UNIVERSITÀ DEGLI STUDI DI PERUGIA

DIPARTIMENTO DI MATEMATICA E INFORMATICA

Prima esercitazione:

CALCOLO DISTRIBUITO E PARALLELO

RELALIZZAZIONE DI UN SISTEMA DI CALCOLO DISTRIBUITO MEDIANTE IL SOFTWARE HTCONDOR

Studente: Manuel Severi

INDICE

1 | INTRODUZIONE

Con la seguente relazione, andremo ad analizzare l'obbiettivo del nostro progetto, ponendo l'attenzione anche sulle differenze tra l calcolo parallelo e calcolo distribuito e sui software utilizzati

1.1 OBBIETTIVO

L'obbiettivo del seguente progetto è di realizzare un sistema di calcolo tramite l'utilizzo di macchine virtuali, configurate appositamente per lo scopo, e di utilizzare il framework di software HTCondor che ci permette di lavorare in parallelo sulle varie macchine, nel nostro progetto configureremo solo due macchine.

1.2 DEFINIZIONE E DIFFERENZE TRA CALCOLO PARALLELO E DISTRIBUITO

- **Calcolo parallelo**: con il seguente termine si indica l'utilizzo di più CPU all'interno della stessa macchina, quindi utilizzare più unità computazionali, per risolvere un determinato carico di lavoro. Il lavoro (o Job), viene eseguito in maniera concorrente suddividendolo in più sotto lavori. Ogni istruzione presente nel lavoro da svolgere viene eseguita simultaneamente su varie CPU differenti.
- **Calcolo distribuito**: Per calcolo distribuito si intende, un sistema composto da vari calcolatori collegati tra loro attraverso la rete ed impiegati per risolvere obbiettivi comuni. Il lavoro viene suddiviso tra i vari device ed ognuno di esso esegue parte del lavoro. I software che permettono di fare ciò vengono definiti "Software Distribuiti" e la

programmazione volta a sfruttare tale potenzialità prende il nome di “programmazione distribuita”.

- **Differenze tra i due approcci**: Nel calcolo parallelo abbiamo un'unica memoria condivisa utilizzabile da tutti i processori che vogliano occuparsi del lavoro da svolgere. Mentre nel calcolo distribuito, tutti i processori hanno memoria distribuita, una sorta di memoria privata per ogni precessore, e comunicano tra loro grazie al passaggio di messaggi

2 | SOFTWARE UTILIZZATI

2.1 HTCONDOR

HTCondor è un framework di software di elaborazione ad alta velocità open source per la parallelizzazione distribuita a elevata intensità computazionale. Può essere utilizzato per gestire il carico di lavoro su un cluster dedicato di computer o per estrarre il lavoro per i computer desktop inattivi, il cosiddetto ciclo di scavenging. HTCondor funziona su sistemi operativi Linux, Unix, Mac OS X, FreeBSD e Microsoft Windows. HTCondor può integrare sia risorse dedicate (cluster montati su rack) sia macchine desktop non dedicate in un ambiente informatico. Il funzionamento è semplice, gli utenti inviano i loro lavori, che volgiono che siano svolti in parallelo, al software, che li inserisce in una cosda e sceglie, in base alle sue politiche di gestione delle risorse, quando e chi deve eseguire i lavori monitorando adeguatamente l'esecuzione e informando l'utente del termine. Quindi perché utilizzare HTCondor?

HTCondor è utile se dobbiamo elaborare una grossa mole di dati e lavori che si vogliono svolgere in tempo breve. Inoltre, Condor ha la possibilità di monitorare i lavori in corso, e riesce ad inviare un avviso via mail (o altri servizzi di comunicazione) quando si completa un lavoro.

HTCondor ha diversi demoni che lavorano in background come:

- **Master** utilizzato per amministrare il sistema. Gestisce il resto dei demoni di condor e controlla periodicamente i timestamp degli altri demoni così da assicurarsi il corretto funzionamento di tutte le macchine del Pool di calcolo. In caso di malfunzionamenti anomali invia dei report all'amministratore della macchina che ha registrato il problema.
- **Stard** rappresenta una macchina nel Pool Condor. Contiene la descrizione dell'elaboratore e l'esecuzione di questo demone consente alla macchina di eseguire lavori. Questo demone è anche responsabile della politica della vita di un lavoro remoto e quando il demone è pronto genera il lavoro su un computer remoto.
- **Schedd** rappresenta i valori nella pool di Condor. Tutti i valori dei lavori vengono inviati a questo demone che si occupa della memorizzazione. È possibile visualizzare e manipolare la coda dei valori con i comandi: *condor-submit, condor-q, condor-rm.*
- **Collector** responsabile della raccolta di tutte le informazioni sullo stato di un pool di Condor. Tutti gli altri demoni inviano degli aggiornamenti periodici definiti *ClassAd* a questo demone. Questo demone, per capire il suo funzionamento, può essere paragonato ad un database dinamico che viene interrogato per avere, come risposta, informazioni specifiche sulle varie parti di Condor. Collector viene eseguito sulla macchina designata come principale.

