# ls -l apparmor.d
total 48
drwxr-xr-x 2 root root 4096 Apr 20 09:40 abstractions
-rw-r--r-- 1 root root  732 Feb 21  2009 bin.ping
drwxr-xr-x 2 root root   27 Apr 20 09:40 program-chunks
-rw-r--r-- 1 root root  788 Feb 21  2009 sbin.klogd
-rw-r--r-- 1 root root 1130 Feb 21  2009 sbin.syslog-ng
-rw-r--r-- 1 root root 1087 Feb 21  2009 sbin.syslogd
drwxr-xr-x 2 root root   52 Apr 20 09:40 tunables
-rw-r--r-- 1 root root  696 Feb 21  2009 usr.sbin.avahi-daemon
-rw-r--r-- 1 root root  842 Feb 21  2009 usr.sbin.identd
-rw-r--r-- 1 root root  853 Feb 21  2009 usr.sbin.mdnsd
-rw-r--r-- 1 root root 1098 May 15  2007 usr.sbin.named
-rw-r--r-- 1 root root 1246 Feb 21  2009 usr.sbin.nscd
-rw-r--r-- 1 root root 1861 Feb 21  2009 usr.sbin.ntpd
-rw-r--r-- 1 root root  724 Feb 21  2009 usr.sbin.traceroute
#
```

Abstractions and program-chunks have common rules as includes

Extra profiles in /etc/apparmor

```
# ls -R apparmor
apparmor:
logprof.conf  profiles  reports.conf  reports.crontab  severity.db  subdomain.conf

