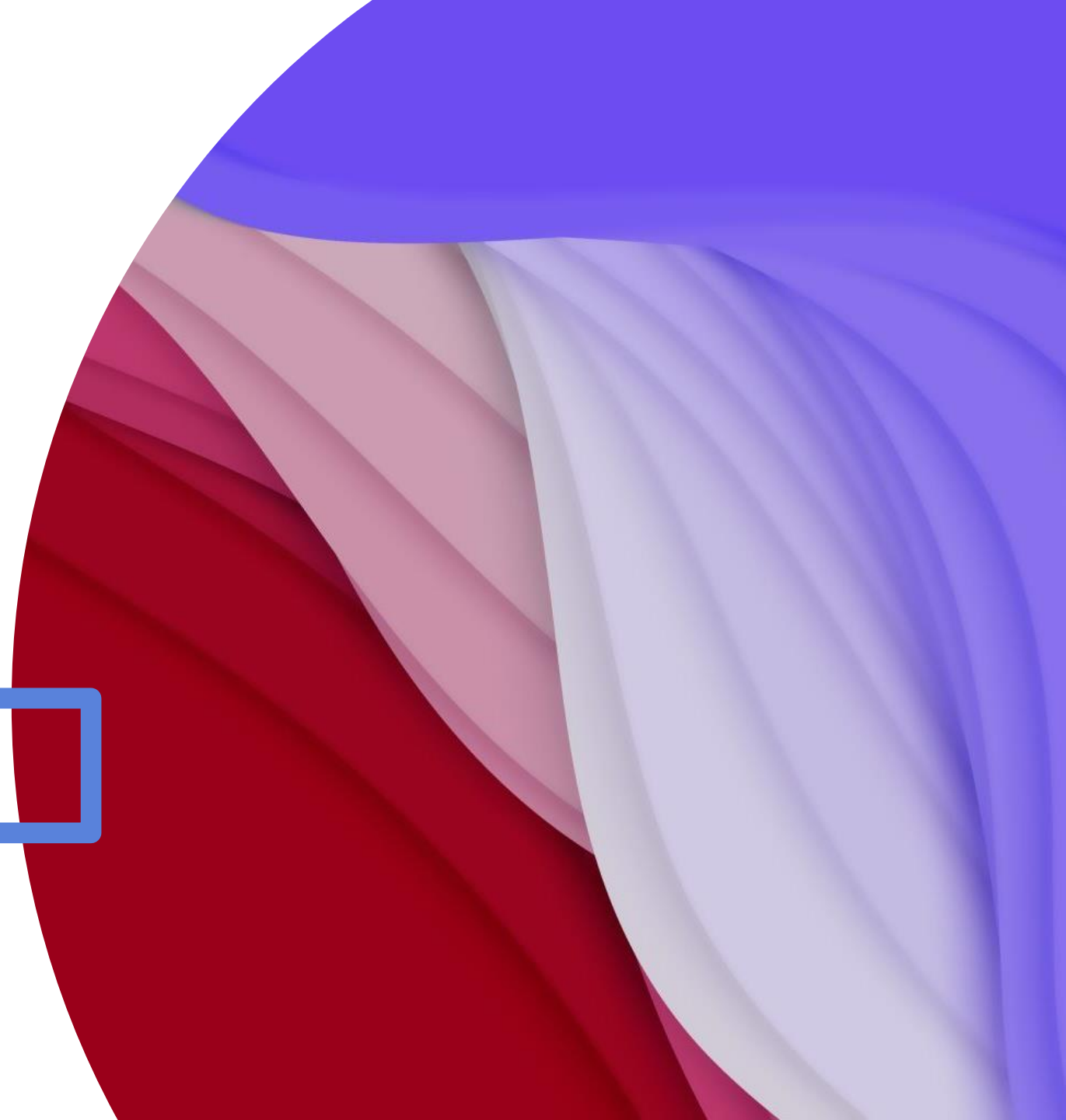
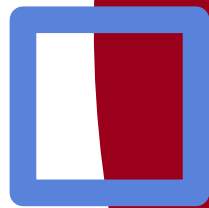
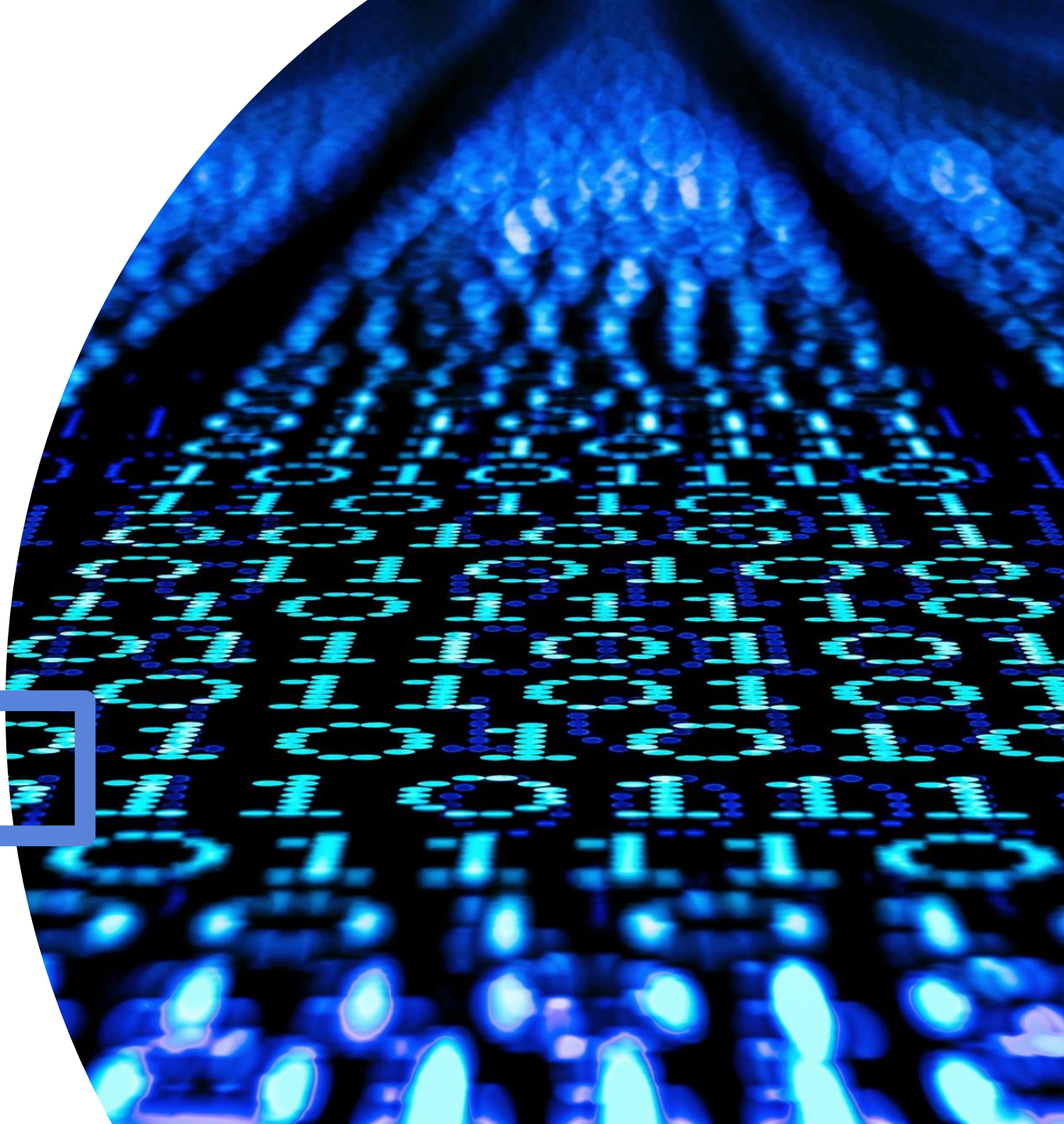


Appunti.odt



Funzionamento
del sistema
GNU/Linux



Funzionamento
di base



Moduli per la gestione hardware



Dispositivi a blocchi: usano buffer e cache per ottimizzare il trasferimento di blocchi di una certa dimensione



Dispositivi a caratteri: trasferiscono dati un carattere alla volta, possono consumarli per farci qualcosa o fornirli se ne hanno disponibili, lasciando il consumatore in attesa se non ne hanno



Dispositivi vari



Moduli per la gestione hardware



- Un bit speciale per identificare se sono a blocchi o a caratteri
- Un major number per identificare quale device driver usare
- Un minor number per indicare l'istanza di dispositivo di quella classe
- `/dev/zero` produce uno stream infinito di zeri binari
- `/dev/null` restituisce EOF a ogni read e ogni write viene scartata
- `/dev/random` produce byte casuali ad alta entropia, bloccante ove non ce ne sia abbastanza
- `/dev/urandom` produce stream pseudocasuale infinito
- `/dev/tty*` terminali fisici del sistema
- `/dev/pts/*` pseudo-terminali
- `/dev/sd*` dischi e partizioni

Il sistema di storage

- Partizionamento: suddivisione in sottoinsiemi di blocchi
 - MBR in cui si ha una tabella principale con massimo 4 partizioni primarie , e una di queste può essere segnata come partizione estesa contenente fino a 12 unità logiche
 - GPT usato specialmente con UEFI, permette di avere un massimo di 128 partizioni da 8 ZB invece delle 15 da 2 TB offerte da MBR





Il sistema di storage

- Formattazione: creazione del filesystem per organizzare lo spazio in modo comprensibile





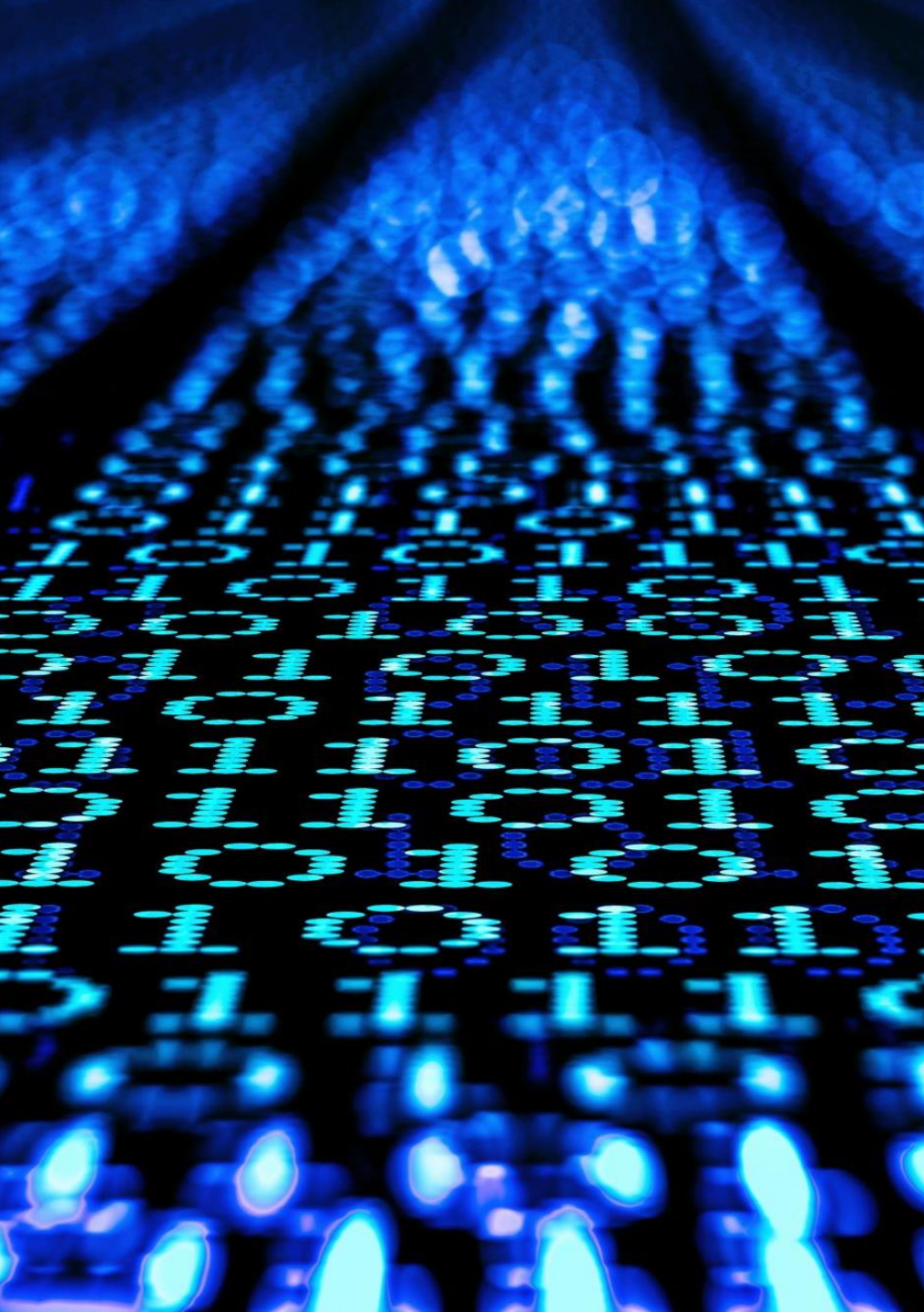
Il sistema di storage

- Mount: innestamento della gerarchia locale creata dal partizionamento nella gerarchia globale
 - 1) `mount /dev/sda1 /`
 - 2) `mount /dev/sda5 /home`

Il processo di avvio

- il BIOS individua i dispositivi di possibile caricamento di bootloader per esaminarli in un certo ordine
- il bootloader sceglie il sistema operativo
- il sistema operativo carica di device driver e lancia il processo init
- init avvia i servizi necessari all'inizializzazione del sistema nell'ordine corretto





Collocazione delle risorse

- Dati statici / variabili
- Dati condivisibili / non condivisibili



Gestione dei pacchetti software

- Ogni sistema ha bisogno di un'oculata gestione del software installato, dal momento della sua installazione fino a quello della disinstallazione: generalmente parlando è sempre opportuno disinstallare software che non ci serve più
- La fase dell'aggiornamento dei pacchetti è particolarmente critica, in quanto nuove versioni potrebbero interrompere il supporto ad alcune componenti hardware o software del sistema, o causare problemi con altri pacchetti con cui c'è dipendenza
- La configurazione dei software è altrettanto importante in quanto deve tenere conto delle politiche di utilizzo ed eventualmente prevedere particolari casi d'uso o caratteristiche del sistema

Installazione manuale



reperimento software

E' importante autenticare i pacchetti usati, solitamente mediante una chiave pubblica fidata o come minimo verificando il fingerprint del file



estrazione pacchetto

I file vengono tipicamente prima compattati in un solo tar, e poi compressi



esame delle scelte offerte

Nella directory data dall'estrazione è opportuno leggere eventuali README e INSTALL



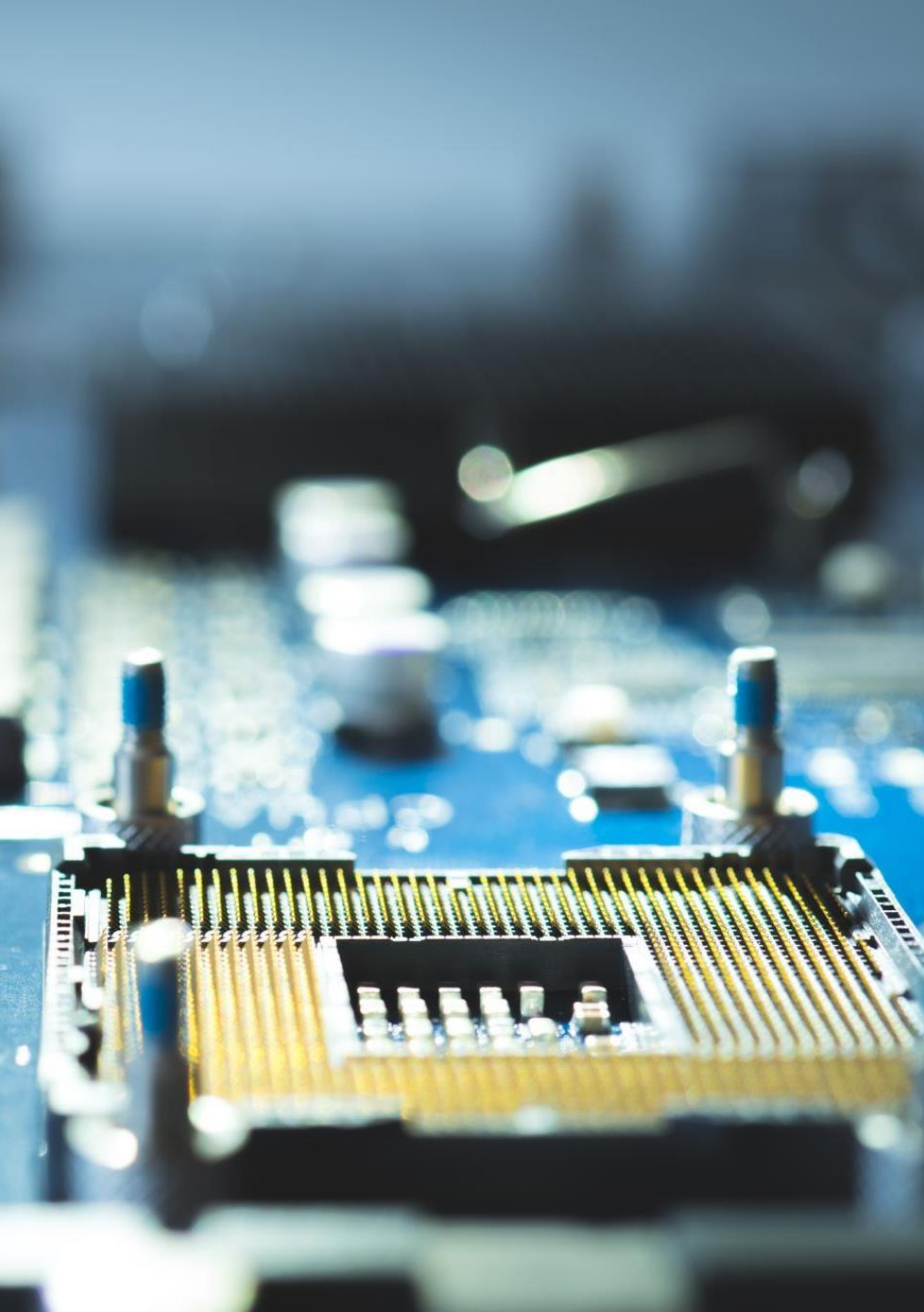
configurazione sorgenti

Si lancia configure con i parametri scelti, risolvendo eventuali problemi evidenziati



compilazione

Si lancia make o si seguono le indicazioni del config.log

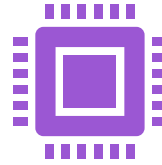


Installazione manuale

- installazione, usando `sudo make install`



Installazione assistita



L'installazione assistita si effettua tipicamente usando un package manager, nelle distribuzioni Linux



Il tool si fa carico di verificare le dipendenze, scorrendo il grafo delle dipendenze di un dato pacchetto individuando tutti i componenti di cui ha bisogno



Un pacchetto tipicamente si presenta come un singolo file che contiene sia il software precompilato che i criteri per la verifica di compatibilità e prerequisiti



Distribuzioni

- Architetture supportate: tutte le distribuzioni supportano Intel 32bit, alcune 64bit, altre sono disponibili per la grande varietà di processori su cui è stato portato il kernel, ma è bene ricordare che il supporto a pacchetti di terze parti non è garantito per architetture esotiche
- Stabilità vs aggiornamento: in GNU/Linux il processo di rilascio frequente di software porta a dover confrontare i benefici dell'avere versioni più recenti del software, con più funzionalità, con la stabilità di versioni già ampiamente testate; e la stabilità è importante su un server in produzione
- Version vs rolling: Le distribuzioni con versione forniscono solo aggiornamenti correttivi durante tutto il loro ciclo di vita





Distribuzioni

- Supporto e durata: Il supporto garantito è tipico delle distribuzioni commerciali, per quelle gratuite dipende dalla dimensione della comunità di utenti



Pacchetti



prerequisiti: pacchetti necessari perché il candidato funzioni, come per l'installazione



configurazione: si pensi a modifiche al formato delle configurazioni già messe a punto per la versione funzionante



dipendenze di altri software: modifiche alle interfacce mostrate potrebbero influire sul funzionamento di altri software

Vagrant e
SSH





SSH

- Negoziazione dei cifrari disponibili
- Autenticazione host remoto per mezzo della sua chiave pubblica
- Inizializzazione canale comunicazione cifrata
- Negoziazione dei metodi disponibili per l'autenticazione utente
- Autenticazione utente





SSH

- generare una coppia di chiavi asimmetriche
- installare sull'host remoto la chiave pubblica
- copia su host remoto
- append alla lista di utenti autorizzati



Vagrant



Vagrant è una libreria per la gestione di macchine virtuali che permette di rendere portabile la configurazione rispetto a diversi virtualizzatori



L'immagine della VM di base può essere depositata e facilmente importata da un deposito, i parametri per istanziare una VM sono definiti in un file



Un V.File contiene informazioni sulla box di base, sui vari tipi di interfacce di rete, la mappatura di cartelle condivise host-guest, il provider del virtualizzatore e la configurazione dell'HW virtuale, oltre che la configurazione d'ambiente, il provisioning: una serie di comandi eseguiti da vagrant nella shell del guest al primo avvio

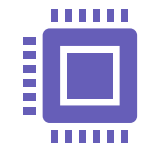
Vagrant + SSH



vagrant up crea automaticamente una chiave privata sull'host, nella directory in cui viene fatta init, e installa la chiave pubblica nella VM, predisponendo una mappatura di rete da una porta host alla 22 guest



vagrant ssh usa in automatico queste impostazioni per connettersi alla VM come user vagrant, senza password, abilitato a sudo



vagrant ssh-config mostra come configurare il client ssh per comportarsi come vagrant ssh, in modo da poter usare ssh originale per operazioni non supportate da vagrant ssh, come scp



Gestione servizi

- Ogni utente ha la propria crontab , non importa sapere dove si trovano perché ognuno può configurarle usando crontab -e che apre editor
- I task di sistema si trovano in /etc/crontab, questo editabile solo da root e solo direttamente; a differenza di crontab contiene un campo in più che indica a nome di quale utente dovrà essere eseguito un certo comando, perché se root vuole pianificare task per altro utente non va a modificare il crontab di quell'utente ma modifica /etc/crontab
- Ogni crontab ha un elenco di direttive nella forma



Gestione servizi

- atd è il demone che si occupa di gestire code di compiti da svolgere in momenti prefissati
- ATJOB=\$ catturo il job id per salvarlo da qualche parte
- Watchdog: Utile se si vuole porre un limite al tempo di esecuzione di un task, senza ricorrere a un busy wait



