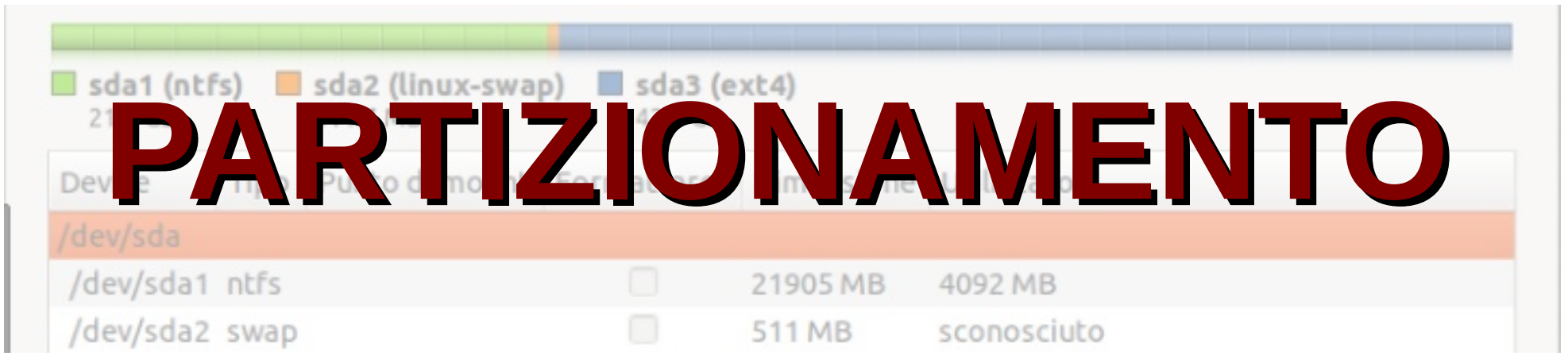
AVVIO DI UN SISTEMA OPERATIVO



PARTIZIONAMENTO

BIOS (Basic Input-Output System)

- All'accensione la cpu, se non vi sono errori hardware, legge le istruzioni da un particolare chip presente sulla MB, detto volgarmente **BIOS**.

- Il **BIOS** è un programma scritto in una memoria permanente (una volta erano **ROM** poi **EPROM**, poi **EEPROM**) e controlla la prima fase del processo di avvio:

  - Effettua un test del sistema,ad esempio un blando memtest, il check delle ventole ecc.
  - cerca e controlla le varie periferiche,
    ed infine cerca una unità di massa da cui avviare il sistema operativo (hd, usb pen, dvd ecc).

  - Da qualche anno il concetto di **BIOS** (detto ora **legacy**) è stato sostituito dal sistema **BIOS UEFI (Unified Extensible Firmware Interface).**

BIOS UEFI

- **UEFI** supera le limitazioni del vecchio **BIOS** (Legacy **MBR**) consentendo, tra l'altro:
  - **Avvio da dischi particolarmente capienti** (di capacità superiore ai **2 Terabytes**) utilizzando **GPT** (**GUID Partition Table**).
    - **GPT** è parte integrante dello standard **UEFI** e porta con sé tutta una serie di novità rispetto all'utilizzo del **MBR** (Master Boot Record).
  - Vantaggi di **UEFI/GPT**
    - **Architettura indipendente** da CPU e driver.
    - **Ambiente preOS** (accessibile cioè fuori del sistema operativo, prima della fase di boot di quest'ultimo) molto più completo e versatile rispetto al passato. Si pensi che di solito **UEFI** offre anche il supporto della connettività di rete.
    - **Design modulare**.
    - Eliminazione della necessità di un bootloader (fatta eccezione per utilizzi più avanzati).
    - Esecuzione di moduli firmati (funzionalità **Secure Boot**).
  - A differenza del sistema precedente (basato su **MBR**), utilizza dei moduli (firmati) per la definzione dei boot record associati all'avvio dei sistemi operativi.