apparmor/profiles:
extras

apparmor/profiles/extras:
README                                  usr.lib.postfix.lmtp
bin.netstat                             usr.lib.postfix.local
etc.cron.daily.logrotate                usr.lib.postfix.master
etc.cron.daily.slocate.cron             usr.lib.postfix.nqmgr
etc.cron.daily.tmpwatch                 usr.lib.postfix.oqmgr
sbin.dhclient                           usr.lib.postfix.pickup
sbin.dhcpcd                             usr.lib.postfix.pipe
sbin.portmap                            usr.lib.postfix.proxymap
sbin.resmgrd                            usr.lib.postfix.qmgr
sbin.rpc.lockd                          usr.lib.postfix.qmqpd
sbin.rpc.statd                          usr.lib.postfix.scache
usr.NX.bin.nxclient                     usr.lib.postfix.showq
usr.bin.acroread                        usr.lib.postfix.smtp
usr.bin.apropos                         usr.lib.postfix.smtpd
usr.bin.evolution-2.10                  usr.lib.postfix.spawn
usr.bin.fam                             usr.lib.postfix.tlsmgr
usr.bin.freshclam                       usr.lib.postfix.trivial-rewrite
usr.bin.gaim                            usr.lib.postfix.verify
usr.bin.man                             usr.lib.postfix.virtual
usr.bin.mlmmj-bounce                    usr.lib64.GConf.2.gconfd-2
usr.bin.mlmmj-maintd                    usr.sbin.cupsd
usr.bin.mlmmj-make-ml.sh                usr.sbin.dhcpd
usr.bin.mlmmj-process                   usr.sbin.httpd2-prefork
