

Ubuntu Cloud

**Tutorial e Manuale di utilizzo per la
creazione di un Cloud privato con
Ubuntu Server 12.04LTS**

Giuseppe Calsolaro

Sommario

- Ubuntu Cloud 3**
 - Canonical Ubuntu e il software libero 3**
 - MAAS: Metal as a Service..... 4**
 - Quando usare MAAS?..... 5**
 - Una configurazione tipica MAAS 5**
 - MAAS Region Controller..... 6**
 - Cluster Controller..... 6**
 - Node Controller 7**
 - TFTP (PXE) e DHCP 7**
 - Juju Orchestration..... 9**
- MAAS Private Cloud 11**
 - I componenti 11**
 - Dietro le quinte..... 13**
 - Requisiti Hardware..... 14**
 - Requisiti Software..... 14**
 - Distribuzione dell'hardware sulle macchine virtuali 14**
- Creare un Cloud Privato con MAAS..... 16**
 - Installazione e prima configurazione delle macchine 16**
 - Post Installazione..... 18**
 - Aggiungere i Nodi al Cloud 21**
 - Juju - Il Deploy dei charms..... 24**
 - Il Deploy dei servizi 28**
 - Terminare un servizio..... 37**
- Troubleshooting..... 39**
 - Problema di perdita delle informazioni di rete 39**
 - Problema di inoltro delle chiavi..... 39**
 - Username e password del superuser MAAS dimenticati 39**
 - Problema del trasferimento di chiavi 40**
- Ottimizzazioni di MySQL..... 41**

Ubuntu Cloud

Canonical Ubuntu e il software libero

Canonical¹, Corporate Sponsor della distribuzione di Linux nota come *Ubuntu*, ha svolto in questi anni un interessante lavoro dedicato allo sviluppo di applicazioni distribuite su numerosi ambienti di calcolo. Ubuntu, secondo Canonical, è sistema operativo più popolare al mondo impiegato per Cloud pubblici, privati e ibridi e sta cercando di renderlo il sistema operativo numero uno per il numero di macchine cooperanti in Cloud.

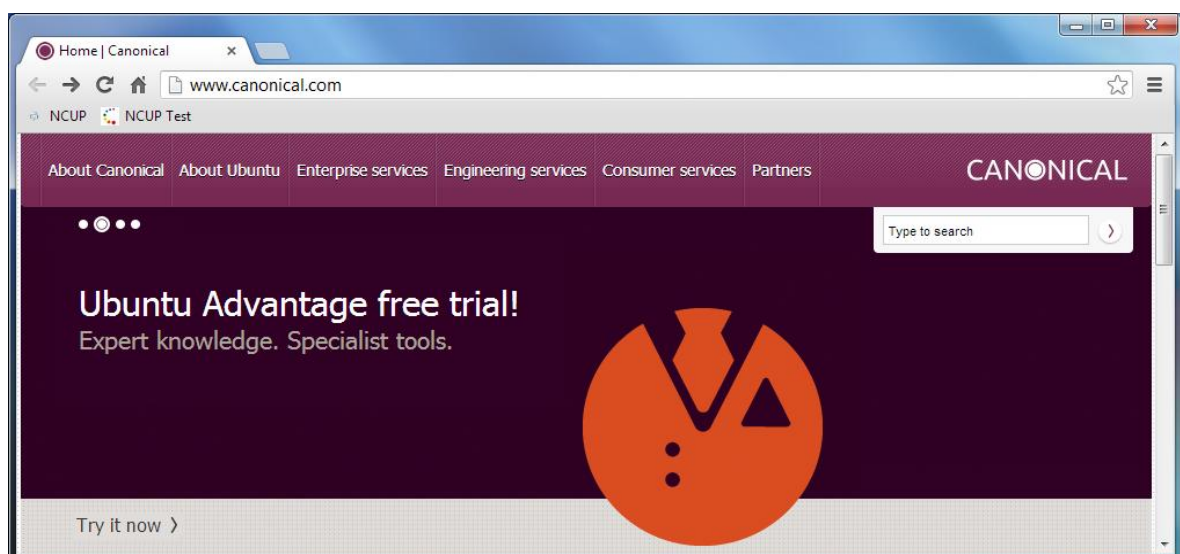
L'idea di base è che l'hardware è sempre più una *Commodity* che si compra per quello che offre, non per quello che è. In un ambiente Cloud, non ci si interessa troppo sui dettagli tecnici della CPU, della velocità dei bus, o dei canali di memoria. Si vuole semplicemente potenza di calcolo, Storage e Networking. Secondo Canonical, MAAS è un modo per astrarre tutti i dettagli fisici dei computer in ciò che veramente importa.

Secondo *Matt Raveil*², MAAS si colloca a un livello sottostante l'*IAAS (Infrastructure-as-a-Service)*. L'idea è quella di utilizzare MAAS per gestire l'hardware, e utilizzare Juju per coordinare le applicazioni e i carichi di lavoro. In teoria, è possibile utilizzare MAAS e Juju in maniera cooperante per implementare di simile a *OpenStack*³ ma semplicemente di riuscire a configurare le istanze in maniera molto più rapida.

Esso rende facile configurare l'hardware su cui distribuire qualsiasi servizio che ha bisogno di scalare su e giù in modo dinamico (il Cloud è solo un esempio). Consente di allocare i server in maniera dinamica, proprio come le istanze di un Cloud, solo che in questo caso, stiamo parlando di veri e propri Nodi fisici. Aggiungi un altro nodo al cluster, e assicurarsi che abbia almeno 16 GB di RAM (ad esempio) diventa facile come richiederlo.

Con una semplice interfaccia Web, è possibile aggiungere, commissionare, aggiornare e riciclare server a volontà. Al mutare delle esigenze è possibile rispondere rapidamente con l'aggiunta di nuovi Nodi e/o riallocarli in maniera dinamica tra i servizi. Quando arriva il momento, i Nodi possono essere estratti per l'uso al di fuori di MAAS.

MAAS è disponibile con la versione beta di *Ubuntu 12.04 LTS*.



¹ <http://www.canonical.com/>

² Product Manager di Canonical MAAS

³ OpenStack è un progetto IAAS Cloud computing di [Rackspace Cloud](#) e [NASA](#). E' un software libero open source rilasciato sotto licenza Apache.

MAAS: Metal as a Service

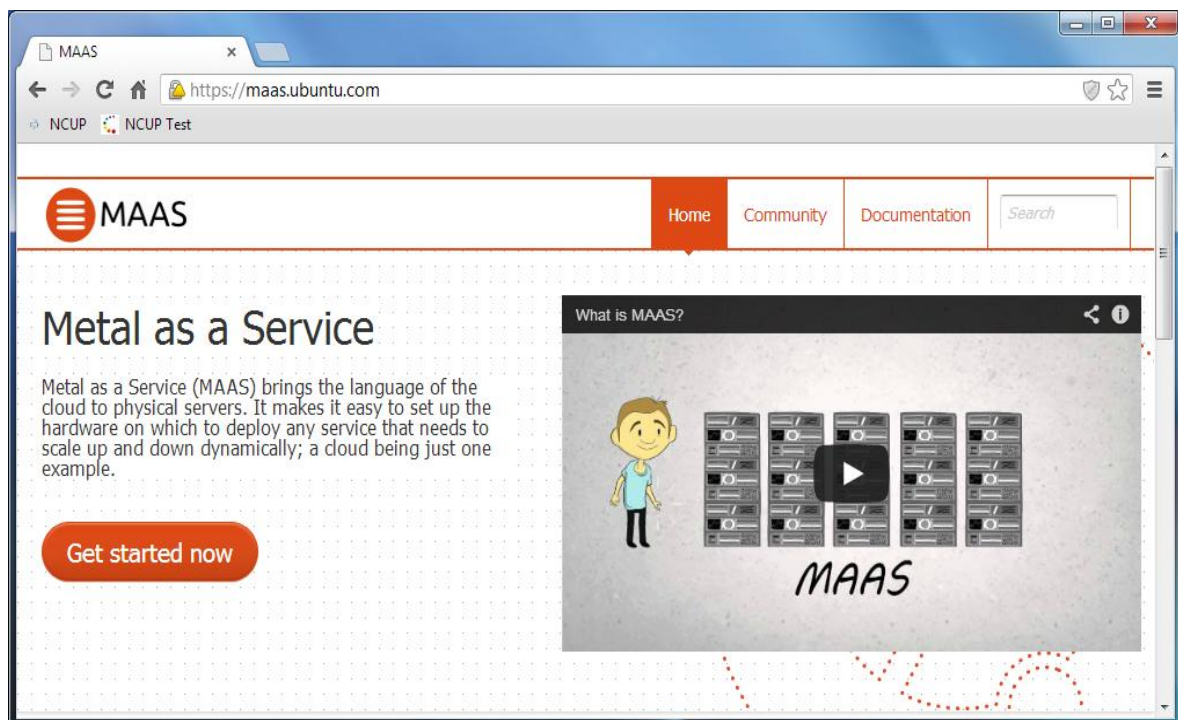
Il Cloud ha rivoluzionato l'approvvigionamento e la fornitura di istanze virtuali e Storage su richiesta in pochi minuti o secondi. Al centro del Cloud c'è la virtualizzazione: le risorse sono tutte virtualizzate, così che possano essere recuperate su richiesta.

Molti servizi IT moderni seguono la filosofia Cloud: intere distese di server, che fanno più o meno la stessa cosa, hanno scalabilità orizzontale piuttosto che su scala industriale. In tali ambienti, è utile pensare alla collezione di macchine fisiche, come una nuvola, anche se non è virtualizzate. Così Canonical ha creato "*Metal as a Service (MAAS)*"⁴, un sistema che rende rapida e facile da installare, la base fisica hardware su cui implementare servizi complessi che hanno bisogno di scalare su e giù in modo dinamico.

Canonical MAAS collega il dinamismo del Cloud Computing al mondo della fornitura di macchine fisiche di Ubuntu. Connettere, commissionare e distribuire i server fisici a tempo di record, riassegnare i Nodi tra i servizi in modo dinamico e tenerli aggiornati e, a tempo debito, ritirarsi dal loro uso.

MAAS è un nuovo modo di pensare l'infrastruttura fisica. Elaborazione, Storage e network sono delle Commodities nel mondo virtuale, e per implementazioni su larga scala, lo stesso vale per MAAS. "*Metal as a Service*" consente di gestire svariati server come una risorsa malleabile per l'assegnazione a problemi specifici, e riallocazione su base dinamica.

In combinazione con il software di orchestrazione di Juju, MAAS permette di ottenere il massimo dal proprio hardware fisico e implementare servizi complessi in maniera dinamica con facilità e sicurezza.



⁴ <https://MAAS.Ubuntu.com/>

Quando usare MAAS?

Probabilmente si dovrebbe utilizzare MAAS se le proprie esigenze corrispondono a una di queste affermazioni:

- Si sta tentando di gestire molti server fisici.
- Si desidera implementare servizi con il minimo sforzo.
- Si desidera ottenere il massimo dalle proprie risorse.
- Si vuole lavorare in modo replicabile e affidabile.

Viceversa, non c'è bisogno di MAAS se una o tutte queste affermazioni risultano essere vere:

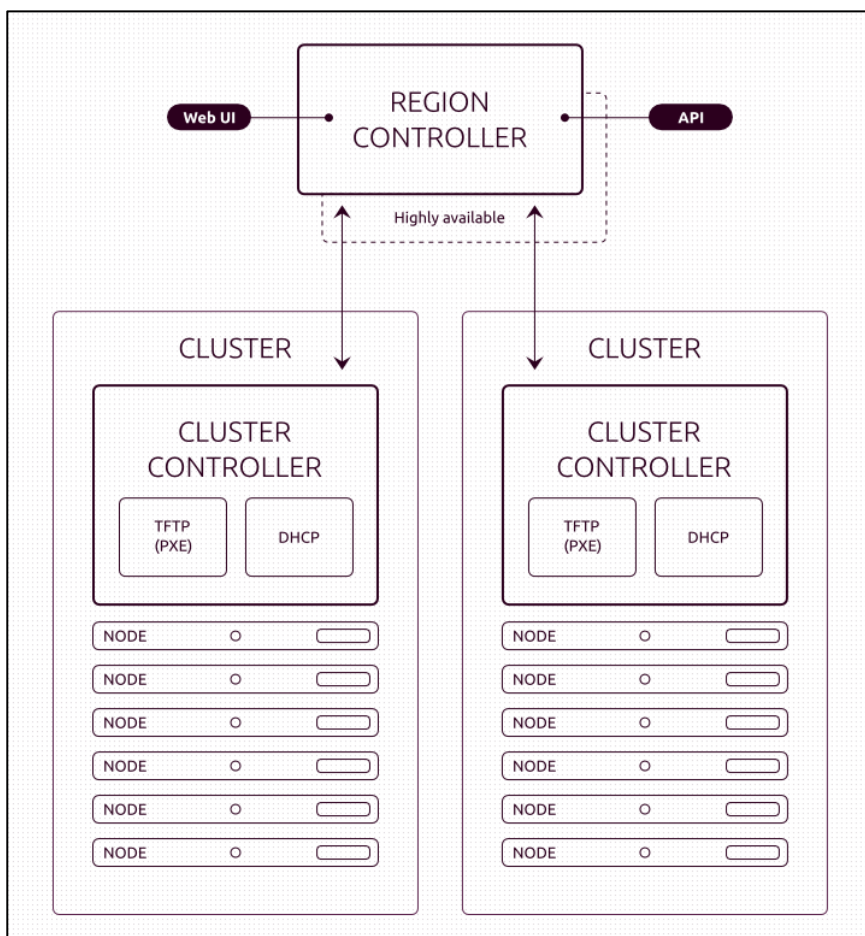
- Non si vuole gestire l'hardware fisico.
- Non si hanno problemi a trascorrere molto tempo in una stanza server.
- Si vuole provare a configurare servizi complicati senza nessun aiuto.

Una configurazione tipica MAAS

MAAS è progettato per funzionare con l'hardware fisico, indipendente se il setup include migliaia di server o solo pochi. I componenti chiave del software MAAS sono:

- *Region controller.*
- *Cluster controller(s)*
- *Nodes*

Per i piccoli (in termini di numero di Nodi) configurazioni, probabilmente basta installare il Region Controller Cluster Controller sullo stesso server: non è assolutamente produttivo ed utile avere più Region Controller se si ha il bisogno di organizzare i Nodi in subnets diverse (ad esempio, se si hanno molti Nodi).



MAAS Region Controller

Il *Region Controller*, o *Cloud Controller (CLC)*, è colui che gestisce il front-end per l'intera infrastruttura Cloud. Esso fornisce un'istanza Web compatibile con i servizi di interfaccia per gli strumenti del client da una parte e interagisce con il resto dei componenti dell'infrastruttura Cloud dall'altra. Fornisce, inoltre, un'interfaccia Web per gli utenti i quali possono gestire svariati aspetti dell'infrastruttura.

Le sue funzioni:

- Monitorare la disponibilità di risorse sui vari componenti dell'infrastruttura Cloud, compresi i Nodi supervisor utilizzati per mettere a disposizione le istanze dei Nodi e i Cluster Controller che gestiscono i Nodi veri e propri.
- Arbitrario delle risorse: decidere quale cluster verrà utilizzato per il fornire le istanze.
- Monitorare le istanze in esecuzione.

In breve, il Region Controller ha una conoscenza completa della disponibilità e dell'utilizzo delle risorse nel Cloud e del suo intero stato.

Cluster Controller

Il Cluster Controller gestisce uno o più *Node Controllers* e distribuisce/gestisce le istanze su di essi. Esso gestisce anche il collegamento in rete per le istanze in esecuzione sui Nodi. Esso comunica col Region Controller (CLC) su un lato e sul Node Controller dall'altro. Le risorse gestite includono i Nodi stessi, gli orari e le reti per le macchine virtuali.