Gestione servizi

- `at` TIME prende come parametro quando fare un job, quando si fa invio poi mostra altro prompt, perché si aspetta l'inserimento dei comandi da eseguire POI sullo stdin
- `atq` elenca i comandi in coda, formato
- `atrm job` rimuove comandi dalla coda
- AT E CRON SONO DEMONI, gli stream vanno obbligatoriamente ridiretti, se non ci interessano si ridirigono a `/dev/null`
 - `PATHCOMANDO="$ ")` salva nella variabile il percorso assoluto dello script in cui è contenuto

Systemd

- gestire le dipendenze tra i servizi: ha senso avviare alcuni servizi soltanto se sono già avviati i servizi su cui si basano così come quando si installa software ha senso installare un pacchettoB solo se esiste già pacchettoA da cui dipende
- avviare servizi a richiesta: se in una sequenza di passi c'è dipendenza ad es
- logging precoce: il sistema di log è un demone, ma anche se lo lancio presto, tutto ciò che è partito prima di lui non può essere registrato



Systemd

- `systemctl` è il comando con cui si controlla il funzionamento del sistema



Systemd



`systemctl list-units` mostra tutte le unit gestite di tutti i tipi elencati



`systemctl -t type` mostra tutte le unit attive del tipo specificato



`systemctl list-unit-files` mostra tutte le unit installate del tipo specificato



`systemctl --state state` mostra tutte le unit che si trovano nello stato specificato



Systemd


- Requires elenco di altre unit di cui questa necessita: se l'avvio di queste fallisce, questa viene arrestata
- Wantsdipendenze di questa meno strette di Requires, viene tentato l'avvio delle altre unit ma se fallisce, questa viene avviata comunque
- Conflicts vincolo negativo per rendere unit mutualmente esclusive
- OnFailure unit da avviare quando questa fallisce
- RequiredBy / WantedBy specifica dipendenza inversa: quando si installa questa unit vengono informate le altre che hanno bisogno di lei che esiste; aggiungendo automaticamente entry Requires / Wants
- Restart riavvia il servizio in caso di terminazione; è l'equivalente del respawn di init che si occupava di garantire il riavvio dei servizi fondamentali

Scrittura di unit file

- Description= Descrizione
- Requires/Wants= da chi dipende questa unit
- Documentation= inserire indicazione su dove trovare documentazione agevola la vita al sistemista
- Type= tipo di avvio
- ExecStart= comando da lanciare all'avvio
- Exec = opzionali, comandi per stop e reload
- Restart= opzionale, reazione alla terminazione



Scrittura di unit file

- WantedBy= chi dipende da questa unit
 - Alias= nome con cui la unit è nota a systemd
- 



Scrittura di unit file

- Type=simple default, se la exec fallisce, non ce ne accorgiamo
- Type=forking fa eseguire un processo in background, con un pid che si può catturare e mettere in un file
- Type=oneshot per lanciare un processo una volta e poi uscire, es





Monitoraggio



- systemd ha integrato il logging di Linux, tenuto in un journal
 - I messaggi dai processi vengono serializzati, poi
 - viene posto un timestamp: importante per stabilire relazione causale tra due eventi, quando si diagnostica un problema, poi
 - vengono classificati: permette di gestire dove finiscono i messaggi inviandoli a diverse destinazioni



Monitoraggio

- `/etc/syslog.conf` contiene le regole di configurazione per lo smistamento
 - `kern.* /dev/console` invia su device speciale che fa vedere i messaggi sulla console tutti i messaggi di facility kern di qualsiasi priorità
 - `*.info;mail.none; /var/log/messages` invia su quel file messaggi con qualunque facility con importanza info o superiore, con esclusione di quelli che hanno facility mail
 - `*.emerg *` invia ai terminali di tutti gli utenti messaggi con priority pari o superiore ad emerg
 - `kern.crit "| /usr/bin/alerter"` | significa che lancia processo figlio e tutti messaggi vengono inviati su stdin del processo indicato
 - `*.=warning @loghost` invia al server di nome loghost SOLO i messaggi con priority warning



Monitoraggio

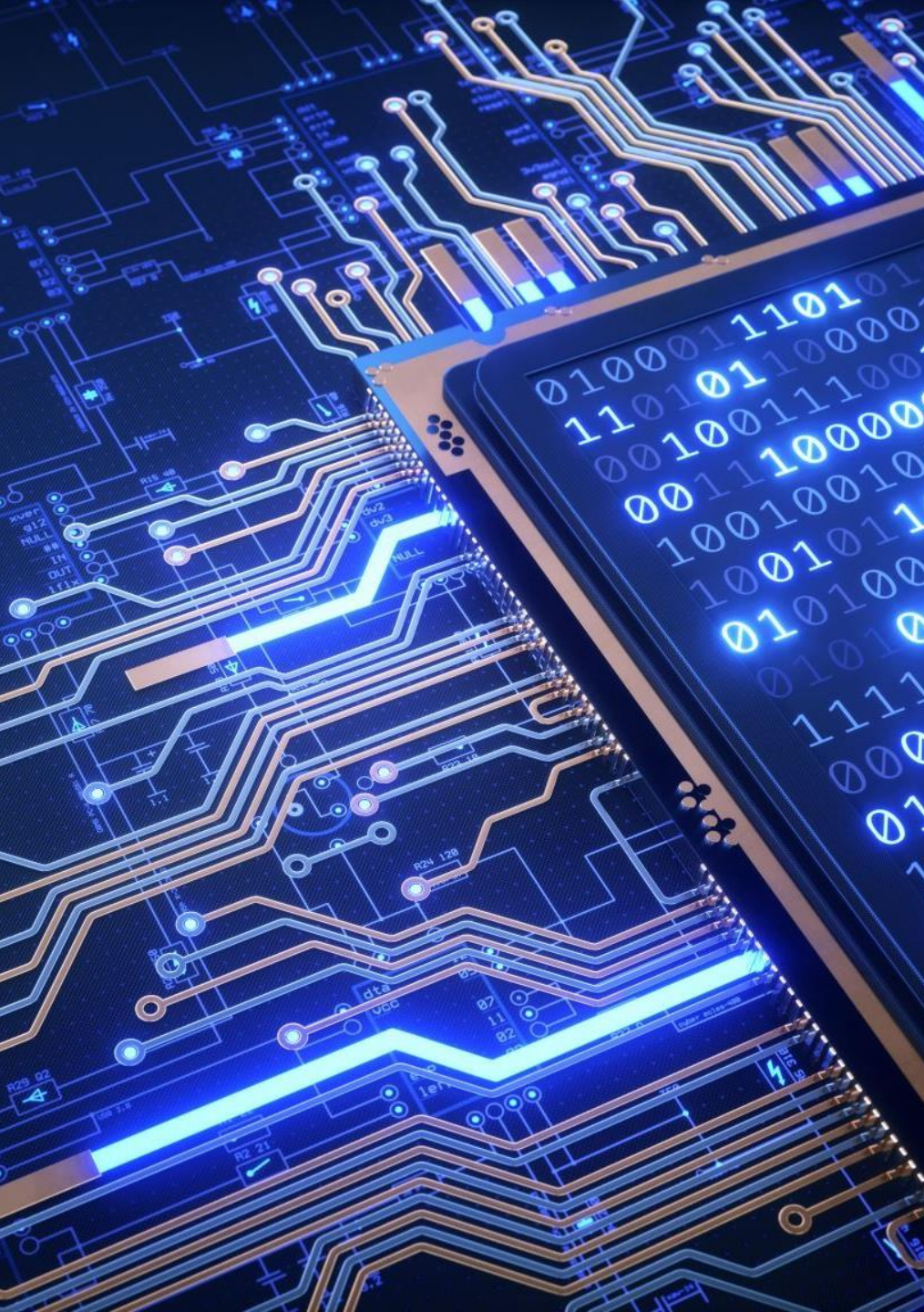


- rsyslog è un'evoluzione, può ricevere messaggi anche da socket, api di identificazione.
 - attivazione della ricezione di messaggi via rete
 - `$ModLoad imudp / module`
 - `$UDPServerRun 514 / input`
 - integrazione del kernel logging
 - `$ModLoad imklog / module`
- logger è il comando per inviare messaggi di log a rsyslog, quello che si invia al syslog non è esattamente ciò che finisce nel file: viene aggiunta una serie di informazioni utili alla gestione del sistema
 - un timestamp
 - il nome della macchina che ha prodotto il messaggio
 - il nome dell'utente che ha prodotto il messaggio, poi duepunti e il messaggio vero e proprio

Monitoraggio

- ps ha moltissime opzioni avendo tre diverse sintassi
 - la sezione PROCESS SELECTION BY LIST mostra come ottenere una lista di processi secondo le loro proprietà , molto meglio che fare grep dell'intera lista di processi
 - la sezione OUTPUT FORMAT CONTROL mostra come formattare la lista prodotta -o, o, --format seguite da una stringa di specificatori documentata nella sezione STANDARD FORMAT SPECIFIERS permettono un controllo completo sui campi che si vogliono far comparire nella lista)
 - la sezione PROCESS STATE CODES spiega il significato della colonna STAT e dà un'indicazione fondamentale dello stato del processo
- uptime riporta il tempo trascorso dal boot e il carico del sistema: lo scheduler registra il numero totale di processi in stato R o D






Monitoraggio

- free mostra informazioni sulla memoria
- top riassume ps, uptime, free + uso dettagliato cpu, è interattivo e viene aggiornato regolarmente, permette di interagire coi processi
 - CPU scarica – molti processi in D → un dispositivo non risponde
 - CPU usata principalmente in userspace – sistema CPU-bound
 - CPU usata principalmente in iowait – sistema I/O-bound
 - indagini più approfondite con vmstat e iostat disponendo di una baseline



Monitoraggio

- File indicato con nome assoluto
 - STDIN di un processo, identificato da pipe verso il programma da lanciare
 - Utenti collegati username, o * per tutti
 - Server di syslog remoto@indirizzo o @nome
 - A default comunicazione su UDP porta 514, dove ci sarà un suo syslog che gestisce, l'unica cosa che non potrà fare è fare un altro salto
- 

Utenti e file

- useradd serve per creare nuovi utenti, con granularità fine e grande possibilità di automazione
 - -m crea la home dell'utente seguendo come template i file dentro /etc/skel
 - -s assegna la shell all'utente, le possibili shell sono indicate dentro il file /etc/shells, altrimenti prende il default
 - -U crea un gruppo con lo stesso nome dell'utente
 - -K con questo parametro è possibile specificare la `UMASK=0077`
 - -G per assegnare l'utente ad un gruppo supplementare alla creazione
 - /etc/passwd, formatoprandini:x:500:500:Marco Prandini:/fat/home:/bin/bash
 - Accessibile a tutti, tutti i dati pubblici dell'utente



Utenti e file

- useradd serve per creare nuovi utenti, con granularità fine e grande possibilità di automazione
 - Primo argomento è username, il secondo parametro è una x che ricorda la retrocompatibilità con il campo con password, importante tenerlo così che il sistema cerchi corrispondenza in /etc/shadow
 - Poi userid numerico e groupid numerico
 - Poi campo informazioni, home directory dell'utente, cwd da cui parte all'avvio il sistema
 - Infine il programma avviato normalmente al login utente
- /etc/shadow, formato
prandini:\$1\$/PBy29Md\$kjC1F8dvHxKhvMTWelnX/:12156:0:99999:7:
 - Linee corrispondenti a passwd ma accessibile solo a root
 - Lasciare passwd pubblico è pratico: ad esempio per leggere gli id numerici utente, da usare in diversi contesti per identificare gli utenti

Utenti e file

- useradd serve per creare nuovi utenti, con granularità fine e grande possibilità di automazione
 - Contiene dati sulla validità temporale della password, esaminabili e modificabili con chage: quando si crea un nuovo utente, i valori di default vanno presi da qualche parte: alcuni sono in /etc/login.defs , altri in /etc/default/useradd , utili per configurare il sistema con valori arbitrari
- id <USER> riporta tutte le informazioni di identità
- usermod permette di modificare, coi suoi diversi parametri, tutte le caratteristiche dell'utente (come useradd ma può essere usato solo da root. Esiste anche una serie di comandi specifici per cambiare singole proprietà, usabili da utenti standard per agire sul proprio account
- chsh modifica la shell di login

```
mirror_mod = modifier_ob.  
set mirror object to mirror.  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.  
mirror_ob.select = 0  
= bpy.context.selected_object  
data.objects[one.name].select  
print("please select exactly  
-- OPERATOR CLASSES --  
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
context):  
context.active_object is not
```

Utenti e file

- `chfn` modifica del nome reale
- `passwd` modifica della password
- `adduser` non è standard in tutte le distro e chiede i dettagli interattivamente, quindi utile se usato in maniera estemporanea, ma non se dobbiamo scriptare creazione utenti
- `groupadd` e `addgroup` per la creazione di gruppi, analogamente
- `gpasswd` modifica password e lista utenti di un gruppo
- `getent` interroga il db utenti o gruppi
- `last` mostra i login effettuati sul sistema, `lastlog` mostra la data di ultimo login di ogni utente, `faillog` mostra i login falliti sul sistema



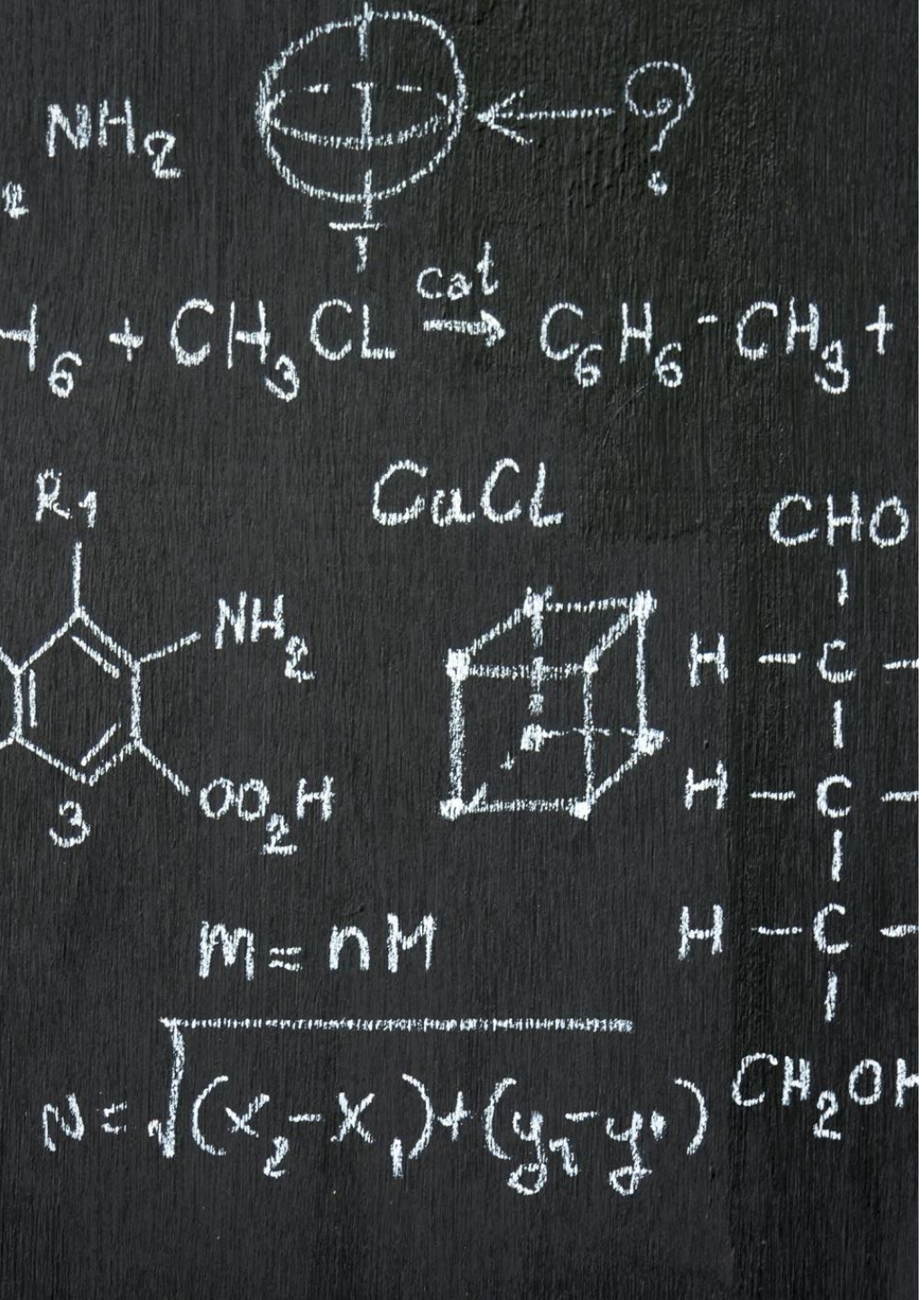
Utenti e file

- R read: permette la lettura di un file, per le directory equivale a leggere la prima colonna
- W write: scrittura dentro un file, aggiunta/cancellazione/rinomina di file in una directory
- X execute: esegue il file come programma, permette lookup tra nome e inode della directory
- viene fatta alla creazione di un file, l'utente creatore è assegnato come proprietario del file, il gruppo attivo dell'utente creatore è assegnato come gruppo proprietario
- Per cambiarla dopo la creazione del file, si usano
 - `chown : <file>` modifica owner e/o group owner del file
 - `chgrp <file>` cambia group owner del file
- alla creazione di file viene fatta seguendo logica negativa, ovvero con tutti quelli sensati meno quelli della `umask`



Utenti e file

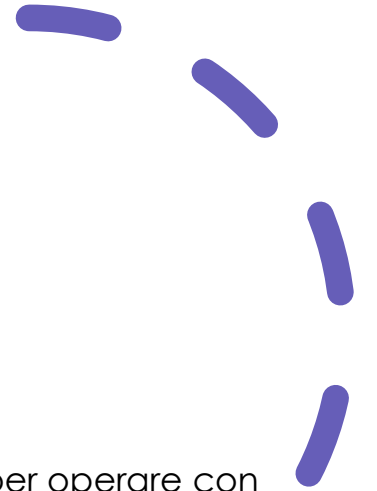
- Successivamente alla creazione, si usa `chmod` per modificare i permessi
 - modo numerico in base ottale, es
 - `chmod 2770 miadirectory` sarà SUID SGID STICKY `rwx rwx -`
 - `chmod 4555 miocomando` sarà SUID SGID STICKY `r-x r-x r-x`
 - modo simbolico utile per modifiche a singoli bit
 - soggetto dice in quale terzina stiamo operando
 - operatore
 - permesso
 - esempio `chmod 'a=r,g-rx,u+rw'` file all'esatto ad `r`, group privato di `r` e `x`, user dotato di `r` e `w`





Utenti e file

- real user id U
- real group id G
- effective user id , identità assunta dal processo per operare con soggetto diverso da U
- effective group id , stessa cosa ma per identità di gruppo assunta dal processo per operare come soggetto diverso da G
- Applicati ai file hanno senso solo se eseguibile, possono fare in modo che euid e/o egid siano diversi dai corrispondenti ruid/rgid
 - bit 11 SUID 1 fa sì che un programma eseguito da utente qualsiasi, venga lanciato in processo che gira con identità dell'utente proprietario
 - bit 10 SGUID stessa cosa ma per identità del gruppo
 - bit 9 STICKY bit è obsoleto, dice al SO di tenere in cache una copia del programma...