usr.bin.mlmmj-recieve                   usr.sbin.imapd
usr.bin.mlmmj-send                      usr.sbin.in.fingerd
usr.bin.mlmmj-sub                       usr.sbin.in.ftpd
usr.bin.mlmmj-unsub                     usr.sbin.in.ntalkd
usr.bin.opera                           usr.sbin.ipop2d
```

etc

Copy rule(s) to /etc/apparmor.d to activate

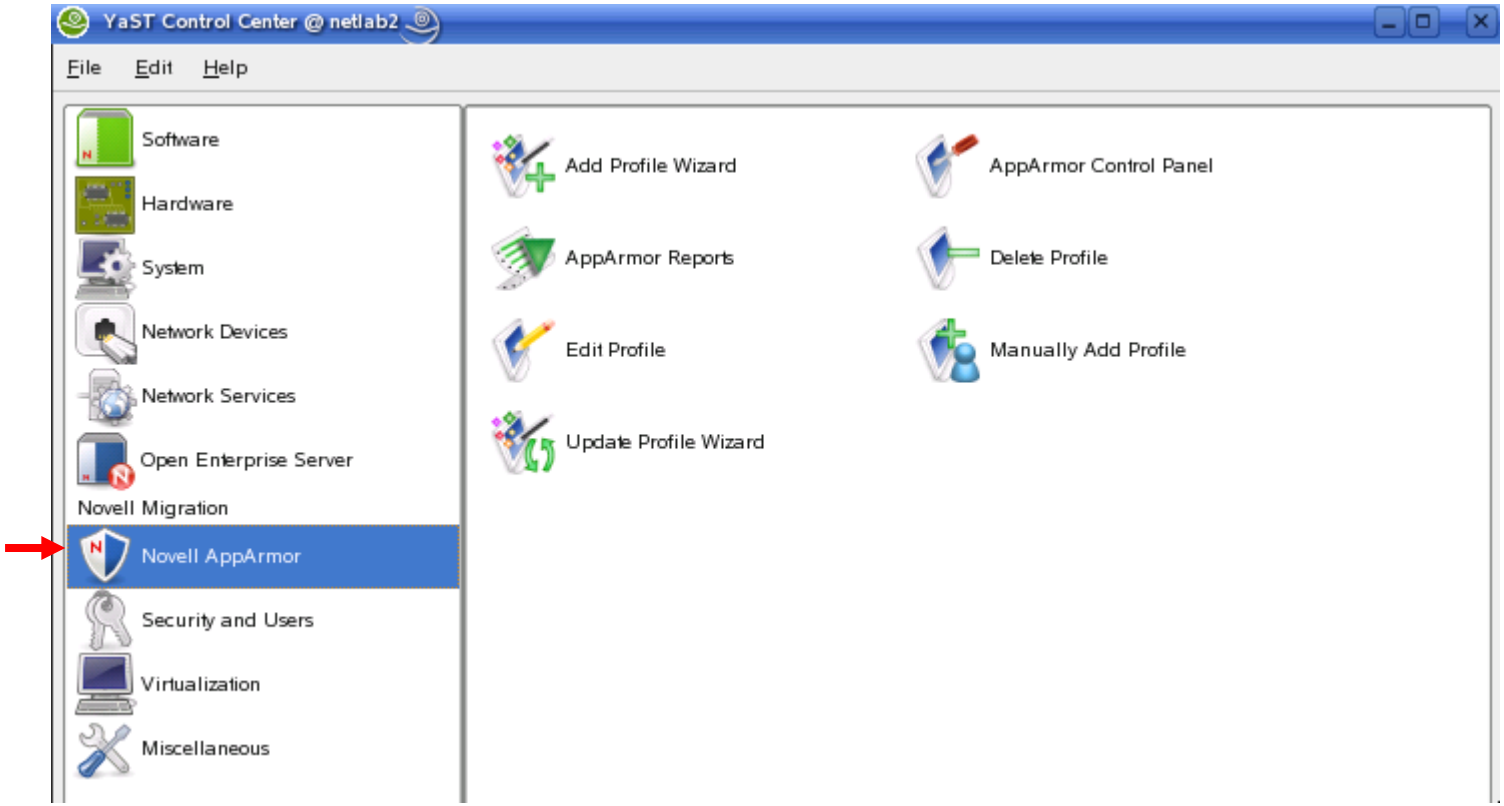
Profile names

A profile name is the *file path* to the program and then the program name, dotted

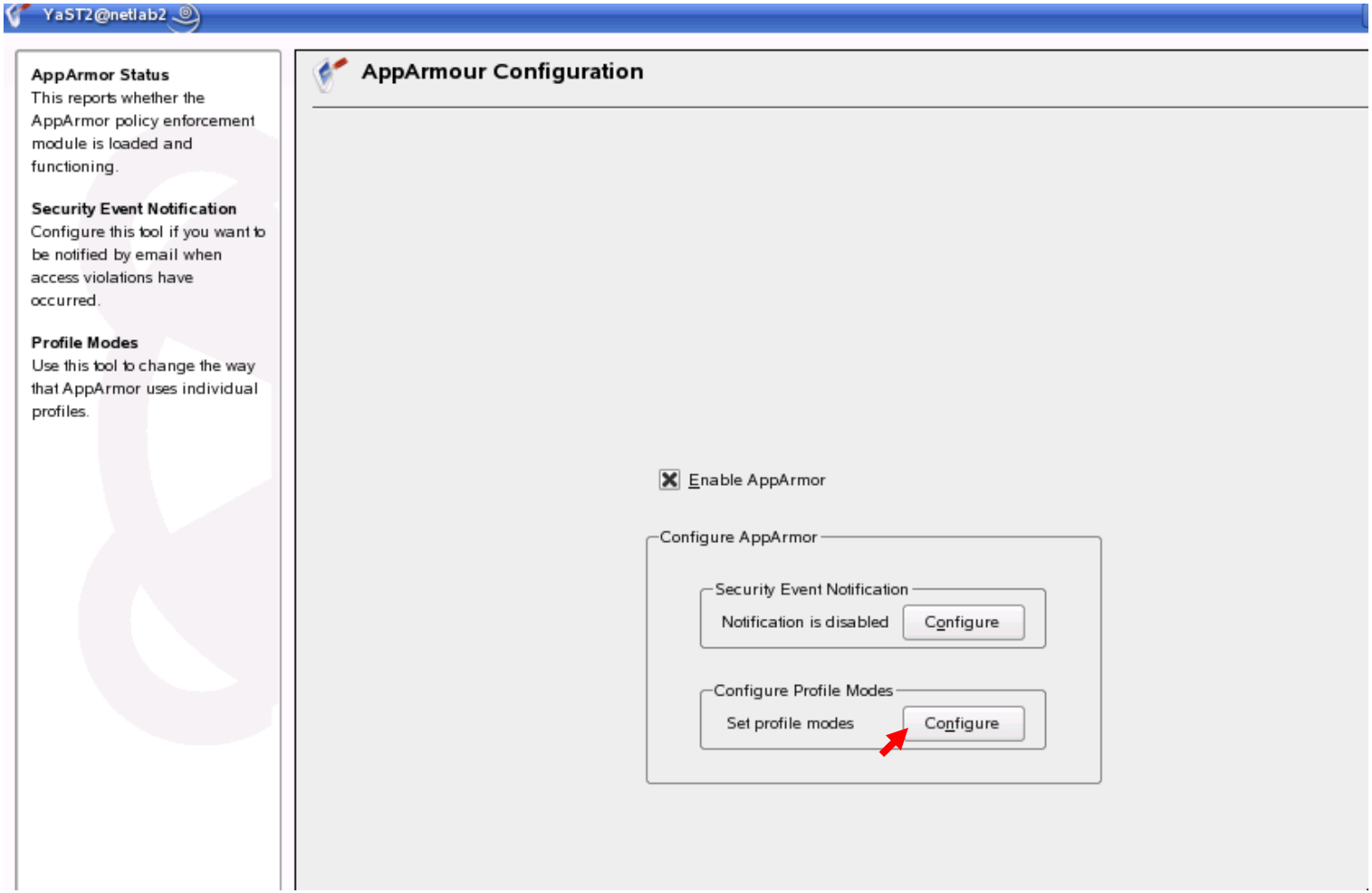
/usr/sbin/named has its rules in *usr.sbin.named*

These files are text, which we may edit or invoke a wizard

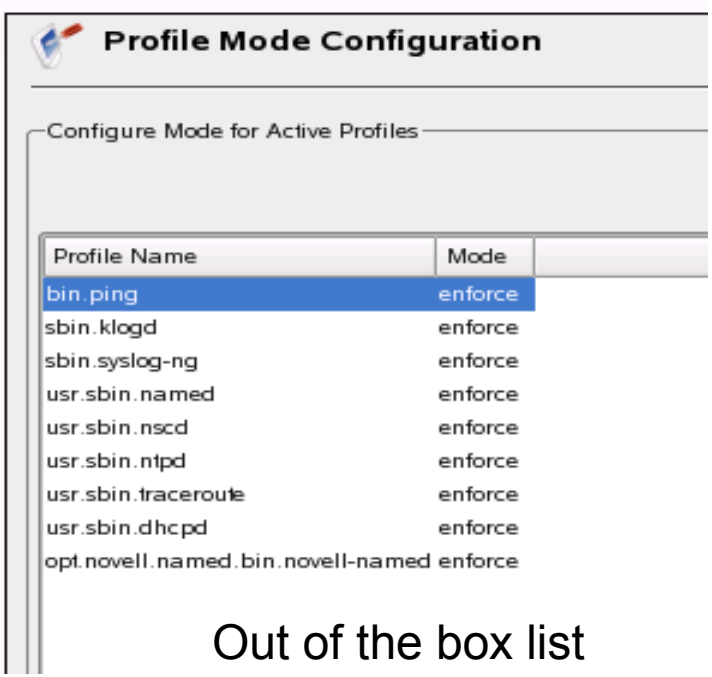
Apparmor within YaST



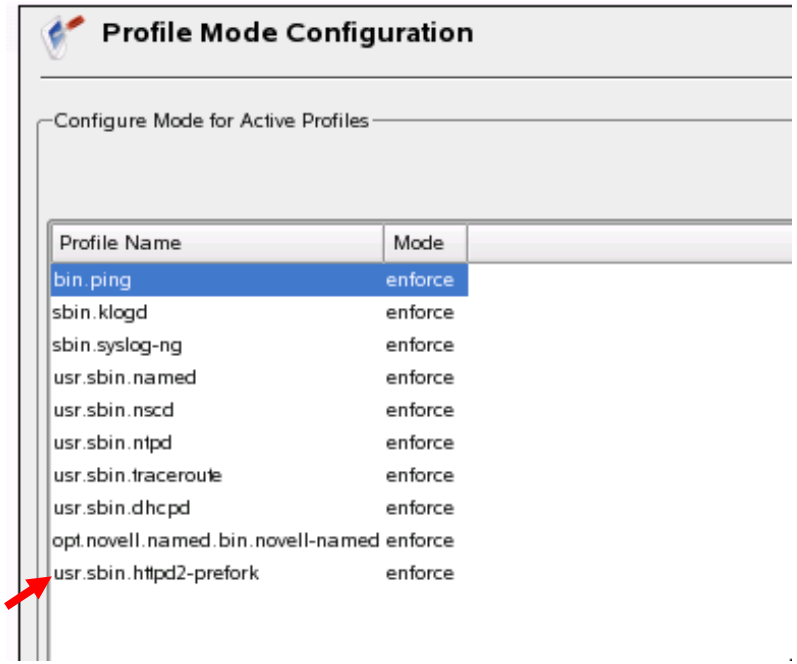
Configure it



See what is being watched



Out of the box list



After copying profile *usr.sbin.httpd2-prefork* to active directory apparmor.d

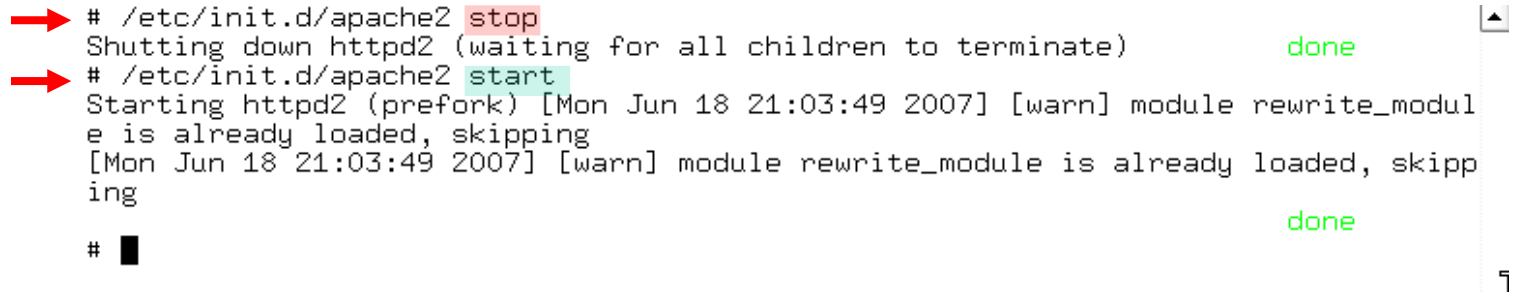
Notice these buttons:

On to experimenting…

An application gains a profile only when it is started

Let's restart Apache2 and see what happens

Recall, the profile is in "enforce" mode, so fingers crossed and here we go…



My goodness, we got away with it, thus far…

But we have not asked Apache to fetch files, yet

Make a new URL available

Create file /etc/apache2/conf.d/test.conf to view /usr/src

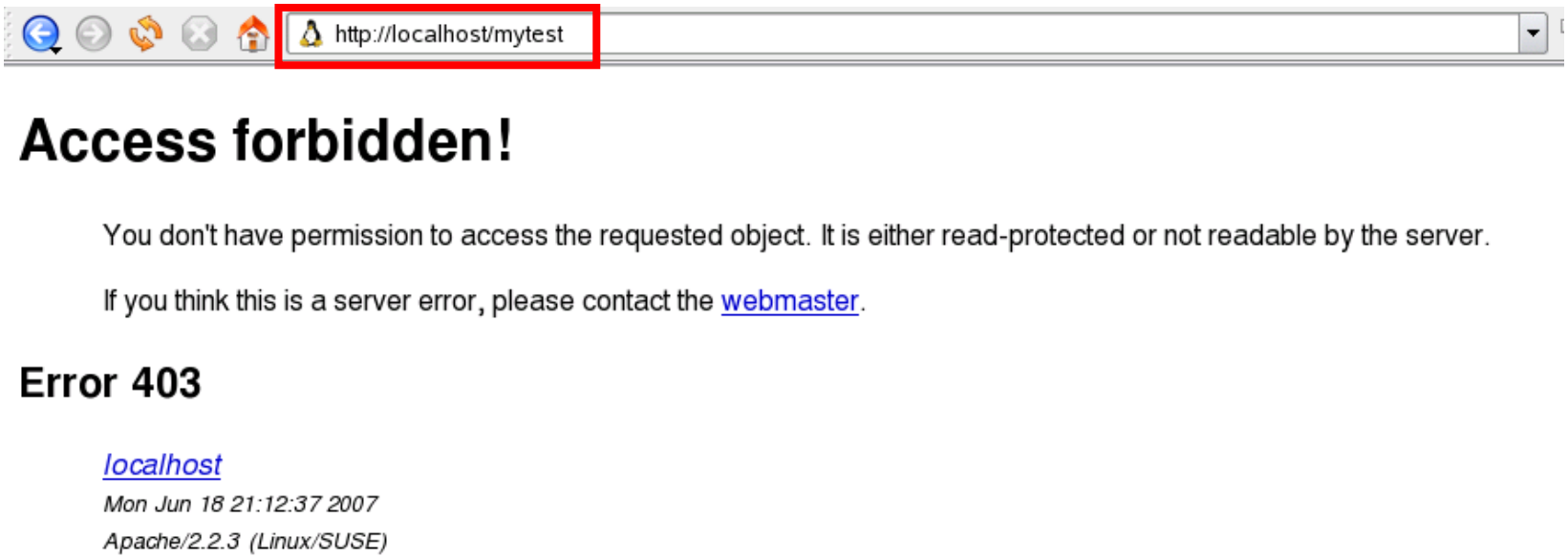
```
# A test configuration script
    Alias /mytest  /usr/src
    <Directory /usr/src>
        Options +Indexes +FollowSymLinks
        IndexOptions +NameWidth=*