Il controller del cluster deve avere la comunicazione con il controller Cloud, e deve avere una comunicazione con il controller di ogni nodo nel suo cluster. Esso è strettamente un componente interno, e quindi non dovrebbe avere una comunicazione diretta con il mondo esterno. Infatti, tutte le attività degli utenti che richiedono un intervento da parte del controller del cluster utilizzeranno i comandi del controller di Cloud, che a sua volta comunicherà direttamente le azioni da intraprendere al controller del cluster.

Esso riceve anche gli ordini del controllore Cloud. Gli ordini che il controller del cluster riceve dal controller Cloud hanno lo scopo di gestire l'esecuzione delle istanze nella loro zona di disponibilità.

Il Region Controller infatti, non specifica quale nodo deve compiere un'azione, ma solo il Cluster delegato a tale operazione. Ad esempio, dopo il Cluster Controller ha ricevuto un ordine di eseguire dal Region Controller nella propria zona di competenza, decide su quale nodo, che possiede quell'istanza, girare la richiesta e ne invia l'ordine di esecuzione del caso in cui le richieste sono multiple.

Le sue funzioni:

- Ricevere richieste dal Region Controller per allocare l'istanza di esecuzione.
- Decidere quale Node Controller utilizzare per invocare l'istanza richiesta.
- Controllare la rete virtuale disponibile per le istanze.
- Raccogliere informazioni sui Node Controllers registrati comunicandoli al Region Controller.

Node Controller

Un nodo è un virtual server attivo, in grado di eseguire delle KVM⁵ come hypervisor. MAAS installa automaticamente una KVM quando l'utente sceglie di installare il nodo. Le macchine virtuali in esecuzione controllate da MAAS sono chiamate istanze. Canonical ha scelto KVM come hypervisor predefinito nonostante sia possibile sceglierne di diversi.

Il Node Controller viene eseguito su ciascun nodo e controlla il ciclo di vita delle istanze in esecuzione sul nodo. Esso interagisce con il sistema operativo e l'hypervisor in esecuzione sul nodo da un lato e il Cluster Controller dall'altro.

Il Node Controller interroga il sistema operativo in esecuzione sul nodo per scoprirne le sue caratteristiche fisico (il numero di core, la dimensione della memoria, lo spazio su disco disponibile e anche per conoscere lo stato delle istanze delle macchine virtuali in esecuzione sul nodo e propaga i dati fino al Region Controller.

Le sue funzioni:

- Raccoglie dati relativi alla disponibilità delle risorse e l'utilizzo delle stesse sul nodo comunicando i dati al Cluster Controller.
- Gestisce il ciclo di vita dell'istanza.

TFTP (PXE) e DHCP

Il Boot delle macchine come Nodi e il riconoscimento delle stesse al Region Controller avviene tramite gli strumenti di *DHCP* e *PXE* situati all'interno di ciascun Cluster Controller.

Di solito, gli amministratori dei datacenter hanno per anni usato *PXE*⁶ per eseguire a distanza installazioni automatiche di software sulle macchine.

Sono di seguito elencati i passi che esso esegue quando viene avviato un server secondario, a patto che quello primario, nel nostro caso quello in cui è situato il Region Controller, sia attualmente in esecuzione:

- Il BIOS cerca un server DHCP, e ottiene alcuni parametri come l'indirizzo IP e l'indirizzo di un server TFTP.
- Ottiene un file di configurazione dal server TFTP con un menu di avvio (pxelinux.0).
- Scarica il programma di installazione del sistema operativo e installa sulla macchina.
- Al riavvio della macchina, l'OS viene installato.

Il tutto è allettante, ma è un sistema difficile da configurare e costoso da mantenere.

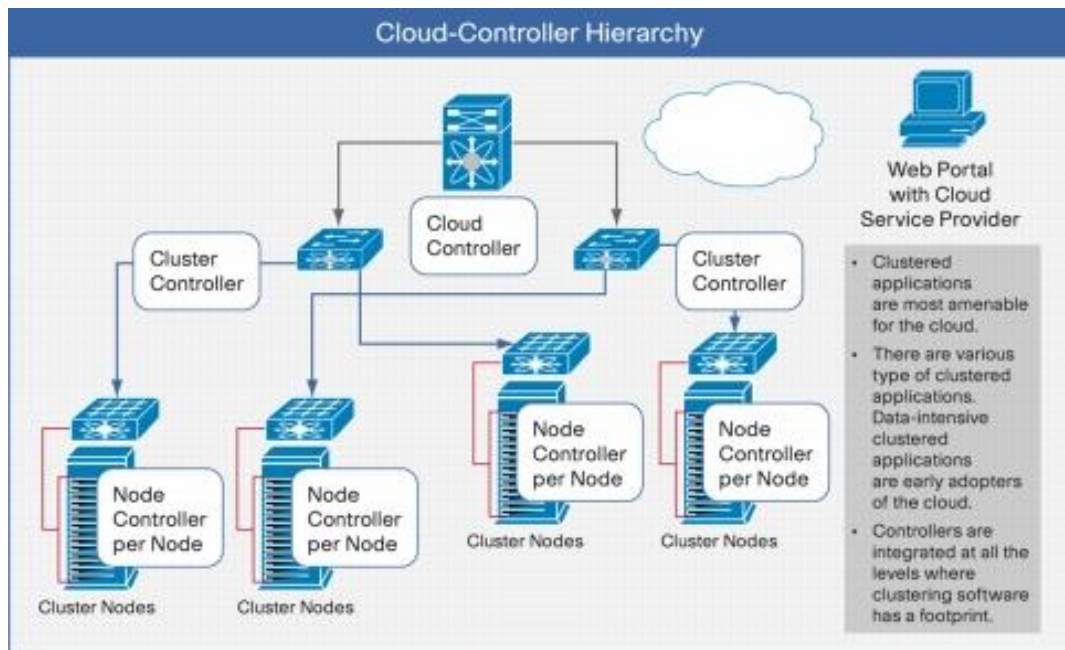
A questo punto interviene *Cobbler*⁷. Cobbler è un server di installazione di Linux che permette la configurazione rapida di ambienti di installazione di rete. Esso collega e automatizza svariati task Linux in rete in modo da non dover saltare tra molti comandi e collegandoli alle varie applicazioni durante la distribuzione di nuovi sistemi. Cobbler aiuta nella gestione DNS e DHCP. In pratica, nel nostro caso, consente di definire profili per i diversi tipi di macchine o di ruoli e si occupa di allocare una macchina all'interno del namespace degli indirizzi IP dedicati ai client.

Quindi, MAAS è fondamentalmente una interfaccia Web che fa il PXE / DHCP di configurazione server, gestisce le ISO di installazione sui Nodi, e si integra con Juju per la gestione dei pacchetti dei servizi.

⁵ Kernel-based Virtual Machine

⁶ PreBoot Execution Environment

⁷ <http://cobbler.github.com/> Cobbler è una versione server di Linux che permette un rapido setup di un network.



Juju Orchestration

Per molto tempo, le distribuzioni server di Linux, son state orientate verso la collaborazione e interoperabilità di più macchine fisiche. In alcuni casi si trattava di server differenti su ognuno dei quali girava un differente set di applicazioni, portando si organizzazione, ma anche isolamento, risorse riservate, e altre caratteristiche simili. In altre situazioni i server erano configurati in maniera molto simile, in modo che il sistema diventasse più scalabile avendo carico distribuito di risorse, e in modo che l'intero sistema diventasse più affidabile dato che il guasto di una singola macchina non influenzava il gruppo nel suo complesso. In questo modo, gli amministratori di server diventano degli esperti che orchestravano il posizionamento e la connettività dei servizi all'interno del gruppo di server.

Dato questo scenario, è sorprendente che la maggior parte degli sforzi per promuovere la gestione della configurazione del software sono ancora legati alle singole macchine. Ulteriori sforzi sono rivolti al problema di gestire più macchine come una singola unità, ed è interessante notare, che esse possiedono ancora un meccanismo per gestire dei servizi in maniera individuale. In altre parole, esse permettono all'amministratore di modificare la configurazione individuale dei servizi in una sola volta, ma difficilmente garantiscono l'orchestrazione immediata di molteplici servizi distribuiti.

Questa è la sfida che ha motivato la ricerca attraverso il progetto *Juju*⁸ di Canonical. Juju vuole essere al tempo stesso un fornitore di servizi e strumento di orchestrazione degli stessi, che consente lo stesso tipo di collaborazione e la facilità d'uso che si vede oggi in giro per la gestione dei pacchetti, ma su un livello più alto, sui servizi. Con Juju, chiunque è in grado di creare servizi in modo indipendente, e fare in modo che comunichino attraverso un protocollo di configurazione semplice. Quindi, gli utenti possono utilizzare due o più prodotti di Juju distribuirli comodamente in un ambiente, in modo simile a come si installa un pacchetto attraverso la rete col comando "apt".

Juju è un *Framework di Deployment*⁹ e gestione dei servizi di nuova generazione. E 'stato paragonato ad APT¹⁰ per il Cloud. La vera innovazione di Juju consiste in quello che viene definito "charm". Esso può essere visto come un pacchetto che contiene tutto il necessario per implementare un particolare servizio. Con Juju, ognuno è in grado di creare il proprio charm in maniera completamente indipendente dall'architettura fisica, e renderlo fruibile e interoperabile con altri servizi, comunicando tramite un semplice protocollo. Juju è stato sviluppato utilizzando *Python*. Tuttavia un charm realizzato per Juju può essere scritto in qualsiasi linguaggio di programmazione. Attualmente ogni unità di servizio è Deployata su un nodo diverso, il quale può essere strutturato in modo tale da comunicare con gli altri attorno a lui. Il motivo che ha portato a questa decisione è stato quello di mantenere Juju in uno stato di lavoro molto veloce. In futuro non si esclude che una singola macchina possa ospitare molteplici servizi.

Il 3 Aprile 2012 è stato creato il "Juju Charms Store"¹¹. Stiamo parlando di un vero e proprio negozio virtuale gratuito, un repository nel quale vengono messi a disposizione svariate decise di charms, per ognuno dei quali è presente una documentazione per la procedura di Deploy, scaricamento e aggiornamento di pacchetti. Ogni charms viene gestito al pari di una feature gratuita alla quale si può accedere e scegliere di installare sulla propria macchina in pochi secondi.

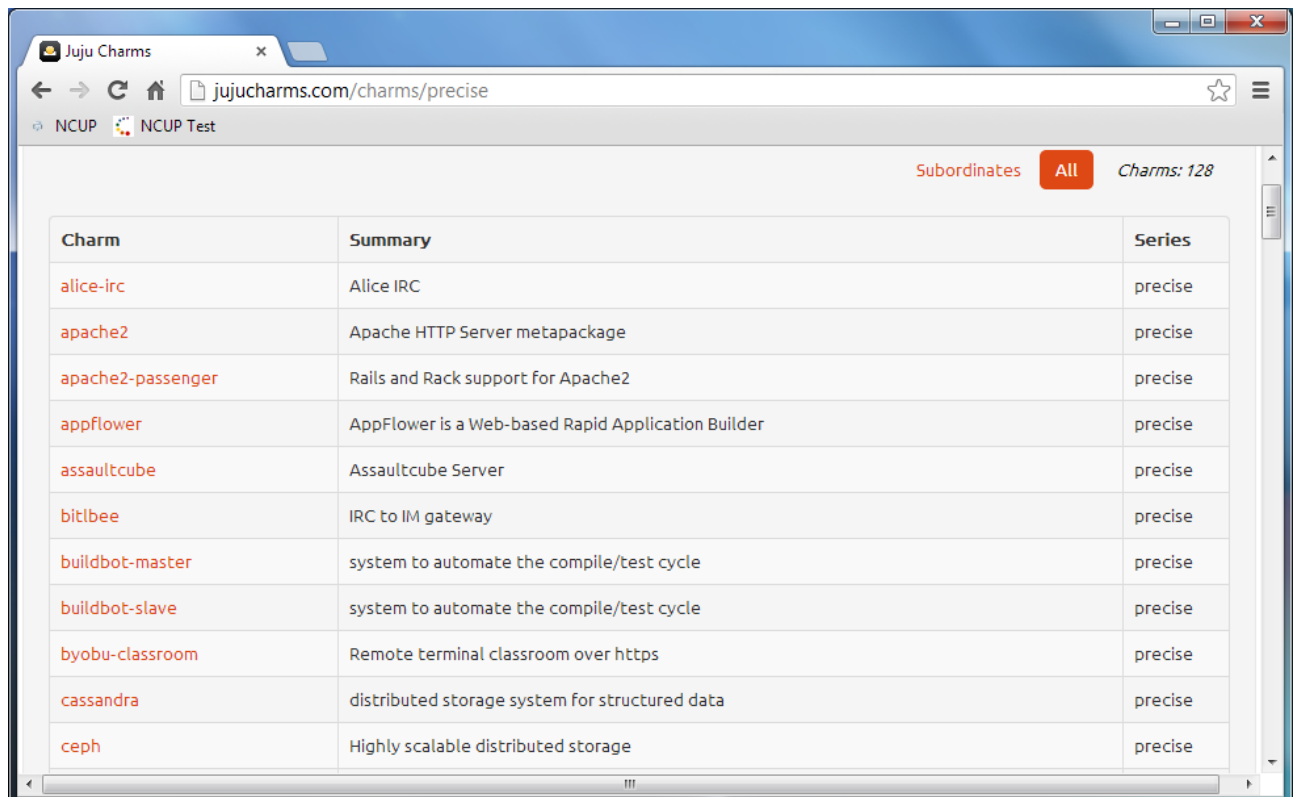
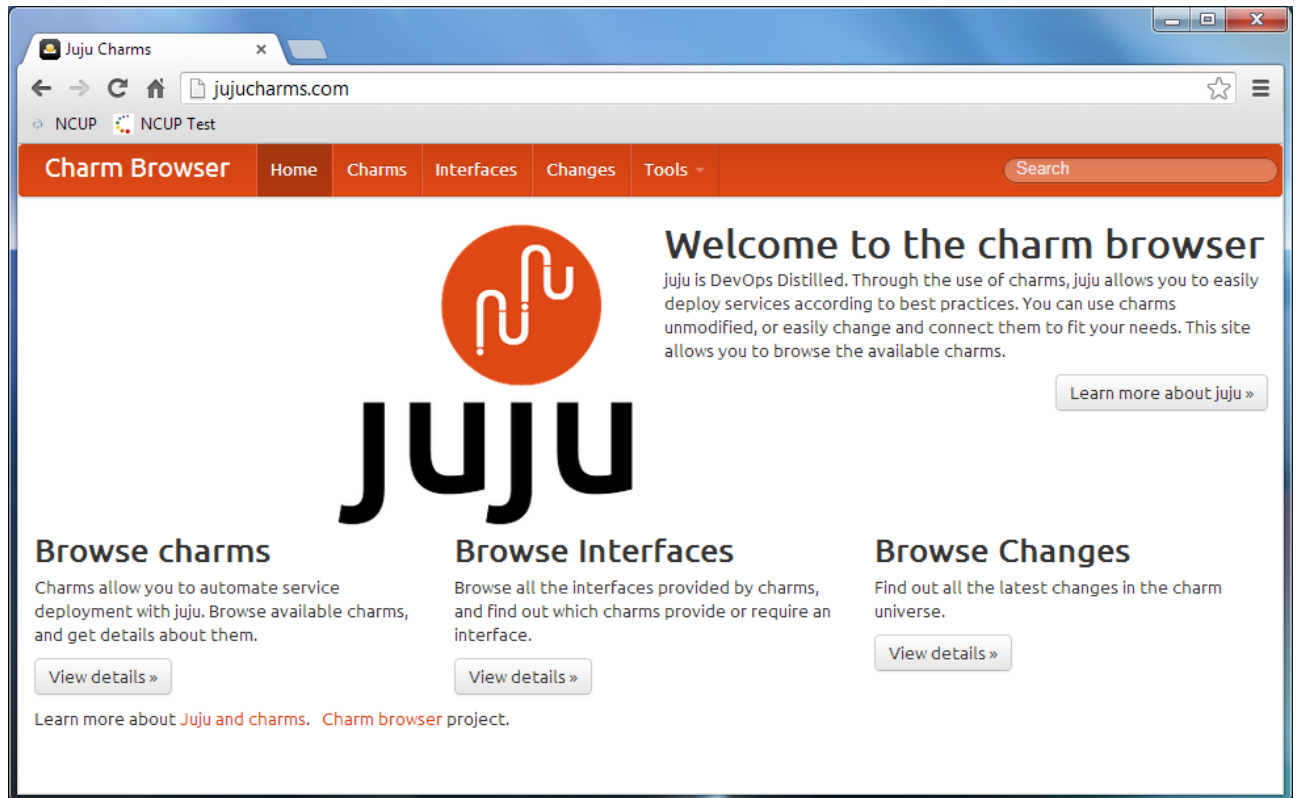
⁸ <https://Juju.Ubuntu.com/>

⁹ http://www.youtube.com/watch?v=1yoldgdqzLk&feature=player_embedded

¹⁰ Advanced Packaging Tool.