2.2 VIRTUAL BOX

Oracle VM VirtualBox, è un software open source che consente di emulare sul proprio computer varie macchine virtuali. è programmato per supportare architetture x86 e x64 bit. Supporta l'istallazione e l'esecuzione della maggior parte dei sistemi operativi come quelli che fanno parte delle famiglie di Windows, Linux e macOS. Inoltre supporta la virtualizzazione di hardware Intel e AMD.

Tra l'hardware supportato da VirtualBox troviamo

- **HDD**: gli hard disk vengono emulati con uno speciale formato chiamato *Virtual Disk Image,* dall'estensione VDI che attualmente è soggetto al grave problema della mancata compatibilità con altre soluzioni di virtualizzazione.
- **Scheda grafica**: come scheda grafica, per impostazione predefinita fornisce una periferica *VESA con 12 MB di RAM* configurabili. È possibile installare un driver speciale fornito dalla Guest Additions che fornisce maggiori performance.
- **Scheda di rete**: come schede di rete Ethernet, il software ne fornisce molteplici suddivisibili in:
  - NAT
  - Bridge
  - Interna
  - Solo host
  - Driver generico
- **USB:** installando il pacchetto proprietario di estensioni per VirtualBox, viene emulato un controller USB, così che

qualunque periferica viene collegata al sistema possa essere gestita.

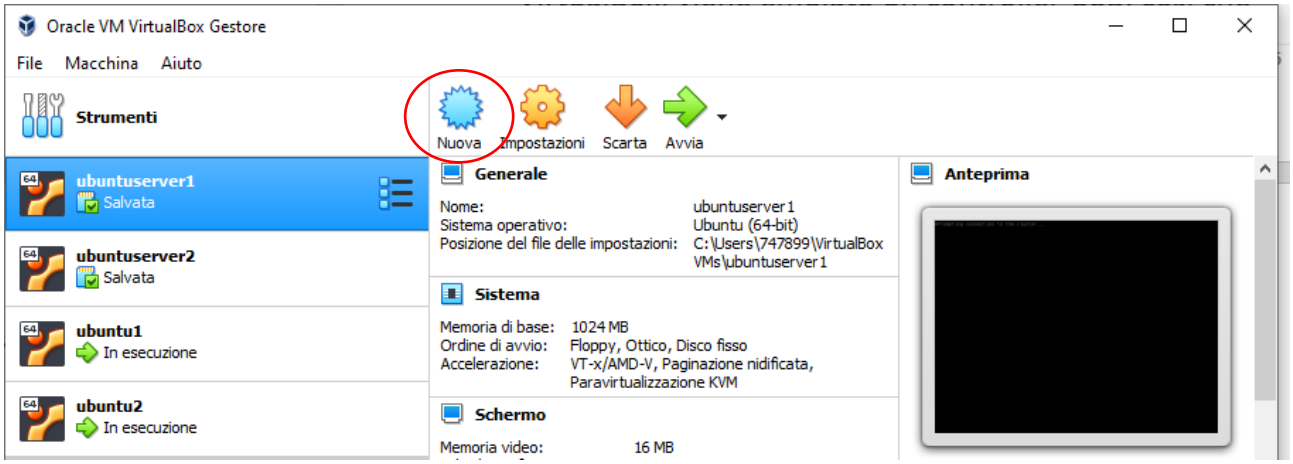
2.3 LUBUNTU

Lubuntu è una distribuzione derivata da Ubuntu. Fino alla versione 18.04 ha utilizzato LXDE come desktop environment. A partire dalla versione 18.10 usa LXQt come ambiente predefinito. È software libero pubblicato sotto i termini della GNU General Public License. Questa distribuzione utilizza come window manager Openbox e risulta molto leggera ed adatta anche ai PC più datati, netbook e dispositivi mobili. Una prima recensione sosteneva che Lubuntu usasse la metà della RAM utilizzata da Xubuntu e da Ubuntu su una normale installazione o nell'utilizzo tipico. Il seguente sistema operativo è stato scelto perché molto leggero, permette a computer non potentissimi come quello su cui abbiamo svolto la relazione, di gestire due macchine virtuali contemporaneamente senza presentare particolari problemi.