        Order allow,deny
        Allow from all
    </Directory>
~
~
```

Gives Apache access to /usr/src, but Apparmor has yet to agree

Restart Apache to re-read its configuration: that also worked

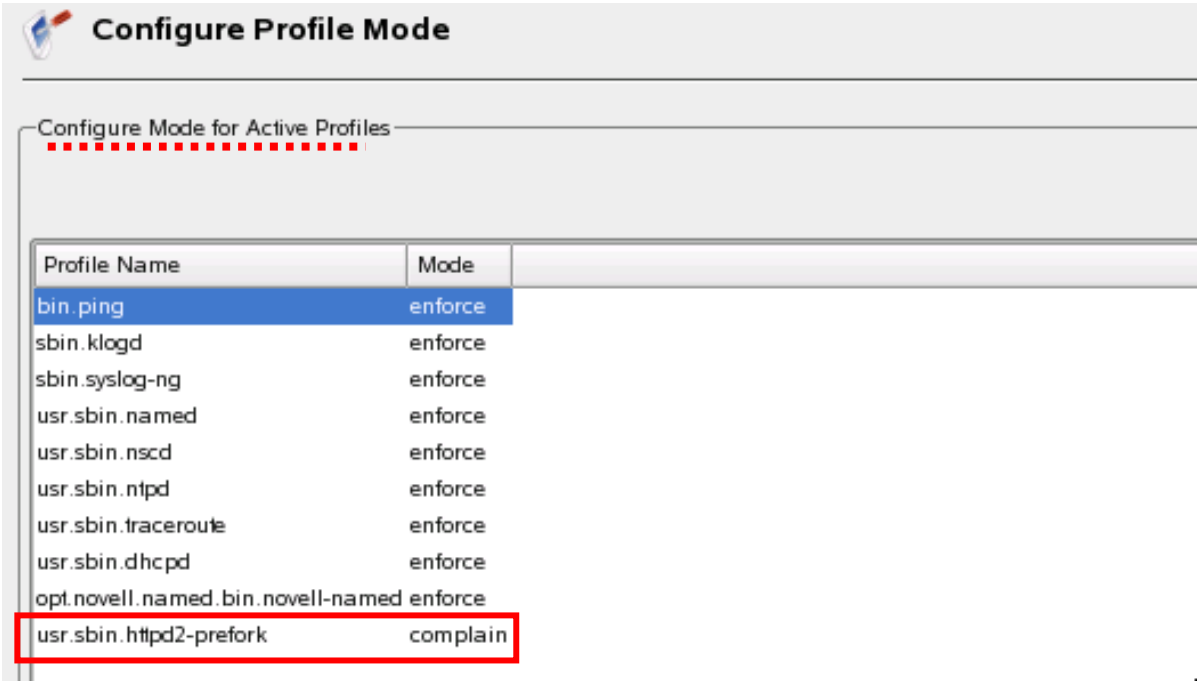
Try our test area



Hmm, well, that's right. Apache was monitored and had no Apparmor permission to look in /usr/src

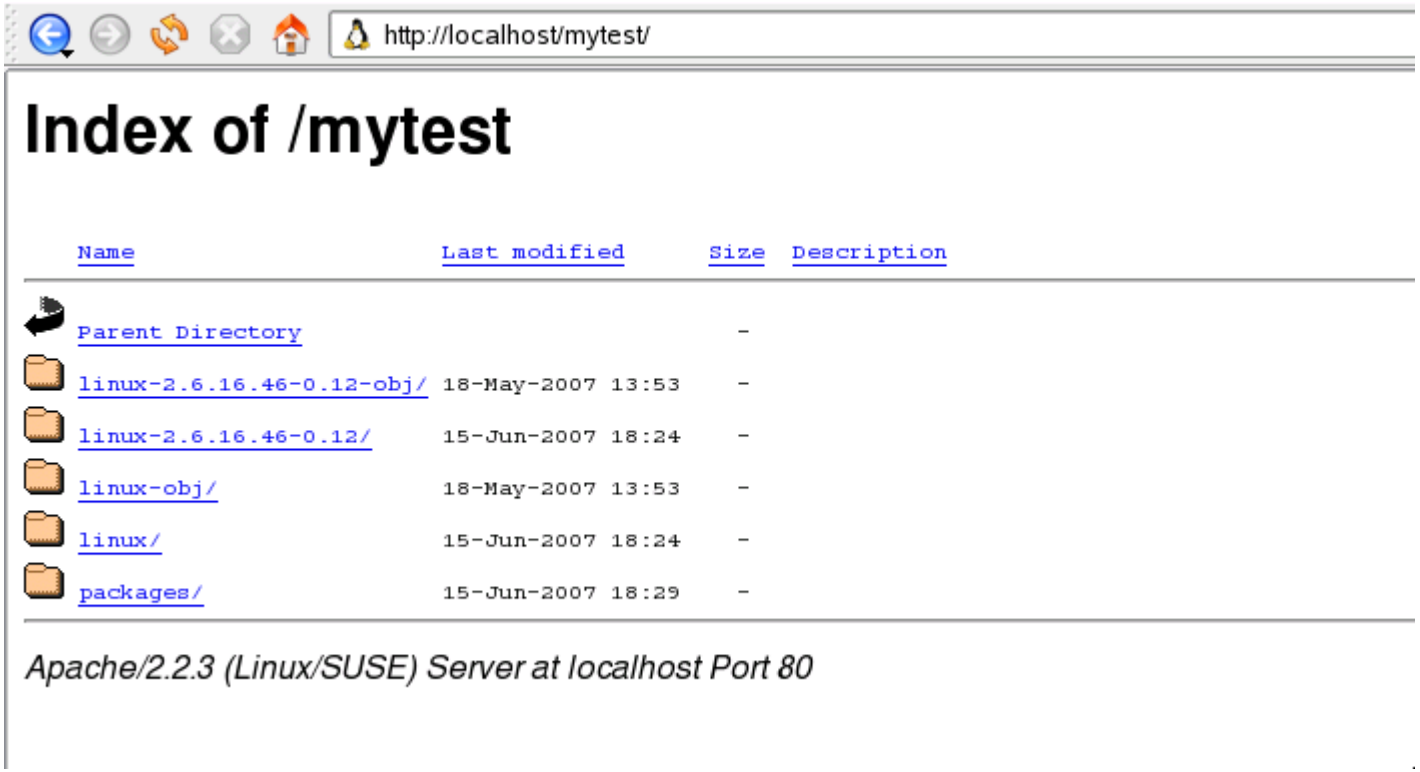
Use “complain” mode

Command complain http2-prefork



This permits access, but writes complaints to log for later rule forming by the wizard

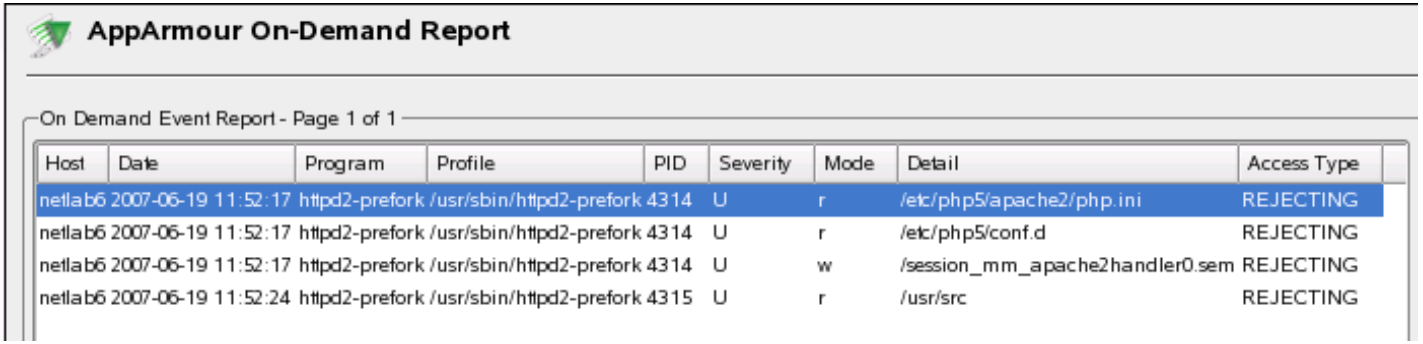
The test URL worked, expected



Now go back to “enforce” mode and fix up rules

Command enforce http2-prefork

Check /var/log/messages



AppArmour On-Demand Report

On Demand Event Report - Page 1 of 1

Host	Date	Program	Profile	PID	Severity	Mode	Detail	Access Type
netlab6	2007-06-19 11:52:17	httpd2-prefork	/usr/sbin/httpd2-prefork	4314	U	r	/etc/php5/apache2/php.ini	REJECTING
netlab6	2007-06-19 11:52:17	httpd2-prefork	/usr/sbin/httpd2-prefork	4314	U	r	/etc/php5/conf.d	REJECTING
netlab6	2007-06-19 11:52:17	httpd2-prefork	/usr/sbin/httpd2-prefork	4314	U	w	/session_mm_apache2handler0.sem	REJECTING
netlab6	2007-06-19 11:52:24	httpd2-prefork	/usr/sbin/httpd2-prefork	4315	U	r	/usr/src	REJECTING

Hmmm, some PHP grumbles, a handler problem, and our test case.