¹¹ <http://Jujucharms.com/>

Parliamo ad esempio di *mysql*, *phpmyadmin*, *wordpress*, *django*, *joomla*, svariati server applicativi, server apache come *tomcat* e *cassandra*, server ftp, server di posta, server dedicati allo storage per gestire documenti personali, musica e molto altro.



MAAS Private Cloud

I componenti

Come già spiegato, MAAS consente di trattare server fisici come macchine virtuali nel Cloud. Invece di dover gestire individualmente i singoli server, MAAS trasforma il nostro hardware in un Cloud dinamico al pari di una singola risorsa. Che cosa significa in pratica? Si comunica a MAAS quali macchine si desidera di gestire, provvede ad avviarle, controlla l'integrità dell'hardware, e le tiene in attesa per quando se ne ha bisogno. È quindi possibile creare Nodi, distruggerli e ridistribuirli proprio come è possibile con le macchine virtuali nel Cloud.

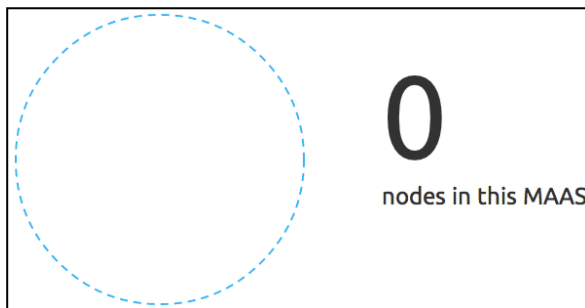
Quando si è pronti a Deployare un servizio, MAAS alloca a Juju i Nodi di cui ha bisogno per alimentare tale servizio. Al mutare delle esigenze, si può facilmente scalare i servizi: si ha bisogno di più potenza per un determinato cluster al fine di gestire la priorità dei servizi? Basta deallocare uno dei Nodi di elaborazione e ridistribuire le risorse. Quando si ha finito, è altrettanto facile riallocare le risorse.

MAAS è ideale se si desidera la flessibilità e la potenza del Cloud, ma è necessario implementare tutto da zero.

Come ho potuto aver modo di vedere e capire, MAAS esibisce un comportamento diverso a secondo dello stato del nodo. Ecco l'elenco degli stati che il nodo può assumere in MAAS:

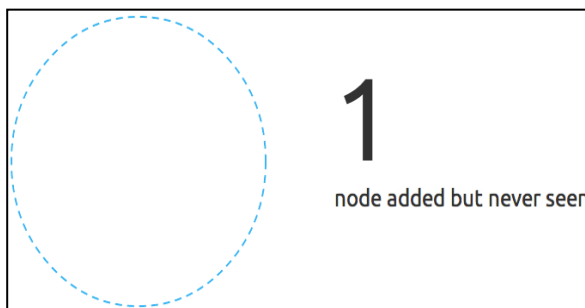
- Offline (aggiunto il MAC-Address) – Commissionato.
- Aggiunto ma mai visto (richiedere un avvio PXE) – Dichiarato.
- In coda – Pronto.
- Deployato – Allocated a MAAS.

Di seguito il dettaglio di ogni stato:



0 Nodi nel MAAS.

Ecco lo stato iniziale con cui si presenta MAAS. Viene fornita una vista grafica dell'ambiente di lavoro con un indicatore numerico che fornisce indicazione sul numero di Nodi presenti nel nostro Cloud. In questo stato, semplicemente, non è avvenuto ancora nulla.

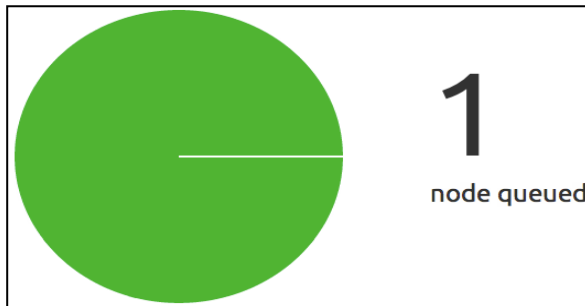


Mai visto. Dichiarato.

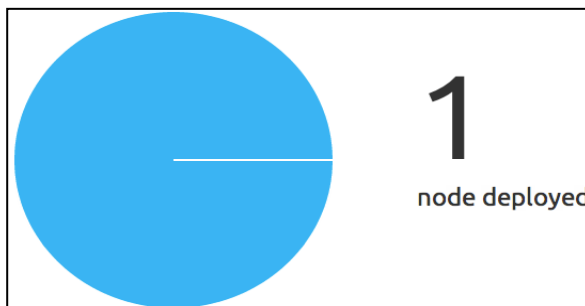
Questo stato viene visualizzato quando un nodo non registrato in MAAS è in attesa di avviarsi in PXE. È possibile scegliere l'opzione che arruolare acquisirà il server per MAAS. L'unica cosa da fare è accettare il nodo tramite la GUI Web. Quindi, fare clic su "Accept & commission".

**Offline. Commissionato.**

Il nodo è stato aggiunto a MAAS ed è in attesa per essere avviato. Dopo la prima sequenza di avvio è necessario rilanciare la macchina al fine di avviare la negoziazione col server MAAS.

**In coda. Ready.**

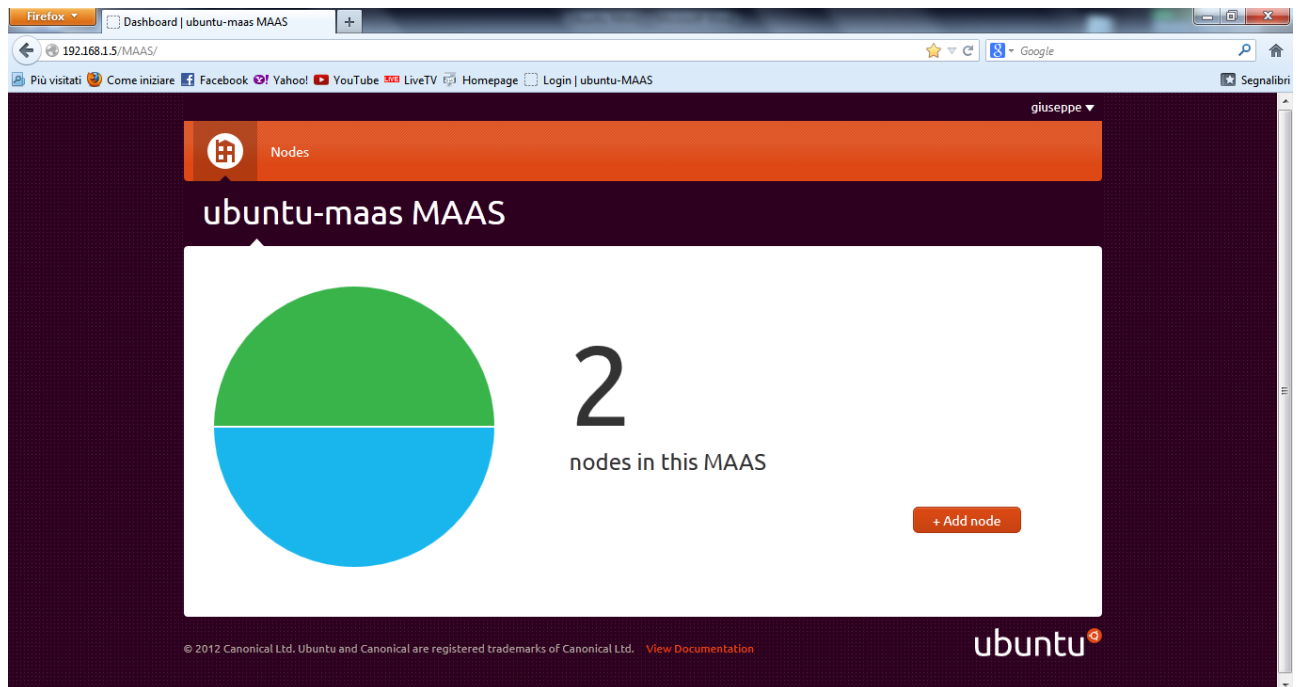
Il nodo è stato avviato tramite PXE, ha ottenuto un indirizzo IP dal server MAAS col quale avviare la negoziazione con esso. Ora è il momento di installare il sistema operativo. Questa fase, nella migliore delle ipotesi si prolungherà per almeno 65 minuti!

**Deployato. Allocato a MAAS.**

Questa è la fase finale. Ciò significa che il nodo contrassegnato come “Deployed” ha il compito di negoziare con Juju tramite un’operazione di Bootstrap. Questo nodo è pronto, è possibile utilizzarlo per un Juju charm. E’ da sottolineare che a quanto pare, il nodo è stato interamente acquisito da Juju e

non è possibile rimuoverlo. L'unico modo è distruggere l’unità con Juju.

Ed ecco qui una vista finale del *Dashboard* fornito dal Region Controller di MAAS tramite interfaccia Web. Da questa vista, è possibile aggiungere Nodi, allocarli a MAAS, ottenere informazioni su di essi e organizzarli in modo da renderli disponibili per l’utilizzo. E’ inoltre possibile avere accesso a una sezione di gestione degli account che fornisce un modo grafico per poter generare chiavi di sicurezza SSH, certificati di negoziazione, e tanto altro.



Dietro le quinte

Ultimo step prima di avviare la descrizione di come ho utilizzato Ubuntu MAAS per generare il mio Cloud privato. E' di fatti molto importante essere consapevoli di ciò che accade dietro le quinte mentre MAAS è in esecuzione. Tutto comincia dal server MAAS. Durante il processo di installazione della distribuzione 12.04LTS di Ubuntu, ci viene chiesto di scegliere tra un'installazione base della versione server e un'installazione Server MAAS, con l'”*enlistement*”¹² dei Nodi annessi. Installando MAAS, verranno annessi all'installazione un sacco di servizi come apache, DNS, *TFTP*¹³, una gestione opzionale DHCP. MAAS è scritto in Python e l'interfaccia grafica Web utilizza il Framework *Django*¹⁴. MAAS utilizza un database locale in cui memorizzare tutti i Nodi da Deployare. Un nodo è identificato dal suo indirizzo MAC-Address. Il modo migliore per configurare e gestire il server MAAS è utilizzare l'interfaccia grafica Web. Il server sarà gestito da un superutente, che è obbligatorio, e senza il quale non sarebbe possibile avviare la gestione del server MAAS.

I passi base effettuati (in automatico e non) per la configurazione di MAAS sono:

1. Registrare i Nodi tramite MAC-Address in MAAS
2. Avviare il node in modalità PXE Boot.
3. Negoziazione tra MAAS e il nodo tramite DHCP.
4. L'IP ottenuto dal nodo durante la negoziazione sarà inserito all'interno di un range di indirizzi dedicati tramite DHCP. Il nodo otterrà inoltre un Hostname visibile a partire dall'interfaccia Web che avrà la seguente forma “node-a482df4c-9458-11e1-a951-000c290cdd88.local”
5. Il nodo resta nello stato di “enlist” fino a che il server non carica l'immagine ISO degli OS. Questa operazione dura alcuni minuti e comporta lo spegnimento automatico della macchina che ospita il nodo, una volta forniti al server i propri parametri e indirizzi.

¹² Traducibile in Italiano con “Arruolamento”.

¹³ http://it.wikipedia.org/wiki/Trivial_File_Transfer_Protocol Trivial File Transfer Protocol

¹⁴ Framework open source per sviluppo di applicazione Web, scritto in Python.

<https://www.djangoproject.com/>

- 6. Il Boot tramite PXE avvierà l’importazione sul nodo dell’immagine ISO presente sul server e la successiva installazione della stessa. Al termine verrà generato uno “user profile” relativo all’installazione effettuata, il quale sarà accessibile via Web a un indirizzo simile a questo: “http://192.168.1.100/cblr/svc/op/ks/system/node-a482df4c-9458-11e1-a951-000c290cdd88”.
- 7. Il log di qualunque operazione effettuata viene salvata nei seguenti percorsi “/var/log/MAAS/rsyslog” e in “/var/log/squid-deb-proxy/access.log”.

Requisiti Harwdare

Sistema Operativo	Windows® 7 Professional 64 bit
Processore	Intel® Core™ i7 Processore 2670 : CPU @ 2.20 GHz
Chipset	Mobile Intel® HM65 Express Chipset
Memoria Principale	DDR3 1333 MHz SDRAM, 2 x SODIMM 8GB SDRAM
Display	15.6" Full HD/HD (1366x768) /16:9
Scheda Video	NVIDIA GeForce® GT 540M, 2GB DDR3 VRAM
Disco Fisso	500GB,7200rpm
LAN/WLAN	10/100/1000 Base 802.11 b/g/n

Requisiti Software

Software di virtualizzazione	OracleVirtualBox (with LAN configuration extension pack)
Network configuration	Bridge Adapter, Realtek PCI GBE Family Controller (PCnet-FAST III)
Modalità promiscua	Permetti tutto.
Distribuzione Linux	Ubuntu Server 12.04.2-i386-LTS

Distribuzione dell’hardware sulle macchine virtuali ¹⁵

I dati riportati di seguito fanno riferimento a una configurazione minimale che mi è stato possibile gestire con il pc a mia disposizione. Ovviamente, trattandosi di distribuzioni server di linux e contando che ogni servizio Deployato richiede un nodo a sé stante, si capisce che le risorse di calcolo necessarie dovrebbero essere enormemente superiori. Quindi la configurazione riportata di seguito è ben lontana da quella che nella realtà si dovrebbe avere.

- Server MAAS

Hardware	Utilizzato	Suggerito	Note
CPU	1GHz	2 x 2GHz	
Memoria	2GB RAM	8GB RAM	
Spazio su disco	16GB	200GB	8GB basta solo per installare l’immagine, la cache.
Networking	100Mbps	1000Mbps	L’installazione delle immagini sui Nodi via PXE richiede molte risorse.