BIOS Legacy, MBR

- Vedremo il sistema di avvio basato su **MBR**, come quello dei vecchi BIOS pre-UEFI, perchè più semplice. **UEFI** è compatibile (lo emula) anche con **MBR.**
- Una volta finiti check e controlli il **BIOS** cerca un **MBR** (**Master Boot Record**) di solito memorizzato nel primo settore del disco, ne carica il contenuto in memoria e gli passa il controllo delle operazioni.
- **MBR**, a cui il **BIOS** aveva ceduto il controllo, cerca la prima **partizione attiva(*)**.
  - Ne legge il record di avvio che contiene le istruzioni su come caricare il **boot loader** per avviare il SO.
  - Successivamente **MBR** carica il **boot loader** che assume il controllo del processo di avvio.

(*)**Partizione**: una porzione del disco. È attiva se tra le sue proprietà vi è il **flag di boot** posto a 1.

BOOT di Windows

- Il compito di *boot loader* veniva svolto dal file **NTLDR.exe** che controllava il processo di selezione del sistema operativo e il rilevamento dell'hardware prima dell'inizializzazione del kernel.
- **NTLDR.exe** può visualizzare un menù da cui si può selezionare il sistema operativo. La videata è basata sulle informazioni che si trovano nel file **boot.ini.**
- **boot.ini** è un semplice file di testo nascosto e a sola lettura.
- Per saperne di più
  http://support.microsoft.com/default.aspx?scid=kb;EN-US;q99743
  http://support.microsoft.com/default.aspx?scid=kb;EN-S;q102873

BOOT di Linux

- **LiLO.**
- **GRUB.**
- **LoadLin.**

  - (**LinuxLOader - LiLO**) era il kernel loader di Linux ed è composto da due parti: una avviabile e una che provvede ad installarlo. La parte che provvede ad installare **Lilo** si appoggia sul file `/etc/lilo.conf`.

  - Al giorno d'oggi decisamente limitato...

GRUB

- **GRand Unified Bootloader:** se avete installato Linux, o nei pc del laboratorio, è quel **menù che vi permette di scegliere se avviare Windows o Linux**.
- Gestisce facilmente il boot di più sistemi operativi anche se da dischi diversi.
- In generale **grub** si configura tramite un file presente in `/boot/grub` di nome `menu.lst`
- Per installarlo è sufficiente usare il comando:

  ```
  # grub-install /dev/sda
  ```

- Ubuntu ha un suo sistema particolare: trovate le opzioni di default nel file

  `/etc/default/grub`. Altre configurazioni le trovate in `/etc/grub.d/`. Qui trovate diversi file, simili a *script* che potete disattivare semplicemente togliendo i permessi di esecuzione (`chmod -x ...`). Per fissare le modifiche:

  ```
  # update-grub
  ```

- Per approfondire:

  `http://wiki.ubuntu-it.org/AmministrazioneSistema/Grub`

Fase di Boot - Kernel

- Dopo aver scelto un’opzione del boot loader, il sistema carica in memoria il **kernel del sistema operativo**.
- Il **kernel** si occupa di controllare che periferiche ci sono nel computer, carica i driver relativi, effettua delle operazioni di check preliminari, riconosce le partizioni e cede il controllo al processo di **init**.
- **init ≠ initrd: init è un processo, initrd è un file.**
- **initrd**: è un piccolo *file system temporaneo* che funge da file system root, viene caricato in memoria nella fase iniziale della procedura di boot. La funzione di **initrd** è quella di fornire al kernel i moduli che potrebbero non essere presenti al suo interno, impedendo a quest'ultimo di riconoscere, e quindi di far funzionare, una determinata periferica.

Processo di Init

- **SysV Init** era il processo standard nel mondo Linux per controllare l'esecuzione del software all'avvio, nei primi Linux funzionava così:
  - Il kernel cercava **init** nella directory `/sbin` e lo avviava cedendogli il controllo della fase di avvio.
  - **init** avviava lo script `/etc/rc.d/rc.sysinit`.
    - **rc.sysinit** gestiva quasi tutti i processi di avvio ed eseguiva `rc.serial`.
  - **init** eseguiva tutti gli script per il runlevel di default.
  - **init** eseguiva`/etc/rc.d/rc.local`.
- Nei so unix moderni **init** poggia sul concetto di **runlevel**.