```
Jun 19 11:52:17 netlab6 kernel: audit(1182250337.831:3): REJECTING r access to /
etc/php5/apache2/php.ini (httpd2-prefork(4314) profile /usr/sbin/httpd2-prefork
active /usr/sbin/httpd2-prefork)
Jun 19 11:52:17 netlab6 kernel: audit(1182250337.831:4): REJECTING r access to /
etc/php5/conf.d (httpd2-prefork(4314) profile /usr/sbin/httpd2-prefork active /u
sr/sbin/httpd2-prefork)
Jun 19 11:52:17 netlab6 kernel: audit(1182250337.831:5): REJECTING w access to /
session_mm_apache2handler0.sem (httpd2-prefork(4314) profile /usr/sbin/httpd2-pr
efork active /usr/sbin/httpd2-prefork)
Jun 19 11:52:24 netlab6 kernel: audit(1182250344.667:6): REJECTING r access to /
usr/src (httpd2-prefork(4315) profile /usr/sbin/httpd2-prefork active /usr/sbin/
httpd2-prefork)
Jun 19 11:53:52 netlab6 kernel: audit(1182250432.393:7): PERMITTING r access to
/usr/src (httpd2-prefork(4316) profile /usr/sbin/httpd2-prefork active /usr/sbin
/httpd2-prefork)
```

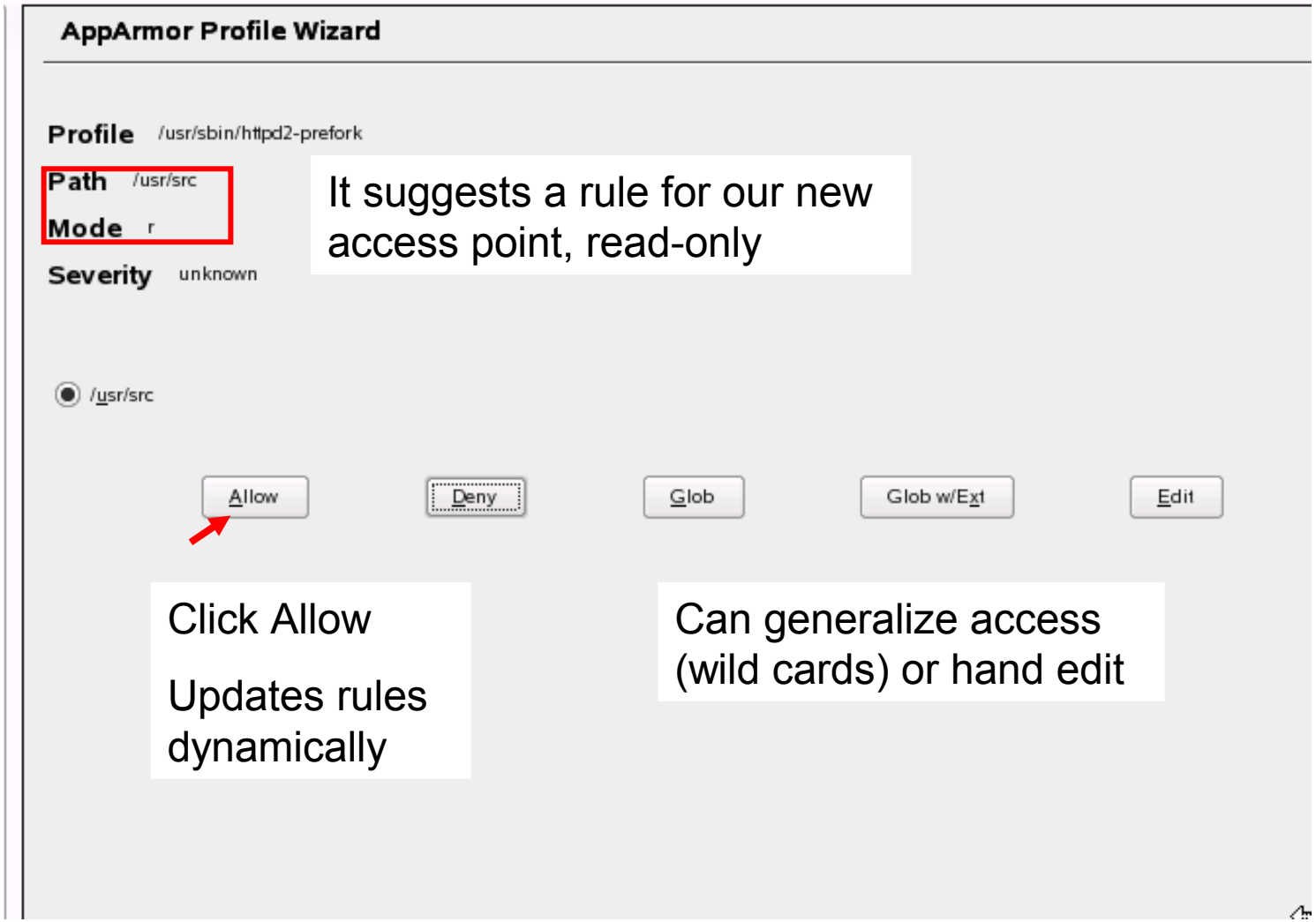
Note well: Apparmor requires auditd be running (the default state)

/var/log/audit/audit.log