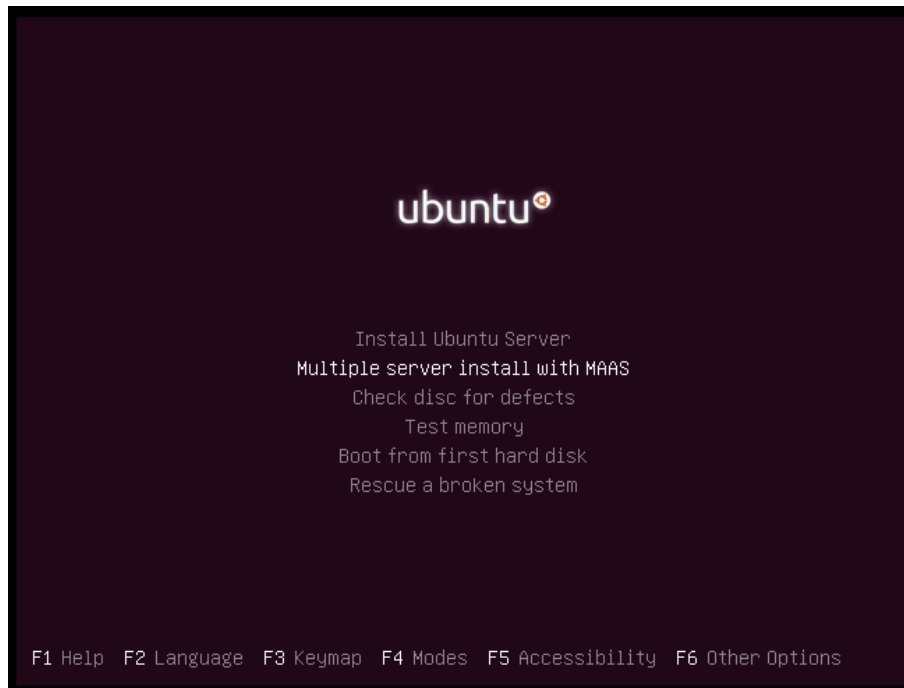
¹⁵ I requisiti “suggeriti” fanno riferimento alla guida ufficiale di Ubuntu Cloud Eucalyptus della distribuzione 10.04 di Ubuntu. <https://help.Ubuntu.com/community/UEC/CDInstall>

- Server Cluster/Nodo

Hardware	Utilizzato	Suggerito	Note
CPU	VT	VT, 64-bit	
Memoria	1GB RAM	4GB RAM	Più memoria significa molti più utenti gestibili.
Spazio su disco	8GB	100GB	
Networking	100Mbps	1000Mbps	L'installazione delle immagini sui Nodi via PXE richiede molte risorse.

Creare un Cloud Privato con MAAS

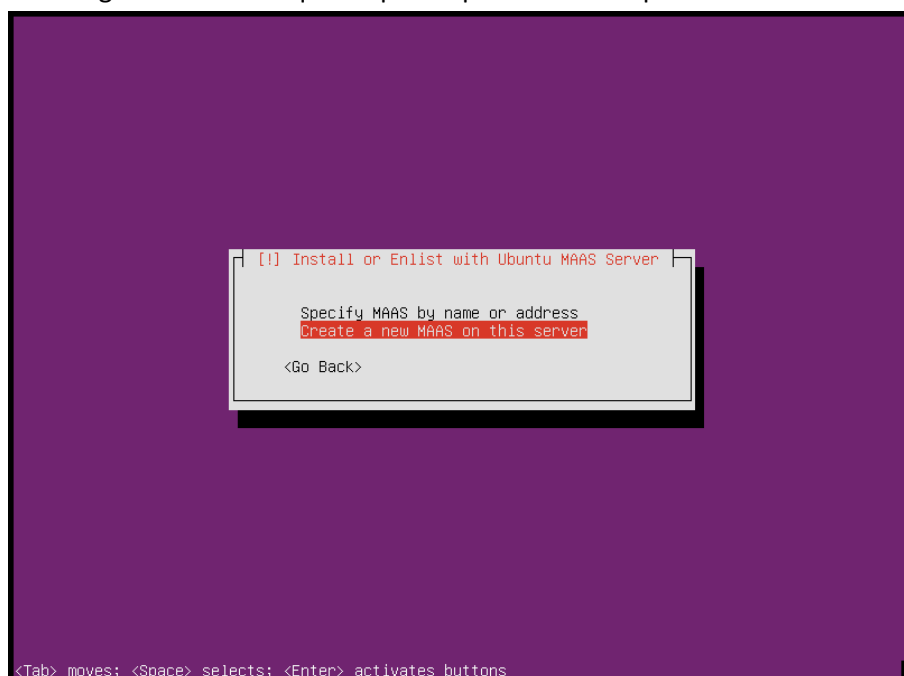
Installazione e prima configurazione delle macchine



Comincio illustrando le prime fasi di installazione sulla macchina virtuale dedicata ad ospitare il server principale, il server MAAS. Imposto come ordine di Boot il drive del lettore cd o direttamente il percorso dell'immagine ISO a mia disposizione. Si presuppone ovviamente, già configurata la sezione relativa

all'hardware di rete come evidenziato nella sezione precedente dei requisiti software. La procedura iniziale di installazione è molto semplice. Come proposto dall'immagine, viene chiesto quale tipo di installazione selezionare, se la classica versione server di Ubuntu oppure se installare server multipli utilizzando MAAS. Le fasi successive riguardano la scelta della lingua, dell'orologio, del sistema di base da installare e della configurazione della rete che sarà automaticamente risolta dal DHCP di Ubuntu. Niente di difficoltoso.

Scelto il nome da assegnare al mio Host ("*ubuntu-maas*"), si giunge alla fase dell'installazione in cui bisogna selezionare quale tipo di operazione intraprendere con MAAS. La scelta ricade su



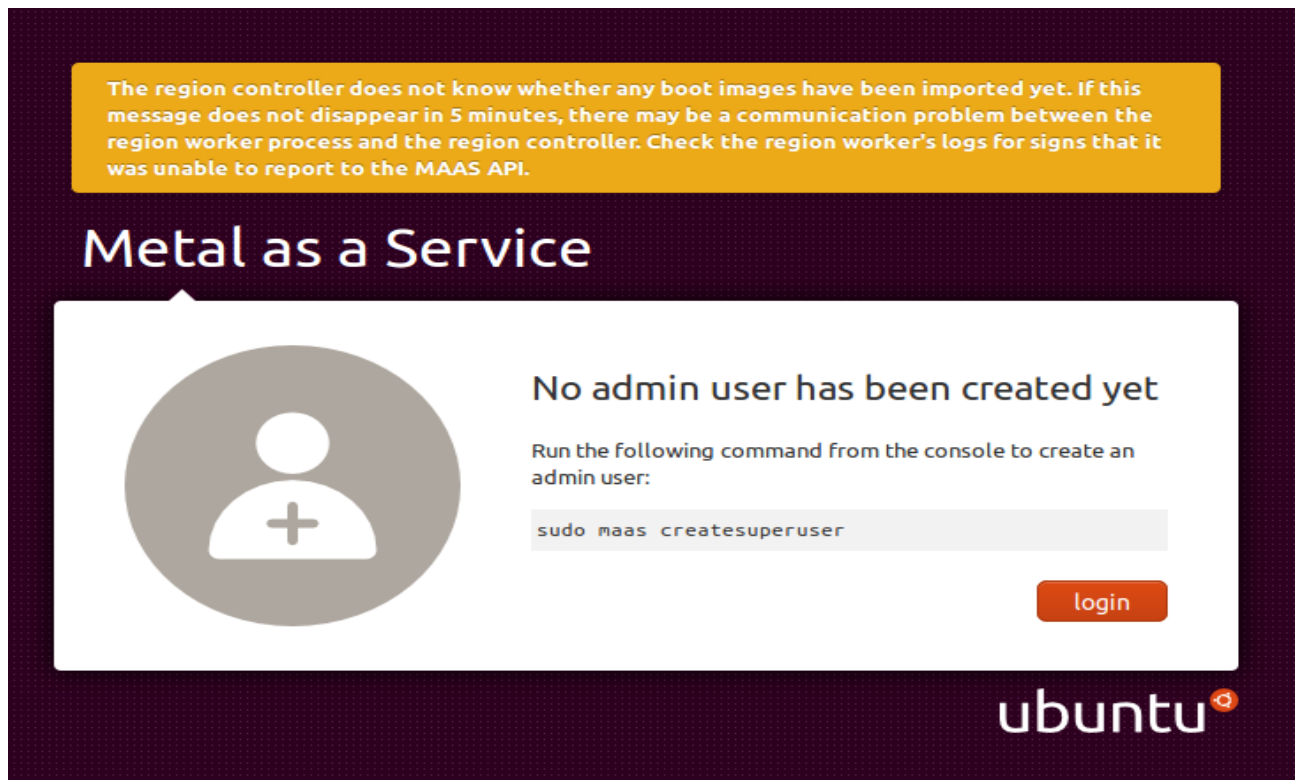
"*Create a new MAAS on this server*". Sto quindi selezionando di installare il Region Controller di MAAS sulla mia macchina. La seconda opzione, che per ora tralascio, ricorrerà nel momento in cui avrò bisogno di aggiungere dei Nodi al mio Cloud e pertanto avrò bisogno di

effettuare il “*discovery*” del server principale specificando il suo indirizzo IP di riferimento. Senza entrare nel merito, riprenderò il discorso a configurazione ultimata, in fase di aggiunta dei miei Nodi.

A questo punto l’interazione diventa ridotta: verranno installati i componenti di base del sistema, la lingua selezionata, un eventuale proxy di rete (se presente), e impostazioni di carattere generale come ad esempio la creazione dell’utente e il set della password di accesso al sistema. Al termine della procedura, se tutto è andato a buon fine verrà comunicato l’esito positivo dell’installazione e la seguente schermata indicherà l’indirizzo IP di riferimento mediante al quale, con qualunque browser, posso accedere alla parte di front-end del Region Controller, ovvero il Dashboard. Esso fornisce informazioni di carattere generale sullo stato dei Nodi, sulla loro configurazione, sugli utenti iscritti al sistema, sulle chiavi di accesso e molto altro. Inserisco l’indirizzo nel mio browser e la seguente schermata mi segnala l’avvenuta installazione del sistema.

Ricordiamo quanto accennato precedentemente e ciò spiega anche il significato dello Screenshot:

- E’ obbligatoria la presenza di un superutente che gestisca il Region Controller.
- MAAS lavora utilizzando delle immagini ISO che, su richiesta, copierà sui Nodi annessi eseguendo il Boot del sistema via PXE. E’ pertanto necessario importare e conservare le immagini sul server principale di riferimento.



D'ora in poi, è bene utilizzare un collegamento con sessione SSH alla macchina per diversi motivi. Prima cosa perché è più conveniente lavorare con una sola interfaccia che passa dalla visualizzazione di una macchina all'altra; secondo motivo, perché è possibile utilizzare l'opzione copia e incolla per i comandi che seguono la configurazione iniziale, che vedremo nel post-installazione. In questo modo si prevengono errori di ortografia che potrebbero capitare dato che avremo a che fare con comandi di gestione degli indirizzi IP, chiavi SSH, e configurazioni generali. A questo scopo, e lavorando in Windows con macchine virtuali Linux, Putty andrà benissimo.

```

giuseppe@ubuntu-maas: ~
login as: giuseppe
giuseppe@192.168.1.4's password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-26-generic i686)

 * Documentation:  https://help.ubuntu.com/
Last login: Tue Mar 26 12:13:22 2013
giuseppe@ubuntu-maas:~$ ifconfig
eth0      Link encap:Ethernet  IndirizzoHW 08:00:27:e2:15:fc
          indirizzo inet:192.168.1.4  Bcast:192.168.1.255  Maschera:255.255.255.
0
          indirizzo inet6: fe80::a00:27ff:fee2:15fc/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:81 errors:0 dropped:0 overruns:0 frame:0
          TX packets:93 errors:0 dropped:0 overruns:0 carrier:0
          collisioni:0 txqueuelen:1000
          Byte RX:8801 (8.8 KB)  Byte TX:15261 (15.2 KB)
          Interrupt:10  Indirizzo base:0xd020

lo        Link encap:Loopback locale
          indirizzo inet:127.0.0.1  Maschera:255.0.0.0
          indirizzo inet6: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:240 errors:0 dropped:0 overruns:0 frame:0
          TX packets:240 errors:0 dropped:0 overruns:0 carrier:0
          collisioni:0 txqueuelen:0
          Byte RX:37174 (37.1 KB)  Byte TX:37174 (37.1 KB)

giuseppe@ubuntu-maas:~$

```

Post Installazione

A questo punto, la rete è stata installata automaticamente in modalità Bridge, il sistema è visibile tramite Web e abbiamo accesso al terminale del server tramite un client SSH.

Le prime fasi del processo di configurazione prevedono innanzitutto l'aggiornamento del sistema alla sua versione più recente:

```

$ sudo apt-get update && sudo apt-get upgrade && sudo apt-get dist-upgrade
$ sudo reboot

```

Dopo l'aggiornamento si potrebbe verificare una riconfigurazione automatica degli indirizzi IP, pertanto potrebbe essere necessario riconfigurarli in maniera adeguata:

```

$ sudo dpkg-reconfigure maas
$ sudo dpkg-reconfigure maas-provision
$ sudo reboot

```

Aggiungo delle regole sul Firewall in modo tale che il dnsmasq dei Nodi non vada in conflitto con quello del server MAAS:

```

$ sudo apt-get install ebtables
$ sudo iptables -I INPUT -p udp --dport 69 -j REJECT
$ sudo ip6tables -I INPUT -p udp --dport 69 -j REJECT
$ for i in AA:BB:CC:DD:EE:F0 AA:BB:CC:DD:EE:F1 AA:BB:CC:DD:EE:F2 AA:BB:CC:DD:EE:F3
; do \ sudo ebtables -I INPUT -p IPv4 --ip-prot udp --ip-dport 67 -s $i -j DROP ;
done

```

Dove nel ciclo for bisogna sostituire gli effettivi MAC-Address dei Nodi installati.