È stata usata l'ultima versione disponibile del sistema operativo, con al versione del kernel numero 4.15.0-70-generic facilmente reperibile sfruttando il comando da terminale *uname*

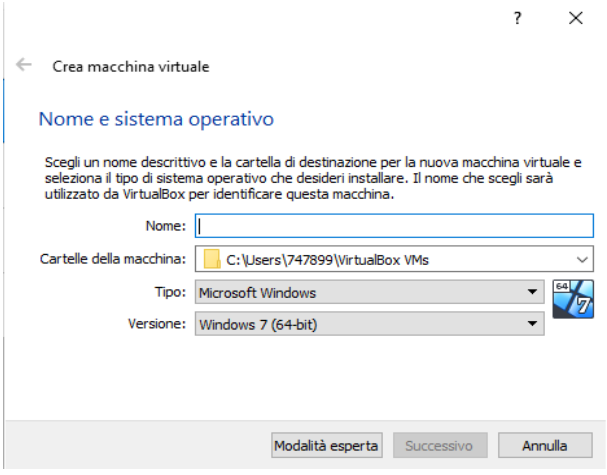
3 | INSTALLAZIONE MACCHINE VIRTUALI

Per la creazione della nostra macchina virtuale abbiamo utilizzato, come descritto in precedenza, il software VirtualBox.
Il primo passo è quello di avviare il software e cliccare sulla stella blu per creare una nuova macchina ed entrare nel menù di setup della macchina

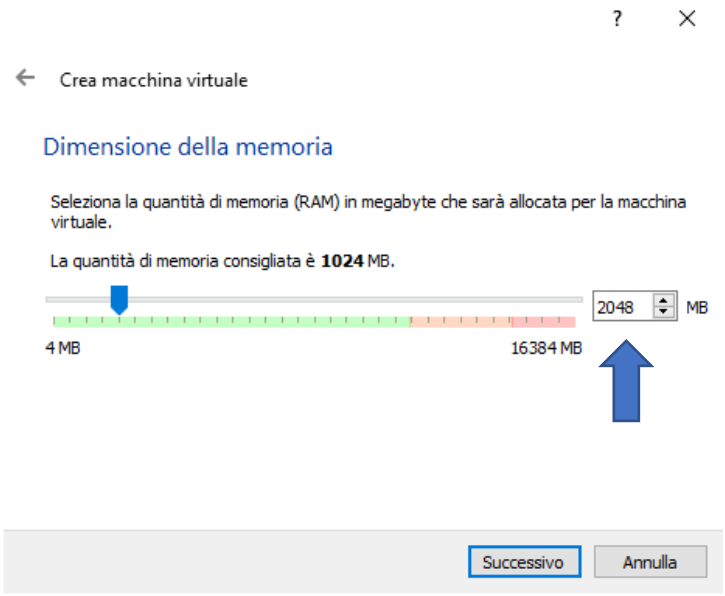


Una volta cliccato nell'area selezionata, della schermata precedente, andiamo a seguire la procedura guidata seguendo i prossimi passi:

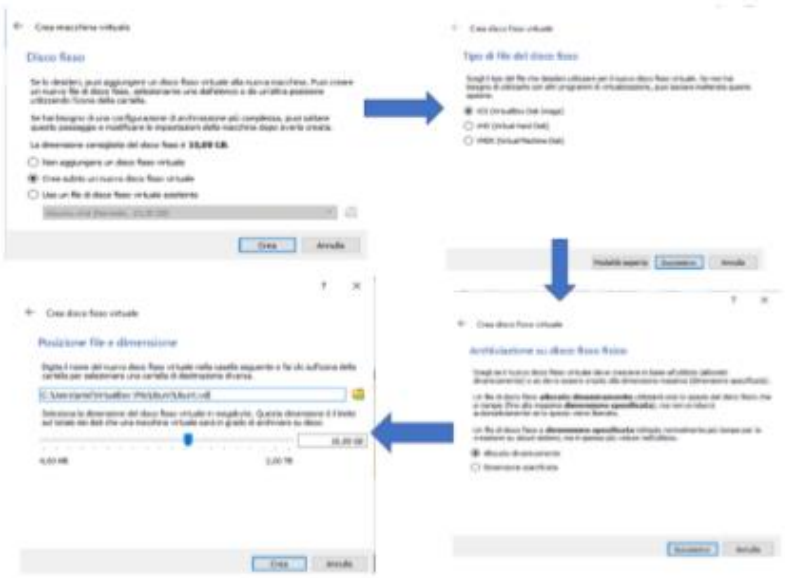
1. Assegniamo un percorso di salvataggio della nostra virtualmachine e diamo il nome alla macchina



2. Assegniamo la RAM utilizzabile della macchina virtuale, in questo progetto raddoppiamo quella consigliata di default andando a scrivere 2048 nella finestra indicata dalla freccia blu.



3. Ora continuiamo con tutte le catatteristiche del disco fisso lasciando le opzioni consigliate da VirtualBox. Le schermate che vedremo sono come quelle sottostanti.