Runlevel

- **Runlevel** è un particolare configurazione software del sistema che permette l'esistenza di un solo gruppo di processi: è **uno stato del sistema che permette solo determinate azioni**.
- Vi sono **sette** runlevel, numerati da **0** a **6.**
  - Se avete avviato Ubuntu dovreste essere in runlevel 5.. o 3 se non è partita l'interfaccia grafica.

- Da root potete cambiare runlevel con il comando `init` e controllare in che livello siete con il comando `runlevel`:

```
#runlevel
N 5
#init 3 ; passa a runlevel 3
```

Processo di init in dettaglio

- Il **kernel** individua **init** nella directory **/sbin** e lo esegue cedendogli il controllo della fase di avvio.
- Appena eseguito **init** **diventa padre di tutti i processi avviati automaticamente dal sistema**. Tra i processi avviati vi sono l'attivazione dello swap, il check dei dischi ecc ecc.
- **init** legge il file **`/etc/inittab:`** una tabella che descrive la *configurazione* di ogni **runlevel**.
- Successivamente **init** avvia tutti i processi in background necessari al sistema cercando nella directory `/etc/rcX.d` dove `X` è il **runlevel** predefinito di **`inittab`**.
  In particolare termina tutti i processi descritti negli script il cui nome comincia con **K** e avvia quelli il cui nome comincia con **S**.
- Infine **init** si occupa di avviare i processi delle console di sistema (*getty*) ed esegue `/etc/rc.local`.

Esempio di /etc/inittab

```
# inittab       This file describes how the INIT process should

#               set up the system in a certain run-level.

#[...]

# id:3:initdefault: (Default runlevel).

# The runlevels used by RHS are:

#   0 - halt (Do NOT set initdefault to this)

#   1 - Single user mode

#   2 - Multiuser, without NFS (The same as 3, if you do not have networking)

#   3 - Full multiuser mode

#   4 - unused

#   5 - X11

#   6 - reboot (Do NOT set initdefault to this)
```

Fonte http://openskill.info/infobox.php?ID=365

Systemd e Upstart

- In quasi tutte le moderne distribuzioni Linux, **init** è ormai stato sorpassato da altri sistemi di avvio del SO.
- Due di essi sono:
  - **Upstart**: realizzato dal 2006 da Canonical.
  - **Systemd**: che pian piano ha rimpiazzato il concorrente su tutte le distro (Ubuntu compresa).
- I “nuovi” sistemi promettono maggiore flessibilità e semplicità nella gestione dei *demoni* da avviare e un boot più snello e rapido prevedendo, ad esempio l’avvio asincrono di alcuni demoni fra loro indipendenti.

Installazione di un SO

- In seguito ad una buona analisi inziale, da cui estrapoliamo ciò che ci serve e per quale motivo, l'installazione base di un sistema operativo risulta banale:
  - spesso si tratta di avviare l'installatore del sistema da cd o chiavetta,
  - Cliccare su avanti per un certo numero di volte,
  - Riavviare il nuovo sistema.
  - **Applicare aggiornamenti e patch!**
- Particolarmente importante è comunque la fase di **partizionamento dello storage disponibile**, **che deve essere differente a seconda della tipologia di server da installare.**
  - Tale fase dovrebbe apparire tra un click su avanti e l'altro….

Partizionamento

- È la suddivisione di un’unità di storage in parti (**partizioni**).
- **Partizione: una porzione del disco**. Un disco può contenere diverse partizioni a seconda delle limitazioni del sistema operativo che lo utlizza. Ad esempio nei sistemi PC-Intel (**MBR**) il limite è costituito da **4** partizioni **primare**. Una partizione primaria speciale è quella definita **estesa**, che può contenere fino a **16 unità logiche**. In Linux questo limite è superato usando **LVM**.

Partizionamento

- In realtà ora con lo schema di partizionamento **GPT** questi limiti dovrebbero essere superati…..
- A seconda del tipo di installazione che si deve effettuare è bene suddividere il disco in modo diverso… Vediamo alcuni esempi da prendere con le pinze:
  - PC di Casa
  - Server Web
  - Server Mail
  - Server Home
  - Server DB

Partizionamento – Linux

- **PC di casa**: il default di Ubuntu di solito va bene, al limite si possono tenere separate le home (`/home`) da `/` :

  - `/dev/sda1 /`
  - `/dev/sda5 /home`
  - **`/dev/sda6 swap`**

- Ma cos’è questo/a swap?

