Università degli Studi di Milano
Facoltà di Scienze Matematiche, Fisiche e Naturali
Anno Accademico 2014/2015

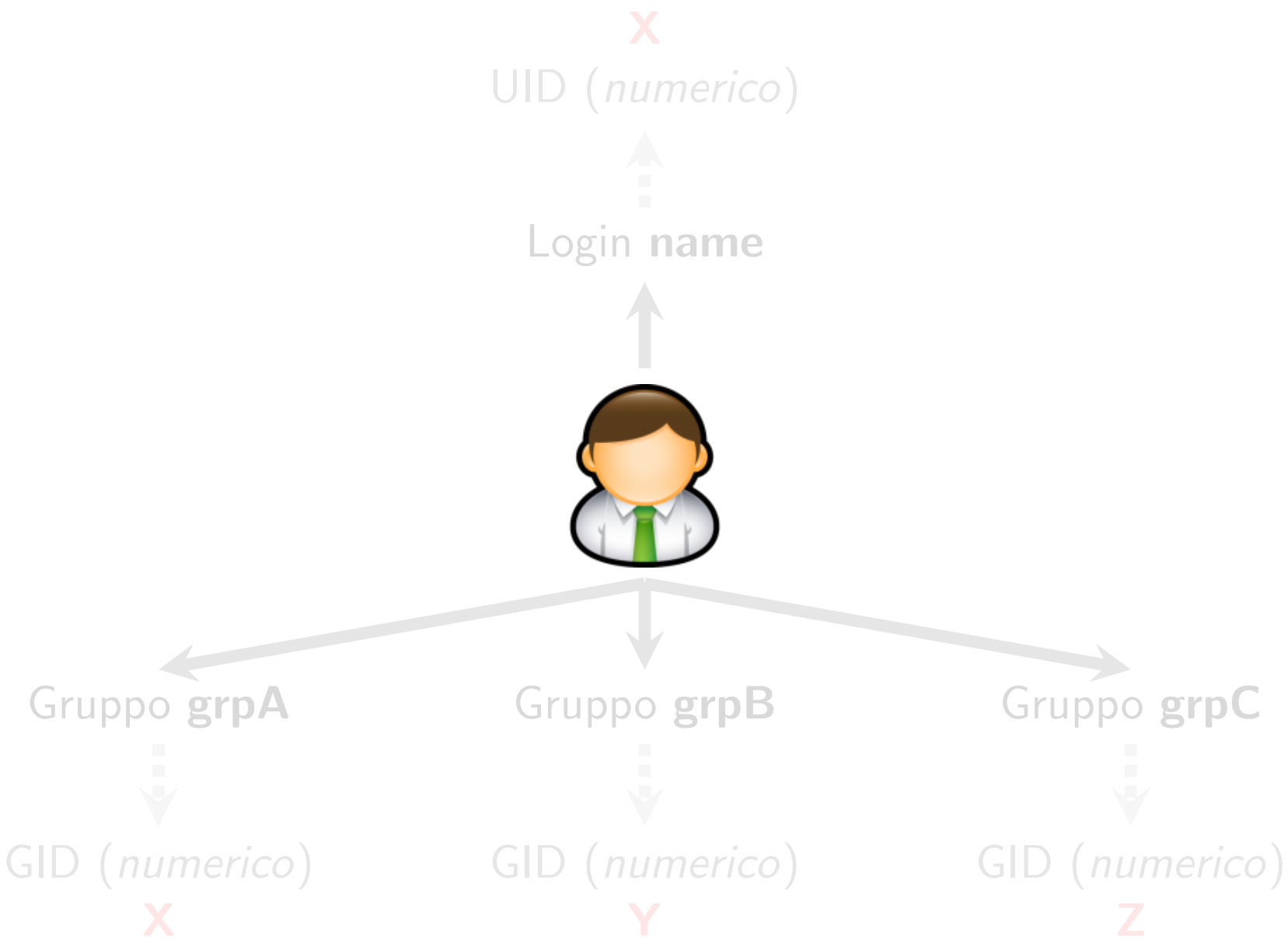
Controllo degli accessi in UNIX - parte II