4 CONFIGURAZIONE SCHEDE DI RETE

Dopo aver creato le due macchine virtuali richieste, e prima di avviarle per la prima vota andiamo a configurare le schede di rete. Le macchine avranno una configurazione in parte diversa. La macchina principale, quella denominata Ubuntu1 che svolgerà il ruolo di Manager, avrà bisogno di tre schede di rete:

1. Una scheda di tipo **NAT** per potersi collegare al web e scaricare i pacchetti necessari e se c'è bisogno effetuare aggiornamenti vari. Questa impostazione vale anche per l'altra macchina
2. Una scheda in modalità "**scheda con Bridge**" che permetterà, la connessione di rete della macchina
3. Una scheda verrà configurata come "**scheda solo Host**" la quale permetterà, la comunicazione interna fra i vari host del sistema distribuito. Anche questa scheda va attivata nella medesima configurazione, nella macchina Ubuntu2

Per configurare le schede occorre entrare nella sezione "impostazioni" della macchina e andare nella scheda "rete" e li selezionare il numero e quale scheda rete usare. Per continuare a seguire la guida con facilità è consigliato assegnare le schede nell'ordine in cui precedentemente sono state menzionate.

5 INSTALLAZIONE LUBUNTU E CONFIGURAZIONE HTCONDOR

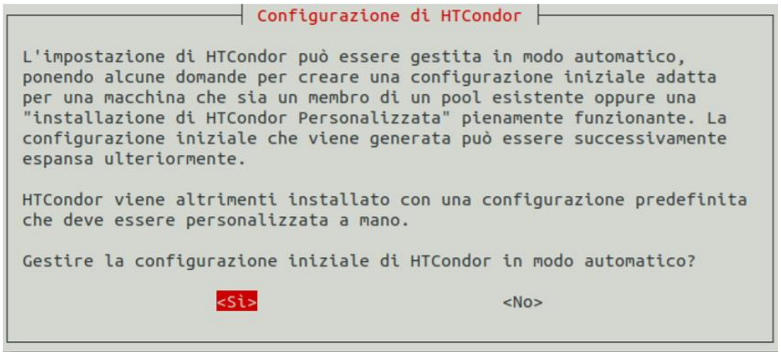
Dopo aver configurato correttamente le macchine virtuali, passiamo all'istallazione del sistema operativo LUbuntu. Scarichiamo la ISO del sistema operativo (comodamente reperibile dal sito ufficiale di Ubuntu). Una volta scaricata procediamo con l'istallazione del sistema operativo. La prima scelta da effettuare è quella della lingua (se si sceglie "italiano" l'istallazione richiede una connessione internet per scaricare il package opportuno). Ora si seleziona "istallazione lubuntu" e si continua seguendo la procedura guidata. Una volta concluse tutte le operazioni è consigliato effettuare un reboot (della macchina oltre a quello che svolge la stessa macchina virtuale) per controllare se la macchina avvia correttamente il sistema operativo al riavvio della macchina. Il processo va ripetuto anche per l'altra macchina virtuale.

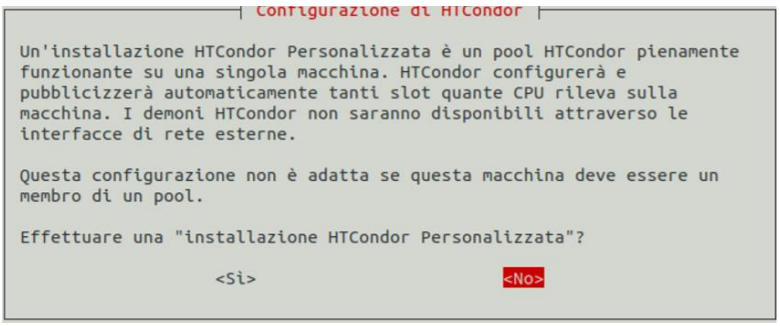
5.1 INSTALLAZIONE E CONFIGURAZIONE DI CONDOR

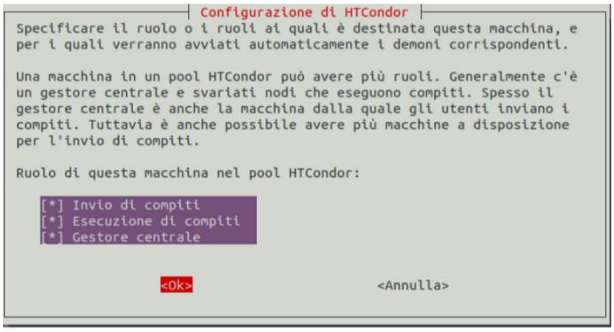
Passiamo ora all'istallazione del software HTCondor su entrambe le macchine e alla sua configurazione su entrambe le macchine. Per fare ciò entriamo nel terminale e scriviamo il comando "*sudo apt-get install htcondor*". Ora vediamo l'istallazione del MASTER, che è la macchina numero 1.