Swap

- Lo **swap** viene utilizzato per liberare memoria RAM: *il sistema operativo salva sul disco una porzione della memoria allocata, che quindi può essere liberata e riallocata per i programmi che ne hanno bisogno*.
- Questa porzione contiene i dati che hanno minore probabilità di essere richiesti nel futuro, e in genere sono quelli meno recentemente utilizzati.
- Nel momento in cui si rende necessaria tale operazione, le prestazioni del sistema crollano bruscamente, essendo la scrittura su hard disk molto più lenta di quella in RAM.
- Due filosofie per la gestione dello swap, ognuna con pregi e difetti:
  - **Area/partizione di Swap**.
  - **File di swap**.

Area vs File (di swap)

Area di Swap	File di swap
Dimensione fissa: per estenderla è necessario aggiundere altri dischi e/o altre partizioni di swap o agire tramire sistemi di ridimensionamento delle partizioni	**Dimensione variabile**: si può adattare a seconda delle necessità. Può essere gestito automaticamente dal SO.
Nessuna frammentazione: essendo una partizione non soffre di frammentazione	**Rischio di frammentazione**: anche se alcune aree della partizione che ospita il file possono essere riservate, non sempre è salvato in zone contigue
Migliori performance: si riducono gli overhead delle operazioni di accesso al file. I dati sono salvati in zone contigue.	**Performance**: vincolate dalle operazioni di I/O su file e dalla frammentazione.

Area di Swap (Linux)

- **Fino a pochi anni fa, l'area di swap doveva essere grande 1,5 volte (e non uguale al-) la dimensione della ram installata nel pc**. Perchè?
  - Storicamente conteneva l'intero dump della memoria in caso di crash del kernel → al riavvio dopo il crash non c'era più spazio swap disponibile.

- Attualmente, data la crescente quantita di ram presente nei server, l'area di swap è dimensionata in modo diverso, spesso è una percentuale della ram (10%, 20%,...50%).

File di Swap (Windows)

- Windows gestisce lo swap tramite un file (**`pagefile.sys`**), che dovrebbe, in una installazione server, essere posizionato in una partizione diversa da `C:\`.
- I dump di memoria in caso di crash sono salvati in file diversi da `pagefile.sys` e di solito si trovano in `C:\Windows\System32`.
  - E' possibile avere dei minidump di memoria invece di dump completi della memoria configurando adeguatamente le opzioni del sistema operativo.

LVM

- **Logical Volume Manager**: è un software per la gestione dello storage realizzato per renderne più flessibile la gestione rispetto al tradizionale partizionamento fisico.
- Permette di ridimensionare ogni filesystem senza dover riavviare il sistema anche quando il filesystem è in uso (**tranne /**).
- È utilizzato su: file server, database server, NAS per gestire lo storage.

Partizionamento – Linux

- In questi esempi di partizionamento si suppone di avere **1TB di storage** e **64Gb di ram**.
- **ATTENZIONE:** un sistema di macchine virtuali può non rispettare questi schemi di partizionamento!!!
- Usate **LVM** per ingrandire o rimpicciolire partizioni all'occorrenza (tranne root).
- **Server Web**
  - / max 50gb
  - /var almeno 50gb se c'è un piccolo dbms
  - **/var/www spaziopersito*numerodisiti+10%**
    10gb * 50 + (500*0,1) = 550GB
  - **/var/log almeno 10gb**
  - **Swap: 64 * 1,5 = 96GB ≃ 100GB (metodo classico)**

Partizionamento – Linux

- **Server Home/SSH**
  - / almeno 50gb
  - **/home quotautente*numerodiutenti+10%**
    10gb * 50 + (500*0,1) = 550GB
  - /var almeno 50gb
  - **/var/log almeno 10gb**
  - **Swap: 64 * 1,5 = 96GB ≃ 100GB (metodo classico)**
  - **Swap: 64*50% = 32GB ≃ 40GB (altra specifica data dalla quantità di ram, sull'esame ve lo dice il prof)**
  - **Swap: 64*10% = 6,4GB ≃ 7GB (altra specifica data dalla quantità di ram, sull'esame ve lo dice il prof)**

- Cos'è SSH? Lo vediamo dopo...