Utenti e file

- Applicati alle directory
 - bit 11 SUID non è usato
 - bit 10 SGID serve a poter cambiare automaticamente membership di un utente quando entra a lavorare in una directory
 - preconditione: SGID settato, l'utente appartiene anche al gruppo proprietario della directory
 - effetto: l'utente assume come gruppo attivo quello proprietario della directory, i file creati hanno quel gruppo proprietario
 - vantaggi, mantenendo una umask 006: nelle aree collaborative i file sono automaticamente resi leggibili e scrivibili da tutti i membri del gruppo
 - bit 9 Temp le directory temporanee , hanno un problema, chiunque può cancellare un file





Comandi utili per lavorare su file

- Elenco e navigazione

- Il filesystem Linux è un unico albero anche con più dischi fisici, la radice è '/' e tutti i path completi iniziano da questa
- pwd mostra directory di lavoro e tutti i path relativi si riferiscono alla pwd, i processi continuano ad occupare la pwd dov'era l'utente che li ha lanciati
- cd <destinazione> per cambiare directory
- In ogni directory D si hanno le subdir '.' che coincide con D e '..' che coincide con la dir superiore



Comandi utili per lavorare su file

- Analisi dei metadati
 - ls per elencare contenuto di directory, opzioni
 - -l abbina al nome le info associate al file
 - -a non nasconde i file che iniziano per '.'
 - -A come -a ma nasconde i file '.' e '..'
 - -F postpone carattere '*' a eseguibili e '/' alle directory
 - -d lista nomi dir senza listarne contenuto al contrario del default, può essere utile quando i nomi sono espansi partendo da wildcard
 - -R percorre ricorsivamente la gerarchia
 - -i mostra gli i-number dei file oltre al nome
 - -r inverte l'ordine elenco

Comandi utili per lavorare su file

- Analisi dei metadati
 - -t lista i file in ordine di data/ora di modifica
 - -l forza output singola colonna
- Metadati per ls -l: formato -rw-r--r-- 1 root root 1514 Mar 29 11:07 /etc/passwd
 - primo bit indica il tipo: - file standard, d directory, l link simbolico, b block special, c character special, p named pipe , s socket
 - il numero dopo i permessi indica i link allo stesso inode, possono aversi più nomi per lo stesso inode
 - seguono utente e gruppo, dimensione, e data di modifica
 - Link simbolici: anche detti softlink, sono un nome diverso da quello principale dato a un file - a differenza dell'hardlink, memorizzato in una directory che punta a un'inode
 - L'hardlink riferenzia un inode direttamente: si possono creare hardlink SOLO a inode esistenti



Comandi utili per lavorare su file

- Analisi dei metadati
 - I softlink sono utili per creare link tra diversi filesystem
- Ogni file ha 3 o 4 timestamp distinti Xtime (
 - mtime modifica contenuto
 - atime accesso al contenuto
 - ctime modifica metadati
 - wtime creazione del file, se supportato)
 - gestiti dal FS, possono essere cambiati a mano con touch
 - per estrarre i metadati, comando stat, formato
 - `stat --format='%U %a %z' filepath` con U utente proprietario, a permessi, z ctime



Comandi utili per lavorare su file

- Creazione e generazione file
 - rm – cancella un file o meglio rimuove il link
 - cp – copia 1+ file in una dir, attenzione con file device e softlink: copia concetto o contenuto?
 - mv – sposta 1+ file in una dir
 - ln – crea un link a file
 - mkdir – crea dir
 - rmdir – cancella dir: deve essere vuota, con opzione -r cancella ricorsivamente

Comandi utili per lavorare su file

- Ricerca nel filesystem
 - `find`, ricerca in tempo reale il FS, occhio al carico indotto
 - `-name 'expr'` Nome contiene espressione, es
 - Timestamp entro periodo
 - `-size +N-M` dimensione entro limiti
 - Tipo specifico
 - Di proprietà di utente o gruppo specificato, o "orfani"
 - Permessi di accesso specificati
 - Esempio: `find /usr/src -name '*.c' -size +100k -print`
 - ricerca sotto `/usr/src` tutti i file che finiscono per `.c`, hanno dimensione maggiore di 100K, ed elencarli sullo standard output

Comandi utili per lavorare su file

- Ricerca nel filesystem
 - E' possibile eseguire un comando su ogni oggetto individuato con l'opzione `-exec comando \`
 - `Es`
 - `Find` trova in / i file regolari, orfani, modificati meno di 2 giorni fa , su ognuno di essi viene invocato `grep` che ricerca TXT nel contenuto
 - Il comando `locate` ricerca su un db indicizzato , con meno carico sul sistema rispetto a `find`, ma si può specificare solo pattern nel nome e le risposte possono essere obsolete per modifiche effettuate dopo l'esplorazione
 - Comando `file` per identificare il contenuto di un file, fa tre test
 - 1) usa `stat` per capire se il file è vuoto o speciale
 - 2) usa il DB dei "magic number"

Comandi utili per lavorare su file

- Ricerca nel filesystem
 - 3) usa metodi empirici per capire se è un file di testo, in tal caso quale sia la lingua naturale o linguaggio di programmazione
- Due formati dei file di testo
 - nei sistemi UNIX le linee sono terminate da un solo carattere, line feed o LF o \n o 0x0A
 - nei sistemi DOS/Windows le linee sono terminate da due caratteri: carriage return line feed o CRLF o \r\n o 0x0D0A
 - senza opportuna conversione i file di origine DOS hanno caratteri extra fine linea
 - conversione: Ubuntu pacchetto tofromdos , comandi todos / fromdos

```
mirror_mod = modifier_ob...
set mirror object to mirror...
mirror_mod.mirror_object...
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True

selection at the end -add
mirror_ob.select= 1
modifier_ob.select=1
context.scene.objects.active
("Selected" + str(modifier...
mirror_ob.select = 0
= bpy.context.selected_obj...
data.objects[one.name].select...

print("please select exactly...

-- OPERATOR CLASSES ----

types.Operator):
X mirror to the selected
object.mirror_mirror_x"
mirror X"

context):
context.active_object is not
```

Comandi utili per lavorare su file

- Trasferimento dati da/per device : i comandi più ovvi non sono pratici per trasferire dati con file speciali, cat e ridirezioni sono utilizzabili in modo "tutto o niente", cp non funziona
 - dd può leggere/scrivere byte da qualsiasi file, bypassa filesystem scrivendo direttamente su memoria fisica, formatodd if=<NOME> of=<NOME>
 - se NOME= - si intende STDIN/STDOUT
 - skip=<N> da che punto leggere
 - seek=<N> in che punto scrivere
 - count<N> quanti dati trasferire
 - bs=<N> dimensione blocco

```
mirror_mod = modifier_ob.  
#set mirror object to mirror  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
#selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.  
mirror_ob.select = 0  
= bpy.context.selected_object  
data.objects[one.name].select  
  
print("please select exactly  
  
-- OPERATOR CLASSES ----  
  
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
  
context):  
context.active_object is not
```



Comandi utili per lavorare su file

- Archiviazione e compressione
 - Per archiviazione si intende prendere una gerarchia di directory e serializzarle
 - -A concatena più tar
 - -c crea nuovo tar
 - -d trova differenze tra archivio tar e FS
 - -r aggiunge file a un tar
 - -t elenca contenuto di un tar
 - -u aggiorna file in un tar
 - -x estrae file da un tar
 - --delete cancella file da un tar



Comandi utili per lavorare su file

- Archiviazione e compressione
 - A default assume che l'archivio sia su /dev/tape, quindi si usa sempre opzione -f <FILENAME>
 - Altre opzioni
 - -p conserva tutte le informazioni di protezione, ma utenti standard sono forzati a dare la loro ownership quando estraggono
 - -v verbose stampa i dettagli
 - -T <ELENCO> prende i file da archiviare da ELENCO invece che da parametri
 - -C <DIR> svolge tutto come dopo cd DIR
 - --files-from=nomelistaindica un file da cui leggere la lista dei file da archiviare
 - Esempi: non è necessario specificare '-' fintanto che non è necessario usare più di un'opzione che richiede parametri
 - Creazione – tar cvpf users.tar /home/* : la barra iniziale verrà rimossa in modo da rendere relativi i path



Comandi utili per lavorare su file

- Archiviazione e compressione
 - Estrazione – `tar -C /newdisk -xvpf users.tar`: poiché i path nell'archivio sono relativi, la directory home viene ricreata dentro /newdisk e tutta la gerarchia sottostante ricostruita
 - In pipeline – `tar cvpf - /home/* | tar -C /newdisk -xvpf`
- Compressione: tar non comprime, i formati di compressione più comuni in Linux sono
 - .gz con comando `gzip`
 - .bz2 con comando `bzip2`
 - .xz con comando `xz`
 - I comandi base prendono un argomento file e lo comprimono aggiungendo estensione
 - con opzione `-d` si decompime ricreando il file e rimuovendo l'estensione
 - con opzione `-c` si riversa su STDOUT invece che su file



Comandi utili per lavorare su file

- Archiviazione e compressione
 - tar ha opzioni per invocare decompressione
 - -z per gzip
 - -j per bzip2
 - -J per xz
 - Es precedente tar cJf archive.tar.xz *



Comandi utili per lavorare su file

- Copia massiva di file
 - Il trasferimento di gerarchie di file e cartelle, contenenti file non standard non è gestito correttamente da tutte le versioni di cp -a e scp -R
 - rsync permette di effettuare più controlli, come la possibilità di non trasferire file già presenti a destinazione, o di configurare un comportamento per file speciali
 - sintassi base rsync SORGENTE DESTINAZIONE sorgente elenco di file e cartelle, destinazione cartella
 - con protocollo nativo
 - rsync HOST::SRCDIR DESTINAZIONE
 - rsync SORGENTE HOST::DESTDIR
 - con SSH , stessa sintassi di sopra ma con un solo : e non 2
 - opzioni
 - Come copiare

Comandi utili per lavorare su file

- Copia massiva di file

- -l / -L copia i link come link/come file puntato
- -p / -o / -g preserva i permessi, il proprietario, il gruppo
- -t / -a / -N preserva i timestamp di modifica/accesso/creazione
- -D preserva i file speciali

- Cosa copiare

- -u salta i file che sono più nuovi a destinazione o che a parità di età hanno stessa dimensione
- -c salta i file che a destinazione hanno lo stesso checksum

```
mirror_mod = modifier_ob.  
#set mirror object to mirror  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
#selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.name))  
mirror_ob.select = 0  
bpy.context.selected_objects  
data.objects[one.name].select  
  
print("please select exactly  
  
-- OPERATOR CLASSES ----  
  
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
  
context):  
context.active_object is not
```



Comandi utili per lavorare su file

- Full backup è la copia completa di ogni singolo file nel/nei filesystem oggetto del backup, ingombrante quindi difficile farlo frequentemente
- Incremental backup è la copia dei soli file cambiati da una data di riferimento, tipicamente quella di esecuzione dell'ultimo full backup

Comandi utili per lavorare su file



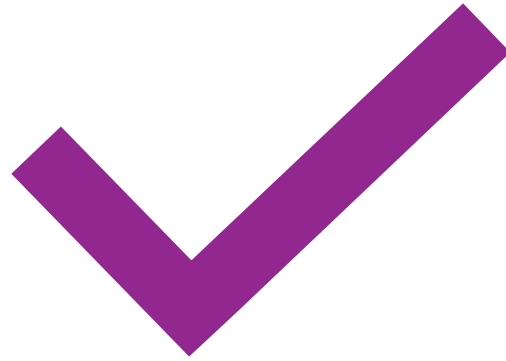
correttezza della copia: si dovrebbe tenere il FS a riposo durante il backup, ma è raro nella pratica quindi va fatta attenzione ai dettagli relativi a lettura di file aperti o di strutture complesse come DB



protezione dei dati: un backup contiene tutti i file del sistema, ma non c'è il sistema operativo a mediare l'accesso quindi in caso di riservatezza dei dati va difeso diversamente



affidabilità dei supporti: con periodicità dipendente dalla criticità dei sistemi, ci si deve assicurare che i dati siano scritti correttamente e siano leggibili per tutta la durata prevista della copia: protezione da fattori tecnologici e da fattori ambientali



Riga di comando

Prontuario riga di comando

whoami –
indica il proprio
username

id – dà
informazioni su
identità e
gruppo di
appartenenza

who – chi è
collegato alla
macchina

shutdown now
– solo root,
spegne la
macchina o la
riavvia , now
indica quando

Prontuario riga di comando

- Keyword
- Builtin
- Comandi esterni
- Alias
- Funzioni

```
mirror_mod = modifier_ob.  
#set mirror object to mirror.  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
#selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.name))  
mirror_ob.select = 0  
= bpy.context.selected_objects  
data.objects[one.name].select  
  
print("please select exactly  
  
-- OPERATOR CLASSES ----  
  
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
  
context):  
context.active_object is not
```

Prontuario riga di comando

\ davanti al
comando
previene solo
l'espansione degli
alias

Keyword builtin
previene
l'espansione degli
alias e l'uso di
funzioni e invoca
l'esecuzione del
builtin specificato

Keyword
command utilizza
un comando
esterno anche se
esiste una
funzione con lo
stesso nome

comando unalias
cancella un alias
definito in
precedenza



Prontuario riga di comando

- assoluto `usr/local/bin/top`
- relativo `./mycommand`



Prontuario riga di comando

- man pages: si invoca con `man <nomepagina>`, ogni applicazione installa pagine di manuale, raggruppate in sezioni
 - User commands
 - System calls
 - Funzioni di libreria
 - File speciali
 - Formati dei file, protocolli, relative strutture in C
 - Giochi
 - Varie: macro, header, filesystem
 - Comandi di amministrazione solo per root
 - `man -a <comando>` : cerca in tutte le sezioni



Prontuario riga di comando

- man pages: si invoca con `man <nomepagina>`, ogni applicazione installa pagine di manuale, raggruppate in sezioni
 - `man <sez> <comando>` : cerca in una sezione
 - `man -k <keyword>`: cerca tutte le pagine attinenti alla parola chiave
- i builtin bash non hanno man page, si possono avere info sommarie con `help <builtin>` o in particolare nella man page bash
- info files: si leggono con il comando `info`, a metà strada tra man page e ipertesto
- HOWTO: si trovano in `/usr/ /doc/HOWTO`, utili per la risoluzione di problemi pratici
- On-line: molte risorse, in particolare tldp.org The Linux Documentation Project



Intro a VIM

- COMMAND
 - Il cursore è posizionato sul testo
 - La tastiera è utilizzabile solo per richiedere l'esecuzione di comandi, e non per introdurre testo
 - I caratteri digitati non vengono visualizzati
 - Si passa a INPUT con oOilaACR
 - Si passa a DIRECTIVE con :/?
 - Si può richiedere lo spostamento del cursore con i comandi di movimento
 - h 1 a sinistra
 - l 1 a destra
 - k 1 linea sopra

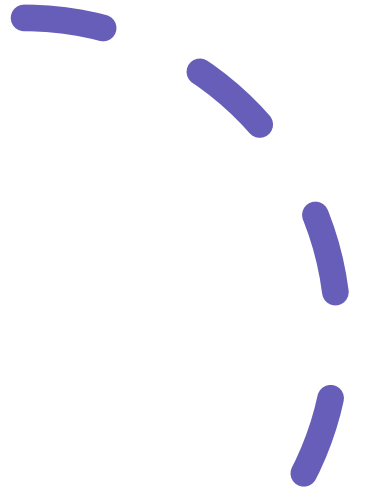




Intro a VIM

- COMMAND

- j 1 linea sotto
- Questi 4 Sostituibili con le frecce
- ^ inizio riga
- \$ fine riga
- gg prima linea del testo
- G ultima linea del testo
- #G sulla linea numero #
- E' anche possibile apportare modifiche
 - x cancella il carattere su cui si trova il cursore



Intro a VIM

- **COMMAND**

- dd cancella la linea su cui si trova il cursore
- rX rimpiazza il carattere sotto il cursore con X
- I successivi sono comandi di modifica, causano l'ingresso in INPUT
- i inserimento nella posizione del cursore
- I inserimento a inizio riga
- a append nella posizione del cursore
- A append a inizio riga
- R replace
- cw change word, elimina la parola che inizia sotto il cursore ed edita





Intro a VIM

- COMMAND
 - C change, elimina fino a fine riga
 - o linea vuota sotto al cursore
 - O linea vuota sopra al cursore
- Ricerca e spostamenti possono essere usati come terminatore per alcuni comandi di modifica
 - d\$ cancella dalla posizione corrente a fine riga
 - dG cancella dalla posizione corrente a fine file
 - c/ciao<RET> cancella dalla posizione corrente alla prima occorrenza della stringa ciao e si porta in insert mode
- Con il punto
- Precedendo un comando con un numero N, il comando verrà eseguito N volte consecutivamente



Intro a VIM

- COMMAND

- u annulla l'ultima azione eseguita
- Copia e incolla
 - La copia viene eseguita con buffer interni a vi, è possibile specificarne altri
 - yy copia la linea corrente nel buffer
 - "ayy copia la linea corrente nel buffer "a"
 - d esegue il cut
 - p incolla dopo la linea corrente
 - P incolla prima della linea corrente
 - Per copiare blocchi di linee si può



Intro a VIM

- COMMAND

- Usare marcatori: ma marca la posizione con il simbolo "a"
- Usando le ripetizioni: ci si posiziona sulla prima linea del blocco, per copiare 10 righe si digita 10yy
- Usando la ricerca: ci si posiziona sulla prima linea del blocco, per copiare fino alla parola 'basta' si digita y/basta<RET>

- INPUT

- Tutti i caratteri digitati vengono visualizzati ed inseriti nel testo
- Si passa a COMMAND con <ESC>



Intro a VIM



- DIRECTIVE
 - Ci si trova posizionati con il cursore nella linea direttive e si possono richiedere tutti i comandi per il controllo del file
 - Si passa a COMMAND con <RET>
 - I comandi per caricare/salvare/uscire sono DIRECTIVE
 - :r <file> inserisce il contenuto di file al punto del cursore
 - :w scrive il file corrente
 - :q esce
 - ZZ scrive ed esce
 - Digitando la barra / si entra in DIRECTIVE per cercare stringhe
 - Con n si passa alla successiva occorrenza, con N alla precedente

Intro a VIM

- DIRECTIVE

- /<RET> e ?
- :s/trova/sostituisci/ cerca "trova" e lo sostituisce con "sostituisci"
- :%s/trova/sostituisci/cgi cerca "trova" in ogni linea del file e lo sostituisce con "sostituisci"
 - Dopo aver chiesto conferma
 - Anche più volte nella stessa linea
 - Case insensitive
 - % è una scorciatoia per 1,\$, in realtà il comando può essere invocato come :l,Fs/trova/sostituisci/ per applicarlo tra le linee l ed F



Composizione comandi e filtri

- Fd 0 STDIN
- Fd 1 STDOUT
- Fd 2 STDERR

Composizione comandi e filtri

- Ridirezione stdout: `ls > miofile`: scrive lo stdout di `ls` su `miofile`, troncandolo
- Ridirezione stderr: `2>` e `2>>`
- Confluenza: `ls > miofile 2>&1`: ridirige stderr dentro stdout, e poi stdout su file
- Ridirezione stdin: `sort < miofile` riversa il contenuto di `miofile` su stdin di `sort`
- Piping: `ls | sort`: bash fa sì che ciò che `ls` produce su stdout venga riportato su stdin di `sort`




Filtri

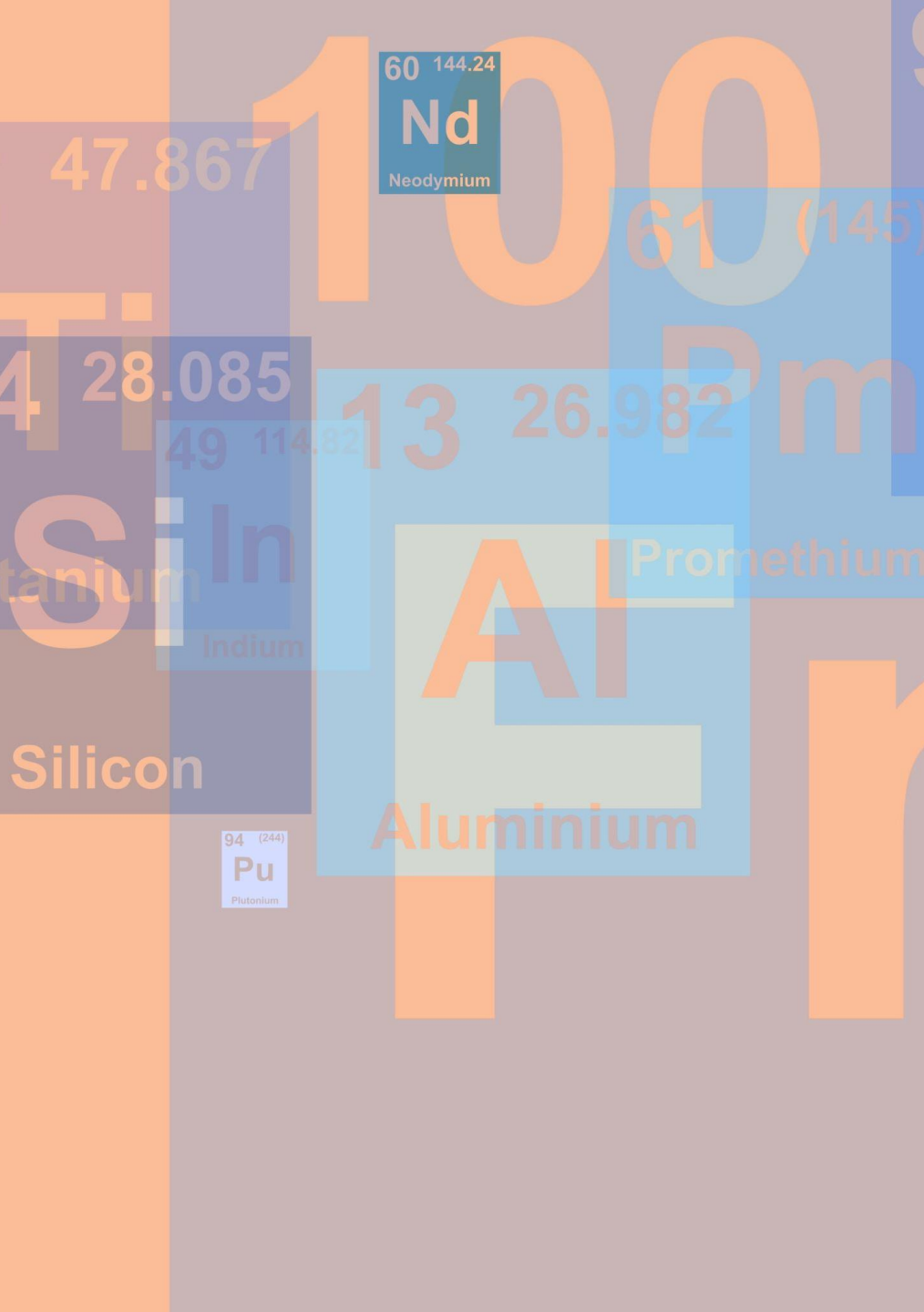
- `cat: copia stdin su stdout`





Filtri

- less: non è un vero filtro essendo l'output destinato al terminale ma è utile per l'uso interattivo
 - -h comandi disponibili
 - frecce – scorrimento
 - <N>g si porta alla riga num <g> default:1
 - G si porta al termine del file
 - /<pattern> cerca la riga successiva al cursore contenente <pattern>
 - <pattern> cerca la riga successiva al cursore contenente <pattern>
 - n ripete la ricerca fatta in precedenza
 - N ripete la ricerca fatta in precedenza, ma nel verso opposto
 - q esce da less
- 



Filtri

- rev: filtro che inverte l'ordine dei caratteri di ogni linea in stdin, su stdout
- head e tail: estraggono la parte iniziale/finale di un file
 - -c NUM i primi/ultimi NUM caratteri, con -/+NUM tutto il file eccetto/a partire da NUM caratteri
 - -n NUM prime/ultime NUM righe, con -/+ NUM tutto il file eccetto/a partire da NUM righe
 - -f per tail, mantiene il file aperto e mostra in tempo reale le ultime righe

Filtri

- cut: taglia parti di righe
 - -cELENCO_POSIZIONI_CARATTERI
intervallo per caratteri scelti
 - Per i file a record , opzioni -d e -f per uno o più campi di ogni record
 - cut -dCARATTERE_DELIMITATORE -fELENCO_CAMPI
 - -s evita che vengano poste in output le righe senza delimitatore
 - Es
 - Es

Periodic Table of the Elements

Number → 1																	
Symbol ← H																	
Name → Hydrogen																	
Atomic Weight ← 1.008																	
						13 IIIA	14 IVA										
						5 B Boron 10.81	6 C Carbon 12.01										
						13 Al Aluminium 26.9815385	14 Si Silicon 28.08										
6 VIB	7 VIIB	8 VIIIB	9 VIIIB	10 VIIIB	11 IB	12 IIB											
24 Cr Chromium 51.9961	25 Mn Manganese 54.938044	26 Fe Iron 55.845	27 Co Cobalt 58.933194	28 Ni Nickel 58.6934	29 Cu Copper 63.546	30 Zn Zinc 65.38	31 Ga Gallium 69.723	32 Ge Germanium 72.63									
42 Mo Molybdenum 95.95	43 Tc Technetium (98)	44 Ru Ruthenium 101.07	45 Rh Rhodium 102.90550	46 Pd Palladium 106.42	47 Ag Silver 107.8682	48 Cd Cadmium 112.414	49 In Indium 114.818	50 Sn Tin 118.71									
74 W Tungsten 183.84	75 Re Rhenium 186.207	76 Os Osmium 190.23	77 Ir Iridium 192.217	78 Pt Platinum 195.084	79 Au Gold 196.966569	80 Hg Mercury 200.592	81 Tl Thallium 204.38	82 Pb Lead 207.2									
106 Sg Seaborgium (269)	107 Bh Bohrium (270)	108 Hs Hassium (269)	109 Mt Meitnerium (278)	110 Ds Darmstadtium (281)	111 Rg Roentgenium (282)	112 Cn Copernicium (285)	113 Nh Nihonium (286)	114 Fl Flerovium (289)									
60 Nd Neodymium 144.242	61 Pm Promethium (145)	62 Sm Samarium 150.36	63 Eu Europium 151.964	64 Gd Gadolinium 157.25	65 Tb Terbium 158.92535	66 Dy Dysprosium 162.500	67 Ho Holmium 164.93033	68 Er Erbium 167.25									
92 U Uranium 238.02891	93 Np Neptunium (237)	94 Pu Plutonium (244)	95 Am Americium (243)	96 Cm Curium (247)	97 Bk Berkelium (247)	98 Cf Californium (251)	99 Es Einsteinium (252)	100 Fr Fermium (257)									