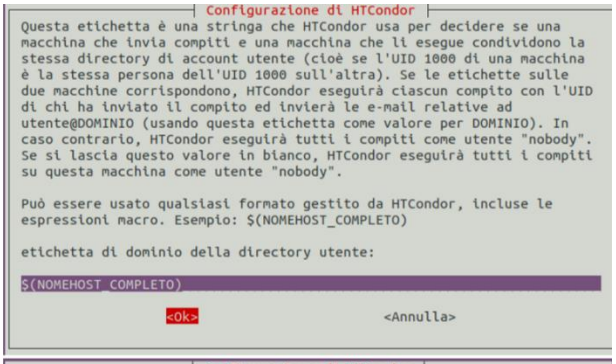
- Istallazione della macchina che fungerà da Master
  Seguiamo i passi che porteranno all'istallazione di HTCondor sulla prima macchina virtuale. Esempio di configurazione è

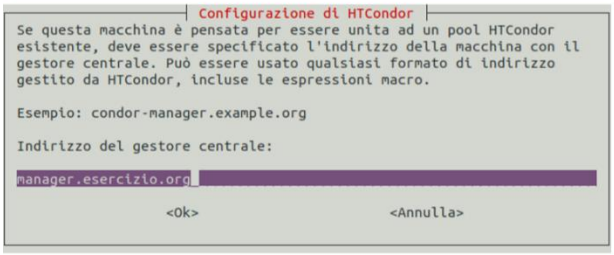
presente nelle prossime immagini.

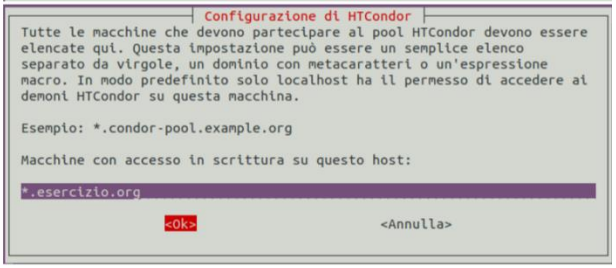








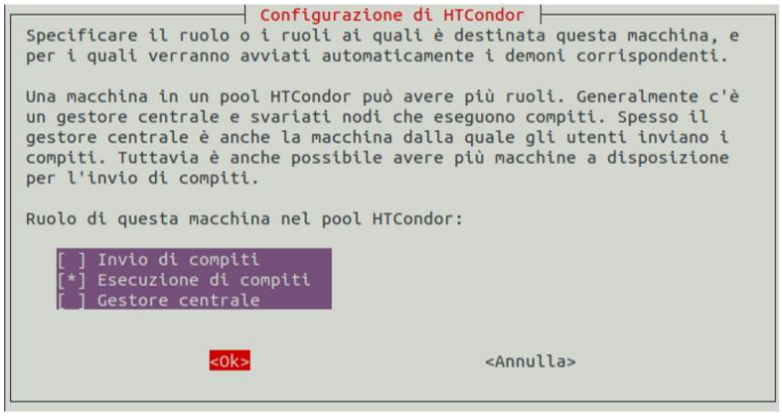




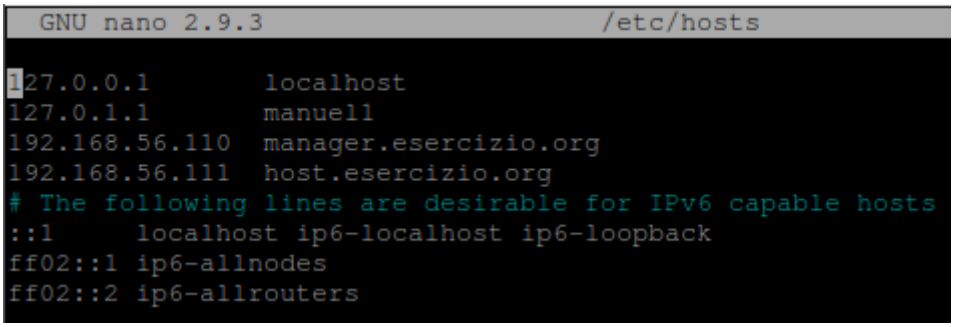
In queste immagini particolare importanza l'assume la schermata numero 3 dove, per il Master, bisogna spuntare, con il tasto space tutte le caselle

- Installazione sulla macchina che verrà utilizzata come Host che cambia solo nel terzo passaggio dove andremo a spuntare solo "*esecuzione di compiti*" come evincibile