Andrea Lanzi

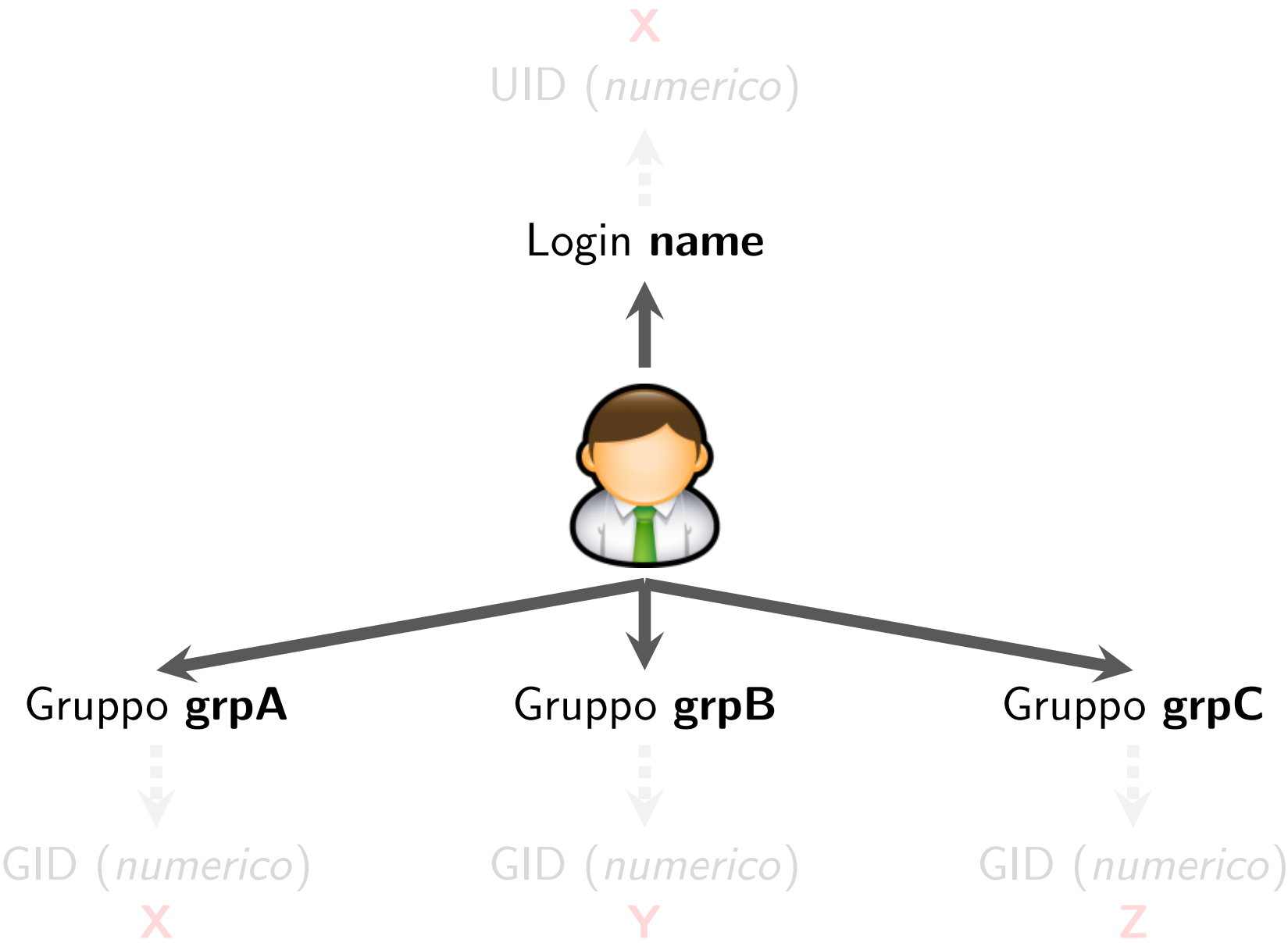
14 Aprile 2013

Sommario

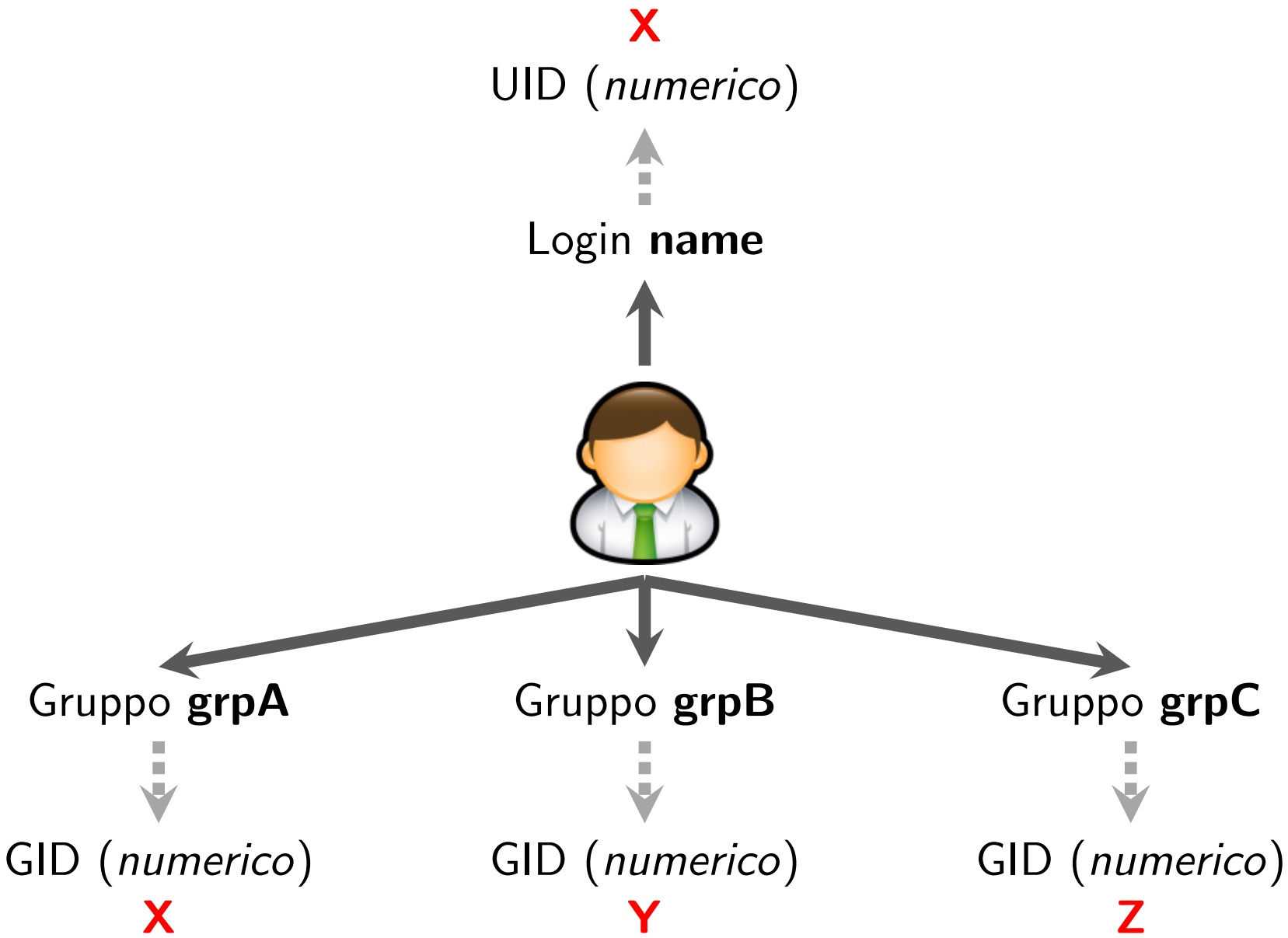
Permessi dei processi - riassunto



Permessi dei processi - riassunto



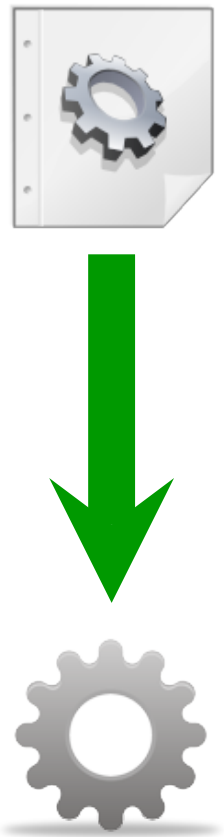
Permessi dei processi - riassunto



Permessi dei processi - riassunto

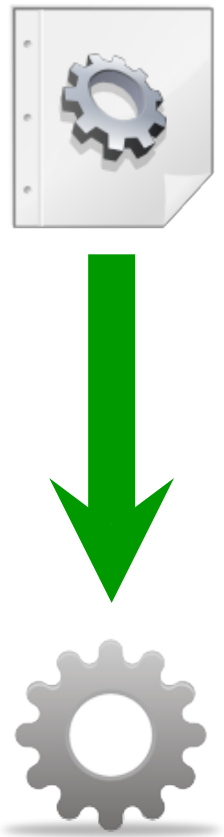
permessi	Permessi `rw{x|s}{t}` definiti per **proprietario**, **gruppo**, gli **altri utenti**
proprietario	Un **unico proprietario** identificato da una stringa
gruppo	Un **unico gruppo** di appartenenza identificato da una stringa

Permessi dei processi - riassunto



permessi	Permessi `rw{x|s}{t}` definiti per **proprietario**, **gruppo**, gli **altri utenti**
proprietario	Un **unico proprietario** identificato da una stringa
gruppo	Un **unico gruppo** di appartenenza identificato da una stringa

Permessi dei processi - riassunto



permessi	Permessi `rw{x|s}{t}` definiti per **proprietario**, **gruppo**, gli **altri utenti**
proprietario	Un **unico proprietario** identificato da una stringa
gruppo	Un **unico gruppo** di appartenenza identificato da una stringa

PID	ID univoco che identifica il processo
UID/GID	UID/GID reale, appartenente all'utente che esegue il processo
EUID/EGID	UID/GID effettivo in un particolare istante di tempo