Partizionamento - Linux

- **Server Database**
  - / almeno 50gb
  - **/var grande (>100gb), può ospitare uno o più DBMS (mysql, postgre)**

    quotaDB * numerodiDB +10%

    10gb * 50 + (500*0,1) = 550GB
  - **/var/log almeno 10gb**
  - **Swap: 64 * 1,5 = 96GB ≃ 100GB (metodo classico)**
  - **Swap: 64 * 75% = 48GB ≃ 50GB (altra specifica data dalla quantità di ram, sull'esame ve lo dice il prof)**

Partizionamento - Linux

- **Server Mail**
  - / almeno 50gb
  - **/mailstore quotamailperutente*numerodiutenti+10%**
    10gb * 50 + (500*0,1) = 550GB
  - /var almeno 50gb
  - **/var/log almeno 10gb**
  - **Swap: 64 * 1,5 = 96GB ≃ 100GB (metodo classico)**
  - **Swap: 64 * 0,25 = 16GB ≃ 20GB (altra specifica data dalla quantità di ram, sull'esame ve lo dice il prof)**

Partizionamento - Esercizio per l’esame

Supponendo di disporre di uno storage da **10 TB** e **512 gb** di RAM, come partizionereste il sistema Linux con autenticazione centralizzata ed **export delle Home**? Per comodità si immagini di dover gestire **50** utenti, si stima che ogni utente possa occupare al massimo **100gb** di spazio. Considerare **swap=0,25 ram**, **swap=0,5 ram** , **swap=0,75 ram**. Calcolare la swap anche con il metodo classico. Quanti utenti può ospitare il sistema?

Soluzioni

È richiesto di partizionare lo storage di un server per la gestione centralizzata degli utenti che fornisce, oltre all'autenticazione anche lo spazio per le home.

Dovendo gestire **50** utenti, dato uno spazio complessivo di **10Tb** e considerando **512gb** di ram, imponendo come spazio circa **100gb** a utente abbiamo diverse soluzioni:

Soluzione A

swap 512GB*1,5 = 768GB circa 1TB

swap 512GB*0,5 = 256GB circa 300GB

/home 50*100GB+10%= 5,5 TB

/ 500GB

/var 1 TB

/var/log 500GB

/usr 1,5TB

Soluzione B

swap 512GB*1,5 = 768GB circa 1TB

swap 512GB*0,75 = 384GB circa 400GB

/home 50*100GB+10%= 5,5 TB

/ 500GB

/var 1 TB

/var/log 500GB

/extra 1,5TB

Soluzione C

swap 512GB*1,5 = 768GB circa 1TB

swap 512GB*0,25 = 128GB circa 130GB

/home 50*100GB+10%= 5,5 TB

/ 500GB

/var 1 TB

/usr/src 500GB

/usr 1,5TB

Quanti utenti può ospitare il sistema?

- Calcolando 100Gb ad utente e considerando la swap = 0,5 ram abbiamo:
  - NumUtenti = (10TB-256GB) / 100GB circa 100 utenti

Considerazione….

Dopo aver calcolato lo spazio necessario per i servizi richiesti restano diverse soluzioni tutte valide.

Aggiungere +10% può sembrare solo una paranoica sicurezza da Sys Admin… ma è bene farlo nella realtà!