```
netlab6:/var/log/audit # pwd
/var/log/audit
netlab6:/var/log/audit # ls -l
total 4
-rw-r----- 1 root root 847 Jun 19 12:14 audit.log
netlab6:/var/log/audit # cat audit.log
type=DAEMON_START msg=audit(1182249430.218:1188) auditd start, ver=1.2.9, format
=raw, auid=4294967295 pid=9503 res=success, auditd pid=95
type=CONFIG_CHANGE msg=audit(1182249430.297:5): audit_enabled=0 old=0 by auid=42
94967295
type=DAEMON_END msg=audit(1182249569.615:1189) auditd normal halt, sending auid=
0 pid=10663 subj= res=success, auditd pid=9503
type=DAEMON_START msg=audit(1182251663.640:2883) auditd start, ver=1.2.9, format
=raw, auid=4294967295 pid=4840 res=success, auditd pid=48
type=CONFIG_CHANGE msg=audit(1182251663.740:13): audit_enabled=0 old=0 by auid=4
294967295
type=CONFIG_CHANGE msg=audit(1182251663.744:14): audit_backlog_limit=256 old=64
by auid=4294967295
type=APPARMOR msg=audit(1182251668.952:15): REJECTING r access to /usr/src (http
d2-prefork(4747) profile /usr/sbin/httpd2-prefork active /usr/sbin/httpd2-prefor
k)
netlab6:/var/log/audit #
```

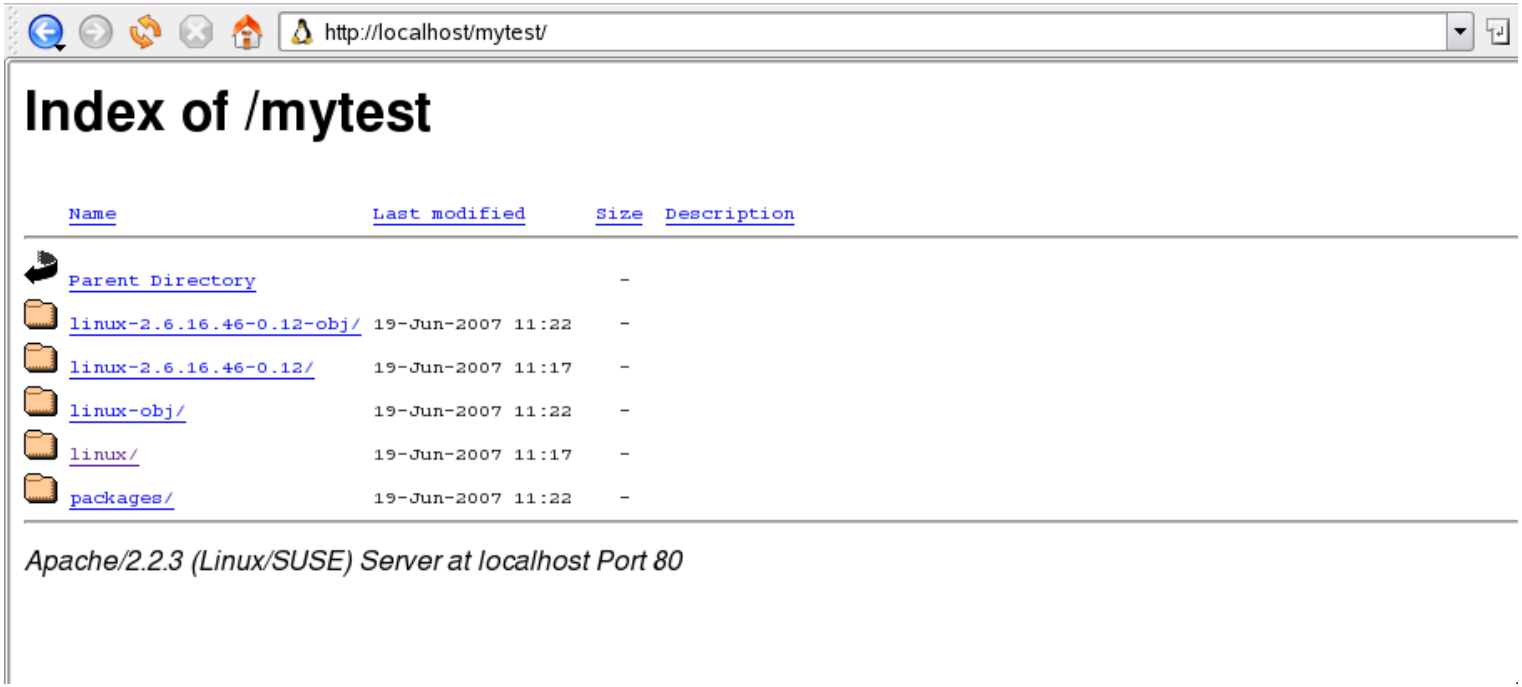
This is where Apparmor prefers to find status messages for use by the rules wizard. Auditd needs to be running.

Update Profile Wizard



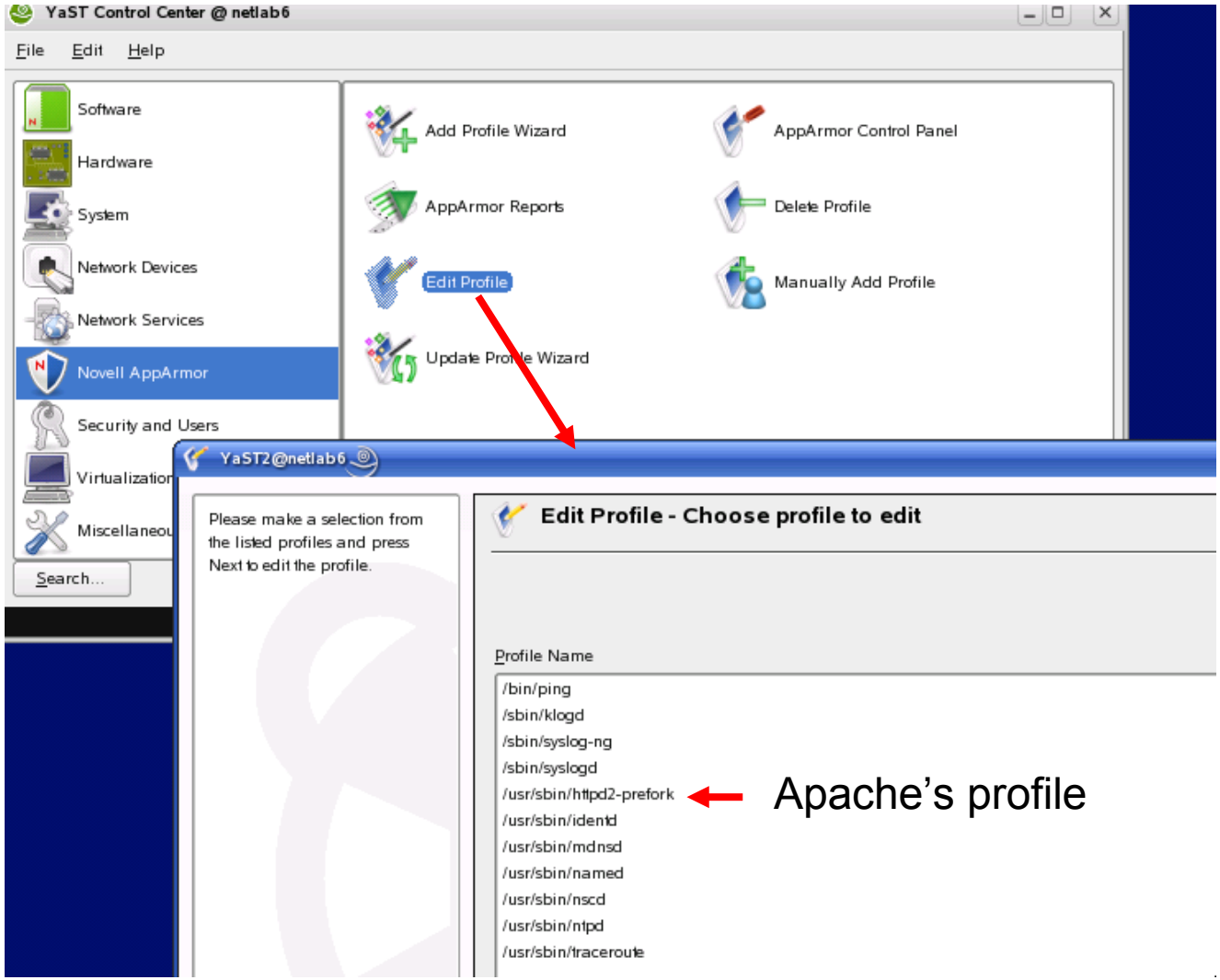
Retry with new rule, enforced

Try web page again without reloading Apache



Success!

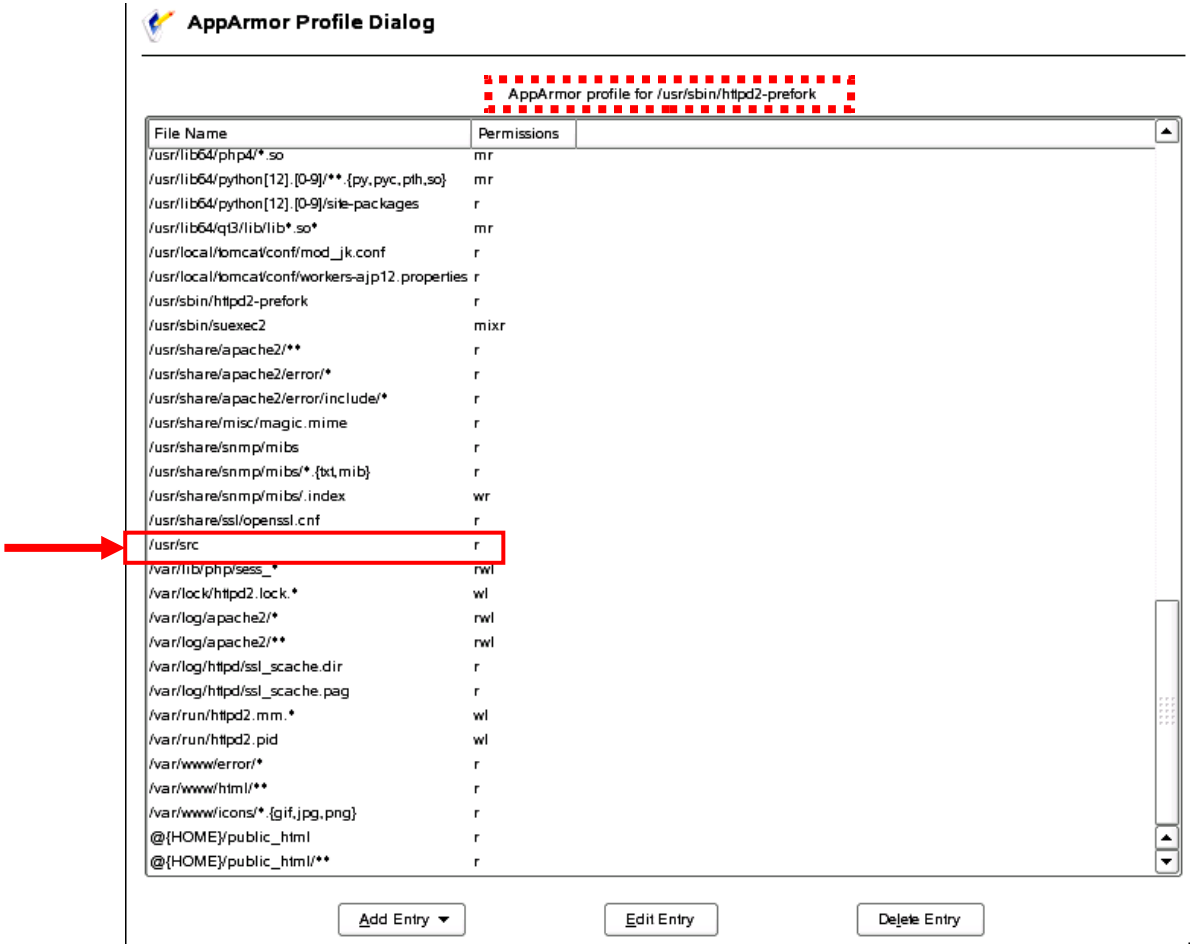
Look at the final profile



Review Apache rules, scroll down

Our new rule for Apache

Others were from the template



The text file of rules for Apache