Filtri

- sort: ordina le linee di uno stream
 - -u elimina entry multiple
 - -r reverse
 - -R random
 - -m merge di file già ordinato
 - -c controlla se il file è già ordinato
 - -b ignora spazi a inizio riga
 - -d considera solo i caratteri alfanumerici e gli spazi
 - -f ignora maiusc/minusc
 - -n interpreta stringhe di numeri per valore numerico




Filtri

- sort: ordina le linee di uno stream
 - -h interpreta numeri leggibili come 2K, 1G...
 - -tSEP imposta SEP come separatore, a default spazi
 - -kKEY chiave di ordinamento, se usato più volte, ordina per la prima chiave, poi la seconda...
 - KEY è nella forma semplificata F] dove
 - F = numero di campo
 - C = posizione in caratteri nel campo
 - OPTS = una delle opzioni di ordinamento
- uniq: elimina i duplicati consecutivi
- wc: con -c conta i caratteri, con -l le linee e con -w le parole





Filtri

- `grep`: esamina le righe in ingresso e produce in uscita quelle che matchano un pattern passato come argomento
- 



Filtri

- Nel caso una RE corrisponda a più di una sottostringa in una stringa, la RE matcha quella che inizia per prima
- A partire da quel punto, se la RE matcha più di una sottostringa, si seleziona la più lunga





Filtri

- Controllo matching
 - -E usa extended RE
 - -F disattiva le RE e usa il parametro come stringa letterale
 - -w/-x fa match solo con RE “whole word” o “whole line”
 - -i rende l’espressione case insensitive
- Controllo input
 - -r cerca ricorsivamente nei file di una cartella
 - -f FILE prende le RE da un FILE invece che come parametro



Filtri

- Controllo output
 - -o restituisce solo le sottostringhe che corrispondono alla RE invece della riga che le contiene, separatamente una per riga di output
 - -v restituisce le linee che non contengono l'espressione
 - -l utile passando a grep più file su cui cercare: restituisce solo i nomi dei file in cui l'espressione è stata trovata
 - -n restituisce anche il numero della riga contenente l'espressione
 - -c restituisce solo il conteggio delle righe che contengono la RE
 - -q quiet, non scrive niente su stdout e esce con 0 appena trova match
 - --line-buffered disattiva il buffering

Espressioni
regolari moderne
: regexp o RE,
documentazione
in man page
regex

RE = uno o più
rami non vuoti
separati da |

Ramo = uno o
più pezzi
concatenati

Pezzo = atomo
eventualmente
con
moltiplicatore

Periodic Table of the Element

Number → 1										← Symbol	
Name → Hydrogen										← Atomic Weight	
1.008											
						13 IIIA	14 IVA				
						5 B Boron 10.81	6 C Carbon 12.01				
						13 Al Aluminium 26.9815385	14 Si Silicon 28.08				
6 VIB	7 VIIB	8 VIIB	9 VIIB	10 VIIB	11 IB	12 IIB					
24 Cr Chromium 51.9961	25 Mn Manganese 54.938044	26 Fe Iron 55.845	27 Co Cobalt 58.933194	28 Ni Nickel 58.6934	29 Cu Copper 63.546	30 Zn Zinc 65.38	31 Ga Gallium 69.723	32 Ge Germanium 72.63			
42 Mo Molybdenum 95.95	43 Tc Technetium (98)	44 Ru Ruthenium 101.07	45 Rh Rhodium 102.90550	46 Pd Palladium 106.42	47 Ag Silver 107.8682	48 Cd Cadmium 112.414	49 In Indium 114.818	50 Sn Tin 118.71			
74 W Tungsten 183.84	75 Re Rhenium 186.207	76 Os Osmium 190.23	77 Ir Iridium 192.217	78 Pt Platinum 195.084	79 Au Gold 196.966569	80 Hg Mercury 200.592	81 Tl Thallium 204.38	82 Pb Lead 207.2			
106 Sg Seaborgium (269)	107 Bh Bohrium (270)	108 Hs Hassium (269)	109 Mt Meitnerium (278)	110 Ds Darmstadtium (281)	111 Rg Roentgenium (282)	112 Cn Copernicium (285)	113 Nh Nihonium (286)	114 Fl Flerovium (289)			
60 Nd Neodymium 144.242	61 Pm Promethium (145)	62 Sm Samarium 150.36	63 Eu Europium 151.964	64 Gd Gadolinium 157.25	65 Tb Terbium 158.92535	66 Dy Dysprosium 162.500	67 Ho Holmium 164.93033	68 Er Erbium 167.25			
92 U Uranium 238.02891	93 Np Neptunium (237)	94 Pu Plutonium (244)	95 Am Americium (243)	96 Cm Curium (247)	97 Bk Berkelium (247)	98 Cf Californium (251)	99 Es Einsteinium (252)	100 Fr Francium (257)			

Espressioni regolari moderne :
regexp o RE, documentazione in
man page regex


- Atomo = uno di
 - contiene RE
 - contiene charset
 - \wedge o $\$$ o
 - Backslash sequence
 - Singolo carattere
 - qualsiasi carattere
 - \wedge inizio linea
 - $\$$ fine linea
 - Backslash sequence



Espressioni regolari moderne : regexp o RE, documentazione in man page regex

- Atomo = uno di
 - `\< - \>` stringa vuota a inizio/fine parola
 - `\b` stringa vuota a confine di parola
 - `\B` stringa vuota a condizione che non sia confine di parola
 - `\w` qualsiasi lettera o numero
 - `\W` qualsiasi carattere non in `\w`
- indica da `n` a `m` occorrenze dell'atomo che lo precede
- indica zero o una occorrenze dell'atomo che lo precede
- `*` zero o più occorrenze atomo precedente
 - `+` una o più occorrenze atomo precedente

Espressioni
regolari moderne
: regexp o RE,
documentazione
in man page
regex



UN qualsiasi
carattere tra
a,b o c

UN qualsiasi
carattere tra a
e z compresi

UN qualsiasi
carattere che
non sia né d
né c



Espressioni regolari moderne : regexp o RE, documentazione in man page regex

- blank lower upper cntrl print xdigit o eventualmente nel locale attivo
 - egrep '^Nel.*vita\.\$' miofileha come output tutte le righe di miofile che iniziano per Nel e finiscono per vita
 - egrep '.es e' miofile ha come output tutte le righe che contengono in qualsiasi posizione la sequenza
 - 1 carattere qualsiasi
 - es
 - una sequenza di 3-5 caratteri potenzialmente diversi uno dall'altro, a patto che ognuno sia diverso da e ed s
 - e
- tee legge stdin e lo scrive sia su stdout che sui file indicati
- sed e awk non sono semplici filtri avendo un vero e proprio linguaggio di programmazione, vediamo solo alcuni esempi pratici
- sed Stream EDitor, segue formato base sed -e 'comando' o sed -f 'script', non useremo script generici ma il solo comando di sostituzione



sed
's/VECCHIO_PATTERN/NUOVO_VALORE/
'

- i case insensitive
- g global
- NUM sostituisce solo l'occorrenza NUM-esima
- -i edita il file dato
- -u unbuffered



```
sed  
's/VECCHIO_PATTERN/NUOVO_VALORE/  
'
```

- awk è un interprete per AWK che useremo solo come evoluzione di cut perché permette di considerare qualsiasi sequenza di caratteri come unico delimitatore
 - stampa il secondo campo del file, purché sia separato dal primo da un numero qualunque di blanks cat personale | awk ''
 - in un file che riporta il risultato di un'operaz
 - a differenza di cut non ha il concetto di "-f 5-" , ma: cat file | awk ') }'
 - tr 'A-Z' 'a-z' trasforma maiuscole in minuscole
 - tr ';;.!?'
 - in generale, se il secondo set è più limitato del primo set, il suo ultimo carattere viene ripetuto quanto basta a generare la corrispondenza 1:1
 - tr ';;.!?'
 - tr -d '\r' elimina ogni occorrenza del carriage return

Shell e gestione processi

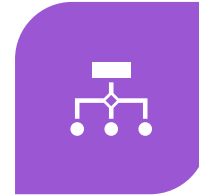
- fork, che crea una copia del processo corrente duplicando tutte le risorse e condivide il codice
- exec, che sostituisce il codice del processo con quello caricato da un programma modificando la text table

```
mirror_mod = modifier_ob.  
#set mirror object to mirror.  
mirror_mod.mirror_object =  
#operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
#operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
#operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
  
#selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.name))  
mirror_ob.select = 0  
= bpy.context.selected_objects  
data.objects[one.name].select  
  
print("please select exactly  
  
-- OPERATOR CLASSES ----  
  
types.Operator):  
# X mirror to the selected  
object.mirror_mirror_x"  
"mirror X"  
  
context):  
context.active_object is not
```

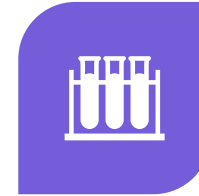
Shell e gestione processi



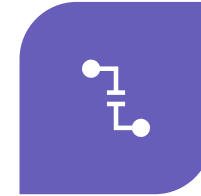
Ridirezione



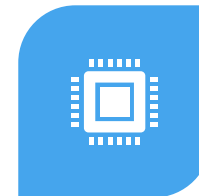
L'STDOUT SI RIDIRIGE CON `> MYFILE` PER APRIRE MYFILE DA FILE POINTER 0 , CON `>>` PER APRIRLO DA FILE POINTER IN FONDO AL FILE; STDERR SI RIDIRIGE CON `2>` O `2>>`; STDIN CON `COMMAND < MYFILE` PORTA IL CONTENUTO DI MYFILE SU STDIN DI `COMMAND`



PARTICOLARE È LA CONFLUENZA: `LS > MYFILE 2>&1` PRIMA RIDIRIGE `STDERR` DI `LS` SU `STDOUT` ; POI `STDOUT` SU `MYFILE`



INOLTRE SI PUÒ EVITARE L'USO DI FILE TEMPORANEI PER INVIARE DATI SU `STDIN` DI UN COMANDO, USANDO `COMMAND <<MARCATORE` O `COMMAND <<<` "TESTO" PER UN SOLA LINEA



PER RIDIRIGERE STREAM PERMANENTEMENTE SI USA `EXEC` , CHE RIDIRIGE GLI STESSI STREAM DELLA SHELL QUINDI TUTTI I COMANDI SUCCESSIVI AVRANNO QUEGLI `FD`



Ridirezione

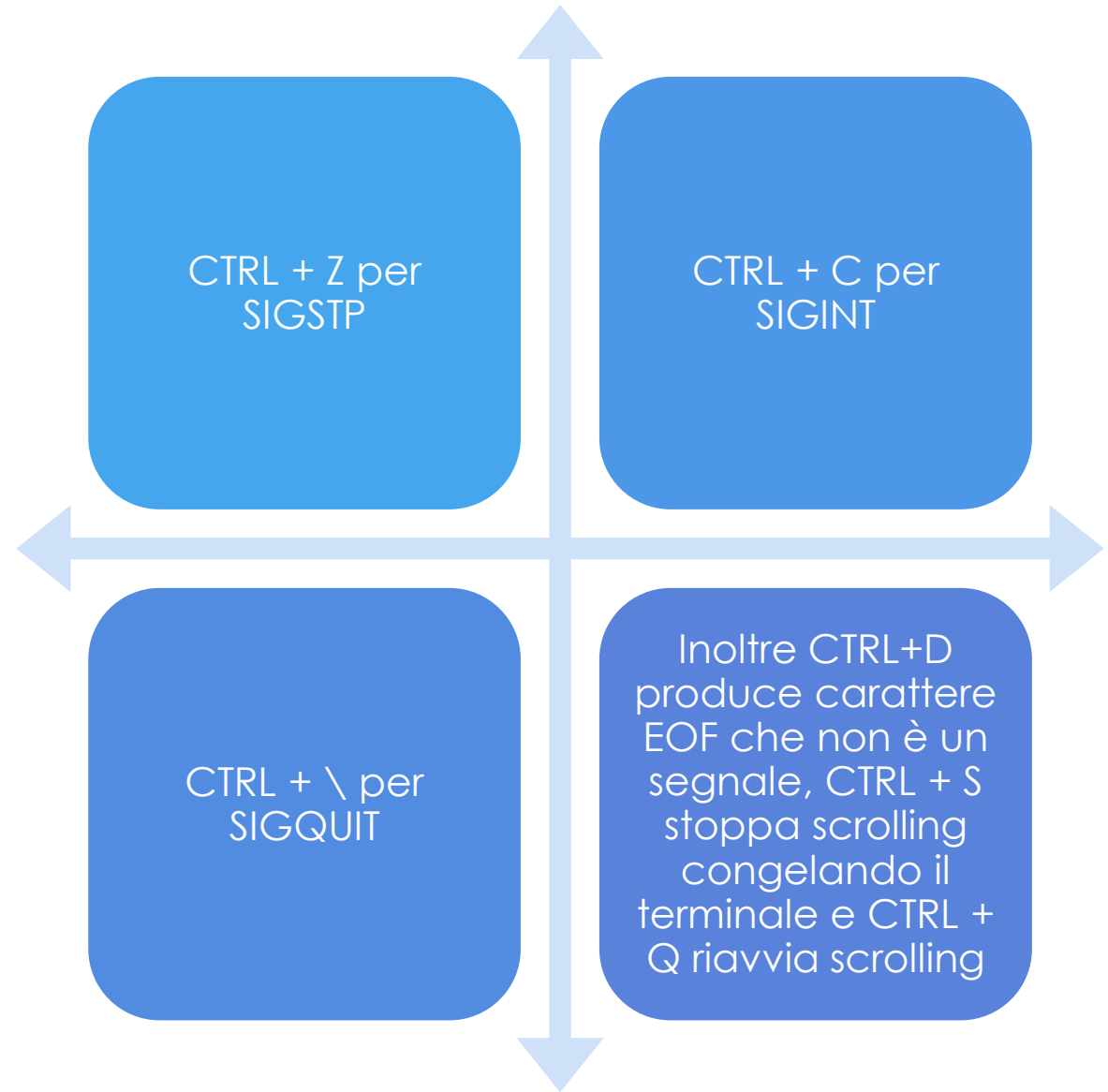
- Con exec si possono anche creare nuovi fd: `exec 3< fin 4> fout 5<> frw` permette letture con `<&3` su `fin`, scritture con `>&4` su `fout` e `frw` sia in lettura che in scrittura; per chiudere un fd si usa `exec fd>&`



Pipe e subshell

- bash fa due fork
- la prima shell figlia chiama `dup2 ,1]` per dirottare il suo `stdout` su `fd` e viceversa la seconda chiama `dup2 ,0)` per dirottare sul suo `stdin` su quello pipe
- infine la prima chiude `fd` e la seconda `fd` per evitare inconsistenze sulla pipe
- poi ognuna fa `exec` del suo comando facendoli partire con gli `fd` già settati in modo da avere trasparenza senza sapere della pipe

Interazione coi processi e segnali





Interazione coi processi e segnali

- watch prende come argomento altri comandi, li lancia periodicamente e aggiorna il suo output
- top mostra i processi più attivi e altre statistiche
- Accenno agli array: gli array in bash usano indici non necessariamente consecutivi
- echo \$ } mostra tutti gli indici validi per l'array di nome PID
- echo \$ } mostra tutti i valori contenuti nell'array di nome PID

Shell scripting

- Tokenizzazione: La riga viene divisa in token usando come separatori i metacaratteri `< > | &}`
- 1st token = alias?
- 1st token = keyword?
- Brace expansion: elementi tra vengono espansi in una lista
- Tilde expansion: se c'è un token nella forma `~username`, viene sostituito con la home directory dell'utente username
- Parameter expansion: carattere `$` può marcare l'inizio di diverse espansioni , la più semplice PE è la sostituzione della stringa `$NAME` con il valore contenuto nella variabile NAME
- Command substitution: il token ``` causa: creazione subshell, esecuzione comando, stdout di command viene posto sulla riga di comando al posto del token originale, a parte eventuali righe vuote alla fine



Shell scripting



Arithmetic expansion:
il token `$()` causa la
valutazione di `expr`,
un'espressione
aritmetica



Process substitution:
con il token `< o >` ,
viene eseguito
comando in modo
concorrente e
asincrono rispetto al
resto della riga



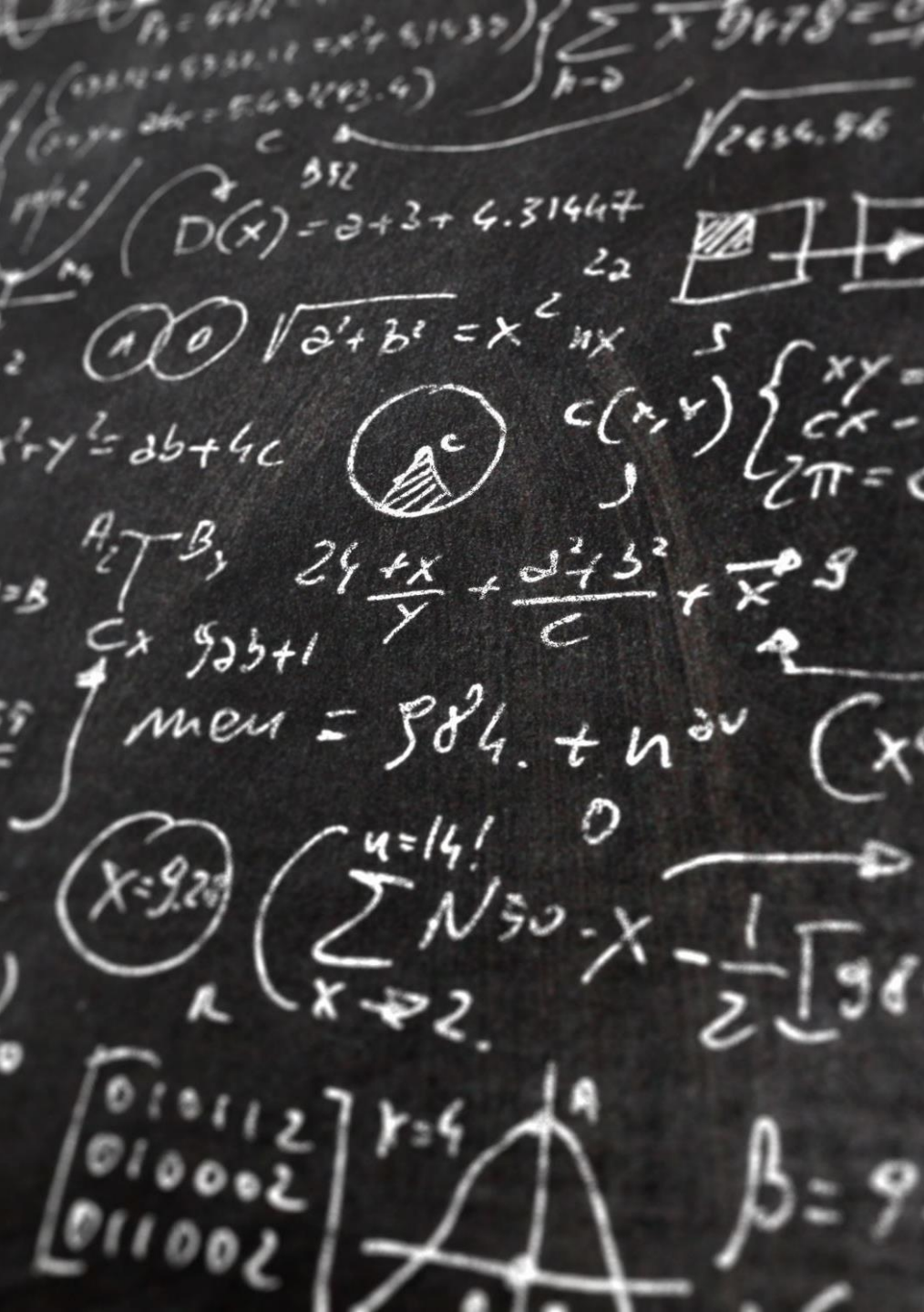
Word splitting: i
risultati dei passi 6..9
sono esaminati e
separati in word
indipendenti



Pathname expansion:
ogni word viene
esaminata e se
contiene uno tra `*`, `?`



Quote removal e
command lookup:
vengono rimosse
tutte le occorrenze di
caratteri di quoting
"usate"
effettivamente ,
vengono impostati gli
stream in caso di
ridirezione, viene
cercato il comando
in ordine tra: funzioni,
builtin, eseguibili in
`$PATH`



Quoting

- \ backslash: protegge solo il carattere successivo, che non verrà interpretato come simbolo speciale
- ' apice: ogni carattere di una stringa racchiusa tra una coppia di apici viene protetto dall'espansione e trattato letteralmente, senza eccezioni
 - 1 → 12
- " doppio apice: ogni carattere di una stringa racchiusa tra una coppia di virgolette viene protetto dall'espansione, con l'eccezione di \$, ` backtick, \ backslash, e altri casi particolari
 - 1 → → 12
 - vengono eseguiti solo i passi 1) Tokenizzazione, 6) Parameter Exp, 7) Command Sub, 8) Arithm
 - vengono saltati 2) Check token alias, 3) Check token keyword, 4) Brace Exp, 5) Tilde Exp e poi 10) Word Split, 11) Path Exp