Protocollo SSH in breve

Fonte: https://www.attachmate.com/it-it/documentation/reflection-desktop-v16/rdesktop-guide/data/secure_shell_r200x_connections_ch.htm

SSH

- **SSH (Secure SHell) è un protocollo che permette di stabilire una sessione remota cifrata tramite interfaccia a riga di comando con un altro host**.
- Una volta si usava telnet, connessione in chiaro.
- Come si usa:

```
# ssh [opzioni] nomeutente@host [comando]
# ssh faromano@squit.dsi.unive.it
# ssh faromano@lab050101
# ssh faromano@157.138.22.21
# ssh las@<ip vostra vm>
```

SSH

- Con il **protocollo ssh** è possibile, inoltre spedire/ricevere file o directory a/da un sistema remoto.
- Cio è possibile tramite il comando `scp`:

```
# scp [options] [user@host:]file1 [user@host:]file
# scp pippo faromano@squit.dsi.unive.it:.
# scp -R pippo/ faromano@lab050101:.
# scp faromano@157.138.22.21:/home/pippo pluto
# scp pippo las@<ip vostra vm>
```

- Esiste anche un programma, `sftp`, che permette di collegarsi ad un server ssh e di inviare/trasferire file e directory tramite comandi simili a quelli del protocollo ftp (put, get...)

SSH - sftp

- Per usare sftp:

```
# sftp user@host
# sftp faromano@squit.dsi.unive.it
# sftp las@<ip vostra vm>
```

- Per Windows/Mac esistono dei client grafici che permettono di utilizzare il trasfermento dati via ssh in modo semplice.
  - Per mac OS: Cyberduck, Fugu, Filezilla.
  - Per Windows: Winscp, Filezilla.
- Anche per Linux vi sono alcuni client grafici, ad esempio per l'interfaccia di Ubuntu, il client grafico ssh è già integrato nel File Manager.
  - Anche per Linux vi è un'implementazione di FileZilla.

SSH – Autenticazione con scambio di chiave

- Quando ci si deve connettere molto spesso ad un server (o a molti server) tramite SSH, può essere tedioso dover inserire ogni volta la password.
- Un modo sicuro per aggirare questo problema è basato sull'**autenticazione tramite una coppia di chiavi (privata e pubblica**).
- Il concetto alla base di questo sistema di autenticazione è semplice: si demanda il compito di verificare i dati di autenticazione direttamente alle **chiavi ssh**, rimuovendo la richiesta della password (meno volte viene digitata, più è difficile che qualche utente male intenzionato la riesca a capire) che viene, eventualmente, sostituita dalla richiesta di una **passphrase** di sblocco della chiave.

SSH – Autenticazione con scambio di chiave

- Generazione delle chiavi:

```
# ssh-keygen -A

# ssh-keygen -t rsa -b 2048
```

- Durante la generazione delle chiavi ci viene chiesto dove salvarle (normalmente è `~/.ssh/id_rsa`): il valore di default va bene.
- Per quanto riguarda la passphrase richiesta, sempre durante la generazione delle chiavi, ci sono due opzioni, entrambe con pregi e difetti:
  - **inserire una passphrase**: dal punto di vista della sicurezza, è ottimo; dal punto di vista pratico, però, si è di fronte al problema che è necessario inserirla ad ogni connessione (nel caso di più host, comunque, rappresenterebbe un sistema molto comodo di accesso tramite la stessa passphrase, invece di una password diversa per ogni host)
  - **inserire una passphrase vuota**: dal punto di vista della sicurezza lascia un po' a desiderare, in quanto il furto della chiave permetterebbe l'accesso incondizionato agli host; dal punto di vista pratico, invece, è comodissimo, in quanto slega l'accesso alla macchina remota dalla richiesta di password.

SSH – Autenticazione con scambio di chiave

- In realtà questo discorso può valere in caso di utilizzo non interattivo come ad esempio in uno script, negli altri casi si può ricorrere a **ssh-agent** per mantenere in cache la passphrase per la sessione corrente:

```
# ssh-add ~/.ssh/id_rsa
```