SUID/SGID	UID/GID salvati
FSUID/SGID	*UID/GID effettivi memorizzati nel file system*

Eventi di sistema

... aka LOG

I messaggi di *log* rappresentano eventi di sistema (programmi) o messaggi del kernel. L'analisi di *log* permette di verificare:

- stato del sistema
- sicurezza del sistema

- `syslogd`: applicazioni user-space
- `klogd`: applicazioni kernel-space

Eventi di sistema

... aka LOG

I messaggi di *log* rappresentano eventi di sistema (programmi) o messaggi del kernel. L'analisi di *log* permette di verificare:

- stato del sistema
- sicurezza del sistema

- `syslogd`: applicazioni user-space
- `klogd`: applicazioni kernel-space

Syslog e i file di log

Cosa è un log?

È un evento a cui è associata una linea di un file di testo, contenente precise informazioni come data, ora, tipo di evento e altri dettagli rilevanti.

Demoni, kernel e servizi producono dati che vengono memorizzati in file di log.
I log sono estremamente utili per scovare e risolvere problemi di configurazione di servizi e periferiche.
Syslogd è il demone principale adibito al logging, ma numerosi servizi e script utilizzano i propri file di log generati *ad hoc*.
I sistemi moderni supportano tool per effettuare rotazione, compressione e monitoring dei file di log, su base giornaliera/settimanale.