Quoting

- es





protetto da virgolette*o da
apici

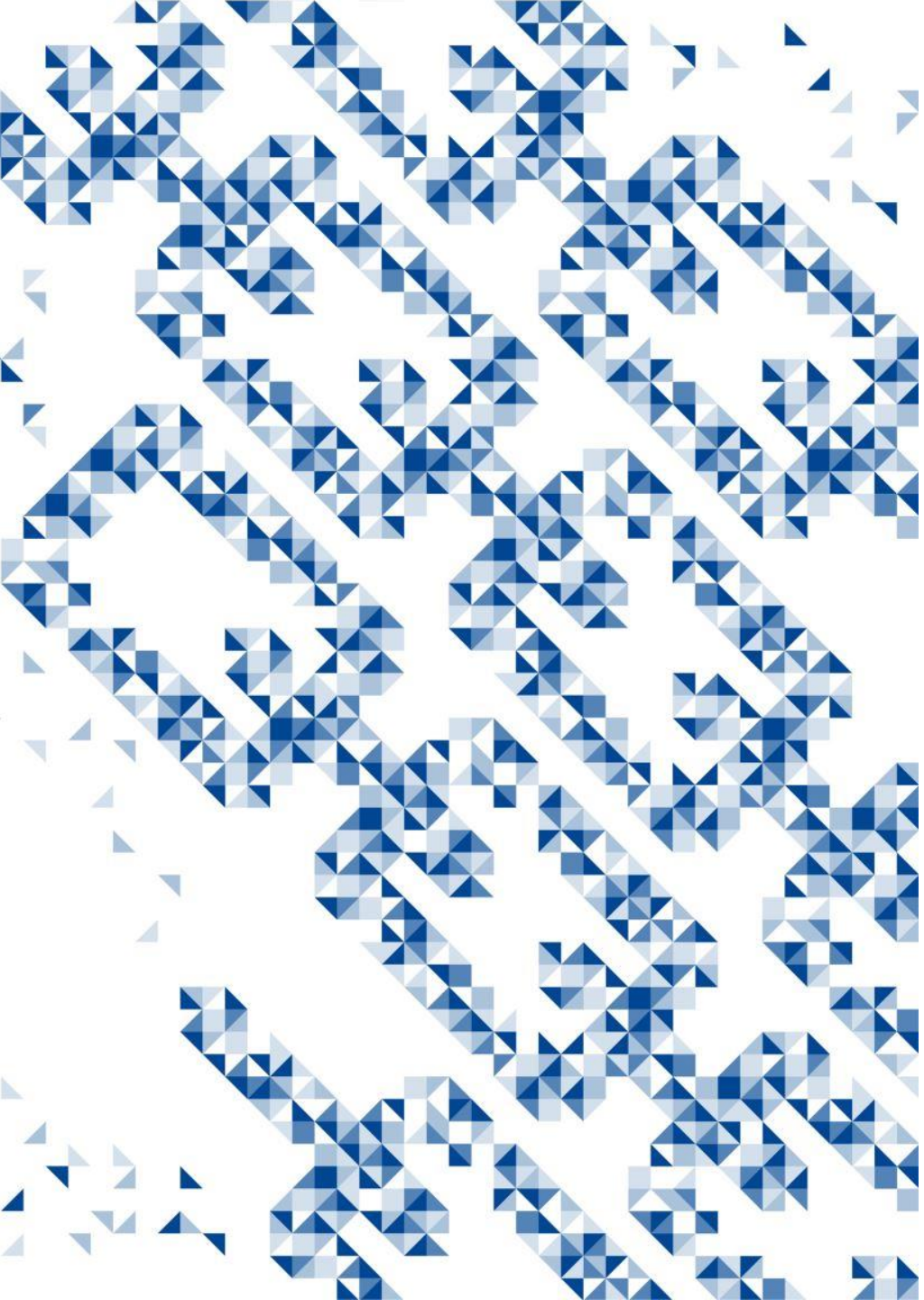
- Il comando echo stampa a video i caratteri che seguono, è utile per visualizzare il valore di una variabile o di una pathname expansion sfruttando l'espansione di bash

Pathname expansion

- * rappresenta una qualunque stringa di zero o più caratteri
- rappresenta un qualunque carattere singolo
- rappresent un carattere appartenente a SET
 - elenco
 - intervallo, il cui ordine dipende dal locale)
 - per negare il contenuto del SET
 -] classe di caratteri come per egrep
 - per includere i caratteri – o], metterli come primi carattere

Pathname expansion

- `echo *` tutti i file del direttorio corrente
- `echo * ?`
- `echo *` fa echo del carattere `*`, non visto come wildcard da bash ma con valore letterale
- `echo * *` elenca tutti i file del direttorio corrente che hanno almeno un carattere
- `echo */**/*` elenca tutti i file dei direttori di secondo livello a partire dalla root



Brace expansion

- Un meccanismo di espansione per generare sequenze di stringhe secondo un pattern con la stessa sintassi della pathname expansion, ma le stringhe sono generate indipendentemente dal fatto che esistano o meno file che rispettano il pattern
- Avviene presto nell'espansione della riga, non si possono usare variabili che contengono valori che la shell espande dopo
- Viene saltata con quoting a doppi apici ""



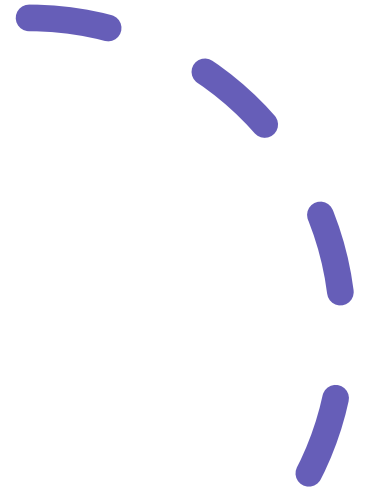
Variabili

- `ls ***` lista tutti i nomi dei file che contengono il carattere `*` in qualunque posizione
- `echo "$A"` stampa esattamente il contenuto della variabile `A`
- `echo '$A'` stampa esattamente `$A`
- `echo ""$A""` Chi vince?
- `echo $; echo "$ "` La differenza è che nel primo viene fatto 10) Word split, nel secondo no quindi tutti gli accapo restano tali e non cambiano in spazi



Variabili

- Settate da bash
 - \$ pid della shell capostipite
 - HOSTNAME nome dell'host
 - RANDOM num
- Usate da bashfor F in "\$ " ; do echo \$F | tr 'a-z' 'A-Z'; done
 - PS0..PS4 prompt in diversi contesti
 - HOME home dir dell'utente





Variabili

- "\$*" viene espanso in "\$1 \$2 \$3"
- "\$@" viene espanso in "\$1" "\$2"
quindi mantiene la protezione per ognuno dei contenuti



Variabili

- manipolare variabili nei processi figli senza perdere i risultati prima di poterli usare → si può usare subshell con `echo ciao |`
- necessità di usare `read` per acquisire dati interattivamente dall'utente in un processo figlio che ha `stdin` alimentato da una pipe anziché da un terminale → creare file descriptor per il terminale con `exec 3<$; exec ciao |`



Variabili

- shift sposta gli argomenti della riga di comando, \$0 rimane il nome dello scripting



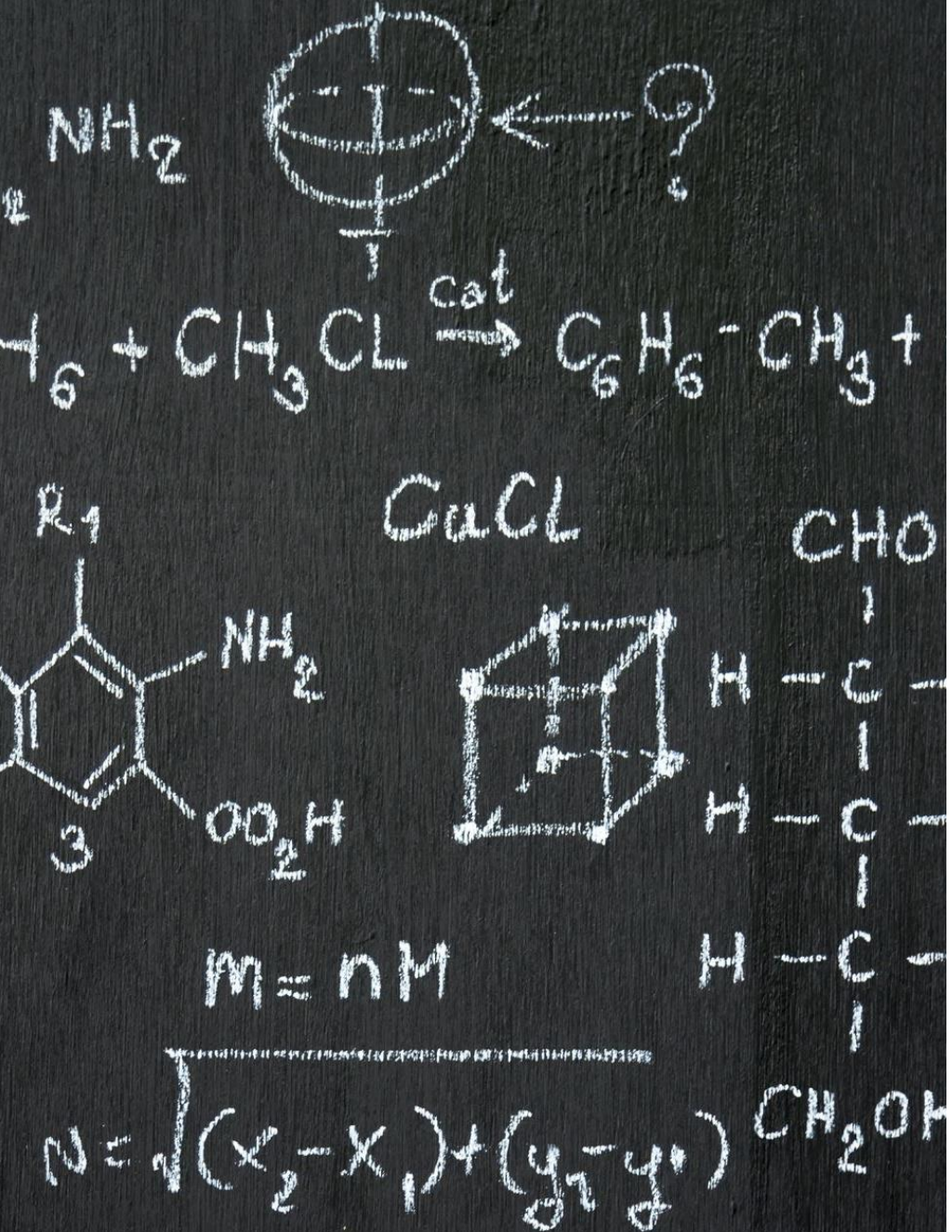


Arithmetic Expansion

- declare -i N per dichiarare variabili come intere
 - La shell non ha concetto di tipi, le variabili sono stringhe
 - Con -p print mostra tipo e valore del simbolo

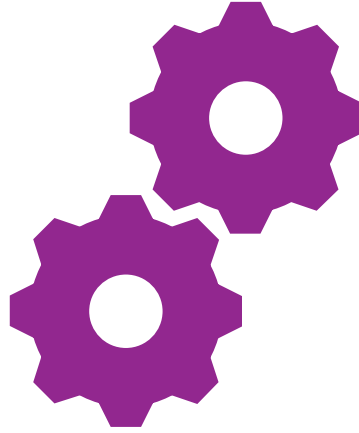
Arithmetic Expansion

- Builtin let, o equivalente comando composto)
 - let N++ l'espressione viene valutata, ha effetti sulle variabili, ritorna true se risultato non è nullo, produce errori su stderr ma nulla su stdout
 -) si comporta come "" per proteggere elementi dell'expr, riconosce variabili anche se non sono dichiarate intere e senza bisogno di prefisso \$ per espansione e interpretandole come zero se non definite
 - Variabili mai definite non danno errore in contesto aritmetico, vengono valutate come fossero
 - id++ ++id post/pre incremento la variabile viene espansa prima/dopo l'incremento
 - + - * / somma sottraz
 - ** % elevamento a potenza, modulo
 - ~ & | ^ NOT NOT AND OR XOR bit a bit
 - << >> shift binario a sx/dx
 - = *= /= += -= <<= >>= &= ^= |= assegnamenti



Arithmetic Expansion

- Builtin let, o equivalente comando composto)
 - `<= >= < > == !`
 - `&& ||` AND OR logico
 - `expr?`
- I numeri si possono esprimere tra base 2 e base 64 con prefisso `B#num` → base B
 - default 10, prefisso 0 ottale, prefisso 0x esadecimale
 - cifre utilizzabili 0..9a..z..Z@_



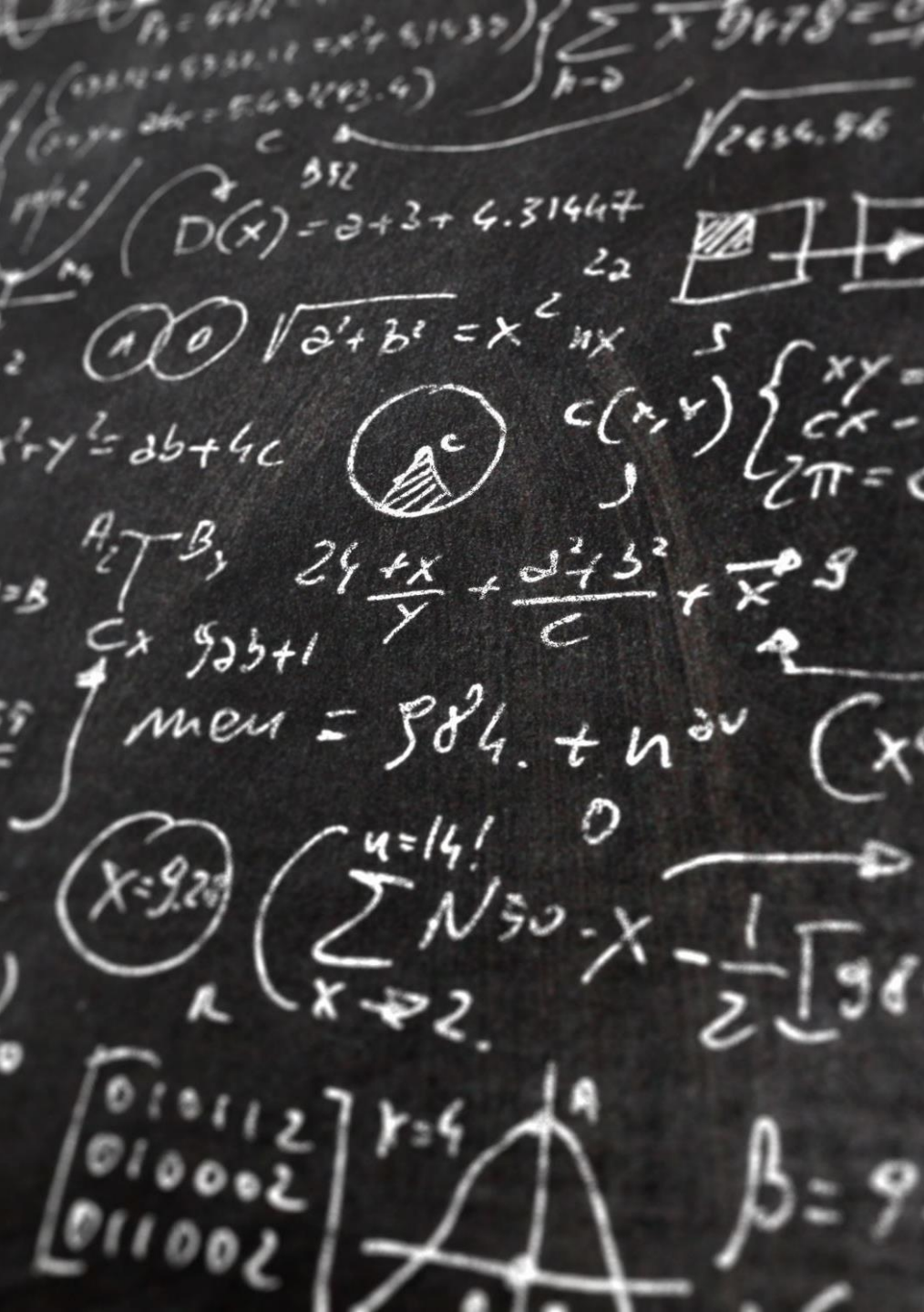
Controllo di flusso



Sequenze

- `'\x0a'` produce ASCII newline

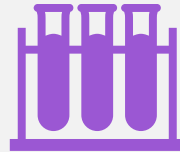




Funzioni

- Le funzioni sono sequenze con un nome function NOME
- Possono ricevere parametri, utilizzabili all'interno allo stesso modo dei parametri posizionali degli script \$1,\$2... ma appunto per questo i valori passati allo script non possono essere visti internamente alla funzione, solo \$0 resta impostato al nome dello script
- Per le invocazioni in pipeline la funzione verrà eseguita dalla shell figlia creata in automatico: scrivendo cat miofile | function | sort si ha creazione automatica di 3 shell; in questo caso non è più vero che lo spazio di memoria è lo stesso del chiamante!

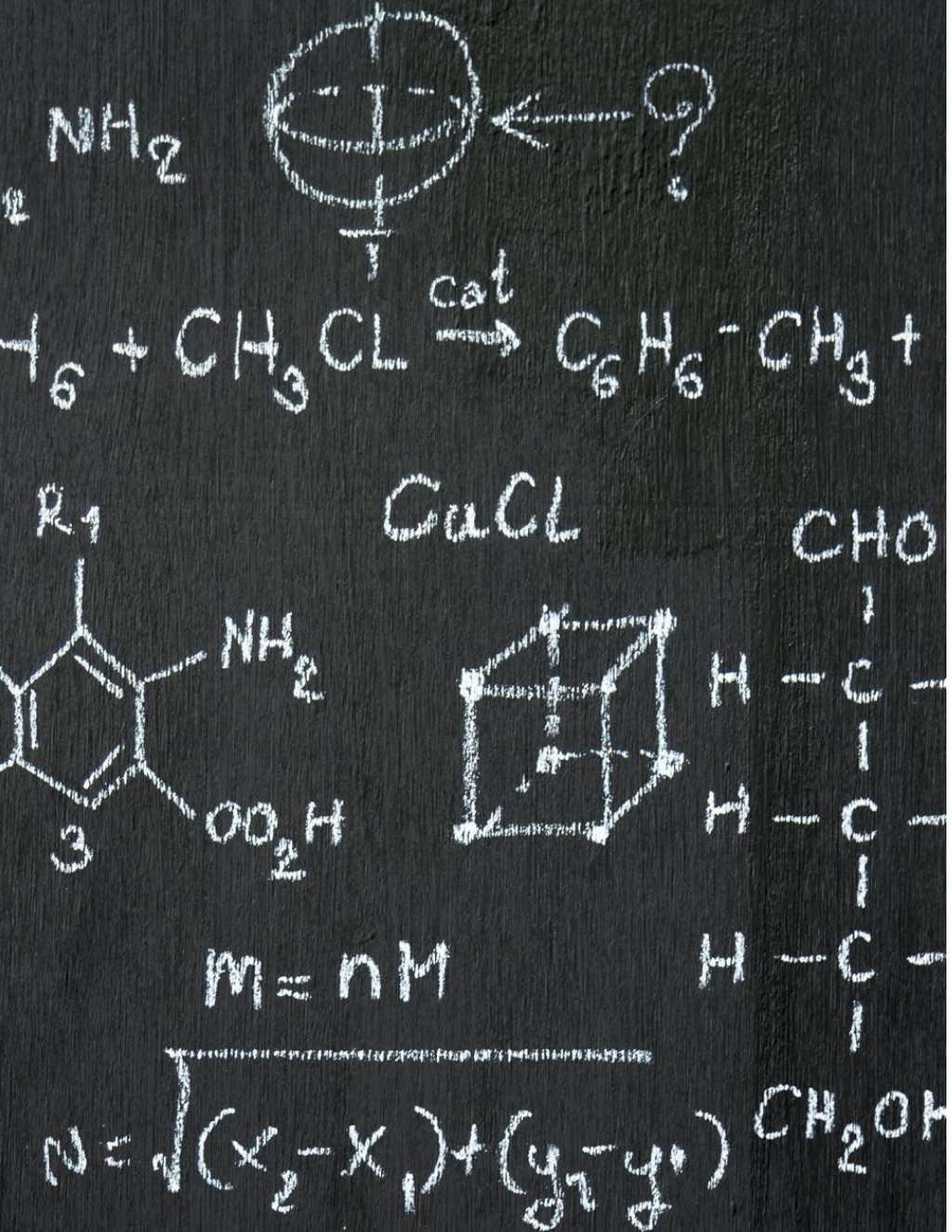
Valutazione delle condizioni



Built-in test, ,]



Comandi test



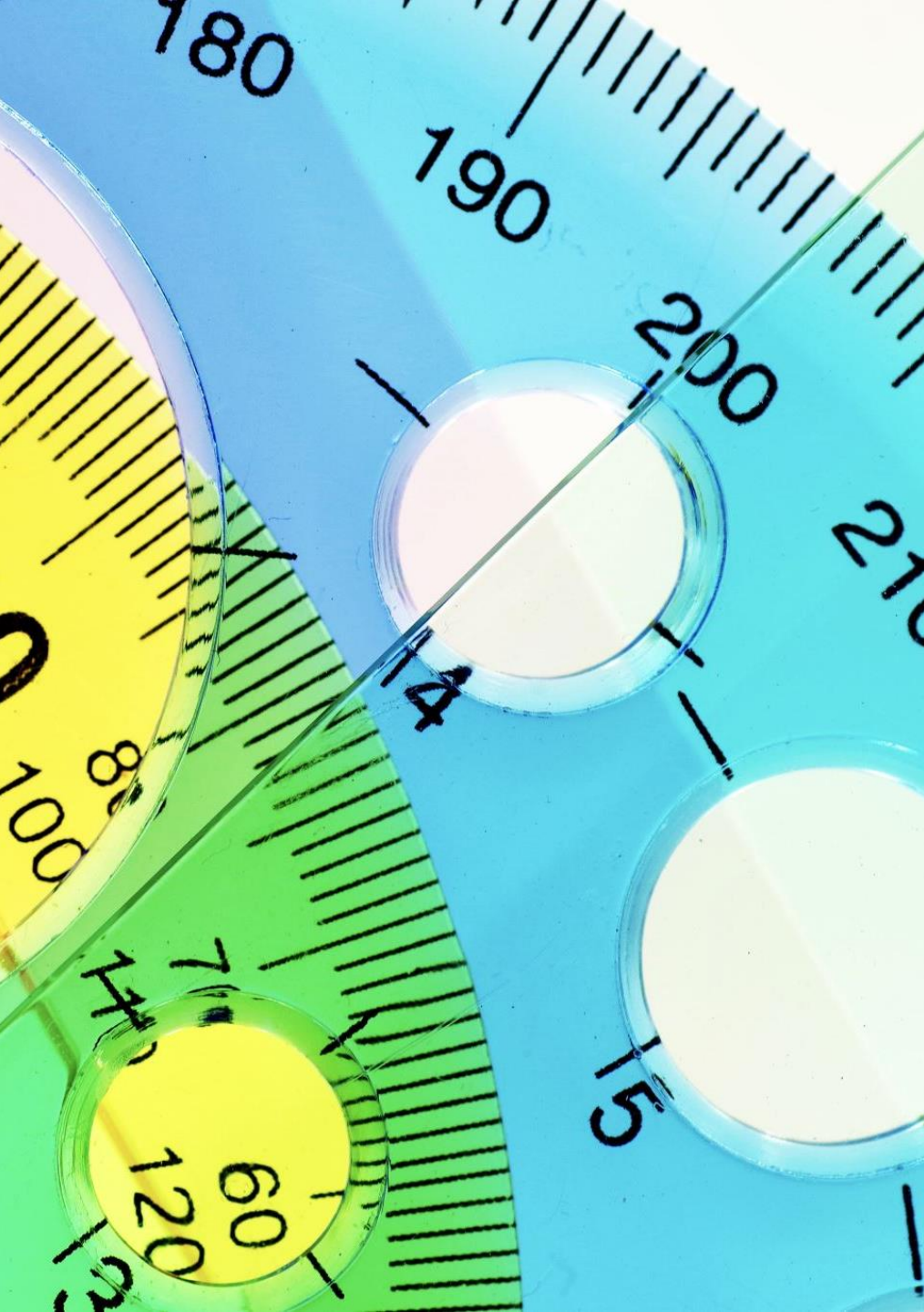
Valutazione delle condizioni

- test e come ultimo parametro) e seguiti da un'espressione la valutano e ritornano 0 o 1 in base al risultato
 - Test unari, un solo parametro: test OP ARG
 - su stringhe
 - -z true se stringa è vuota
 - -n true se string è non vuota
 - su file
 - -e true se file esiste
 - -f true se è file regolare
 - -d true se è directory
 - -s true se file non è vuoto