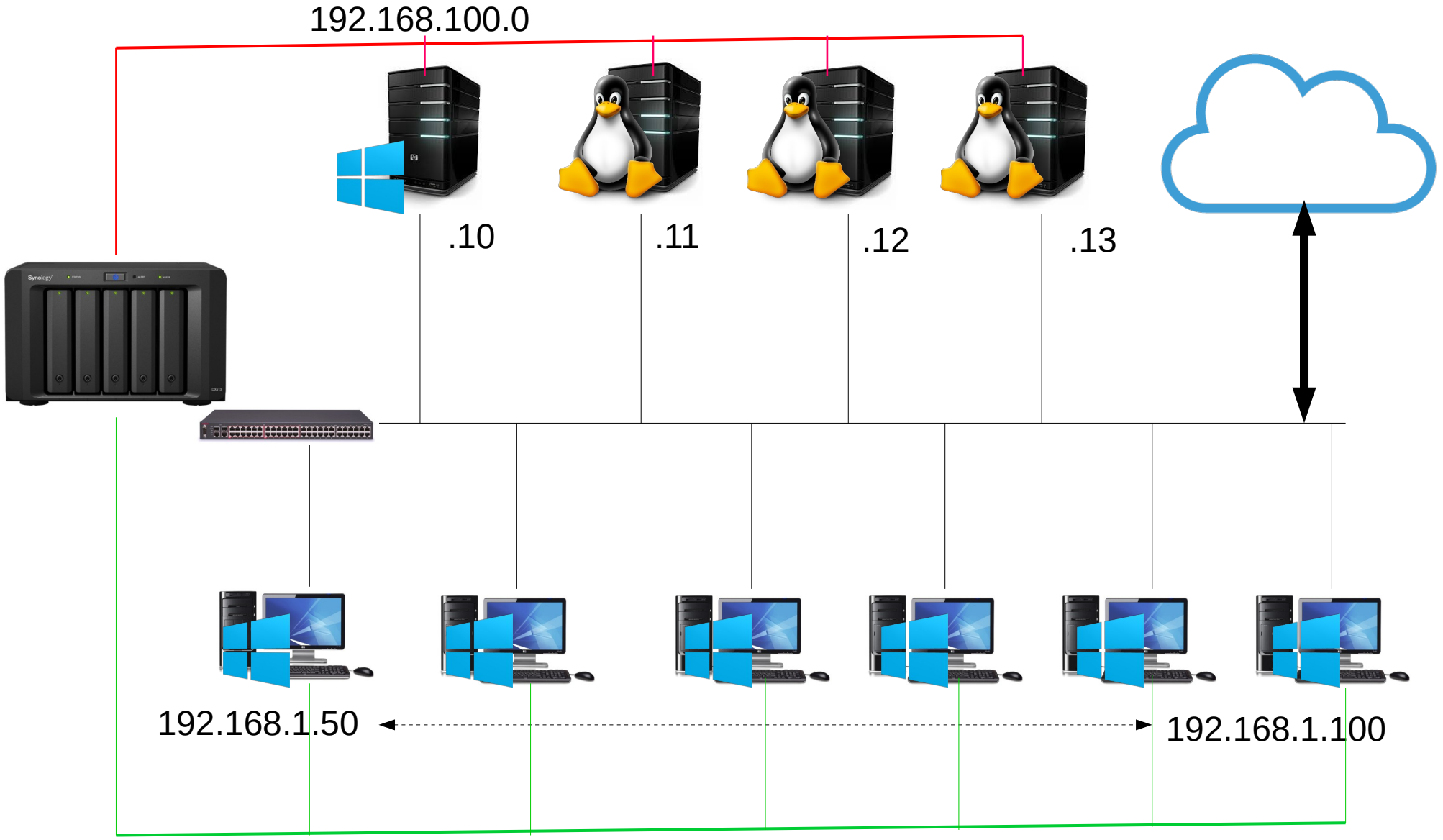
- Copia manuale della chiave pubblica: significa copiare a mano, usando copia & incolla, la chiave pubblica contenuta nel file `~/.ssh/id_rsa.pub` presente sul client nel file `~/.ssh/authorized_keys` presente sul server, nella home relativa all'utente usato su quella macchina.
- **Copia automatica della chiave pubblica**: è possibile usare lo script ssh-copy-id in questo modo dal client:

```
# ssh-copy-id -i ~/.ssh/id_rsa.pub utente@server
```

SSH – equivalente windows di ssh-copy-id

- https://serverfault.com/questions/224810/is-there-an-equivalent-to-ssh-copy-id-for-windows

Avete installato i vs SO preferiti...



Installiamo qualcosa?