Esempi di file di log

cat /var/log/boot

```
Tue Apr 16 15:18:46 2013: Starting VMware services:
Tue Apr 16 15:18:46 2013:    Virtual machine monitor^[[71G done
Tue Apr 16 15:18:46 2013:    Virtual machine communication interface^[[71G done
Tue Apr 16 15:18:46 2013:    VM communication interface socket family^[[71G done
Tue Apr 16 15:18:46 2013:    Blocking file system^[[71G done
Tue Apr 16 15:18:46 2013:    Virtual ethernet^[[71G done
Tue Apr 16 15:18:46 2013:    VMware Authentication Daemon^[[71G done
Tue Apr 16 15:18:46 2013:    Shared Memory Available^[[71G done
Tue Apr 16 15:18:46 2013: [....] Starting MySQL database server: mysqld . . . .
Tue Apr 16 15:18:48 2013: [^[[36minfo^[[39;49m] Checking for tables which need
Tue Apr 16 15:18:48 2013: an upgrade, are corrupt or were not closed cleanly..
Tue Apr 16 15:18:50 2013: [....] Starting MTA: exim4^[[?1c^[7^[[1G[^[[32m ok
Tue Apr 16 15:18:50 2013: EXAMPLE: exim paniclog /var/log/exim4/paniclog has
Tue Apr 16 15:18:50 2013: non-zero size, mail system possibly broken
Tue Apr 16 15:18:50 2013:    Starting Workstation Server:^[[71G done
```

File memorizzati in `/var/log`.

File contenuti: `dpkg.log`, `faillog`, `mail.err`, `syslog`, `Xorg.0.log`, `messages`, ...

File generalmente presenti in /var/log

File	Program	Where[a]	Freq[a]	Systems[a]	Contents
acpid	**acpid**	F	64k	RZ	Power-related events
auth.log	**sudo**, etc.[b]	S	M	U	Authorizations
apache2/*	**httpd** (v2)	F	D	ZU	Apache HTTP server logs (v2)
apt*	APT	F	M	U	Aptitude package installations
boot.log	**rc** scripts	F[c]	M	R	Output from system startup scripts
boot.msg	kernel	H	–	Z	Dump of kernel message buffer
cron, cron/log	**cron**	S	W	RAH	**cron** executions and errors
cups/*	CUPS	F	W	ZRU	Printing-related messages (CUPS)
daemon.log	various	S	W	U	All daemon facility messages
debug	various	S	D	U	Debugging output
dmesg	kernel	H	–	RU	Dump of kernel message buffer
dpkg.log	**dpkg**	F	M	U	Package management log
faillog[d]	**login**	H	W	RZU	Unsuccessful login attempts
httpd/*	**httpd**	F	D	R	Apache HTTP server logs (in **/etc**)
kern.log	kernel	S	W	U	All kern facility messages
lastlog	**login**	H	–	RZ	Last login time per user (binary)
mail*	mail-related	S	W	all	All mail facility messages
messages	various	S	W	RZUS	The main system log file
secure	**sshd**, etc.	S	M	R	Private authorization messages
sulog	**su**	F	–	SAH	**su** successes and failures
syslog*	various	S	W	SUH	The main system log file
wtmp	**login**	H	M	all	Login records (binary)
xen/*	Xen	F	1m	RZU	Xen virtual machine information
Xorg.*n*.log	**Xorg**	F	W	RS	X Windows server errors
yum.log	**yum**	F	M	R	Package management log

a. Where: S = Syslog, H = Hardwired, F = Configuration file
Freq: D = Daily, W = Weekly, M = Monthly, NN[km] = Size-based, in kB or MB
Systems: U = Ubuntu, Z = SUSE, R = Red Hat and CentOS, S = Solaris, H = HP-UX, A = AIX
b. **passwd**, **login**, and **shutdown** also write to the authorization log. It's in **/var/adm**.
c. Actually logs through syslog, but the facility and level are configured in **/etc/initlog.conf**.
d. Binary file that must be read with the **faillog** utility.

Caratteristiche dei file di log

- Proprietario dei file di log è generalmente `root`.
- Possono essere proprietari anche demoni con privilegi ridotti (ES. `httpd`, `mysqld`).
- La dimensione dei file di log può crescere molto velocemente e possono saturare il disco; per questo motivo conservati su partizioni dedicate.
- Sui sistemi Linux i file di `log` sono solitamente memorizzati in `/var/log/`
- `Logrotate`: tool presente su molte distribuzioni Linux per una gestione efficiente dei file di `log`.

Syslog, logger degli eventi del sistema

Syslog, ha due scopi principali:

- semplificare ai programmatori la gestione dei file di log
- consentire agli amministratori del sistema una controllo più efficiente dei file di log

Syslog consente di raggruppare messaggi per *sorgente* e *importanza* ("severity level") e indirizzarli a differenti destinatari: file, terminali, altre macchine.

Syslog si compone di:

1. `syslogd`: demone adibito al logging
2. `openlog`: routine di libreria per inviare messaggi al demone *syslogd*
3. `logger`: user-level utility per interfacciarsi con il demone utilizzando la shell

Segnale HUP ("hangup", segnale #1) per riavviare syslogd.

Configurare syslogd

Il file `/etc/syslog.conf`[1] è un file testuale contenenti righe nel formato:

`selector <Tab> action`

dove:

selector := facility.level

`syslog.conf` per una macchina indipendente:

```
# syslog.conf file for stand-alone machine

# emergencies: tell everyone who is logged on
*.emerg    *
# important messages
*.warning; daemon,auth.info   /var/log/messages
# printer errors
lpr.debug    /var/log/lpd-errs
```

▸ facility OPTIONS ▸ level OPTIONS ▸ action OPTIONS

[1] **EX:** apt-cache show `rsyslog` + INVIO MSG: esempi/configurare_syslog/

Funzionamento:

1. sudo riceve come argomento una linea di comando che deve venire eseguita con i privilegi di un altro utente[a];
2. `sudo` controlla il contenuto di `/etc/sudoers` che attesta: quali utenti possono invocare comandi tramite `sudo` su quali particolari macchine;
3. se l'utente può invocare quel particolare comando
   - l'utente digita la propria password[b]
   - l'utente esegue il comando senza necessità di digitare la password

[a] non necessariamente root!