Valutazione delle condizioni

- test e come ultimo parametro) e seguiti da un'espressione la valutano e ritornano 0 o 1 in base al risultato
 - altri 20 su help test
- Test binari, due parametri: test ARG1 OP ARG2
 - confronto lessicale tra stringhe
 - =, !
 - confronto numerico tra stringhe
 - -eq, -ne equal, not equal
 - -lt, -le less than, less or equal
 - -gt, -ge greater than, greater or equal
 - confronto tra file



Valutazione delle condizioni

- test e come ultimo parametro) e seguiti da un'espressione la valutano e ritornano 0 o 1 in base al risultato
 - -nt newer than
 - -ot older than
 - op binari == e !
 - Es
 - op binario =~ matcha param
 - Es
- `cd "$MYDIR" || echo ; exit` il ramo destro viene eseguito solo se fallisce il sinistro, con messaggio e bloccante
- if è utile tutte le volte che voglio esecuzione condizionata: la keyword if vuole un COMANDO come argomento e solo se il comando ritorna 0 viene considerato come TRUE, qualsiasi altro ritorno viene visto come FALSE

Valutazione delle condizioni

- case permette di esprimere condizioni multiple in modo più leggibile che in una catena di elif
 - WORDS = pattern di pathname expansion
 - itera direttamente sui nomi di file prodotti dall'espansione
- while permette cicli indefiniti basandosi su un comando da eseguire per verificare se l'exit code è TRUE , sintassi

```
mirror_mod = modifier_ob.  
#set mirror object to mirror.  
mirror_mod.mirror_object  
operation == "MIRROR_X":  
mirror_mod.use_x = True  
mirror_mod.use_y = False  
mirror_mod.use_z = False  
operation == "MIRROR_Y":  
mirror_mod.use_x = False  
mirror_mod.use_y = True  
mirror_mod.use_z = False  
operation == "MIRROR_Z":  
mirror_mod.use_x = False  
mirror_mod.use_y = False  
mirror_mod.use_z = True  
#selection at the end -add  
mirror_ob.select= 1  
modifier_ob.select=1  
context.scene.objects.active  
("Selected" + str(modifier_ob.  
mirror_ob.select = 0  
= bpy.context.selected_object  
data.objects[one.name].select  
print("please select exactly  
-- OPERATOR CLASSES --  
types.Operator):  
X mirror to the selected  
object.mirror_mirror_x"  
mirror X"  
context):  
context.active_object is not
```




Valutazione delle condizioni

- while COMANDO ; do LISTA ;COMANDI; ITERATI; done
 - cat /etc/passwd | while read riga ; do echo \$) \$riga ; done
 - cat legge tutto e mette su stdout, pipeline porta stdout su stdin del processo dopo, figlio read mangia una riga di input e restituisce true; while ha ricevuto true e fa iterazione: echo di valore di counter incrementato dopo averlo espanso seguito dal valore di riga
 - Versione con for: for riga in \$; do echo \$) \$riga ; done
 - Il for è vantaggioso perché non c'è pipeline, quindi tutto quello che avviene nelle iterazioni è nella stessa shell che lo lancia: si vede che se si rilancia il comando due volte, parte con il counter dallo stesso valore, la variabile è dello stesso ambiente

Valutazione delle condizioni

- while COMANDO ; do LISTA ;COMANDI; ITERATI; done
 - Se l'elemento informativo è la riga evito di usare un for, con while possiamo sfruttare read che ha il concetto di riga
- break esce da un ciclo for, while o until
- continue salta alla successiva iterazione di un ciclo





Valutazione delle condizioni

- con test /
 - operatori AND, OR, NOT sono rispettivamente -a, -o, !
 - si possono fare raggruppamenti con , vanno protette con backslash , si usano \
- con]
 - operatori AND, OR, NOT sono rispettivamente &&, | |, !
 - si possono fare raggruppamenti con , protette già dalle], si usano così come sono



Valutazione delle condizioni

- WORDS = parametri della command line
 - for PAR in "\$@" ; do echo "\$PAR" ; done
- WORDS = command substitution
 - for USER in \$
- WORDS = brace expansion
 - for ITEM in item_
- Si possono impiegare sequenze numeriche usando NON MOSTRATO
 - for BACKWARDSTENTHS in \$ dove



Valutazione delle condizioni

- Nelle versioni recenti di bash sintassi C-like NON MOSTRATO
 - for)
 - Espressioni di inizializzazione, test di terminazione , espressioni eseguite ad ogni iterazione



Accorgimenti utili

- source : utile per condividere parametri tra script correlati tra loro e per creare librerie di funzioni importabili



Accorgimenti utili

- su è utile da root per eseguire comandi con altre identità, ma non è exec: non si mettono comandi a seguire, apre una shell e finché non si esce da quella non prosegue nel contesto chiamante
 - con -s può impostare orologio di sistema
 - può restituire l'orologio in formati diversi
 - date +FORMAT permette di selezionare cosa e come visualizzare
 - FORMAT è una stringa in cui vengono interpretate sequenze speciali contrassegnate dal carattere % es
 - con -d si può fornire un timestamp da usare al posto del tempo corrente dà ora di evento interessante convertito in epoch,) modifica tempo, date -d "@\$N" stampa in modo leggibile) @ operatore per cambiare da epoch a timestamp
 - -d crea una directory
 - -p DIR crea all'interno di DIR invece che in /tmp
 - Es



Accorgimenti utili

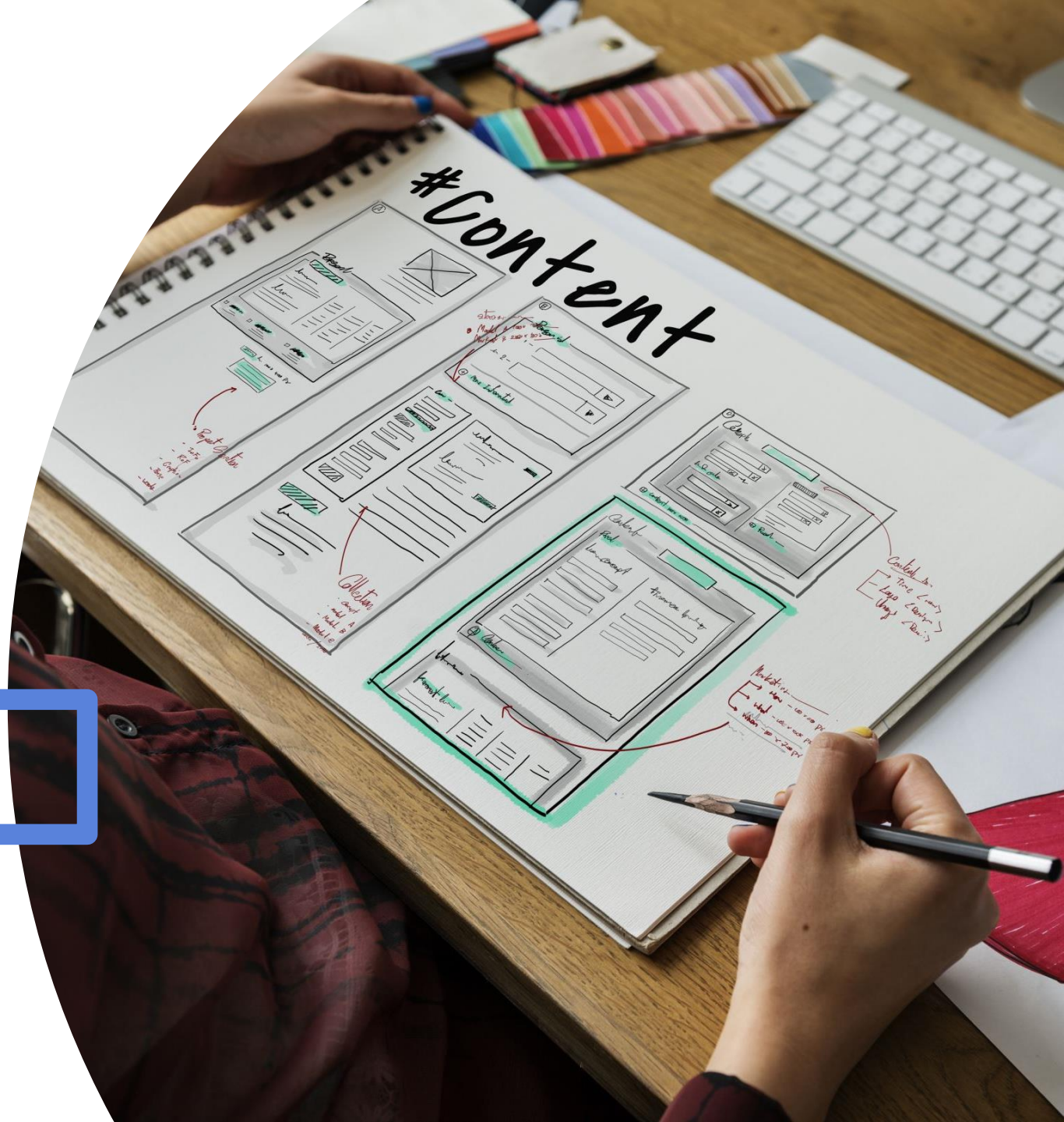
- `dirname` rimuove l'ultimo componente del percorso da un nome di file
 - Es
- `eval` permette di processare un file come se fosse uno script, sottoponendolo ai 12 passi di valutazione



Funzionamento
in rete



Configurazione di rete





Richiami di reti, reti locali



- NAT
 - A default c'è una sola interfaccia in NAT: permette al guest di chiedere direttamente all'hypervisor un indirizzo per uscire sulla rete



Richiami di reti, reti locali

- Bridged
 - Si comportano come se fossero connesse all'interfaccia dell'host attraverso un bridge/hub, quindi come se attestate sulla stessa LAN dell'host
 - Vagrant può assegnare un'interfaccia bridged ad una VM con direttiva `config.vm.network "public_network"` Nel vagrantfile, entro il ciclo di configurazione `do | config |` se c'è questa direttiva, viene matchata con interfaccia bridged di virtualbox
 - `config.vm.network "public_network", ip: "192.168.0.17"` si può assegnare IP statico che verrà dato alla VM in fase di provisioning; usando vagrant per configurare il guest, assegnandogli direttamente un indirizzo IP
 - `config.vm.network "public_network", bridge: "eth0"` si può selezionare a quale interfaccia host agganciarsi



Richiami di reti, reti locali

- Bridged
 - `config.vm.network "public_network", auto_config: false` si può disabilitare l'automatismo per lasciare che il guest si occupi della configurazione, utile ad esempio nel caso nella VM ci sia un client DHCP: servirà un server DHCP sulla rete reale che fornisca configurazione alla VM
- Host-only
 - Hypervisor genera interfaccia virtuale sull'host e le assegna un IP di una specifica subnet, poi connette alla LAN virtuale la VM, così che questa possa comunicare solo con l'host
 - Vagrant assegna interfaccia host-only a VM con direttiva `config.vm.network "private_network"`
 - `config.vm.network "private_network", name: "vboxnet3"` si può creare più di una rete host-only specificando il nome
 - valgono opzioni `ip` e `auto_config` come per `bridged`



Richiami di reti, reti locali

- Internal
 - Hypervisore assegna queste interfacce a LAN completamente virtuale, e fa sì che solo interfacce della stessa internal network possano comunicare tra loro
 - Vagrant le tratta come caso speciale di `private_networkconfig.vm.network "private_network", virtualbox__intnet: "LAN1"`
 - Valgono opzioni `ip` e `auto_config`, potrebbero essere diverse con altri provider)

Reti globali



Le isole sono interconnesse da appalti che fanno da ponte spesso realizzati con tecnologie diverse da quelle dell'isola



L'obiettivo di IP è rendere possibile il dialogo tra network a prescindere dall'implementazione, è realizzato per lavorare indifferentemente su tecnologie diverse

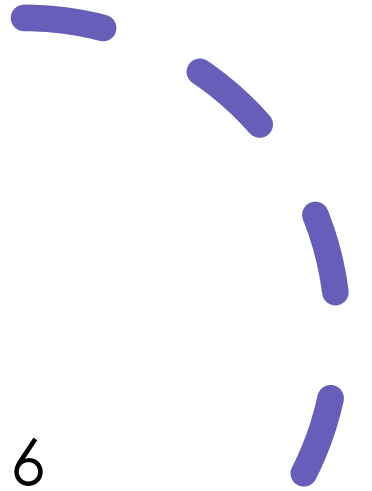


LAN con switch separati, interconnesse mediante un router, hanno possibilità di broadcast separato e possono contare su maggiore sicurezza



NAT

- 10.0.0.0/8
- 172.16.0.0/16 – 172.31.0.0/16
- 192.168.0.0/24 – 192.168.255.0/24



NAT

- Vagrant stesso da fa SNAT-Router
 - la VM viene configurata per usare come default gateway un'interfaccia virtuale che consegna i pacchetti al processo VirtualBox
 - I parametri dell'interfaccia sono assegnati automaticamente, ma si possono configurare con `config.vm.base_maceconfig.vm.base_address`
- Si può configurare VirtualBox perché si comporti anche da DNAT-Router
 - Il processo Vbox si mette in ascolto su una porta TCP o UDP dell'host, il traffico entrante viene modificato assegnando come destinazione l'IP della scheda virtuale NAT del guest
 - Le mappature si configurano con, ad esempio `config.vm.network "forwarded_port", guest: 80, host: 8080`
 - il parametro opzionale `host_ip` può essere usato per limitare raggiungibilità della porta, es. `host_ip: "127.0.0.1"`

Configurazione



- Configurazione runtime
 - suite iproute2 formata da comando ip e sottocomandi, permette controllo completo di tutti gli aspetti più avanzati
 - sottocomando address manipola interfacce
 - visualizzazione ip a
 - assegnazione ip a add <address>/<mask> dev <interface>
 - rimozione ip a del <address>/<mask> dev <interface>
 - sottocomando route manipola tabelle di indirizzamento
 - visualizzazione ip r
 - routing via gateway ip r add <dst_net>/<mask> via <gw_addr>
 - routing via dev ip r add <dst_net>/<mask> dev <interface>)

Configurazione

- Configurazione runtime
 - rimozione ip a del <address>/<mask>
- Configurazione persistente
 - file /etc/network/interfaces vedi man 5 interfaces
 - snippet nella cartella /etc/network/interfaces.d/
 - Esempio: auto eth0# attiva con ifup -a

Configurazione



ping <IP>verifica base della connettività



tracert <IP>verifica del percorso dei pacchetti



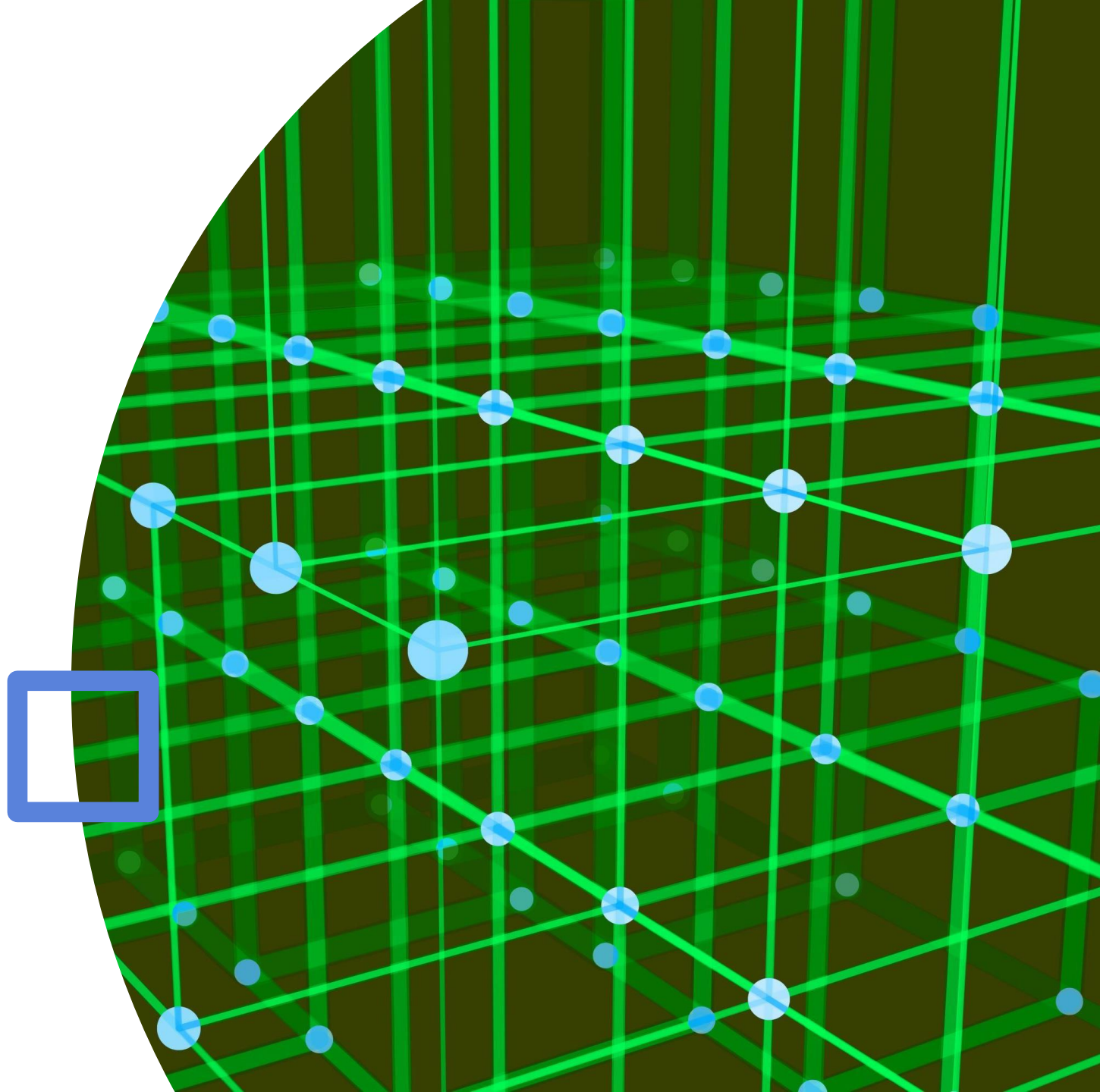
ssverifica stato delle connessioni

- t / -u TCP/UDP only
- l / -astato LISTEN /ALL
- nnon risolve indirizzi/porte in nomi simbolici
- pmostra processi che usano la socket



tcpdump e wireshark per intercettare contenuto dei pacchetti: lo stato delle socket è su ogni endpoint, con questi due invece possiamo vedere proprio i pacchetti in transito

Servizi di rete di base





NSS

- success, rispondere che lo trova, e la catena di chiamate torna all'origine col risultato
- unavail, risultato preliminare che vuol dire che ldap non riesce a rispondere: si passa a controllare dns
- tryagain, riprovare più tardi
- not found, per com'è scritta la regola, se ldap dà not found, c'è il return e si ferma la ricerca; quindi si dà massima autorità a ldap stesso



Risoluzione nomi via NSS

- `getent <dbname>`
`<keyword>` nome del db da interrogare e `keyword` su cui fare la ricerca