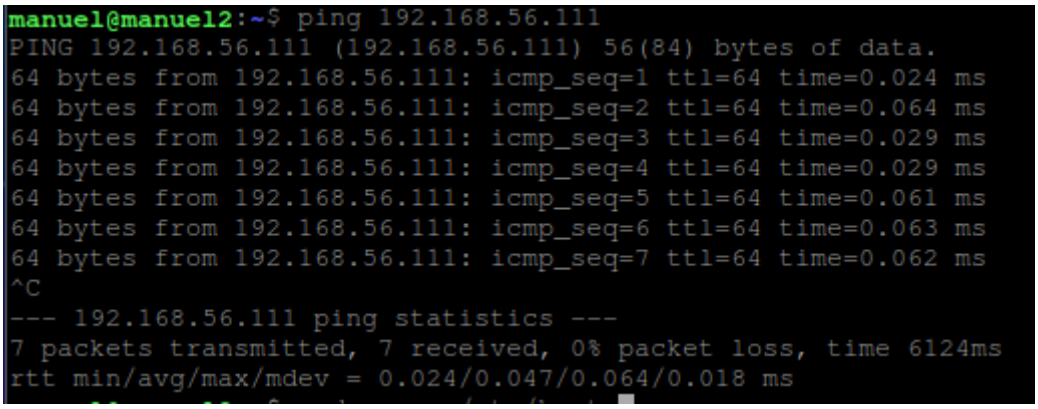
dall’immagine sottostante



Procediamo configurando Condor, una vota che abbiamo finito di installarlo sulle due macchine, settiamo gli indirizzi ip modificando il file hosts (nella directory /etc) tramite il comando “*sudo nano /etc/hosts/hosts*” come mostrato nella schermata sottostante



Ora per controllare che le due macchine riescano a comunicare eseguiamo il ping su entrambe. Nella schermata successiva mostriamo solo il risultato del comando tra la macchina numero 2 verso la numero 1 .



Se entrambi i ping non generano errori passiamo alla configurazione file.

5.2 CONFIGURAZIONE FILE MASTER E SLAVE

Andremo ad apportare delle modifiche al file *condor_config.local* di entrambe le macchine. Tramite i comandi "*sudo /usr/sbin/condor_master*" e "*sudo nano /etc/condor/condor_config.local*" entremo nel file di configurazione delle impostazioni del file di Condor. Aggiungeremo i seguenti parametri per fornire i permessi di scrittura, lettura e autenticazione dei vari Host indicando anche il nome del manager. Di seguito le due configurazioni

- **MANAGER:**
  ALLOW_WRITE = $(CONDOR_HOST) $(IP_ADDRESS) 127.* 192.168.* *.esercizio.org ALLOW_READ = $(CONDOR_HOST) $(IP_ADDRESS) 127.* 192.168.* *.esercizio.org SEC_READ_AUTHENTICATION_METHODS = $(SEC_DEFAULT_AUTHENTICATION_METHODS), ANONYMOUS SEC_CLIENT_AUTHENTICATION_METHODS = $(SEC_DEFAULT_AUTHENTICATION_METHODS),

ANONYMOUS SEC_DEFAULT_AUTHENTICATION_METHODS = FS, PASSWORD ALLOW_ADMINISTRATOR = * ALLOW_DEAMON = * ALLOW_ADVERTISE_MASTER = * ALLOW_NEGOTIATOR = * CONDOR_HOST = manager.esercizio.org BIND_ALL_INTERFACES = True

- **SLAVE:**
  ALLOW_WRITE = $(CONDOR_HOST) $(IP_ADDRESS) 127.* 192.168.* *.esercizio.org ALLOW_READ = $(CONDOR_HOST) $(IP_ADDRESS) 127.* 192.168.* *.esercizio.org SEC_READ_AUTHENTICATION_METHODS = $(SEC_DEFAULT_AUTHENTICATION_METHODS), ANONYMOUS SEC_CLIENT_AUTHENTICATION_METHODS = $(SEC_DEFAULT_AUTHENTICATION_METHODS), ANONYMOUS SEC_DEFAULT_AUTHENTICATION_METHODS = FS, PASSWORD DAEMON_LIST = MASTER, STARTD ALLOW_ADMINISTRATOR = * ALLOW_DEAMON = * ALLOW_ADVERTISE_MASTER = * ALLOW_NEGOTIATOR = * CONDOR_HOST = manager.esercizio.org BIND_ALL_INTERFACES = True

6 | CREAZIONE E INVIO DI UN JOB

Dove aver configurato correttamente HTCondor passiamo alla creazione di un lavoro e al suo invio alla seconda macchina per svolgerlo.