[b] **EX:** Per quale motivo?

Esempio di messaggio di log prodotto da sudo

```
Apr 18 10:55:20 salvador sudo:   srdjan : 1 incorrect password attempt ;
TTY=pts/0 ; PWD=/home/srdjan ; USER=root ; COMMAND=/bin/bash
```

Esempio di /etc/sudoers

```
# /etc/sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the man page for details on how to write a sudoers file.
#

Defaults        env_reset

# Host alias specification

# User alias specification

# Cmnd alias specification

# User privilege specification
root    ALL=(ALL) ALL
```

/etc/sudoers

Ogni riga che definisce i permessi specifica:

1. gli utenti a cui vengono concessi i permessi
2. gli host sui quali i permessi vengono concessi
3. i comandi che gli utenti indicati possono invocare
4. gli utenti, con i permessi dei quali, il comando verrà eseguito

Esempio

```
# Define aliases for machines in CS & Physics departments
Host_Alias CS = tigger, anchor, piper, moet, sigi
Host_Alias PHYSICS = eprince, pprince, icarus
# Define collections of commands
Cmnd_Alias DUMP = /sbin/dump, /sbin/restore
Cmnd_Alias PRINTING = /usr/sbin/lpc, /usr/bin/lprm
Cmnd_Alias SHELLS = /bin/sh, /bin/tcsh, /bin/bash, /bin/ksh, /bin/bsh, /bin/zsh
# Permissions
mark, ed    PHYSICS = ALL
herb    CS = /usr/sbin/tcpdump : PHYSICS = (operator) DUMP
studente    ALL = (ALL) ALL, !SHELLS
%wheel    ALL, !PHYSICS = NOPASSWD: PRINTING
```

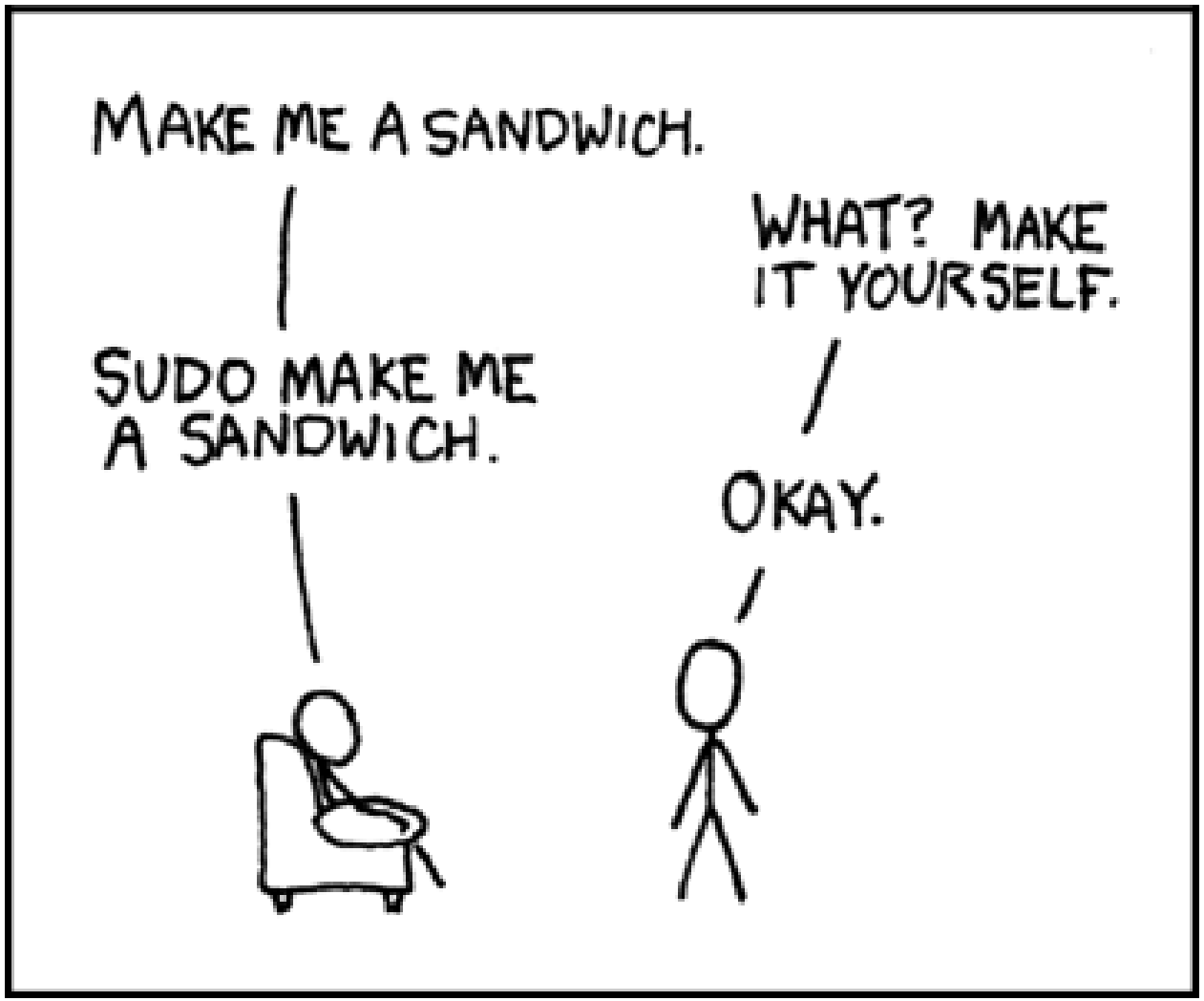
SUDO

Riassunto dei principali vantaggi dovuti all'uso del comando `sudo`:

- logging dei comandi eseguiti (o che si cerca di eseguire!)
- amministratori inesperti potrebbero provocare gravi danni al sistema senza alcuna restrizione sulle operazioni che possono fare
- la password di `root` è a conoscenza di un numero ristretto di persone
- i privilegi concessi all'utente possono venire revocati senza necessità di modificare alcuna password
- esiste una lista. facilmente mantenibile e modificabile, degli utenti che hanno i privilegi di `root`
- si riducono *notevolmente*[2] i rischi di lasciare una shell di `root` incustodita
- un unico file viene utilizzato per la gestione di tutti gli utenti (o addirittura dell'intera rete)

[2] **EX:** In ogni caso???

SUDO - *delega di privilegi*



La gestione degli utenti

Un'*efficiente* gestione degli utenti risulta essere uno dei principali punti di forza di un sistema sicuro.

In un sistema UNIX i tool per la gestione degli utenti sono solitamente: `useradd`, `userdel`, `usermod`.

Numerosi sistemi offrono GUI per semplificare l'aggiunta e la rimozione degli utenti al sistema.

Perchè e come ci si autentica?

Necessità di un meccanismo di identificazione dell'utente all'interno del sistema

- alcune risorse sono riservate *solo* a particolari utenze
- grazie all'identificazione (e quindi ai permessi associati all'utente) si determina la gestione di una particolare risorsa

Identificazione dell'utente all'interno del sistema

- su ogni sistema è presente un'associazione `userid-password`
- **userid**: fornisce l'identificativo dell'utente sul sistema
- **password**: è il token di autorizzazione utilizzato per confermare la propria identità ed accedere a particolari risorse all'interno del sistema

Perchè e come ci si autentica?

Necessità di un meccanismo di identificazione dell'utente all'interno del sistema

- alcune risorse sono riservate *solo* a particolari utenze
- grazie all'identificazione (e quindi ai permessi associati all'utente) si determina la gestione di una particolare risorsa

Identificazione dell'utente all'interno del sistema

- su ogni sistema è presente un'associazione `userid-password`
- **userid**: fornisce l'identificativo dell'utente sul sistema
- **password**: è il token di autorizzazione utilizzato per confermare la propria identità ed accedere a particolari risorse all'interno del sistema

Il file /etc/passwd

cat /etc/passwd

```
root:x:0:0:root:/root:/bin/zsh
studente:x:1000:1000:studente,,,:/home/studente:/bin/bash
...
```

Contiene una riga per ciascun utente presente sul sistema.
Ogni riga, si compone di sette campi (separati da ":").
I campi contengono in ordine:

- login name (username)
- *password cifrata*[3]
- UID
- GID
- informazioni sull'utente (campo GECOS) → comando `chfn`
- home directory
- shell di login

[3] presente solo nelle prime versioni...

Schema di autenticazione utente-password

File necessari per la fase di autenticazione:

- **/etc/passwd**: è leggibile da *tutti* gli utenti; contiene le informazioni *non* sensibili associate a ciascun utente
- **/etc/shadow**: è leggibile solo dall'amministratore del sistema; contiene la password (*criptata*) di ciascun utente e altre informazioni non sensibili

Schema di autenticazione utente-password

File necessari per la fase di autenticazione:

- **/etc/passwd**: è leggibile da *tutti* gli utenti; contiene le informazioni *non* sensibili associate a ciascun utente
- **/etc/shadow**: è leggibile solo dall'amministratore del sistema; contiene la password (*criptata*) di ciascun utente e altre informazioni non sensibili

PROBLEMA:

Ma se solo *root* può leggere il file... Come può un utente generico (ES. *studente*) modificare la propria password senza l'ausilio dell'amministratore???

Schema di autenticazione utente-password

File necessari per la fase di autenticazione:

- **/etc/passwd**: è leggibile da *tutti* gli utenti; contiene le informazioni *non* sensibili associate a ciascun utente
- **/etc/shadow**: è leggibile solo dall'amministratore del sistema; contiene la password (*criptata*) di ciascun utente e altre informazioni non sensibili

PROBLEMA:

Ma se solo *root* può leggere il file... Come può un utente generico (ES. *studente*) modificare la propria password senza l'ausilio dell'amministratore???

ls -l /usr/bin/passwd

```
-rwsr-xr-x 1 root root 45396 May 25 2012 /usr/bin/passwd
```

Schema di autenticazione: cifratura segreto

Cifratura segreto

Vengono utilizzati principalmente due tipi di cifratura per memorizzare la password:

- **DES modificato**[a]: utilizzo di password salting + algoritmo iterato 25 volte
- **hashing**: dato l'hash `Y` è computazionalmente intrattabile trovare un messaggio `M` tale che `hash(M)` = `Y` (ES. MD5, SHA)

[a] oppure il più moderno Blowfish

All'interno del sistema Debian, la scelta dello schema di cifratura viene effettuata modificando `/etc/pam.d/common-password`
Di default viene utilizzato l'algoritmo `SHA512`.