Risoluzione nomi DNS diretta

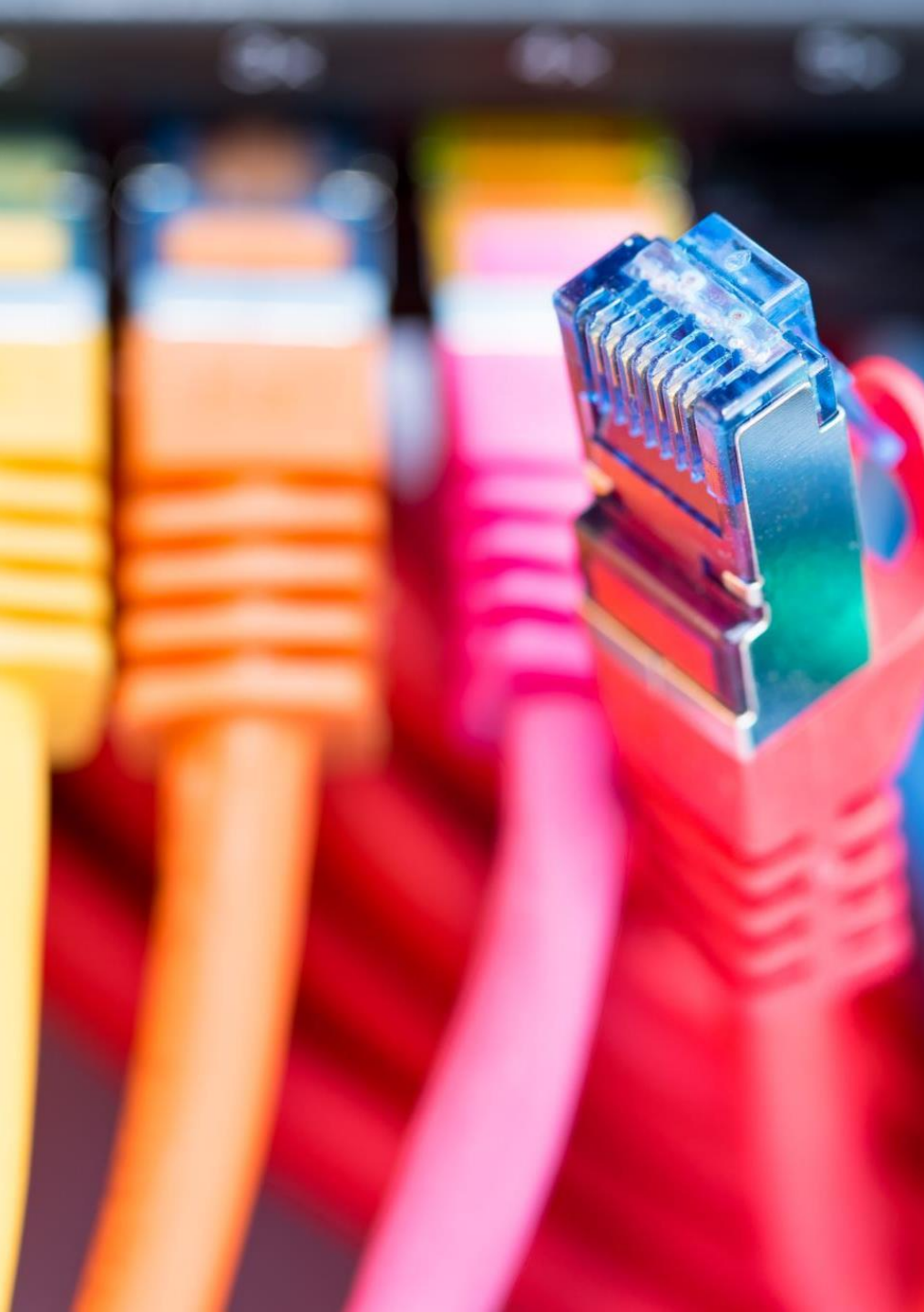
- hostsi usa tipicamente per conversioni IP ← → nome
 - query di un nome: host `www.unibo.it` interroga il server indicato in `/etc/resolv.conf` e restituisce risultato, se lo trova
 - query di server specifico: host `www.unibo.it 8.8.8.8` così salta il file `resolv.conf` e va a chiedere direttamente al name server indicato
- dig per ottenere informazioni legate a un dominio, diverse da nomi host: i record `ns` dicono qual è il name server di un dominio, i record `mx` quali sono i mail exchanger
 - conoscere i Mail eXchanger: `dig mx example.com`
 - conoscere i Name Server: `dig ns example.com`



Zeroconf

- link-local addressing: per determinare automaticamente un indirizzo di rete, assegnazione di indirizzi validi sulla LAN: gli indirizzi di layer 2 per definizione valgono solo localmente
 - Link-local IPv4: riservata a questo scopo la classe 169.254/16, best practice per l'uso degli indirizzi sono:
 - evitare l'assegnamento a interfacce che hanno indirizzi instradabili, non distribuirli con DHCP, non associarli stabilmente a nomi DNS
 - Viene scelto IP random nel range 169.254.1.0 – 168.254.254.255, con seed legato a caratteristica univoca
 - Poi si verifica che qualcuno non abbia già l'indirizzo con ARP probe
 - Infine si annuncia acquisizione con gratuitous ARP





Zeroconf


- link-local addressing: per determinare automaticamente un indirizzo di rete, assegnazione di indirizzi validi sulla LAN: gli indirizzi di layer 2 per definizione valgono solo localmente
 - Link-local IPv6: gli indirizzi IPv6 sono divisi in 64 bit di Subnet Prefix e 64 bit di Interface ID. IPv6 definisce un range per indirizzi link-local, con prefix FE80::/
 - Si prende il MAC (, si allunga di 2 byte mettendo in mezzo i due byte fissi :ff:fe
 - Si inverte il 7° bit del primo byte
 - Il risultato è l'Interface ID da concatenare a FE80
 - IPv6 ha sistema flessibile per determinare se indirizzo è libero e valutare se rete locale è raggiungibile dall'esterno: SLAAC è l'algoritmo per costruire indirizzi link-local validi globalmente configurando host in automatico

Zeroconf

- link-local addressing: per determinare automaticamente un indirizzo di rete, assegnazione di indirizzi validi sulla LAN: gli indirizzi di layer 2 per definizione valgono solo localmente
 - Definizione token identificativo interfaccia
 - generazione indirizzo IPv6 link-local con prefix FE80::/10
 - invio di Neighbor Solicitation , se nessuno risponde si avvanza, altrimenti indirizzo è occupato e processo automatico si ferma
 - invio di Router Solicitation all'indirizzo multicast di tutti i router
 - Se c'è risposta, si valuta la risposta
 - Risposta M: è necessario DHCP, si va nello stesso stato di mancanza router
 - Risposta A : concatenazione del prefisso con indirizzo globale ottenuto



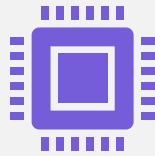
Zeroconf

- link-local addressing; per determinare automaticamente un indirizzo di rete, assegnazione di indirizzi validi sulla LAN: gli indirizzi di layer 2 per definizione valgono solo localmente
 - Se non c'è risposta, si fa richiesta in broadcast per scoprire se c'è server DHCPv6 che può darmi un indirizzo
 - Risposta ricevuta, si assegna e usa l'indirizzo ricevuto
 - Nessuna risposta, si mantiene indirizzo link-local
 - multicast DNS; per la traduzione di nomi in indirizzi in mancanza di DNS unicast configurato manualmente
 - service discovery :basato su server DNS aggiornabile dinamicamente per registrare servizi, per scoprire nomi di servizi anche senza un DNS indicato specificamente
- 

Sincronizzazione



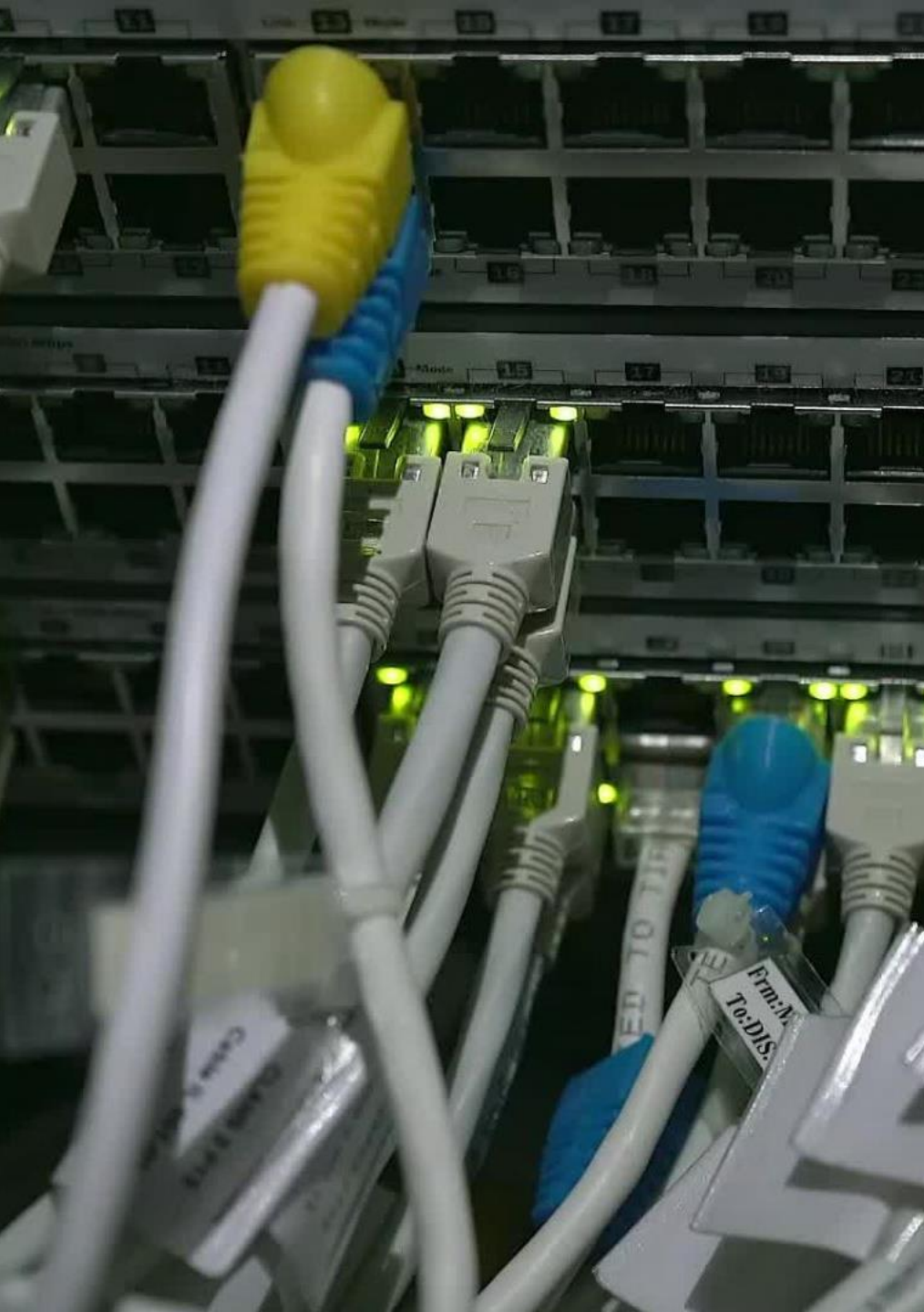
L'allineamento dell'ora di un sistema ad un orologio di riferimento è molto importante, ad esempio per la diagnostica dei problemi , o per i protocolli di autenticazione e autorizzazione



E' possibile usare un protocollo, detto NTP , che compensa i ritardi di rete per ottenere informazioni precise via internet



NTP è preciso ; standard , scalabile e affidabile



Strumenti Linux per i servizi di rete

- Client side
 - Avahi
 - ISC dhcp)
 - systemd.network
 - systemd-resolved
 - ntpd /ntpdate



Strumenti Linux per i servizi di rete

- Server side
 - dnsmasq
 - inoltre PXE e TFTP
 - systemd.dnnsd
 - ntpd

Periodic Table of the Elements

Number →																		1		← Symbol																	
Name →																		H		← Atomic Weight																	
																		Hydrogen		1.008																	
6 VIB						7 VIIB		8 VIIIB		9 VIIIB		10 VIIIB		11 IB		12 IIB		13 IIIA		14 IVA																	
24 Cr		25 Mn		26 Fe		27 Co		28 Ni		29 Cu		30 Zn		31 Ga		32 Ge		33 As		34 Se																	
Chromium		Manganese		Iron		Cobalt		Nickel		Copper		Zinc		Gallium		Germanium		Arsenic		Selenium																	
51.9961		54.938044		55.845		58.933194		58.6934		63.546		65.38		69.723		72.63		74.9216		78.96																	
42 Mo		43 Tc		44 Ru		45 Rh		46 Pd		47 Ag		48 Cd		49 In		50 Sn		51 Sb		52 Te																	
Molybdenum		Technetium		Ruthenium		Rhodium		Palladium		Silver		Cadmium		Indium		Tin		Antimony		Tellurium																	
95.95		(98)		101.07		102.90550		106.42		107.8682		112.414		114.818		118.710		121.757		127.60																	
74 W		75 Re		76 Os		77 Ir		78 Pt		79 Au		80 Hg		81 Tl		82 Pb		83 Bi		84 Po																	
Tungsten		Rhenium		Osmium		Iridium		Platinum		Gold		Mercury		Thallium		Lead		Bismuth		Polonium																	
183.84		186.207		190.23		192.227		195.084		196.966569		200.592		204.38		207.2		208.9804		209																	
106 Sg		107 Bh		108 Hs		109 Mt		110 Ds		111 Rg		112 Cn		113 Nh		114 Fl		115 Mc		116 Lv																	
Seaborgium		Bohrium		Hassium		Meitnerium		Darmstadtium		Roentgenium		Copernicium		Nihonium		Flerovium		Moscovium		Livermorium																	
(269)		(270)		(269)		(278)		(281)		(282)		(285)		(286)		(289)		(292)		(293)																	
60 Nd		61 Pm		62 Sm		63 Eu		64 Gd		65 Tb		66 Dy		67 Ho		68 Er		69 Tm		70 Yb																	
Neodymium		Promethium		Samarium		Europium		Gadolinium		Terbium		Dysprosium		Holmium		Erbium		Thulium		Ytterbium																	
144.242		(145)		150.36		151.964		157.25		158.92535		162.500		164.93033		167.259		168.934		173.054																	
92 U		93 Np		94 Pu		95 Am		96 Cm		97 Bk		98 Cf		99 Es		100 Fm		101 Md		102 No																	
Uranium		Neptunium		Plutonium		Americium		Curium		Berkelium		Californium		Einsteinium		Fermium		Mendelevium		Nobelium																	
238.02891		(237)		(244)		(243)		(247)		(247)		(251)		(252)		(257)		(258)		(259)																	

Strumenti Linux per i servizi di rete

- bind-interface evita conflitti in caso si vogliono usare più istanze di dnsmasq per reti diverse connesse al server
- interface=<interface name> e
- listen-address=<ipaddr> mettono dnsmasq in ascolto solo sull'interfaccia o indirizzo indicati
- user / group / pid utente e gruppo UNIX del processo, file in cui salvare il PID



Strumenti Linux per i servizi di rete

- dnsmasq può fare da DHCP, il server DHCP è disabilitato se non sono specificate le opzioni descritte di seguito
 - `dhcp-range=<start-addr>]`
 - questo specifica la rete da assegnare mediante indirizzo inizio, fine e netmask
 - indirizzi tra `<start-addr>` e `<end-addr>`, se specificato imposta il `<lease time>`
 - la `<netmask>` è opzionale per reti connesse direttamente al server
 - al posto di `<end addr>`, `<mode>` può essere `static` per abilitare il server sulla rete indicata senza servire indirizzi dinamici, ma solo quelli specificati con opzione `dhcp-host`
 - `dhcp-host=`
 - questo caso serve per assegnare indirizzi statici
 - assegna `<hostname>`, `<ipaddr>` e `<lease time>` stabili all'host con `MAC=<hwaddr>`
 - con `ignore` non fornirà mai un lease all'host indicato

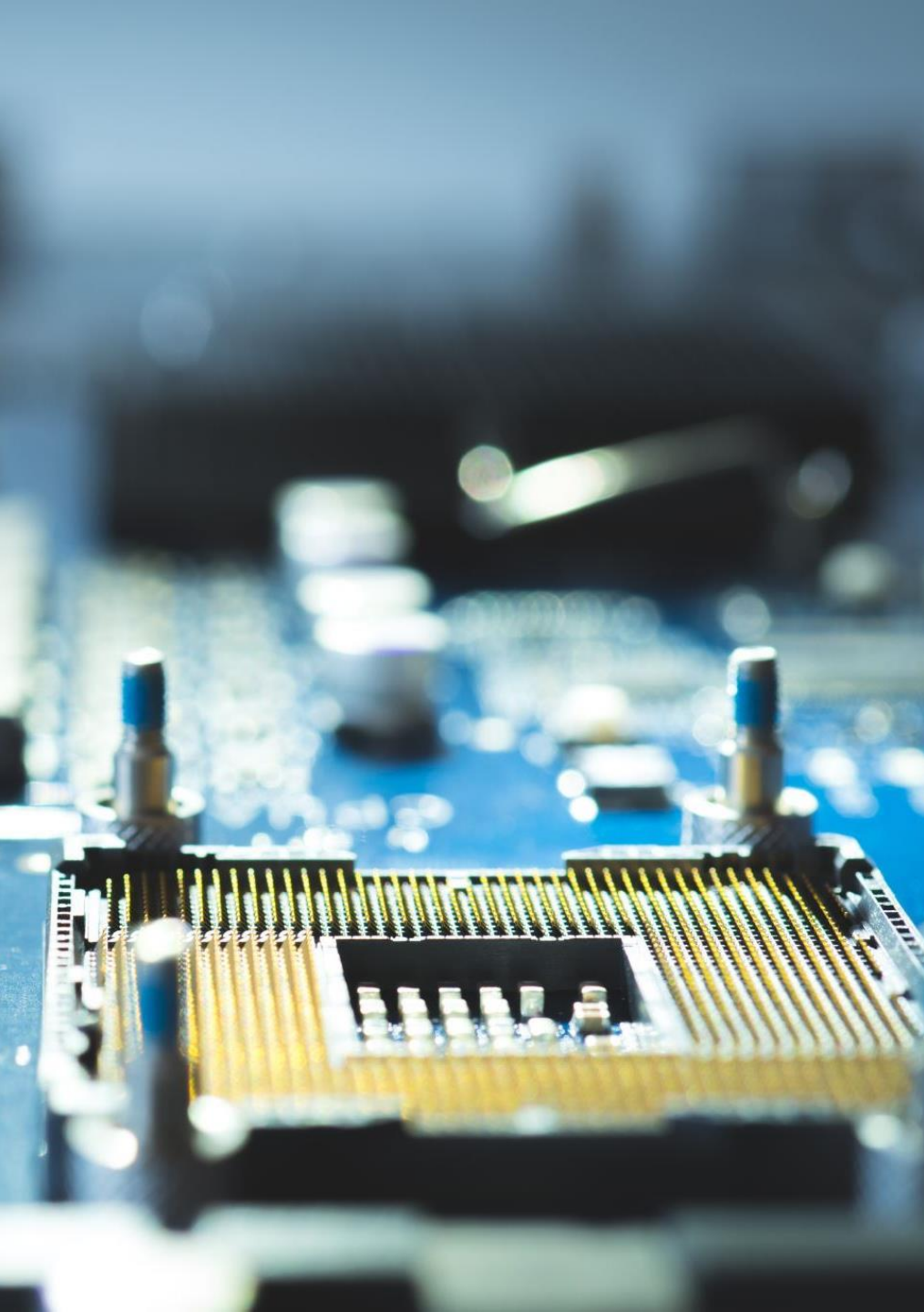
Strumenti Linux per i servizi di rete

- dnsmasq può fare da DHCP, il server DHCP è disabilitato se non sono specificate le opzioni descritte di seguito
 - dhcp-hostfile permette di specificare una directory di file contenenti informazioni formattate come la parte a destra dell' '=' di dhcp-host
 - dhcp-option= ,]uso più comune: dhcp-option=<opt>,<value>
 - dnsmasq fornisce un sacco di info, come convenzione si indica il gateway con opzione -3, per indicare rotta di default
 - Opzioni comuni
 - 1 netmask
 - 2 fuso orario
 - 3 default gateway
 - 4 time server
 - 6 DNS server



Strumenti Linux per i servizi di rete

- dnsmasq può fare da DHCP, il server DHCP è disabilitato se non sono specificate le opzioni descritte di seguito
 - 12host name
 - 15domain name
 - 121static route
 - possibile definire anche rotte statiche:
es
- port=<dns server port>
 - di default 53, se impostato a 0 disabilita il server DNS
- local-service
 - accetta query DNS solo dagli host delle subnet locali al server: tipica di default, previene DNS amplification attacks
 - ha effetto solo se non è specificata nessuna delle opzioni seguenti



Strumenti Linux per i servizi di rete

- dnsmasq può fare da DHCP, il server DHCP è disabilitato se non sono specificate le opzioni descritte di seguito
 - interface
 - except-interface
 - listen-address
 - auth-server options
- ntpd è client e/o server in funzione della configurazione, fatta su `/etc/ntp.conf`

Strumenti Linux per i servizi di rete

- `conresolv-file=<file>`
 - sopprime l'uso di `/etc/resolv.conf`
- `conserver= <ipaddr>`
 - si deve aggiungere `no-resolv`, per evitare uso di `/etc/resolv.conf`

Strumenti Linux per i servizi di rete

- DHCP
 - Pacchetto isc-dhcp-client fornisce comando dhclient, che lanciato senza parametri avvia un demone che tenta di configurare tutte le interfacce
 - File /etc/dhcp/dhclient.conf per parametri impostabili
 - Hook per esecuzione automatica di script al cambio di stato interfaccia, nelle dir
 - /etc/dhcp/dhclient-enter-hooks.d/*
 - /etc/dhcp/dhclient-exit-hooks.d/*



Strumenti Linux per i servizi di rete

- zeroconf
 - framework avahi offre stack completo mDNS/DNS-SD con API per integrazione delle funzionalità con qualsiasi programma C; un demone per gestire registrazione nuovi servizi in modo orchestrato da qualsiasi programma non scritto in C ; un client/wrapper C che semplifica uso di D-Bus; adattatori per integrare le API di avahi negli event loop dei sistemi grafici come GNOME e KDE
 - il demone è responsabile ad esempio della scoperta automatica di stampanti in una rete locale
 - sono disponibili pacchetti con strumenti per svolgere funzioni singole specifiche
- link local
 - pacchetto avahi-autoipd fornisce comando omonimo che implementa IPv4 Link Local, mediante demone indipendente, oppure, nel file /etc/network/interfaces



Strumenti Linux per i servizi di rete

- ad ogni cambio stato dell'interfaccia invoca `/etc/avahi/avahi-autoipd.action`
- può essere usato come fallback se DHCP fallisce

SNMP



Essere meno invasivo possibile, aspetto di sicurezza: certe cose non si possono fare quindi certi errori non si possono fare, anche volendo



Costi: ogni genere di apparato di rete oggi disponibile a basso costo, quindi senza un microprocessore con SO



il modello dei dati, per cui qualunque proprietà di rete mi interessi identificare, viene rappresentata come oggetto

La proprietà ha ricevuto nome univoco e ha ricevuto una sintassi standard



Il modello di interazione: questi dati dove stanno?