6.1 CREAZIONE DI UN JOB

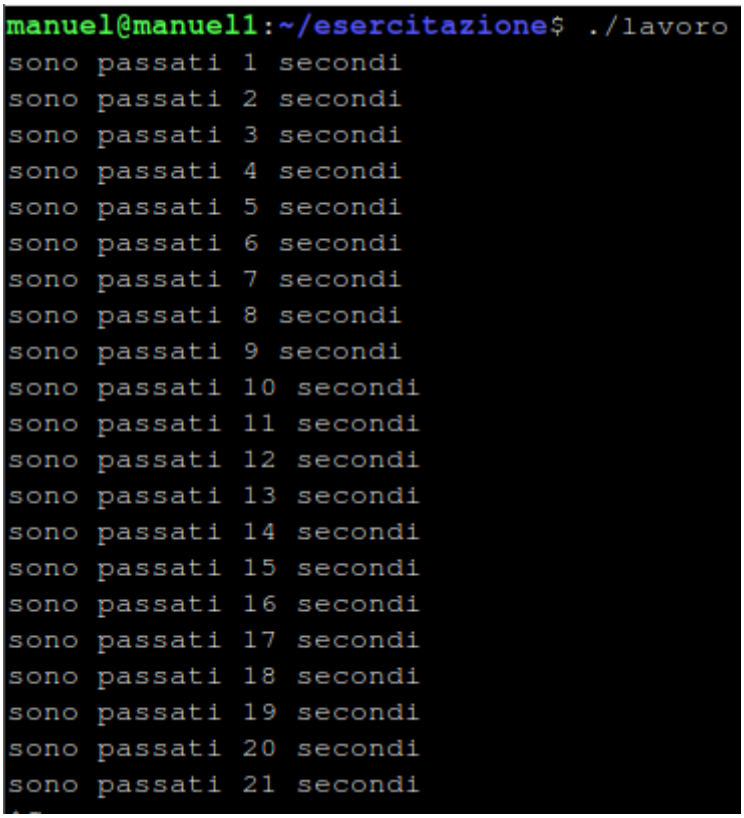
Iniziamo creando una cartella tramite il comando "*sudo mkdir esercitazione*" successivamente, entriamo nella stessa cartella tramite "*cd esercitazione*" e creiamo un file in linguaggio C con "*sudo nano lavoro.c*" dove andiamo a scrivere un semplice lavoro da far svolger alle due macchine. Il codice di esempio è sotto riportato

```c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(){
        int i = 0;
        for(int i=0;1<100;i++){
                sleep(1);
                printf("sono passati %d secondi \n",i+1);
                fflush(stdin);
        }
}
```

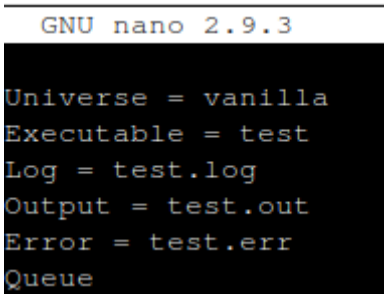
Testiamo il codice creando prima l'eseguibile, controllando eventuali errori nella stesura del codice, tramite il comando "*sudo gcc -o lavoro lavoro.c*" (gcc potrebbe non essere installato in quel caso basta installare i pacchetti tramite il solito comando). Successivamente andiamo ad eseguire il nostro mini job. Il codice

genera delle stringhe a video, e nella schermata sottostante vediamo alcuni secondi del nostro file lavoro.



6.2 INVIO DEL JOB

Eseguiamo il programma tramite la Worker di HTCondor.
All'interno della cartella, nella quale siamo posizionati, ovvero esercitazione, andriamo a creare un file dal nome submit tramite il comando *"sudo nano submit"*. All'interno del file appena creato andremo ad inserire i seguenti parametri:



Analizziamo ora i parametri appena settati.

- UNIVERSE:
  esistono svariati tipi di universi la nostra scelta è caduta su *vanilla* perché è l'universo basilare ed accetta qualsiasi tipo di job. Nei universe sono importanti i *CHEKPOINT* perché se per qualsiasi motivo l'utente commette degli errori non perderà tutto il lavoro ma solo quello eseguito dopo l'ultimo checkpoint raggiunto.
- EXECUTABILE:
  Determiniamo quale file sarà sottomesso al lavoro.
- LOG:
  indica il file sul quale verranno registrate le operazioni svolte durante il lavoro
- OUTPUT
  Il file dove viene inserito l'output del programma che abbiamo sottomesso
- ERROR:
  vengono inseriti, al suo interno, tutti gli errori che si sono verificati durante l'esecuzione di un lavoro. La speranza di ogni programmatore ed utilizzatore che questo file rimanga sempre vuoto
- QUEUE:
  indica il job che si intende mettere in coda.