Schema MD5 (o SHA256, o SHA512)

Schema MD5

- Passi dell'algoritmo:
  - L'utente digita la propria password
  - Viene calcolata la funzione hash MD5 sulla password inserita
  - Viene memorizzato il risultato della funzione all'interno del file `/etc/shadow`
- L'algoritmo MD5 risulta più robusto in quanto prende in considerazione l'effettiva lunghezza della password, a differenza della funzione `crypt` che utilizza come chiave di DES la password dell'utente troncata ad 8 caratteri[a].

[a] **EX:** limiti DES: esempi/schema_md5/pwd_len_crypto.c ; man crypt

Il file /etc/shadow

cat /etc/shadow

```
eve:$6$VmuDqVAO$zfZ6nESpA.27cyTEdTBTTwV0xamV...KEAA06VrAAsw4uf7LLpdAR.eNm.:15797:0:99999:7:::
marco:$1$TAAH8OXd$GdtEkNMu/n2p7Z1qgfflY.:15805:0:99999:7:::
paola:$1$iwUBw.ni$vsopxMc4msUSBNnycAJqb1:15805:0:99999:7:::
luca:$5$19glj0mK$YxPyECEnd2ARlOgbZlanNLBL1T4QL3iNl/HeNbFkEbD:15805:0:99999:7::
```

Ogni riga, si compone di 9 campi separati da (separati da ":").
I campi contengono in ordine:

- loging name
- password criptata
- data in cui la password è stata modificata l'ultima volta
- minimo numero di giorni che devono decorrere prima di poter modificare la password
- massimo numero di giorni che devono decorrere prima di poter modificare la password
- numero di giorni di preavviso con cui avvisare l'utente
- numero di giorni, successivi alla scadenza della password, che decorrono prima che l'account venga disabilitato (Linux)
- data di scadenza dell'account
- *campo riservato per usi futuri*

Il file /etc/group

cat /etc/group

studtriennale:x:1020:marco,paolo,giovanni,clarissa
studmagistrale:x:1032:giacomo,ab123456

Gli utenti vengono organizzati in gruppi.
Nelle prime versioni di UNIX un utente poteva appartenere ad un *unico gruppo*.
Ogni riga, si compone di 4 campi separati da (separati da ":").
I campi contengono in ordine:

- nome del gruppo
- *password cifrata** *(opzionale)*
- GID
- lista degli utenti appartenenti al gruppo

Analogamente a `/etc/passwd`, le password possono opzionalmente venire memorizzate in `/etc/gshadow`.

Cracker

- I programmi di Cracking di password sono utili sia per poter verificare le debolezze delle password del proprio sistema, sia per effettuare attacchi informatici.
- Uno dei più veloci cracker di password per sistemi UNIX, ma non solo è *John the Ripper*.
- *oclHashcat-plus* è un innovativo cracker di password che sfrutta le potenzialità della GPU.

John The Ripper

- *John the Ripper* supporta diversi formati di cifratura tra cui: DES, MD5, Blowfish ed è stato testato su diverse architetture: x86, Alpha, SPARC.
- John the Ripper è reperibile all'indirizzo `http://www.openwall.com/john`
  oppure per debian: *apt-get install john*

Modalità operative di John

Modalità operative

- *"Wordlistmode"*: attacco a dizionario puro consiste nel verificare tramite un file dizionario e alcune regole, le password del sistema.
- *"Single crack Mode"*: reperisce dal campo GECOS le informazioni da utilizzare per la costruzione delle password tramite il file delle password da analizzare.
- *"Incremental mode"*: attacco di forza bruta sul file delle password attraverso alcune regole stabilite dal file di conf.
- *"External mode"*: Modalità definita all'esterno e poi passata al programma john.

Per ulteriori informazioni fare riferimento a
`http://www.openwall.com/john/doc/MODES.shtml`

File di configurazione di john

File di configurazione (`/etc/john/john.conf`)

- C'è una sezione dedicata alle opzioni generali
- Regole per la modalità wordlist e single crack
- Regole per l'incremental
- Regole per la modalità esterna

Ulteriori dettagli sulle regole:
`http://www.openwall.com/john/doc/RULES.shtml`

Regole Wordlist

Sezione[List.Rules:Wordlist]

- *-c[rules]*:Non usare la regola se il cifrario non è case-sensitive.
- $>$ *3*: significa rifiuta la parola se è minore di 3 caratteri.
- *l*: rendi tutti i caratteri minuscoli.

Regole Single

Sezione[List.Rules:Single]

- *"1"*: solo la prima parola dell'account.
- *"2"*: solo la seconda parola dell'account.
- *"+"*: concatenazione delle parole.

Regole Incremental

Sezione[Incremental:NOME], NOME id

- *"MinLen"*: lunghezza minima di password da cercare.
- *"Maxlen"*: lunghezza massima di password da cercare.
- *"Charcount"*: numero massimo di caratteri diversi da impiegare nella ricerca.
- *"Extra"*: aggiunge un altro set di caratteri per la composizione della stringa di ricerca.