```
# ------------------------------------------------------------------
# apache-default-uri is used as a fallback SubDomain hat to use when
# no specific URI, Location, or Directory hat name fits.
# it is included in both the main profile and the DEFAULT_URI subprofile
# so that use of mod_change_hat is optional with the default profile.
/usr/sbin/httpd2-prefork flags=(complain) {
  #include <abstractions/base>
  #include <abstractions/consoles>
  #include <abstractions/kerberosclient>
  #include <abstractions/nameservice>
  #include <abstractions/perl>
  #include <program-chunks/apache-default-uri>
  #include <program-chunks/httpd>

  capability dac_override,
  capability kill,
  capability net_bind_service,
  capability setgid,
  capability setuid,
  capability sys_tty_config,

  /dev/random r,
  /etc/.nam_nss_sock w,
  /etc/apache2/*.conf r,
  /etc/apache2/magic r,
  /etc/apache2/mod_perl-startup.pl r,
  /etc/apache2/ssl.crt/server.crt r,
  /etc/apache2/ssl.key/server.key r,
```

flags = (complain) denotes “complain” mode.

#includes of profile fragments dealing with common accesses

Capabilities are system call permissions. Kill permits reception of kill -9

Apache rules file continued

```
^DEFAULT_URI flags=(complain) {
  #include <program-chunks/apache-default-uri>
  #include <program-chunks/base-files>
  #include <program-chunks/nameservice-files>

  /etc/mtab r,
  /opt/novell/netstorage/webapp r,
  /opt/novell/netstorage/webapp/* r,
  /opt/novell/netstorage/webapp/images/* r,
  /proc/meminfo r,
  /tmp/* w,
  /usr/share/susehelp/docserver/* r,
  /var/lib/qfsearch/** r,
  /var/opt/novell/eGuide/look/default/browser/* r,
  /var/opt/novell/iManager/** r,
  /var/opt/novell/iprint/htdocs/** r,
  /var/opt/novell/iprint/mod_ipp/* rw,
  /var/opt/novell/tomcat4/** r,
  /var/opt/novell/xtier/xsrvd/* w,
}

^HANDLING_UNTRUSTED_INPUT flags=(complain) {
  #include <abstractions/bash>
  #include <program-chunks/nameservice-files>

  /**.htaccess r,
  /bin/bash ix,
  /bin/grep ixr,
  /dev/null w,
  /dev/random r,
    ...
```

Areas for “hats”
Denoted by clauses
^HAT-NAME {…}

Hats are new rule sets added on the fly when Apache changes from a simple mode. Easy to have per-URL rule sets and some for PHP apps.

Process security checking

Command "unconfined" lists network processes which are, or not, protected by apparmor. ps –eM looks at running procs

```
# ./boot.apparmor start
Mounting securityfs on /sys/kernel/security
Loading AppArmor profiles
# unconfined
2398 /sbin/rpcbind not confined
2398 /sbin/rpcbind not confined
2398 /sbin/rpcbind not confined
2912 /usr/lib/postfix/master not confined
2961 /usr/sbin/ntpd not confined
2961 /usr/sbin/ntpd not confined
2961 /usr/sbin/ntpd not confined
2961 /usr/sbin/ntpd not confined
2961 /usr/sbin/ntpd not confined
2961 /usr/sbin/ntpd not confined
3020 /usr/sbin/xinetd not confined
3020 /usr/sbin/xinetd not confined
3028 /usr/sbin/sshd not confined
```

Uses netstat to find listeners

ps –eM checks all running processes

```
# ps -eM
LABEL                          PID TTY          TIME CMD
unconfined                       1 ?        00:00:01 init
unconfined                       2 ?        00:00:00 kthreadd
unconfined                       3 ?        00:00:00 migration/0
unconfined                       4 ?        00:00:00 ksoftirqd/0
unconfined                       5 ?        00:00:00 events/0
unconfined                       6 ?        00:00:00 khelper
```

```
# enforce
Please enter the program to switch to enforce mode:
```

```
# complain
Please enter the program to switch to complain mode:
```

Is this technique practical?

If one were to broadly protect the innards of an OES2/Linux server we are talking about hundreds of executables and tens of thousands of files

The task of detailing all this into rules seems huge

I thought the same, until I undertook exactly this task

After many false starts the light dawned: aggregate similar things (whole directories), inherit perms

With those notions I can do all of OES2 in a morning

A plan of analysis, aggregation

Marking whole directories as available may expose sensitive information in contained files. We must look to be certain the bulk approach is wise.

AppArmor provides several wild card simplifications:

/path/* for all files in the current directory

/path/** for that plus files in subdirectories beneath

/path*.txt and /.txt** select files by pattern

Is this technique practical?

Run in Complain mode, exercise available apps, use wizard to read logs and offer rules

Let the system cook in Complain mode for a week or so to explore corners

Turn on Enforce mode, keep an eye on logs

Performance overhead

Questions have been raised about the system overhead in the Linux Security Module approach

After all, the total quantity of rules may be large

Performance measurements reveal a cost of about 1%, see next slide for numbers

Rules are cached in memory for fast access, the amount of memory used is very small though

Performance overhead

Test Type	2.5.15	2.5.15-lsm	% Overhead with LSM
0K file create	96	96	0%
0K file delete	31	31	0%
10K file create	157	158	0.6%
10K file delete	45	46	2.2%
mmap latency	3246	3158	-2.7%
prot fault	0.899	1.007	12%
page fault	3	3	0%

Local communication bandwidth in MB/s, larger is better:

Test Type	2.5.15	2.5.15-lsm	% Overhead with LSM
pipe	630	597	5.2%
AF Unix	125	125	0%
TCP	222	220	0.9%
file reread	316	313	0.9%
mmap reread	378	368	2.6%
bcopy (libc)	199	191	4%
bcopy (hand)	168	149	11.3%
mem read	378	396	2.6%
mem write	206	197	4.4%

Table 3: LMBench Microbenchmarks, 1 processor machine

Linux Security Modules:
General Security Support for the Linux Kernel

Chris Wright and Crispin Cowan
WireX Communications, Inc.
chris@wirex.com, crispin@wirex.com

Stephen Smalley
NAI Labs, Network Associates, Inc.
sds@tislabs.com

James Morris
Intercode Pty Ltd
jmorris@intercode.com.au

Greg Kroah-Hartman
IBM Linux Technology Center
gregkh@us.ibm.com

USENIX Security Symposium, August 2002

MindWorks Inc. Ltd
210 Burnley Road
Weir
Bacup
OL13 8QE UK

Telephone: +44 (0) 170 687 1900

Fax: +44 (0) 170 687 8203

Web: www.mindworksuk.com

Email: training@mindworksuk.com