Dopo aver settato tutti i parametri, creiamo dei file vuoti (con lo stesso nome ed estensione precedentemente indicati) all'interno della cartella esercitazione con i comandi:

- *Sudo nano test.log*
- *Sudo nano test.out*

- *Sudo nano test.err*

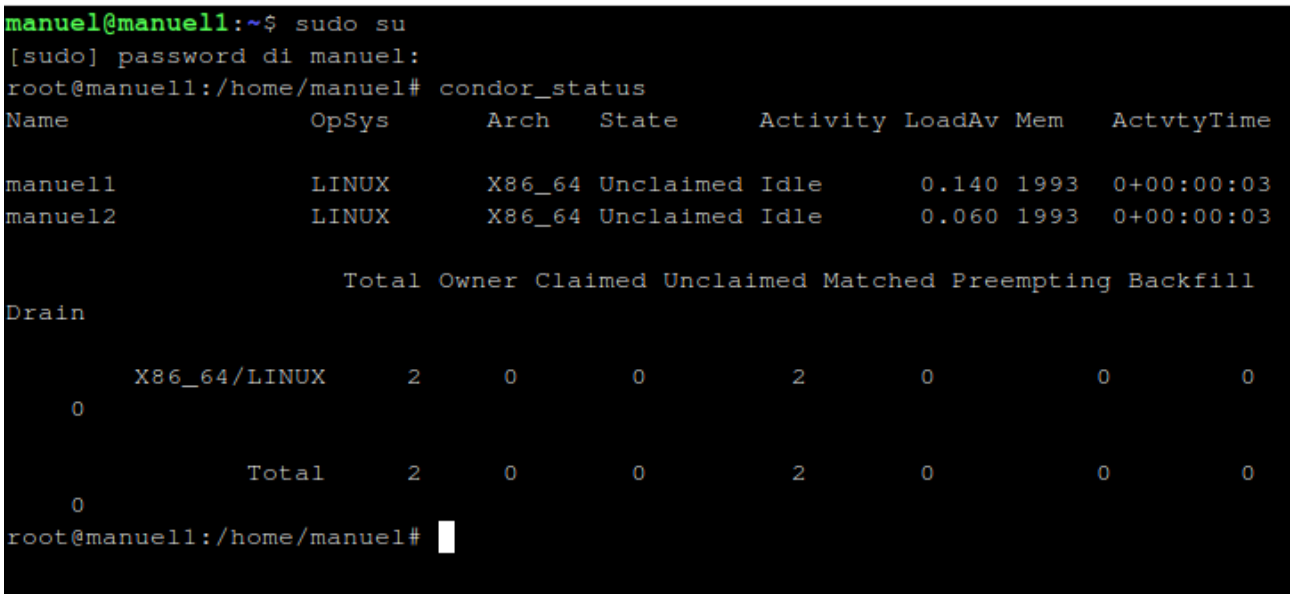
Una volta creati questi file si vanno a modificare i permessi di lettura e scrittura tramite i comandi

- *Sudo chmod 777 test.log*
- *Sudo chmod 777 test.out*
- *Sudo chmod 777 test.err*

chmod (abbreviazione dalla lingua inglese di change mode, cambia modalità) è un comando dei sistemi operativi Unix e Unix-like, e più in generale dei sistemi POSIX, che modifica i permessi di file e directory. Chmod è anche il nome di una chiamata di sistema, definita dallo standard POSIX, che modifica i permessi di un file o directory. Di fatto il comando chmod opera invocando l'omonima chiamata di sistema. Il comando è apparso per la prima volta nella prima versione Unix di AT&T ed è presente nei sistemi operativi Unix e Unix-like.

7 DIMOSTRAZIONE DEL FUNZIONAMENTO

Dopo aver effettuato, tramite i passaggi precedenti, una corretta configurazione su entrambe le macchine andiamo ad effettuare il reboot della macchina. Una volta terminato il reboot apriamo il terminale entriamo nella "versione amministratore" tramite il comando "*sudo su*" avviamo condor e inviamo il lavoro tramite la macchina Master con il comando "*condor_submit submit*" e controlliamo lo stato con "*condoor_status*". I lavori in coda vengono tenuti dal condor che ha l'activity "idle" come dimostrato nella figura sottostante



Il condor con lo stato "idle" diventerà "bussy" ed eseguirà un lavoro mentre andrà in "idle" e terrà un lavoro in coda. Succesivamente anche il secondo condor diventerà "bussy" e perciò potremmo dedurre che i lavori sono terminati. Le schermate sono omesse perché pressoché identiche alla precedente.

8 CONCLUSIONE

In conclusione posso dire che il condor da me installato e configurato esegue correttamente le sue funzioni, e grazie ai file che ho creato nel file submit, posso monitorare lo stato di ogni lavoro, da chi viene eseguito quel lavoro e se ci sono errori durante l'esecuzione degli stessi. Ho trovato molto utile la creazione di questi file, anche se la loro creazione non è obbligatoria ma facoltativa, per il continuo monitoraggio dell'intero condor.