Esempi di utilizzo di john

l'utilizzo di `john` per l'estrazione del file password:

```
# unshadow /etc/passwd /etc/shadow > passwordFILE
```

l'utilizzo di john per la modalità wordlist:

```
# john -wordfile:wordsFILE -rules passwordFILE
```

l'utilizzo di john per la modalità single crack:

```
# john -single passwordFILE
```

Esempi di utilizzo di john

l'utilizzo di john per la modalità incremental:

```
# john -incremental:alpha passwordFILE
```

l'utilizzo di john per mostrare le password trovate finora:

```
# john -show passwordFILE
```

l'utilizzo di john per ripristinare la fase di cracking:

```
# john -restore
```

esercizio logging

0x00

Identificare il file contenente le informazioni di logging associate al processo sudo.
All'interno di tale file trovare tutte le invocazioni del comando sudo, e memorizzarle all'interno del file `/tmp/sudo_invocations.txt`.
Effettuare la stessa operazione individuando le invocazioni di sudo che hanno avuto successo e quelle che non sono andate a buon fine; memorizzare l'output rispettivamente in `/tmp/sudo_successful.txt` e `/tmp/sudo_unsuccessful`.

esercizio logging+sudo

0x01

Consentire all'utente *studente* di eseguire `/bin/bash` utilizzando `sudo` dopo aver digitato la password.
Realizzare uno script che monitora le invocazioni di `sudo` effettuate dall'account `studente`; qualora dopo 3 invocazioni a distanza di 1 minuto, il comando non abbia avuto successo:

- l'utente viene disconnesso
- l'account di `studente` viene temporaneamente disabilitato

HINT

Comandi utili:

- `cat`, `grep`, `usermod`, `date`, `tail`

Esempio:

```
echo $(( $(date -d ‘‘Apr 18 10:55:26’’ +%s)-$(date -d
‘‘Apr 18 10:55:20’’ +%s) ))
```

esercizio sudo

0x02

Utilizzando il file `/etc/sudoers` simile a quello visto nell'▸ Esempio, ottenere una shell di `root` utilizzando l'account di studente.

HINT

È sufficiente fare riferimento alla riga di permessi associata all'utente `studente`...

esercizio sudo

0x03

Consentire a `bob`, `alice` ed `eve` di utilizzare utility per la creazione di archivi (`zip`, `tar`, ...) memorizzando i log contenuti in `/var/log/` e relativi all'invio di posta elettronica.

esercizio gestione utenti+sudo

0x04

Utilizzare il comando `newusers` per aggiungere al sistema gli utenti: `john`, `paul`, `ringo`, `george`.
Modificare opportunamente il file `sudoers` per consentire ai 4 nuovi utenti di creare file system di tipo ext sulla prima partizione del device `/dev/sdb`.

esercizio cracking

0x05

Utilizzando le diverse modalità di `John the Ripper` cercare di individuare il maggior numero possibile di password relative al file: `http://security.di.unimi.it/~srdjan/crack-me.txt`

Esercizio setuid+logging*

douser

Realizzare un programma set-user-ID-root simile a *sudo*. Il programma deve ricevere in input da linea di comando comandi e opzioni nel seguente formato:
`$ ./douser [-u user ] program-file arg1 arg2 ...`
Il programma *douser* esegue *program-file*, con i rispettivi argomenti, come se quest'ultimo fosse eseguito dall'utente *user*. (Se l'opzione *"-u user"*, viene omessa, l'utente di default è *root*). Prima di eseguire *program-file*, il programma *douser* deve richiedere all'utente di autenticarsi utilizzando la propria password (e verificando la correttezza di questa nell'apposito file) e solo in seguito ad una corretta autenticazione, impostare per il processo i corretti valori di user e group ID associati a quel particolare utente[a].

[a] Esercizio 38-2, preso da pag. 796 "The Linux Programming Interface", Michael Kerrisk, ed. No Starch Press, ottobre 2010

Extra:

Con l'ausilio della system call `system` e dell'utility `logger`, modificare il precedente programma per inviare messaggi di logging a `syslogd`.

Link e materiale utile

- "Unix and Linux System Administration Handbook", Evi Nemeth - Garth Snyder - Trent R. Hein - Ben Whaley, ED. Pretience Hall, 4th ed.
- "The Linux Programming Interface", Michael Kerrisk, ED. no starch press
- "Linux Command Line and Shell Scripting Bible", Richard Blum, ED. Whiley

APPENDICE - Syslog facility names

Facility	Programs that use it
*	All facilities except "mark"
auth	Security and authorization-related commands
authpriv	Sensitive/private authorization messages
cron	The **cron** daemon
daemon	System daemons
ftp	The FTP daemon, **ftpd**
kern	The kernel
local0-7	Eight flavors of local message
lpr	The line printer spooling system
mail	**sendmail** and other mail-related software
mark	Time stamps generated at regular intervals
news	The Usenet news system (obsolete)
syslog	**syslogd** internal messages
user	User processes (the default if not specified)
uucp	Obsolete, ignore

▸ Configurare syslogd

APPENDICE - Syslog severity levels (descending)

Level	Approximate meaning
emerg	Panic situations
alert	Urgent situations
crit	Critical conditions
err	Other error conditions
warning	Warning messages
notice	Things that might merit investigation
info	Informational messages
debug	For debugging only

▸ Configurare syslogd

APPENDICE - Syslog actions

Action	Meaning
filename	Appends the message to a file on the local machine
@hostname	Forwards the message to the **syslogd** on *hostname*
@ipaddress	Forwards the message to the **syslogd** on host *ipaddress*
|fifoname	Writes the message to the named pipe *fifoname* [a]
user1,user2,...	Writes the message to the screens of *users* if they are logged in
*	Writes the message to all users who are currently logged in

a. See **info mkfifo** for more information (Linux versions of **syslogd** only).

▸ Configurare syslogd