Modello dei dati: OID e MIB

- da un lato è una collezione teorica che mi dice come sono identificati e definiti anche tutti gli oggetti del mondo ; quindi descrizione dell'intero albero globale degli OID
- dall'altro lato, comprato un apparato, mi chiedo qual è il MIB che descrive proprietà di quell'apparato: sarà un sottoalbero di quello globale, fatto di pezzetti di quello globale

Periodic Table of the Elements

Number → 1																					
Symbol ← H																					
Name → Hydrogen																					
Atomic Weight ← 1.008																					
6 VIB						7 VIIB		8 VIIIB		9 VIIIB		10 VIIIB		11 IB		12 IIB		13 IIIA		14 IVA	
24 Cr Chromium 51.9961		25 Mn Manganese 54.938044		26 Fe Iron 55.845		27 Co Cobalt 58.933194		28 Ni Nickel 58.6934		29 Cu Copper 63.546		30 Zn Zinc 65.38		31 Ga Gallium 69.723		32 Ge Germanium 72.63		5 B Boron 10.81		6 C Carbon 12.01	
42 Mo Molybdenum 95.95		43 Tc Technetium (98)		44 Ru Ruthenium 101.07		45 Rh Rhodium 102.90550		46 Pd Palladium 106.42		47 Ag Silver 107.8682		48 Cd Cadmium 112.414		49 In Indium 114.818		50 Sn Tin 118.71		13 Al Aluminium 26.9815385		14 Si Silicon 28.08	
74 W Tungsten 183.84		75 Re Rhenium 186.207		76 Os Osmium 190.23		77 Ir Iridium 192.217		78 Pt Platinum 195.084		79 Au Gold 196.966569		80 Hg Mercury 200.592		81 Tl Thallium 204.38		82 Pb Lead 207.2		31 Ga Gallium 69.723		32 Ge Germanium 72.63	
106 Sg Seaborgium (269)		107 Bh Bohrium (270)		108 Hs Hassium (269)		109 Mt Meitnerium (278)		110 Ds Darmstadtium (281)		111 Rg Roentgenium (282)		112 Cn Copernicium (285)		113 Nh Nihonium (286)		114 Fl Flerovium (289)		31 Ga Gallium 69.723		32 Ge Germanium 72.63	
60 Nd Neodymium 144.242		61 Pm Promethium (145)		62 Sm Samarium 150.36		63 Eu Europium 151.964		64 Gd Gadolinium 157.25		65 Tb Terbium 158.92535		66 Dy Dysprosium 162.500		67 Ho Holmium 164.93033		68 Er Erbium 167.25		31 Ga Gallium 69.723		32 Ge Germanium 72.63	
92 U Uranium 238.02891		93 Np Neptunium (237)		94 Pu Plutonium (244)		95 Am Americium (243)		96 Cm Curium (247)		97 Bk Berkelium (247)		98 Cf Californium (251)		99 Es Einsteinium (252)		100 Fm Fermium (257)		31 Ga Gallium 69.723		32 Ge Germanium 72.63	

Modello dei dati: OID e MIB

- Tipi di dato semplici
 - interi a 32 bit con segno
 - stringhe di byte
 - OID

Modello dei dati: OID e MIB

- Tipi di dato application-wide
 - network addresses: come IPv4, come generiche stringhe di byte
 - counters: interi a 32/64 bit positivi e crescenti , rollover a zero
 - gauges: interi non negativi con limiti minimo e massimo
 - time ticks: centesimi di secondo trascorsi da un dato evento
 - opaques: stringhe arbitrarie senza controllo di sintassi
 - integers e unsigned integers: ridefiniscono gli interi per avere precisione arbitraria
 - bit strings: stringhe di bit singolarmente identificati
- l'OID rappresenta in astratto la proprietà : es

Modello dei dati: OID e MIB

caso scalare: alla proprietà è associato un valore scalare Per esprimere il concetto di VALORE della proprietà si aggiunge zero all'identificativo: quindi per interrogare il dispositivo, si scrive "nome proprietà.0", che sarà OID per letture/scritture dispositivo

caso tabella: alla proprietà è associata una tabella, per indicarne un valore si aggiunge .colonna.riga per leggere valore di una cella

Modello dei dati: OID e MIB

- MIB-2: e' il sottoinsieme che inizia in 1.3.6.1.2.1; includeva tutti i dati essenziali agli apparati di rete
 - Il gruppo system 1.3.6.1.1.2.1.1, usato per memorizzare dentro l'apparato stesso
 - sysDescr , descrizione libera
 - sysContact , persona responsabile
 - sysName , nome apparato
 - sysLocation , posizione apparato
 - Il gruppo IP 1.3.6.1.2.1.4, formato da 19 scalari + 4 tabelle che descrivono com'è configurato stack IP dell'apparato ; usato per memorizzare parametri generali del protocollo e tabelle con parametri specifici di ogni interfaccia, regola di routing, mappatura MAC
 - ifIndex indice, numero ordinale dell'interfaccia
 - ifDescr descrizione
 - ifType tipo



Modello dei dati: OID e MIB

- MIB-2: e' il sottoinsieme che inizia in 1.3.6.1.2.1; includeva tutti i dati essenziali agli apparati di rete
 - ifMtu
 - ifSpeed velocità
 - ifPhysAddress indirizzo MAC
 - ifOutOctets quanti pacchetti sono usciti dall'interfaccia
- PEN : il sottoalbero su 1.3.6.1.4.1 è dedicato a moduli specifici richiesti da enti privati
 - UCD-SNMP , sotto University of California Davids:in questo modulo hanno definito tutte le proprietà tipiche di un sistema operativo
 - NET-SNMP-EXTEND-MIB è l'output della direttiva extend, qui possono essere fatti comparire dinamicamente nuovi oggetti in base all'output di script di un managed object

Modello di interazione e protocolli

- SNMP è un protocollo a livello applicativo, trasportato su UDP, con agent in ascolto su porta 161 e manager su porta 162 per ricevere le trap
 - Le 4 possibili richieste che un manager può fare ad un agent
 - GetRequestrichiesta base che un manager fa ad un agent per sapere quanto vale un OID; richiede il valore associato al managed object
 - SetRequest richiede di settare il valore associato ad un managed object
 - GetNextRequestrichiede i scoprire qual è OID del managed object successivo quello specificato
 - Dettagli: Getnextrequest implementa la visita dell'albero: questa si può fare in diversi modi
 - o restare al primo livello e chiedere gli altri attaccati alla radice, scoprendo quali sono tutti i nodi del secondo prima di scendere e scoprire tutto il terzo

Modello di interazione e protocolli

- SNMP è un protocollo a livello applicativo, trasportato su UDP, con agent in ascolto su porta 161 e manager su porta 162 per ricevere le trap
 - oppure scendere al primo nodo di questo livello e continuare a scendere finché non trovo una foglia, prima di risalire ed esplorare orizzontalmente
 - getNextRequest permette di dire: caro agent, so che gestisci OID con ident
- GetbulkRequest da v2 in poi è versione ottimizzata di getNextRequest: si evita di avere spedizione pacchetto -> elaborazione -> pacchetto di ritorno; con getbulk si richiede di recuperare tutti gli oggetti successivi a quello indicato, fino a riempire il pacchetto UDP
- Poi ci sono le 4 risposte possibili dall'agent al manager
 - Trapnotifica asincrona dall'agent al manager

Modello di interazione e protocolli

- SNMP è un protocollo a livello applicativo, trasportato su UDP, con agent in ascolto su porta 161 e manager su porta 162 per ricevere le trap
 - InformRequestda v2, come sopra ma con conferma di ricezione: richiedere risposta di conferma non è aggiunta da poco, specie perché SNMP è su UDP
 - Response è usata dall'agent per rispondere a Request
 - Reportsolo v3, usata per comunicazione inter-engine, principalmente per segnalare problemi con elaborazione messaggi ricevuti
- DETTAGLI: SNMP non è molto comprensibile, è efficiente: risale a quando i byte avevano un alto costo

Modello di interazione e protocolli



nuovi data type a 64bit



comandi PDU GetBulk e Inform



party-based security system, molto macchinoso e di scarsa adozione, tanto da portare alla creazione di due sotto-versioni

SNMPv2c
SNMPv2u



un engine: componente software dedicato a invio/ricezione messaggi, estrarre i dati e gestire la sicurezza



più application: implementano algoritmo che permette di confezionare la risposta giusta



LDAP

- Come scegliere localmente la sorgente dei dati
 - NSS è un sistema generale per selezionare i servizi di nomi, abbiamo già visto che permette di scegliere quale fonte usare per individuare concretamente un DB indicato come astratto
 - PAM è un sistema modulare per implementare authentication modules
- Come trasferire i dati dal server centrale ai client
 - LDAP è il protocollo di distribuzione delle informazioni
- fornisce grande velocità di lettura ma minore velocità di scrittura





LDAP

- E' un DB gerarchico, perché ogni nazione ha il suo namespace e perché al suo interno è diviso in diverse parti
 - Pregi
 - se ho modo semplice di immaginare la posizione dell'informazione; questa diventa criterio di ricerca iperefficiente; navigare una gerarchia è semplicissimo
 - molto facile creare delle deleghe nella gestione di sottoalberi: nessun problema di collisione, i nomi devono essere univoci globalmente solo nel loro livello di sottoalbero
 - Svantaggi
 - una volta individuata la struttura della gerarchia, è difficile cambiarla: si ha una visione rigida, con il tradeoff dello scegliere a priori la vista che mi interessa, e avere una ricerca meno efficiente se ce ne interessa un'altra dopo

LDAP

- Non è relazionale, si possono evitare le transazioni e si può usare meccanismo di locking semplice, perché le scritture sono rare e quindi che ce ne siano due nello stesso momento è difficile; un'ipotetica collisione data da modifica dello stesso dato, andrà gestita a mano, il protocollo non se ne occupa
- Deve essere altamente disponibile e quindi poter essere facilmente replicato; avere molte copie è importante

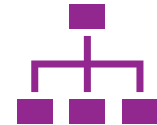
Modello dei dati



LDAP non si preoccupa di come vogliamo memorizzare la base di dati la cosa importante è che ci sia un modulo che LDAP possa usare per accedere fisicamente ai dati, DBI è un'interfaccia per accedere al backend più adatto



DBI gestisce una serie di entry, anche dette object; ogni entry ha una collezione di attribute, ognuno dei quali ha un type e può avere diversi value



Ogni entry ha un identificativo gerarchico detto DN



Replica

- dal lato del server principale, un nodo link permette di sapere a quale altro server subordinato rivolgersi per scendere nella gerarchia e avere i dati del sottoalbero non locale
- dal lato del server che ottiene la delega, il nodo link contiene l'indicazione del server a cui riferirsi per la parte superiore dell'albero: così il server subordinato sa che ha una sua radice ma che non è assoluta; ed in realtà c'è un genitore a cui può rivolgersi per conoscere il resto del DIT

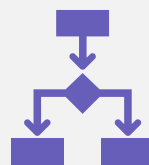
Formato entry



Questi non sono definiti con un nome come nei linguaggi di programmazione, ma si dichiara direttamente il valore etichettato dal tipo



Tra gli attributi, due sono presenti in tutte le entry: dn è nome univoco della entry, objectClass è la classe a cui appartiene la entry



Alcuni attributi sono univoci , altri possono essere multipli



Spazio dei nomi

- Introdurre id univoco, soluzione "da DB"
 - si può effettivamente aggiungere un attributo come `uniqueidentifier`, ma l'idea di LDAP è che gli attributi rappresentino qualcosa di proprio della entry
- Concatenare più coppie attributo-valore
 - Soluzione adottata da LDAP, si concatenano con + più coppie attributo=valore per ottenere RDN localmente univoco

Attributi



SYNTAX ,sintassi:Il vero e proprio tipo di dato nativo



Matching rules:Le regole per stabilire i criteri di confronto tra valori, per valutare uguaglianza e ordinamento nelle ricerche

ORDERING ,ordinamento:es
EQUALITY :es
SUBSTRING:potrei definire che il criterio di confronto è la presenza di una sottostringa



Vincoli d'uso:SINGLE-VALUEammette un solo valore di questo tipo nella entry ,ce ne sono altri come NO-USER-MODIFICATION, USAGE.

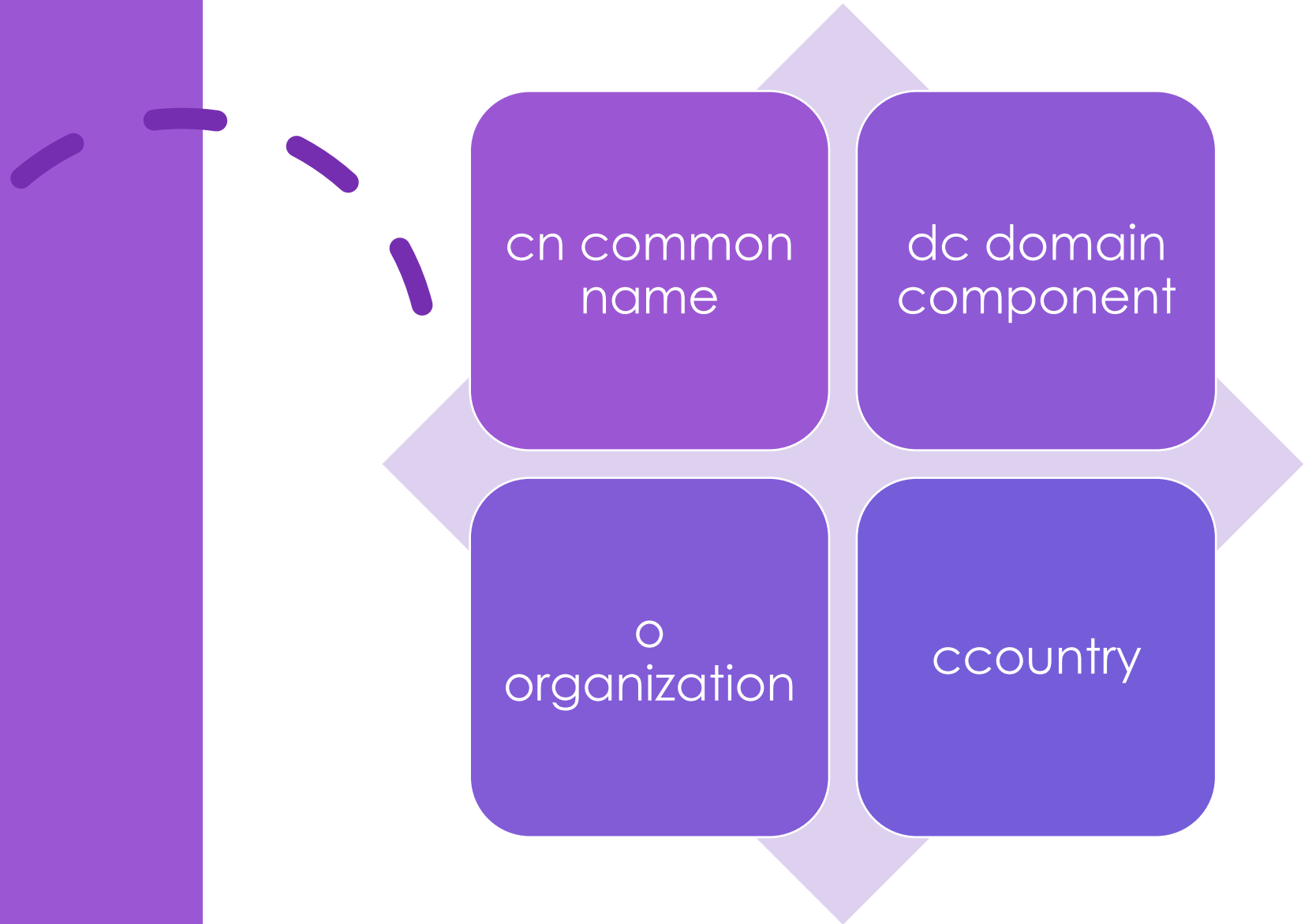


Attributi

- Eventuali dipendenze gerarchiche: con SUP <altro attributeType> il tipo eredita tutte le proprietà del superiore, può ridefinirne alcune, e nelle ricerche per un tipo superiore vengono restituiti tutti i valori dei tipi basati su quello. Quando creo una entry dell'utente marco, se ci metto dentro un telefono e un cellulare, se qualcuno cerca "telefono" verrà fuori sia telefono che cellulare: questo permette di definire in modo ordinato, separatamente, 1 telefono, 1 cellulare e 1 numero di casa; ma di ottenere comunque tutti e tre i risultati se si richiede "telefono", quando non si sa quale si vuole specificamente)



Attributi





Classi

- ABSTRACT servono per creare tassonomia di oggetti, ma sono troppo astratte originare entry
 - es
- STRUCTURAL servono per descrivere categorie di oggetti concreti
 - es
- AUXILIARY possono descrivere collezioni di attributi aggiuntivi che arricchiscono una entry, senza essere collegati a categorie specifiche di oggetti
 - fascicoloSanitario può estendere alcune entry di cane: per definire un'entry di classe cane dovrò/potrò specificare nome, razza,peso;poi potrò dire che una entry cane ha ANCHE objectClass fascicoloSanitario, e questo mi permetterà/obbligherà di inserire anche i dati aggiuntivi
 - Permette di specificare, ad esempio, una classe ausiliaria che prevede attributo colore; senza bisogno di creare due classi separate AutoConColore/AutoSenzaColore



Classi



Se in Java potrei avere il problema di 2 classi che definiscono attributo A con tipi diversi, in LDAP invece attributeType "nome" di classi docente e rettore non è definito nella entry, ma ESTERNAMENTE, nello schema: quando mi riferisco ad un certo tipo, e' per forza QUEL tipo



E' come dire che in JAVA, tutte le variabili che si chiamano A hanno lo stesso tipo



L'effetto è che la entry che usa le diverse classi deve contenere tutti gli attributi MUST di tutte le sue classi, e può contenere tutti gli attributi MAY

Classi





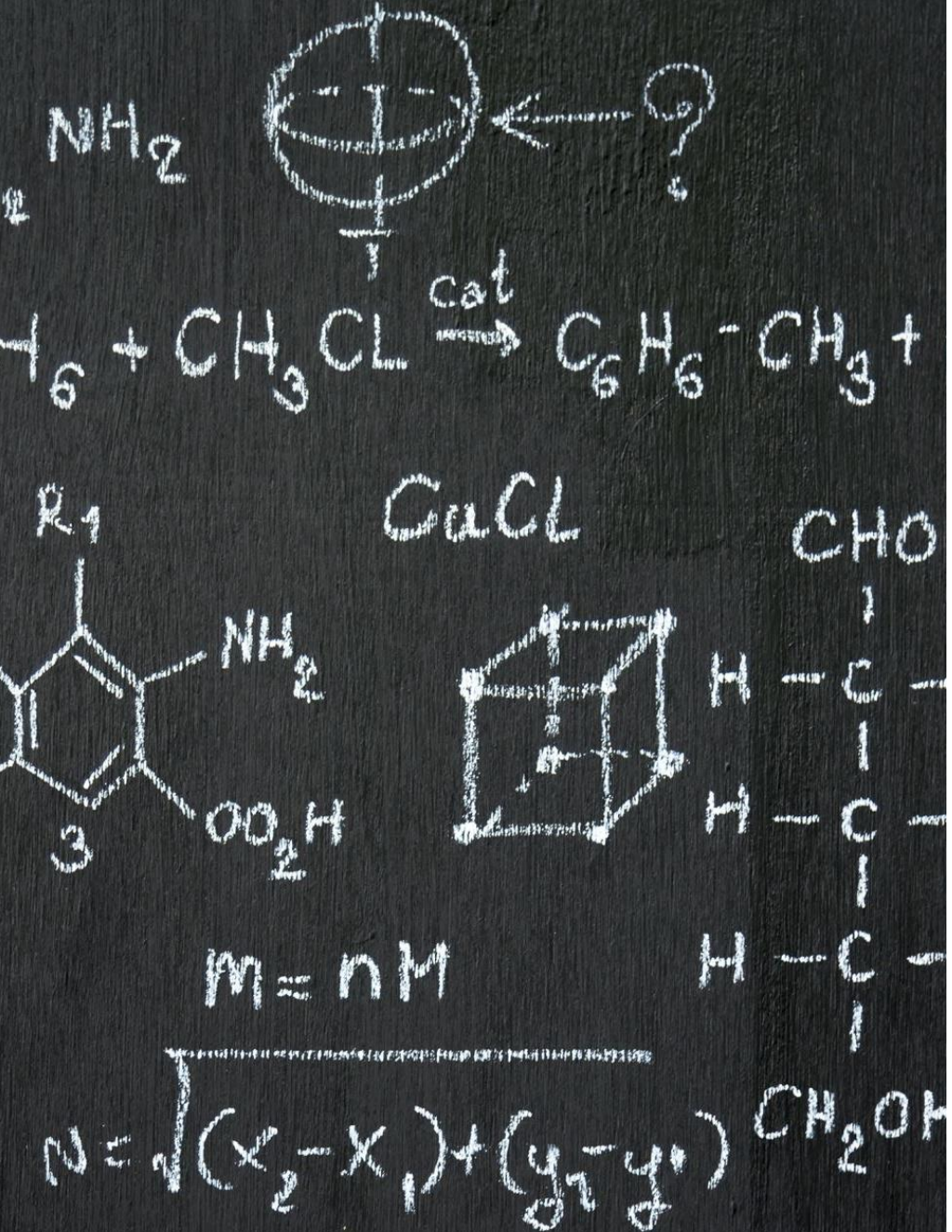
Protocollo e operazioni offerte

- Search
 - La ricerca deve specificare: come tutte le operazioni un bind DN, poi
 - Un base DN :il punto del DIT da cui iniziare la ricerca
 - Uno scope :a partire da base DN, indica quanto estendere la ricerca, con tre valori possibili
 - subdefault, indica l'intero sottoalbero
 - onesolo figli diretti del base DN
 - basesolo il nodo base
 - Eventualmente un filtro :per ricercare in base a contenuto della entry invece che per posizione, usando espressioni logiche in notazione prefissa
 - Sintassi filtri ed esempi:)eventualmente guarda slide
 - tutte le entry nello scope, che hanno come attributo cn la string "Babs Jensen"



Protocollo e operazioni offerte

- Search
 -)operatore "diverso da"
 - * wildcard, qualsiasi stringa
 -))due operandi in and , ognuno tra parentesi: il primo è valore objectClass uguale a person; il secondo è nuovamente una composizione, ma in or , il primo sn uguale a jensen, il secondo cn uguale a babs j* , dove * è qualsiasi stringa
 - Abbiamo quindi possibilità di comporre più espressioni, ponendole come operandi di altre
 -) risultato finale è unica riga con la notazione prefissa
 - comando per ricerca: `ldapsearch -x -b dc=labammsis`
 - -s indica lo scope
- Compare



Protocollo e operazioni offerte

- Add
 - comando per aggiunta: `ldapadd -x -D "cn=admin,dc=labammsis" -w admin`
 - `-f` specifica il file, se omesso usa `stdin`
- Modify
 - comando per modifica: `ldapmodify`
 - Stessi parametri di `ldapadd`, ha più casi: si può usare per aggiunta/modifica/rimozione attributo
- Delete
 - comando per rimozione: `ldapdelete -x -D "cn=admin,dc=labammsis" -w admin DN`
 - Credo che DN sia il il sottoalbero da eliminare ma non ne sono certo
- ModifyRDN