A questo punto cambio i settaggi della configurazione di rete in modo tale da impostare un indirizzo statico per impedire che il DNS cambi l'indirizzo del server MAAS ad ogni avvio dello

stesso. Il server deve essere sempre raggiungibile con gli stessi parametri di configurazione, in ogni modo e in ogni momento. Per prima cosa installo nel sistema un editor di testo¹⁶ che mi permette di modificare i file di configurazione da linea di comando:

```
$ sudo apt-get install vim
```

In accordo con gli indirizzi di rete attualmente assegnati, modifico i seguenti file di configurazione nel seguente modo:

- *“/etc/network/interfaces”*

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
    address 192.168.1.4
    netmask 255.255.255.0
    network 192.168.1.0
    broadcast 192.168.1.255
    gateway 192.168.1.1
    dns-nameservers 192.168.1.1
    dns-search ubuntu-maas
#iface eth0 inet dhcp
```

- *“/etc/resolvconf/resolv.conf.d/base”*

```
search ubuntu-mAAS
nameserver 192.168.1.1
```

Modificando il primo file mi sono assicurato un indirizzo di rete statico impostando manualmente i valori dell'ipv4. Il secondo file invece consente di specificare qual è l'Host dedicato alla risoluzione dei nomi DNS. Pertanto in fase di inserimento di un URL, la *request* contatterà il server dedicato alla risoluzione degli indirizzi per instradarsi verso la destinazione richiesta. Lanciando ora il seguente comando:

```
$ sudo apt-get install maas dnsmasq debmirror
```

sto andando a installare il “dnsmasq”¹⁷, che mi consente l'instradamento automatico delle richieste DNS e la gestione del server DHCP. Questo tool è progettato per supportare reti di piccola dimensione come la mia. Il “debmirror”¹⁸ invece mi consente di creare un repository distribuito nella mia LAN e accessibile a tutti i computer interconnessi, in modo tale che l'aggiornamento di un pacchetto o l'aggiornamento di un parametro di configurazione sul server centrale, si ripercuota automaticamente sui Nodi collegati.

E', a questo punto, importante capire che l'unico server accessibile sulla mia LAN o verso l'esterno, è il server MAAS. Di fatto, anche se i servizi risiedono sui Nodi, la request sarà accolta dal server principale che si occuperà di smistare le richieste verso la loro destinazione finale.¹⁹

Quindi bisogna fare in modo di simulare il funzionamento di un router per rendere possibile l'accesso all'esterno anche alle altre macchine presenti nel Cloud. Attivo pertanto il meccanismo di instradamento dei pacchetti in ingresso e in uscita:

¹⁶ http://it.wikipedia.org/wiki/Vim_%28editor_di_testo%29

¹⁷ https://wiki.archlinux.org/index.php/Dnsmasq_%28Italiano%29

¹⁸ <https://help.ubuntu.com/community/Debmirror>

¹⁹ “Installing and Scaling out Ubuntu Enterprise Cloud in Virtual Environment”
Zoran Pantić and Muhammad Ali Babar
University of Copenhagen, Denmark, 2012

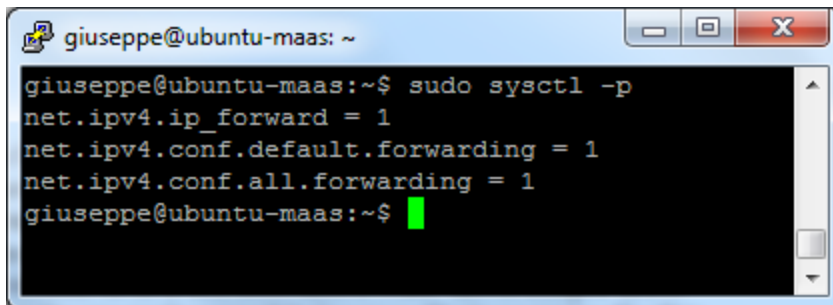
```
$ sudo iptables -t nat --append POSTROUTING --out-interface eth0 -j MASQUERADE
$ sudo iptables --append FORWARD --in-interface tun0 -j ACCEPT
$ sudo bash -c 'echo 1 > /proc/sys/net/ipv4/ip_forward'
```

All'interno del file di configurazione

- `/etc/sysctl.conf`
 1. de-commento la riga `"net.ipv4.ip_forward = 1"`
 2. aggiungo in coda `"net.ipv4.conf.default.forwarding=1"`
`"net.ipv4.conf.all.forwarding=1"`

```
$ sudo sysctl -p
```

Il risultato sarà questo:

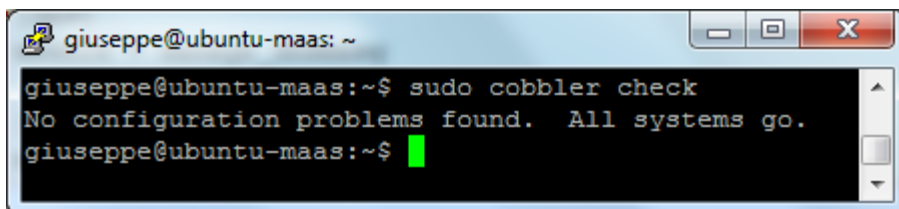


```
giuseppe@ubuntu-maas: ~
giuseppe@ubuntu-maas:~$ sudo sysctl -p
net.ipv4.ip_forward = 1
net.ipv4.conf.default.forwarding = 1
net.ipv4.conf.all.forwarding = 1
giuseppe@ubuntu-maas:~$
```

Al termine, riavvio la rete e il sistema:

```
$ sudo /etc/init.d/cobbler restart
$ sudo cobbler sync
$ sudo reboot
```

Al riavvio faccio il check della configurazione. Se tutto è andato a buon fine uscirà questo messaggio da parte del sistema:



```
giuseppe@ubuntu-maas: ~
giuseppe@ubuntu-maas:~$ sudo cobbler check
No configuration problems found. All systems go.
giuseppe@ubuntu-maas:~$
```

Ultima fase della configurazione della rete prevede l'installazione del DHCP gestito da MAAS:

```
$ sudo apt-get install maas-dhcp
```

Questa fase prevede la scelta:

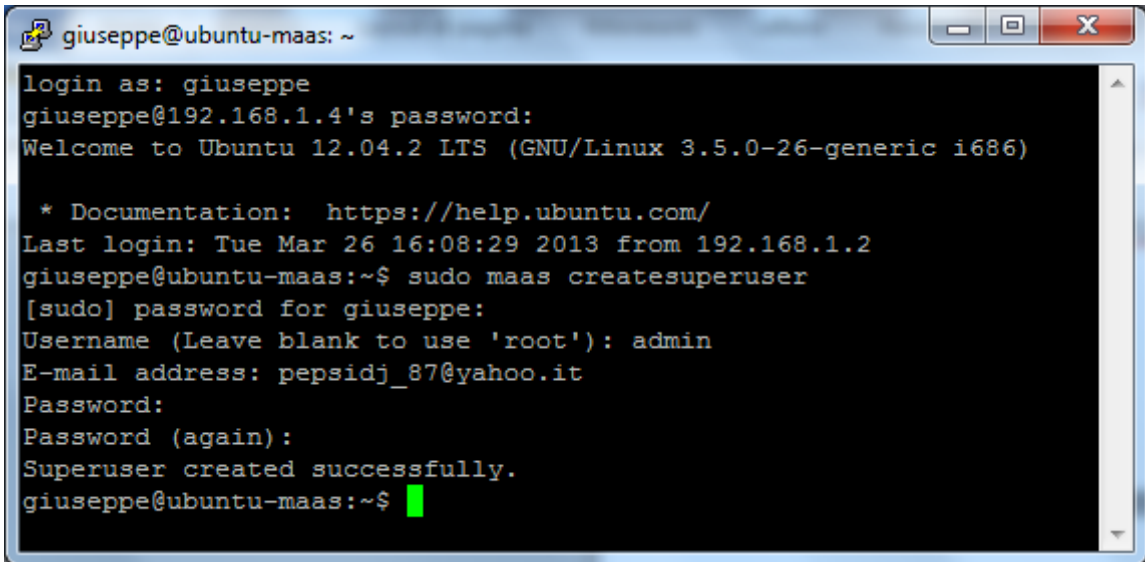
- Di un range di indirizzi IP all'interno del quale MAAS potrà registrare i Nodi. Nel mio caso, fissato `192.168.1.4` l'indirizzo IP del server principale, scelgo un range di valori compresi tra `192.168.1.5` e `192.168.1.200`
- Del Gateway di default per i client DHCP. Imposto il valore a `192.168.1.4`
- Dell'eventuale nome di dominio per i client DHCP. Essendo il Cloud gestito in locale, imposto il valore come `"localdomain"`.

Il sistema si sincronizza. Successivamente verrà riavviato.

La configurazione preliminare del sistema è terminata. Non resta che:

- Creare il superuser di MAAS on la seguente istruzione:

```
$ sudo maas createsuperuser
```



```
giuseppe@ubuntu-maas: ~
login as: giuseppe
giuseppe@192.168.1.4's password:
Welcome to Ubuntu 12.04.2 LTS (GNU/Linux 3.5.0-26-generic i686)

 * Documentation:  https://help.ubuntu.com/
Last login: Tue Mar 26 16:08:29 2013 from 192.168.1.2
giuseppe@ubuntu-maas:~$ sudo maas createsuperuser
[sudo] password for giuseppe:
Username (Leave blank to use 'root'): admin
E-mail address: pepsidj_87@yahoo.it
Password:
Password (again):
Superuser created successfully.
giuseppe@ubuntu-maas:~$
```

- Importare le immagini ISO sul server MAAS che serviranno per l'installazione dei Nodi (l'operazione richiede un po' di tempo, a seconda della velocità della rete a disposizione, visto che andrà a scaricare circa 800MB di immagini):

```
$ sudo maas-import-isos
```

Aggiungere i Nodi al Cloud

A questo punto tramite interfaccia Web si potrà accedere al Dashboard di MAAS.

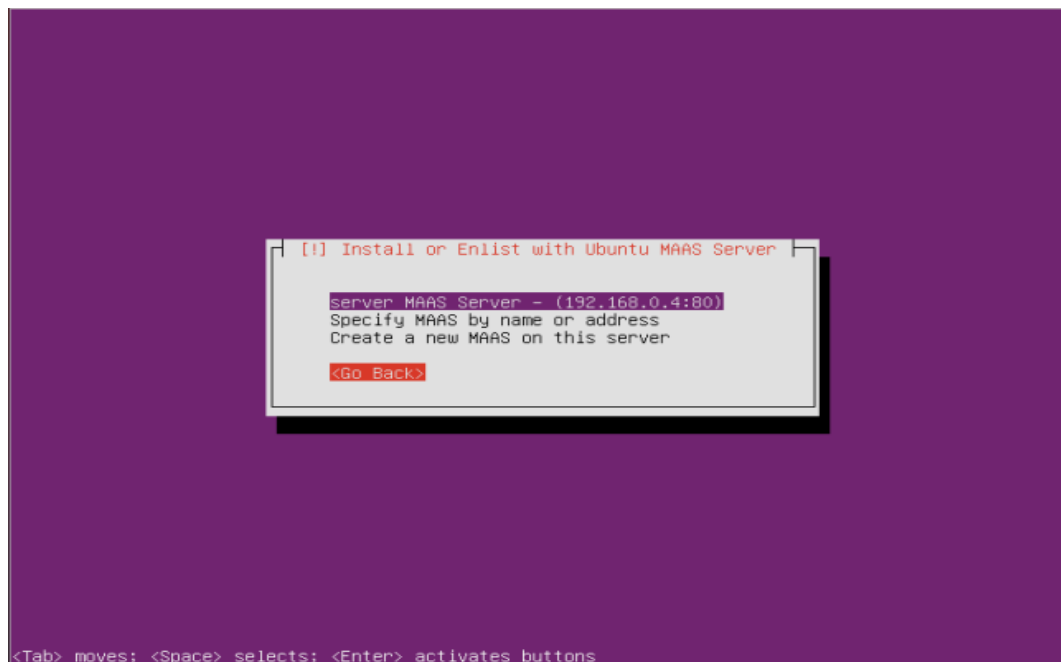
I prossimi passi da effettuare sono:

- Accedere come superuser e creare una nuova utenza personale impostandola come non "admin".
- Accedere con l'utenza appena creata.
- Far eseguire la lettura dei Nodi da parte del server MAAS.

Per i primi due punti non dovrebbe esserci alcuna difficoltà. La creazione del nuovo utente può essere fatta accedendo alla pagina delle impostazioni dell'amministratore e selezionare "+ Add user". Niente di più.

Per quanto riguarda l'aggiunta dei Nodi, questa fase può essere svolta in diverse modalità.

1. Avviare ogni nodo con la distribuzione di Ubuntu utilizzata per l'installazione del Server MAAS.
2. Arrivati al punto indicato dall'immagine, abbiamo ora due diverse possibili scelte:
 - a. Impostare esplicitamente la posizione del Server MAAS tramite indirizzo IP selezionando la voce "Specify MAAS by name or address"
 - b. Selezionare il Server che ci comparirà come prima opzione una volta che il sistema, che risiede sulla stessa LAN, avrà individuato automaticamente la posizione



3. Oppure procedere tramite interfaccia Web. In questo caso è necessario innanzitutto assicurarsi che si sia loggati come utente e non come amministratore e successivamente nella schermata principale del Dashboard selezionare “+Add node” per visualizzare la seguente schermata:

Add node

Mac address

e.g AA:BB:CC:DD:EE:FF

+ Add additional MAC address

Architecture

i386

Hostname (optional)

Default is MAC-based, e.g. "node-aabbccddeeff"

Cancel Add node

Avanzate

Tipo di scheda: PCnet-FAST III (Am79C973)

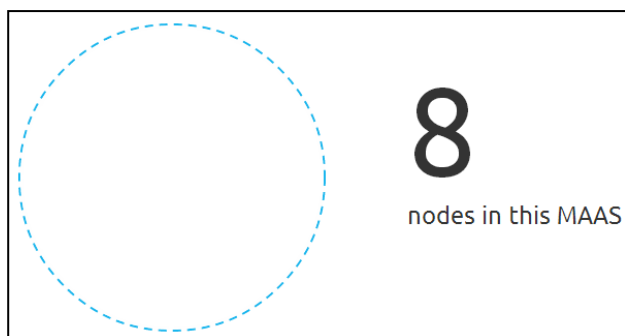
Modalità promiscua: Permetti tutto

Indirizzo MAC: 08002745DA9A

☒ Cavo connesso

Come si può notare dal rettangolo segnato in rosso, va inserito il MAC-Address del nodo da inserire, scegliere eventualmente il tipo di architettura da installare, se a 32 bit o 64 bit, e infine il nome Host che per default prendere il nome del MAC-Address stesso privo di interpunzioni, seguito dalla dicitura “*local*”, per indicare, successivamente, al DNS che vogliamo accedere a un dominio locale. Il MAC-Address di ogni nodo è disponibile nella sezione di configurazione della rete di ogni macchina virtuale creata in Virtual Box, nelle impostazioni avanzate.

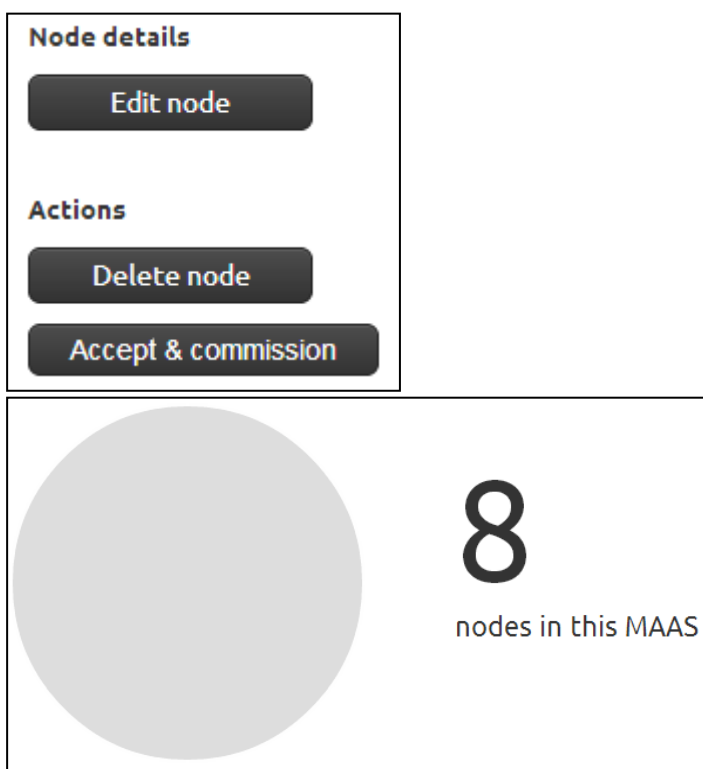
Terminata l'installazione di ogni nodo a disposizione, il risultato sarà uguale a quanto indicato nell'immagine riportata di seguito.



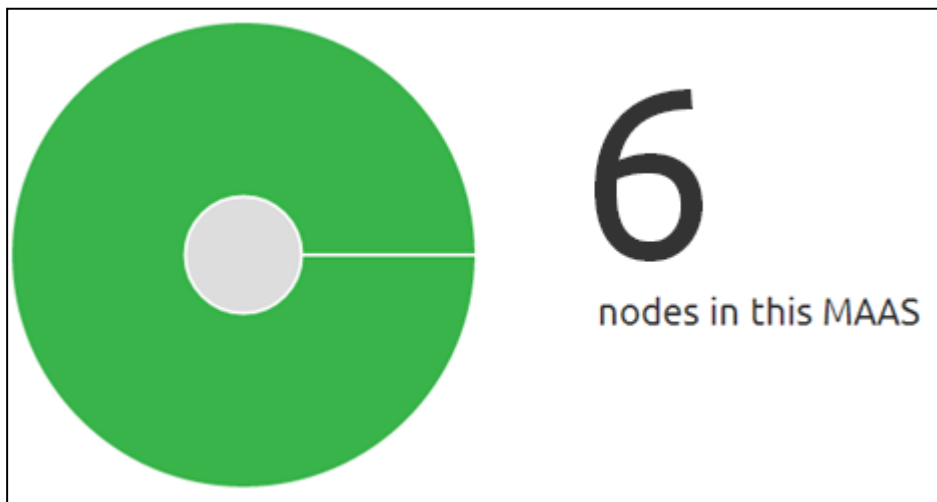
Evito a questo punto di riscrivere il significato della schermata tratteggiata visto che nei capitoli precedenti ho illustrato dettagliatamente il significato di ogni singolo stato che un nodo può assumere.

A questo punto, avendo effettuato l'accesso, e di conseguenza registrato i Nodi con un'utenza non da amministratore, è necessario effettuare il login come tale per procedere alla fase che prevedere di accettare i Nodi e commissionarli.

Per ogni nodo, accedendo all'apposita sezione relativa alla manutenzione, si procede al "commission" tramite apposito pulsante. Il risultato è riportato di seguito nella rappresentazione sulla destra.



L'ultima fase consiste nel fare in modo che il sistema, una volta riconosciuti i Nodi, li prepari in modo tale che siano disponibili per il Deploy delle applicazioni su di essi. Per far ciò è necessario avviare (non per forza contemporaneamente) tutte le macchine virtuali di VirtualBox assegnate ai Nodi in modalità Boot di rete. Come spiegato precedentemente, interverrà il PXE che contatterà il Server tramite LAN per far riconoscere ogni nodo a MAAS, e portarlo nello stato di "Ready". Le varie fasi di vita, inclusi i cambiamenti di stato di ogni nodo, possono essere monitorati in real-time nell'apposita sezione del Dashboard. E infine ecco il risultato: i Nodi commissionati, nello stato Ready, e pronti ad essere presi in consegna da MAAS per il Deploy delle applicazioni.



NB: Come si può notare in maniera visuale, durante il riconoscimento dei Nodi, qualche errore non previsto ha fatto sì che 2 degli 8 Nodi fallissero il processo. I Nodi non riconosciuti sono i quelli evidenziati dal colore grigio, quelli invece verdi sono i Nodi correttamente installati e pronti all'utilizzo. E' buona norma pertanto, creare un numero sufficientemente grande di Nodi (ovviamente relativamente alle nostre esigenze, ad esempio a me ne servono 4 all'incirca, ma prevedendo qualche sorta di contrattempo li ho raddoppiati assicurandomene comunque un buon numero funzionante) in previsione di imprevisti di qualunque genere.

Juju - Il Deploy dei charms

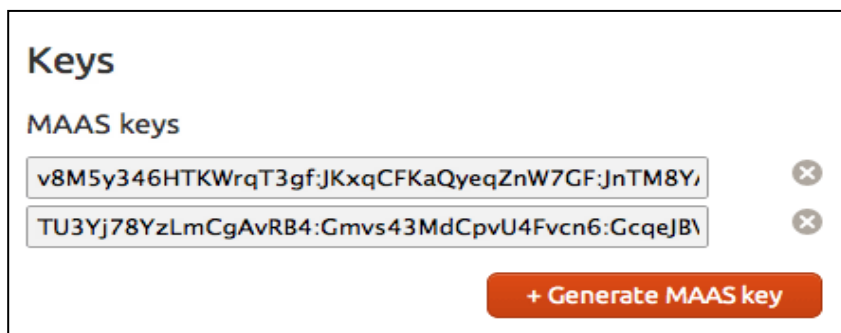
Il sistema a questo punto è pronto per supportare il Deploy delle nostre applicazioni. In questa fase si può notare, in maniera più evidente, come MAAS riesce a reggere e sincronizzare in maniera molto semplice qualunque tipo di operazione si desidera.

Questa è la parte più interessante, l'interazione con Juju. Ovviamente il tutto richiede un preventivo aggiornamento e configurazione del Server MAAS.

Innanzitutto bisogna procurarsi il pacchetto di Deploy di Juju con tanto di aggiornamento dei pacchetti critici necessari al funzionamento, come ad esempio le ultime librerie (compilatore incluso) di Python. Infine bisogna creare una cartella destinata ad ospitare l'*environment* di Juju. Da linea di comando:

```
$ sudo apt-get install python-software-properties -y
$ sudo add-apt-repository ppa:juju/pkg && sudo apt-get update
$ sudo apt-get install juju -y
$ mkdir ~/.juju
```

Ora accedo, tramite interfaccia Web, alla pagina delle preferenze della mia utenza e genero tante MAAS-keys quanti sono gli ambienti (i Nodi) destinati ad ospitare applicazioni fornite da Juju. Basta semplicemente intervenire sul bottone "+ Generate MAAS key":



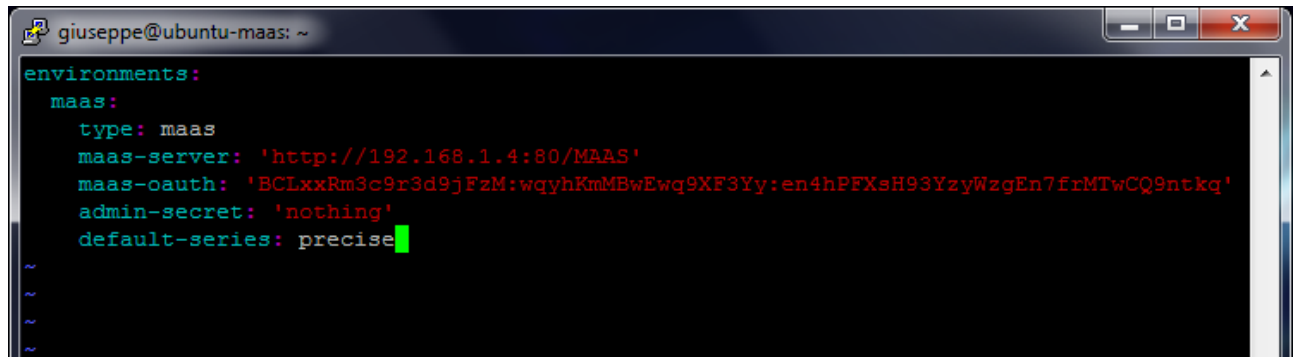
E' fondamentale creare le chiavi in maniera parallela al numero di Nodi, in quanto in fase di Deploy, Juju automaticamente rileverà le chiavi e setterà i parametri di sicurezza per l'accesso alle singole macchine,

sincronizzandole con i valori di sicurezza del server MAAS.

Una prima chiave sarà quella destinata a creare il file di “*environment*”. Un file di *environment* altro non è che un descrittore di Deploy. In esso vengono definite le specifiche relative alla configurazione del Cluster, all’accesso al Server MAAS e alla gestione del repository e delle chiavi di accesso. In fase di interazione con MAAS, Juju interrogherà di continuo il Cluster e quindi il relativo file di *environment*, per autenticarsi e avere accesso alla lista dei Nodi disponibili per il Deploy e quindi renderli disponibili alle nostre esigenze.

Il file di *environment* va generato nella cartella di default di Juju, ovvero quella appena creata. Utilizzando un editor di testo qualsiasi:

```
$ sudo vim ~/.juju/environments.yaml
```



```
giuseppe@ubuntu-maas: ~
environments:
  maas:
    type: maas
    maas-server: 'http://192.168.1.4:80/MAAS'
    maas-oauth: 'BCLxxRm3c9r3d9jFzM:wqyhKmMBwEwq9XF3Yy:en4hPFxsH93YzyWzgEn7frMTwCQ9ntkq'
    admin-secret: 'nothing'
    default-series: precise
```

Ecco il file di *environment* da creare. Le informazioni contenute in esse sono abbastanza auto esplicative.

In ogni caso:

- *type*: descrive il tipo di contesto scelto. I contesti a disposizione sono molteplici ma nel caso in cui si sta procedendo allo sviluppo di un Cloud privato indicheremo come target del contesto “MAAS” stesso. Il discorso cambia quando il Cloud da implementare è di tipo pubblico. In questo caso la dicitura corretta è “*EC2*”²⁰ per indicare un Cloud distribuito in rete con tipi di indirizzi pubblici proiettati sulla rete Web.
- *maas-server*: indirizzo del nostro server MAAS (vale inserire sia l’Hostname che l’indirizzo ip).
- *maas-oauth*: maas-key precedentemente generata tramite interfaccia per l’accesso al Cluster.
- *admin-secret*: chiave di sicurezza dell’amministratore.
- *default-series*: repository di default dei nostri servizi Juju.

Quando ci si deve connettere molto spesso ad un server (o a molti server), può essere fastidioso dover inserire ogni volta la password. Un modo sicuro per aggirare questo problema è basato su SSH²¹, ovvero 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. Esse risolvono il problema degli attacchi di tipo *brute-force* sulle password rendendoli impraticabili.

Pertanto, a meno che non ne esista già una, per collegare Juju all’istanza di *environment* appena creata, bisogna generare una coppia di chiavi SSH.

```
$ sudo ssh-keygen
```

²⁰ Elastic Compute Cloud. E’ un servizio Web di Amazon che offre un servizio di Hosting di indirizzi ip pubblici, destinato proprio al mondo del Cloud. <http://aws.amazon.com/ec2/>

²¹ Secure Shell

```

giuseppe@ubuntu-maas: ~
giuseppe@ubuntu-maas:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/giuseppe/.ssh/id_rsa):
Created directory '/home/giuseppe/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/giuseppe/.ssh/id_rsa.
Your public key has been saved in /home/giuseppe/.ssh/id_rsa.pub.
The key fingerprint is:
3a:87:69:fa:be:22:59:4a:e2:e0:ef:be:c1:e2:84:02 giuseppe@ubuntu-maas
The key's randomart image is:
+--[ RSA 2048 ]-----+
|
|
|
|
|E      S
|= o . +
|*= * * .
|+o= oo o
| .+*o++.
+-----+
giuseppe@ubuntu-maas:~$

```

Lancio infine l'istanza di Juju sul Cluster mediante un'operazione di bootstrapping²²:

```
$ juju bootstrap
```

```

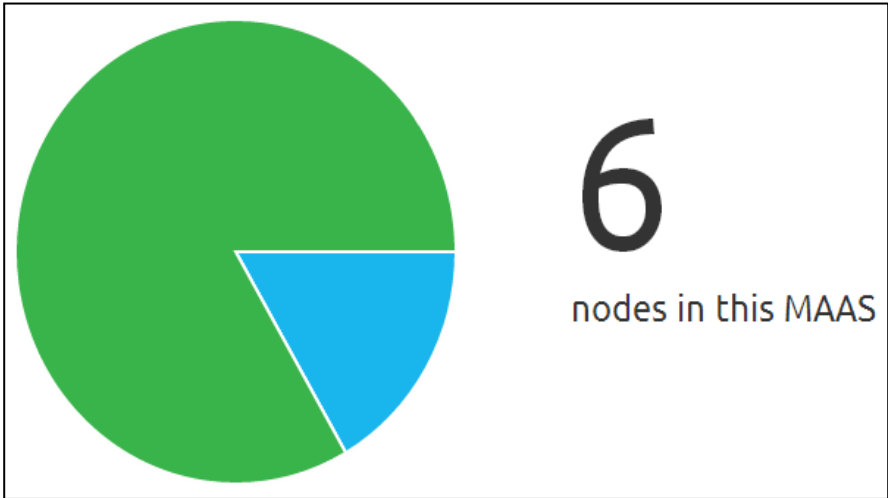
giuseppe@ubuntu-maas: ~
<div class="clear"></div>
</div>
</div>
<div id="footer">
  
  <p>&copy; 2012 Canonical Ltd. Ubuntu and Canonical are registered trademarks of Canonical Ltd.
  <a class="space-left-small" href="https://wiki.ubuntu.com/ServerTeam/MAAS/" target="_blank">View Documentation</a></p>
  <div class="clear"></div>
</div>
</div>
</div>
</body>
</html>

2013-03-27 19:40:09,300 INFO Bootstrapping environment 'maas' (origin: distro type: maas)...
2013-03-27 19:40:13,197 INFO 'bootstrap' command finished successfully
giuseppe@ubuntu-maas:~$

```

Questo è il risultato della procedura di Bootstrap da linea di comando: viene letto il file di *environment* e ne viene fatto il Bootstrap. Questa procedura è da eseguire una sola volta, esclusivamente per il Cluster.

²² Insieme di processi che vengono eseguiti da un computer durante la fase di avvio - <http://it.wikipedia.org/wiki/Boot>



Uno dei 6 Nodi è stato allocato. Questo è il risultato grafico visibile a partire dal Dashboard di MAAS. Il nodo viene allocato per l’utenza che in quel momento ha accesso al sistema. Quindi bisogna assicurarsi di non accedere con quella da amministratore.

MAC	Status
08:00:27:9f:b9:a3 (node-0800279FB9A3.local)	Ready
08:00:27:5e:9f:dc (node-0800275E9FDC.local)	Ready
08:00:27:45:da:9a (node-08002745DA9A.local)	Ready
08:00:27:6e:e8:41 (node-0800276EE841.local)	Ready
08:00:27:e0:12:20 (node-080027E01220.local)	Ready
08:00:27:52:86:45 (node-080027528645.local)	Allocated to giuseppe

Questa invece è la situazione attuale dei Nodi. A questo punto, individuata la macchina virtuale a cui fa riferimento il MAC-Address del nodo allocato, da VirtualBox bisogna avviarlo in modalità PXE. Parte ora il processo più lungo. Deve essere allocato il Cluster, il server di

maggior importanza dopo il Server MAAS, il server dedicato ad orchestrare le applicazioni Juju sui Nodi.

Giunti alla fine dell’installazione del Cluster, il risultato su console della nuova macchina, sarà il seguente:

```
node-080027528645 login: 2013-03-27 15:38:41,790 INFO Initializing zookeeper hierarchy
2013-03-27 15:38:41,861 INFO 'initialize' command finished successfully
juju-machine-agent start/running, process 3521
juju-provision-agent start/running, process 3524
ec2:
ec2: #####
ec2: -----BEGIN SSH HOST KEY FINGERPRINTS-----
ec2: 1024 ce:12:f3:bc:08:74:a9:28:2d:c2:fa:01:9f:a0:98:d5 root@node-080027528645 (DSA)
ec2: 256 61:61:05:13:1b:d5:07:2b:48:75:bd:53:ff:95:1b:67 root@node-080027528645 (ECDSA)
ec2: 2048 2f:74:0a:36:00:c5:5d:4a:62:cb:e2:4a:f7:80:72:40 root@node-080027528645 (RSA)
ec2: -----END SSH HOST KEY FINGERPRINTS-----
ec2: -----BEGIN SSH HOST KEY KEYS-----
ec2: #####
ecdsa-sha2-nistp256 AAAAE2VjZHNhLXNoYTItbGlzdHAyNTYAAAAIbGlzdHAyNTYAAABBBBONhhy
I+k/Uv8qHwCaEK9thuceGFUTJEKx17+GUZeYV+ryCTcBEyx6Wveb2CJ9S/C6/yL3N4+5uG+rWps?lQ=
root@node-080027528645
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDAQMjoozuQDYYiWb8+scaUCnXMggM1TWG/az460m7043
bCsp7k+4snkGmORc7tc429zaXY7jX818bUEdEyzepB7txdPNN3LG+RKDQM5oz5mDLNVq/n4HnUQ03uFq
PX9TaWAdfob6L7AAu5EetmleJIWZ1WUvKSQ8dtbFpUeSrn93eJea2w7PuQoDa94utP2o2Wkcra/W6reG
djuJ7U3fy2KkUgLwNGMc6MXnwaTPSX8Ve0fKXJVTosktSBymPRT1UXLTWU23tcoLxN3XWbL5jJS4200K
wiHsDx/C7xf5U7+UpS26d0FzJAp6xaKiMM6bxgsB0gW/M6FDfUYh6q0d6Fd root@node-080027528645
-----END SSH HOST KEY KEYS-----
cloud-init boot finished at Wed, 27 Mar 2013 19:38:42 +0000. Up 443.97 seconds
```

Le chiavi SSH vengono fatte rimbalzare da Server MAAS a Cluster, vengono accettate e vengono stampate a video. Al termine dell’operazione, il messaggio “Cloud-init Boot finished...” ci segnala l’avvenuto termine del processo di riconoscimento e comunicazione tra le due macchine, con esito positivo. Non resta, a questo punto, che verificarne la comunicazione. Per far ciò bisogna dirigersi

nuovamente sul terminale del Server principale e lanciare un segnale di connessione a Juju per visualizzare l’attuale stato dell’arte e l’avanzamento della configurazione e creazione del nostro Cloud. Da linea di comando:

```
$ juju status
```

Il comando qui riportato (da eseguire sul server MAAS) ricorrerà spesso d'ora in avanti. Tramite questa istruzione è possibile monitorare l'avanzamento e lo stato attuale del sistema. Ecco infine, come si presenta l'ambiente:

```
giuseppe@ubuntu-maas:~$ juju status
2013-03-27 20:52:23,960 INFO Connecting to environment...
The authenticity of host 'node-080027528645.local (192.168.1.46)' can't be established.
ECDSA key fingerprint is 61:61:05:13:1b:d5:07:2b:48:75:bd:53:ff:95:1b:67.
Are you sure you want to continue connecting (yes/no)? yes
2013-03-27 20:52:32,623 INFO Connected to environment.
machines:
  0:
    agent-state: running
    dns-name: node-080027528645.local
    instance-id: /MAAS/api/1.0/nodes/node-940e8dde-963f-11e2-bbae-080027e215fc/
    instance-state: unknown
services: {}
2013-03-27 20:52:32,689 INFO 'status' command finished successfully
giuseppe@ubuntu-maas:~$
```

Il server MAAS si connette al Cluster tramite l'*environment* e ne ricava informazioni. La "macchina 0", che ospita il Cluster è attualmente in esecuzione. E' possibile monitorare i servizi attivi sulle macchine accanto alla voce "services", che ovviamente per ora, risulta vuota.

Termina la fase di configurazione.

L'ultima parte presenterà il Deploy dei servizi.

Il Deploy dei servizi

In precedenza abbiamo parlato del "Juju Charms Store".

Visitando il sito e accedendo all'area dei servizi disponibili, ci si trova di fronte a un'ampia scelta di applicazioni di uso sia domestico ma anche professionale. Questa molteplicità di prodotti è necessaria per soddisfare qualunque richiesta di Cloud a qualunque livello di pubblicazione e per qualunque tipologia di utente. Come si vedrà di seguito, lo "sforzo" maggiore sta nella scelta di ciò che effettivamente abbiamo bisogno e vogliamo mettere a disposizione nel nostro Cloud, in quanto il Deploy dei servizi in sé, risulta estremamente semplice e si può portare a termine in pochi passaggi.

Le operazioni principali di Juju, in fase di Deploy e produzione, sono essenzialmente 3:

- **Il Deploy.**

Deployare un'applicazione significa essenzialmente implementarla all'interno dell'ambiente di sviluppo, soddisfarne le dipendenze e costruire una struttura hardware/software di contorno che la renda robusta, sicura e flessibile ai cambiamenti allo stesso tempo.

Le fasi di un Deploy accurato prevedono:

- a. L'analisi dei requisiti funzionali per capire l'effettivo utilizzo di cui abbiamo bisogno e del come renderlo fruibile.
- b. La scelta del software che soddisfa i nostri obiettivi.
- c. Lo scaricamento/implementazione del servizio nel nostro contesto di lavoro.

- **Le relazioni tra unità.**

Punto forte di Juju, relazionare diversi servizi equivale a tutti gli effetti a coordinarli ed orchestrarli. Questo significa che effettivamente un servizio in sé è poca cosa se non ben supportato e ben fornito di tutte le utility e dei requisiti funzionali richiesti. A tal proposito, in brevi passaggi, Juju permette di specificare delle relazioni più o meno semplici, rimuoverle, modificarle e/o duplicarle.

- **L'esposizione del servizio in rete.**

Una volta terminata la fase di implementazione coordinamento il gioco è fatto. Non resta che segnalare a Juju che il nostro servizio in Cloud è pronto per essere proiettato sulla rete del Web. In sé, in fase di implementazione, il servizio già risiede completamente nella macchina ospitante, ed è a tutti gli effetti funzionante. L'esposizione a questo punto completa il processo aprendo le porte della rete per permettere di raggiungere il servizio e il suo front-end semplicemente attraverso un comunissimo browser.

Ovviamente, detto questo, non bisogna omettere il particolare secondo il quale ogni servizio necessita di vincoli e requisiti propri, non solamente inter relazionali con altre unità: questo significa che ogni servizio gode di proprietà proprie e di comandi esclusivi, comandi "ad hoc" per l'applicazione, che permettono di farne il setup completo e portare a termine in maniera efficiente il processo di configurazione.

Un esempio classico è il settaggio delle password iniziali dei super utenti: come è successo per il server MAAS, per il quale si è dovuto preventivamente creare un superuser, in maniera altrettanto analoga, si verificherà per determinate applicazioni, la necessità di impostare password, opzioni di trasferimento dei dati, tunneling dei protocolli, ecc. attraverso la Shell di Linux.

C'è stato bisogno pertanto di scegliere degli applicativi, che in maniera esemplare, replicassero perfettamente le tre fasi di Juju poc'anzi descritte, proprio perché il lavoro svolto è da intendersi anche come un'utile tutorial del prodotto Cloud di Ubuntu.

Ed ecco i 3 prodotti scelti:

- MySQL
- PhpMyAdmin
- Wordpress

Una brevissima panoramica.

1. **MySQL²³**

Il database Open Source più conosciuto al mondo. Adatto a qualunque tipo di necessità, MySQL è utilizzato e diffuso in tutto il mondo sia in ambito privato, che didattico, che aziendale. E' un *RDBMS*²⁴, composto da un client con interfaccia a riga di comando e un server, entrambi disponibili sia per sistemi Unix che per Windows.

2. **PhpMyAdmin²⁵**

E' un'applicazione PHP gratuita che consente di amministrare in modo semplificato database MySQL tramite un qualsiasi browser. L'applicazione è indirizzata sia agli amministratori del database, che agli utenti. Permette di creare un database da zero, creare le tabelle ed eseguire operazioni di ottimizzazione sulle stesse. Prevede delle funzionalità per l'inserimento dei dati (popolazione del database), per le query, per il backup dei dati, ecc. L'amministratore, ha inoltre a disposizione un'interfaccia grafica per la gestione degli utenti e dei loro permessi sul database.

3. **Wordpress²⁶**

WordPress è una piattaforma software di "*personal publishing*"²⁷ sviluppata in PHP e che usa come database MySQL. Consente la creazione di un sito internet formato da contenuti testuali o multimediali, facilmente gestibili ed aggiornabili. E' distribuito con la licenza GNU²⁸.

Procediamo quindi con l'installazione dei servizi menzionati.

Questo può avvenire in due modi differenti:

²³ <http://www.mysql.it/> - <http://it.wikipedia.org/wiki/MySQL>

²⁴ Relational database management system - http://it.wikipedia.org/wiki/Relational_database_management_system

²⁵ <http://it.wikipedia.org/wiki/PhpMyAdmin>

²⁶ <http://it.wikipedia.org/wiki/WordPress>

²⁷ <http://it.wikipedia.org/wiki/Blog>

²⁸ General Public License - http://it.wikipedia.org/wiki/GNU_General_Public_License - <http://www.gnu.org/>

- Accedendo ai singoli charm direttamente in modalità online dal Juju Store: in questo caso di esegue il download, trasparente all'utente, in una cartella temporanea. Basterà digitare da riga di comando:

```
$ juju deploy mysql
$ juju deploy phpmyadmin
```

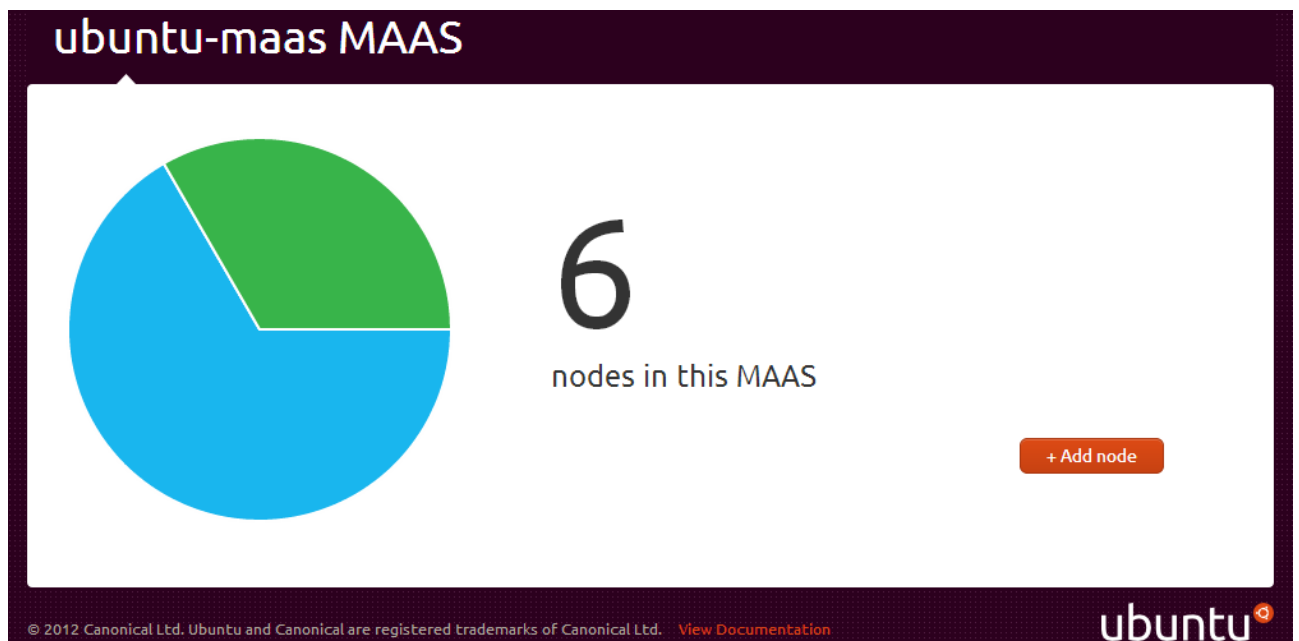
- Creare un repository locale sul proprio server MAAS per conservare i charms e accedervi nuovamente nel caso in cui si voglia procedere una seconda volta ad essi. In questo caso le fasi previste, richiedono la creazione di una cartella, cifrarla se lo si desidera in modo tale da potervi accedere solo come amministratore, scaricarvi le applicazioni e successivamente Deployarle facendo riferimento alla cartella che li contiene. In questo caso si procede nel seguente modo:

```
$ mkdir -p ~/charms/precise
$ cd ~/charms/precise
$ charm get wordpress
$ charm get mysql

$ chmod 600 ~/. charms/precise

$ juju deploy --repository=~/charms local:precise/wordpress
$ juju deploy --repository=~/charms local:precise/mysql
```

Più difficile a dirsi che a farsi. Le due istruzioni sono sufficienti a Deployare le nostre unità di servizio menzionate. La fase di Deploy dura il tempo necessario al download del charm e al Bootstrap automatico che segue. Di fatti, non ci sarà bisogno più di lanciare in maniera esplicita i comandi di Bootstrap come è stato fatto in precedenza per il Cluster. Juju automaticamente andrà ad allocare uno dei Nodi e genererà i codici di accesso. Oltre al messaggio di output sulla console, che indica il buon esito del Deploy, gli sviluppi del processo possono essere seguiti tramite interfaccia Web. Quando viene lanciato il comando precedente, lo stato dei Nodi descritti nel Dashboard di MAAS cambierà. Questo il risultato ottenuto:



Il numero di Nodi Deployato, come si può ben notare, è aumentato.

A questo punto analizziamo la medesima situazione da linea di comando. Per far ciò come al solito bisognerà interrogare, ed è un'operazione utilissima da fare in ogni fase e in ogni momento del ciclo di sviluppo dell'ambiente, il nostro Cluster mediante lo stesso comando utilizzato precedentemente.

Quindi da linea di comando:

```
$ juju status
```

L'output sarà il seguente:

```
giuseppe@ubuntu-maas: ~
2013-03-27 20:59:58,607 INFO 'status' command finished successfully
giuseppe@ubuntu-maas:~$ juju status
2013-03-27 21:00:34,932 INFO Connecting to environment...
2013-03-27 21:00:35,553 INFO Connected to environment.
machines:
  0:
    agent-state: running
    dns-name: node-080027528645.local
    instance-id: /MAAS/api/1.0/nodes/node-940e8dde-963f-11e2-bbae-080027e215fc/
    instance-state: unknown
  1:
    agent-state: not-started
    dns-name: node-080027E01220.local
    instance-id: /MAAS/api/1.0/nodes/node-b7b3f3dc-963f-11e2-bbae-080027e215fc/
    instance-state: unknown
  2:
    agent-state: not-started
    dns-name: node-0800276EE841.local
    instance-id: /MAAS/api/1.0/nodes/node-da3601ca-963f-11e2-bbae-080027e215fc/
    instance-state: unknown
services:
  mysql:
    charm: cs:precise/mysql-16
    relations:
      db-admin:
        - phpmyadmin
    units:
      mysql/0:
        agent-state: pending
        machine: 1
        public-address: null
  phpmyadmin:
    charm: cs:precise/phpmyadmin-3
    relations:
      db-admin:
        - mysql
    units:
      phpmyadmin/0:
        agent-state: pending
        machine: 2
        public-address: null
2013-03-27 21:00:35,695 INFO 'status' command finished successfully
giuseppe@ubuntu-maas:~$
```

Mi sono aiutato utilizzando dei rettangoli rossi per mettere in evidenza le zone di interesse.

Si nota subito che nella parte superiore dell'immagine sono descritte esattamente 3 macchine nel mio *environment* che vanno da 0 a 2.

Ogni macchina è descritta specificandone

- *Lo stato attuale* di ogni macchina, che ha valore duale: avviato, non avviato.
- *Il nome DNS*: è lo stesso Hostname assegnato a ciascun nodo in fase di installazione.
- *L' id dell'istanza*: ovvero un riferimento univoco al nodo.

Si può notare come la prima macchina è in esecuzione (e qui ci si sta riferendo al Cluster che è stato lasciato in esecuzione insieme, ovviamente, al Server MAAS).

Le restanti 2 macchine hanno uno stato che ne indica la presenza ma non il funzionamento: questo perché, una volta allocato, un nodo deve essere avviamento in modalità Boot PXE per poter eseguire l'installazione del sistema di base e del servizio annesso scelto precedentemente. A proposito di servizi, la sezione che precedente era vacante, adesso risulta essere avvalorata con i servizi Deployati un attimo fa.

Ancora qualcosa da notare:

- *Lo stato del servizio: "pending"* in questo caso, e tale resterà fino a che non si avvieranno e installeranno i Nodi e relativi servizi.
- Il *public-address*: per ora "null", sarà avvalorato solamente in fase di esposizione del servizio, in quanto tramite esso potremmo avere accesso al charm installato attraverso interfaccia Web.
- La sezione *"relations"*: contiene le relazioni di cui abbiamo parlato nella sezione precedente. In questo caso è facile intuire come sia stata impostata una relazione tra i due servizi Deployati, dato che ovviamente la rispettiva sezione di relation punta all'altra unità e viceversa.

E di seguito indico il comando utilizzato per collegare le due unità e creare, per l'appunto, la relazione tra esse. Niente di difficile:

```
$ juju add-relation phpmyadmin mysql
```

Il PXE Boot delle macchine Deployate, come noto, riceverà informazioni dal server MAAS su che tipo di installazione effettuare e sul servizio da implementare. A questo punto non resta che aspettare il termine dell'installazione che al più potrà durare all'in circa 30minuti, sicuramente un'operazione più rapida dell'installazione del server principale o del Cluster. I riferimenti temporali ovviamente sono soggetti alle prestazioni hardware assegnate alla macchina virtuale di riferimento.

La solita schermata, quella vista precedentemente dove compariva il processo di generazione delle chiavi SSH, comunicherà il termine dell'installazione avvenuto con successo. Non resta che eseguire ancora una volta il comando di visualizzazione dello stato dell' *environment* creato e il risultato sarà il seguente:

```

giuseppe@ubuntu-maas: ~
2013-03-27 21:30:45,063 INFO Connected to environment.
machines:
  0:
    agent-state: running
    dns-name: node-080027528645.local
    instance-id: /MAAS/api/1.0/nodes/node-940e8dde-963f-11e2-bbae-080027e215fc/
    instance-state: unknown
  1:
    agent-state: running
    dns-name: node-080027E01220.local
    instance-id: /MAAS/api/1.0/nodes/node-b7b3f3dc-963f-11e2-bbae-080027e215fc/
    instance-state: unknown
  2:
    agent-state: running
    dns-name: node-0800276EE841.local
    instance-id: /MAAS/api/1.0/nodes/node-da3601ca-963f-11e2-bbae-080027e215fc/
    instance-state: unknown
services:
  mysql:
    charm: cs:precise/mysql-16
    relations:
      db-admin:
        - phpmyadmin
    units:
      mysql/0:
        agent-state: started
        machine: 1
        public-address: node-080027E01220.local
  phpmyadmin:
    charm: cs:precise/phpmyadmin-3
    exposed: true
    relations:
      db-admin:
        - mysql
    units:
      phpmyadmin/0:
        agent-state: started
        machine: 2
        open-ports:
          - 80/tcp
        public-address: node-0800276EE841.local
2013-03-27 21:30:45,172 INFO 'status' command finished successfully
giuseppe@ubuntu-maas:~$

```

Ecco ciò che è cambiato:

- Le macchine, in alto, che precedentemente venivano segnalate come “*not-started*” sono passate nello stato di “*running*”, ovvero vengono riconosciute come funzionanti. Questo ovviamente è possibile perché esse sono in comunicazione col Cluster e col Server MAAS sulla stessa LAN, pertanto un semplice *ping*²⁹ da quest’ultima è sufficiente per riconoscerne lo stato attuale.
- I servizi Deployati da “*pending*”, ovvero in attesa, sono stati avviati insieme alle macchine che li contengono e pertanto sono pronti ad essere utilizzati.
- Si può notare che la sezione relativa al “*public-address*”, che precedentemente era impostata a “*null*”, ha acquisito automaticamente un valore: è l’indirizzo che, inserito nel browser, ci permette di avere accesso al charm tramite DNS. Ovviamente stiamo parlando di un Cloud privato che vive in una sottorete, pertanto non possiede un dominio pubblico. Nel caso in cui avessi seguito questa strada, il risultato sarebbe differente, l’indirizzo ip pubblico sarebbe del tipo: “*public-address: ec2-www-xxx-yyy-zzz.compute-1.amazonaws.com*”

L’immagine riporta anche un’altra sezione relativa alle “*open-ports*”. Questa sezione fa riferimento a quanto detto prima sulla fase di esposizione di un servizio. Ricordo che un charm, un servizio, per essere utilizzato deve essere esposto, ovvero proiettato sulla rete. Per far ciò, in qualunque fase che va dall’installazione dal Deploy del charm all’installazione della macchina che lo ospita, è possibile lanciare il comando che gestisce, per l’appunto, l’esposizione del servizio di Juju in rete.

²⁹ Packet internet grouper - <http://it.wikipedia.org/wiki/Ping>

E' sufficiente eseguire la riga di comando:

```
$ juju expose phpmyadmin
```

Il servizio sarà automaticamente allocato a una porta di accesso.

Il procedimento di installazione di Juju, dei suoi charms e di MAAS è terminato. Non resta che vedere il risultato del lavoro svolto. Per prima cosa, è bene ricordare che nonostante il Server MAAS svolga al tempo stesso il ruolo di Server DNS e che pertanto sarebbe possibile accedere al servizio mediante il suo Hostname, spesso tuttavia, l'interazione tra un sistema Windows e un sistema Linux in virtuale, impedisce di risolvere correttamente l'indirizzo dei nomi degli Host. Pertanto le uniche soluzioni disponibili prevedono la possibilità di accedere al servizio direttamente attraverso l'ip della macchina ospitante oppure affidarsi a un servizio esterno di gestione del DNS in ambiente Windows, come ad esempio *"No-ip"*³⁰ oppure *"DynDNS"*³¹. Per evitare di installare ulteriormente del software, che alla fine non vincola l'utilizzo del mio Cloud, ho deciso di sfruttare la prima opzione. Pertanto, recupero l'indirizzo ip relativo alla macchina che ospita il servizio di phpmyadmin semplicemente attraverso un *ping* da terminale Linux:

```
giuseppe@ubuntu-maas: ~
giuseppe@ubuntu-maas:~$ ping node-0800276EE841.local
PING node-0800276EE841.local (192.168.1.182) 56(84) bytes of data:
64 bytes from node-0800276EE841.localdomain (192.168.1.182): icmp_req=1 ttl=64 time=0.255 ms
64 bytes from node-0800276EE841.localdomain (192.168.1.182): icmp_req=2 ttl=64 time=0.313 ms
64 bytes from node-0800276EE841.localdomain (192.168.1.182): icmp_req=3 ttl=64 time=0.303 ms
64 bytes from node-0800276EE841.localdomain (192.168.1.182): icmp_req=4 ttl=64 time=0.298 ms
64 bytes from node-0800276EE841.localdomain (192.168.1.182): icmp_req=5 ttl=64 time=0.310 ms
64 bytes from node-0800276EE841.localdomain (192.168.1.182): icmp_req=6 ttl=64 time=0.319 ms

--- node-0800276EE841.local ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5001ms
rtt min/avg/max/mdev = 0.255/0.299/0.319/0.029 ms
giuseppe@ubuntu-maas:~$
```

La trasmissione è avvenuta con successo: ho ottenuto l'indirizzo desiderato.

Aperto un qualunque browser, si inserisce l'indirizzo ip ottenuto, seguito dal servizio richiesto.

Nel caso specifico, la sintassi da riportare nel browser è la seguente:

```
http://192.168.1.182/phpmyadmin
```

Il risultato è confortante. Immediatamente avviene il collegamento, il browser riporta una richiesta di autenticazione per accedere al servizio:

Autenticazione richiesta

Il server <http://192.168.1.182:80> richiede un nome utente e una password. Il server dichiara: phpMyAdmin Setup.

Nome utente:

Password:

Il nome utente è un valore di default proprio del charm. Molti dei charm presenti nello Store del sito condividono la medesima utenza, ovvero: *"juju-admin"*.

³⁰ <http://www.noip.com/>

³¹ <http://dyn.com/dns/>

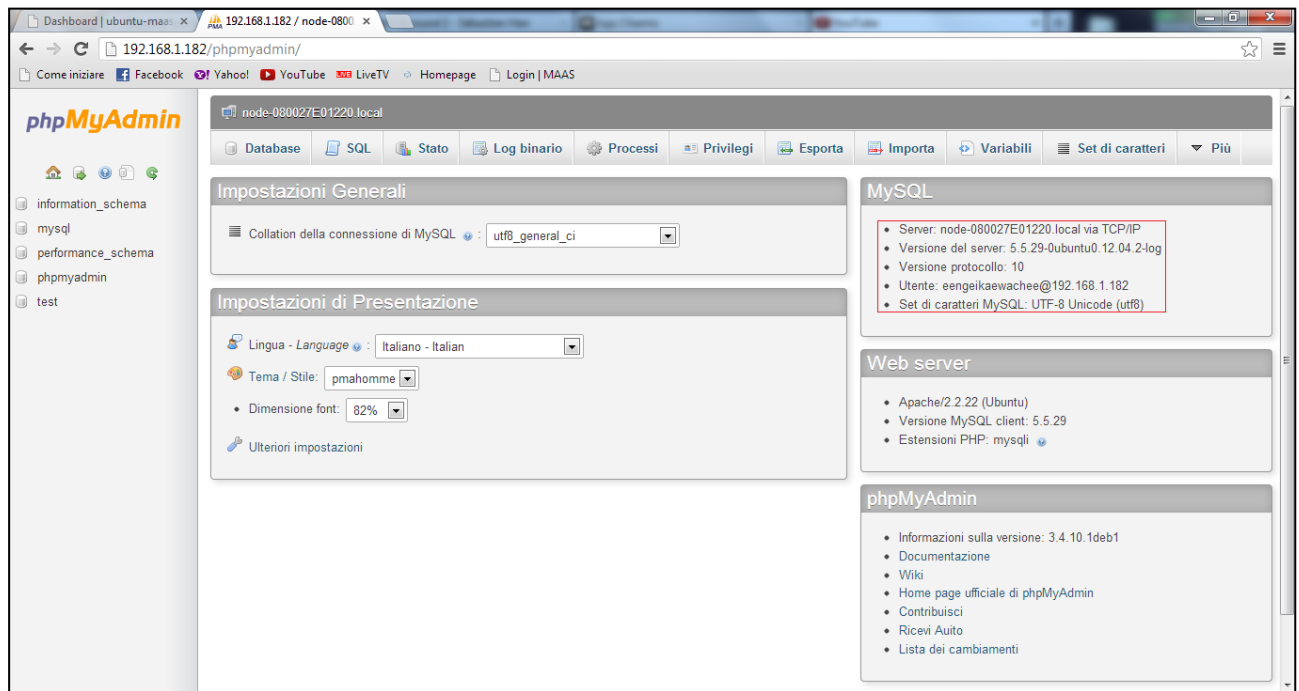
Per quanto riguarda invece la password, tocca a chi Deploya il servizio settarla correttamente con un valore personalizzato. Tornando sul terminale del server MAAS, bisogna eseguire la seguente sintassi:

```
$ juju set phpmyadmin password="password"
```

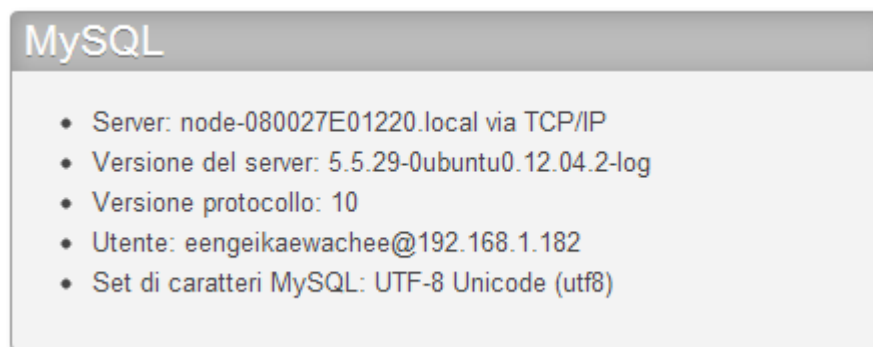
Inserisco i seguenti dati:

- Nome utente: *juju-admin*
- Password: *password*

Ecco il risultato finalmente:



Sulla destra sono riportate informazioni di carattere generale.



Nell'immagine ingrandita è riportato il server contenente MySQL che altro non è che il nodo che ospita il charm MySQL e non l'indirizzo del Cluster o del Server MAAS, attenzione.

Sulla sinistra invece vi sono i database a cui è possibile accedere. Sono esattamente quelli che troviamo comunemente sul database MySQL quando accediamo da linea di comando. La sezione dei database la vedremo tra poco, perché quello che mi interessa di più sta per venire, ovvero, l'interazione con Wordpress.

Per quanto riguarda Wordpress, niente di diverso da quanto già dettagliatamente descritto per il Deploy dei servizi di MySQL e PhpMyAdmin. Ovviamente esso utilizzerà il medesimo database già Deployato, visto che comunque l'obiettivo è verificarne la corretta interazione. Questa volta i comandi da impostare nel terminale sono:

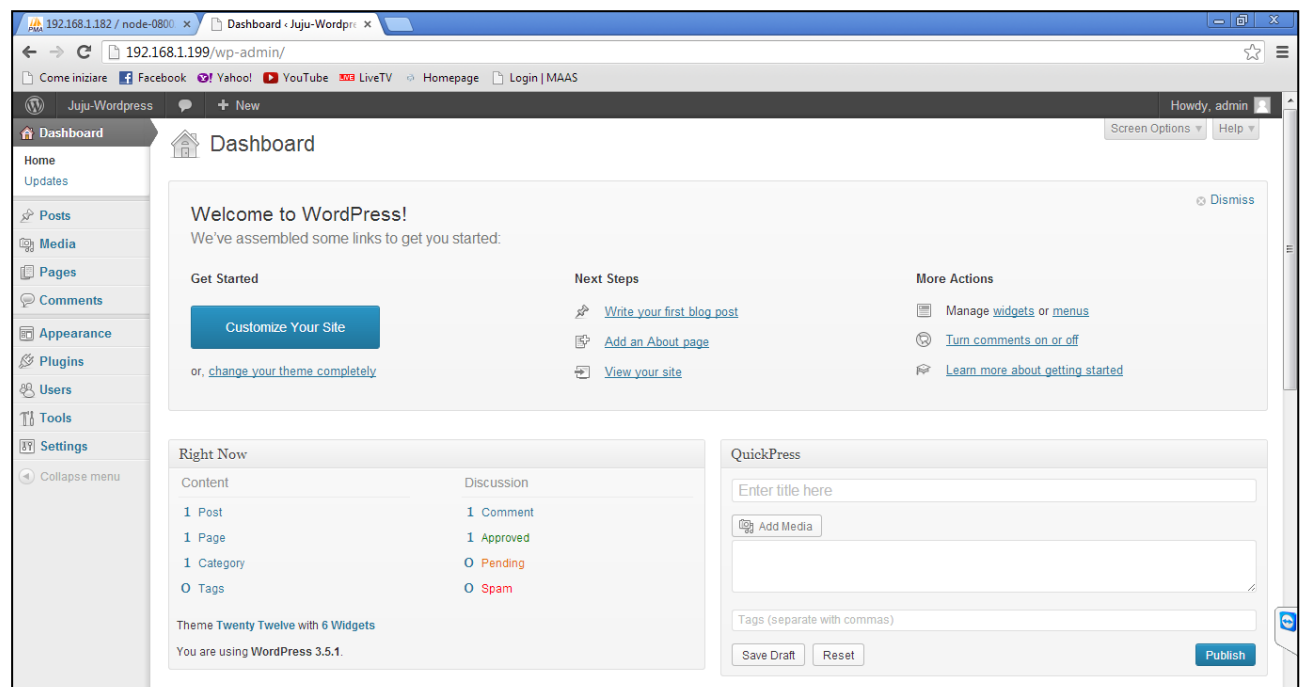
```
$ juju deploy wordpress
$ juju add-relation wordpress mysql
$ juju expose wordpress
```

Avviamo la visualizzazione dello status di Juju:

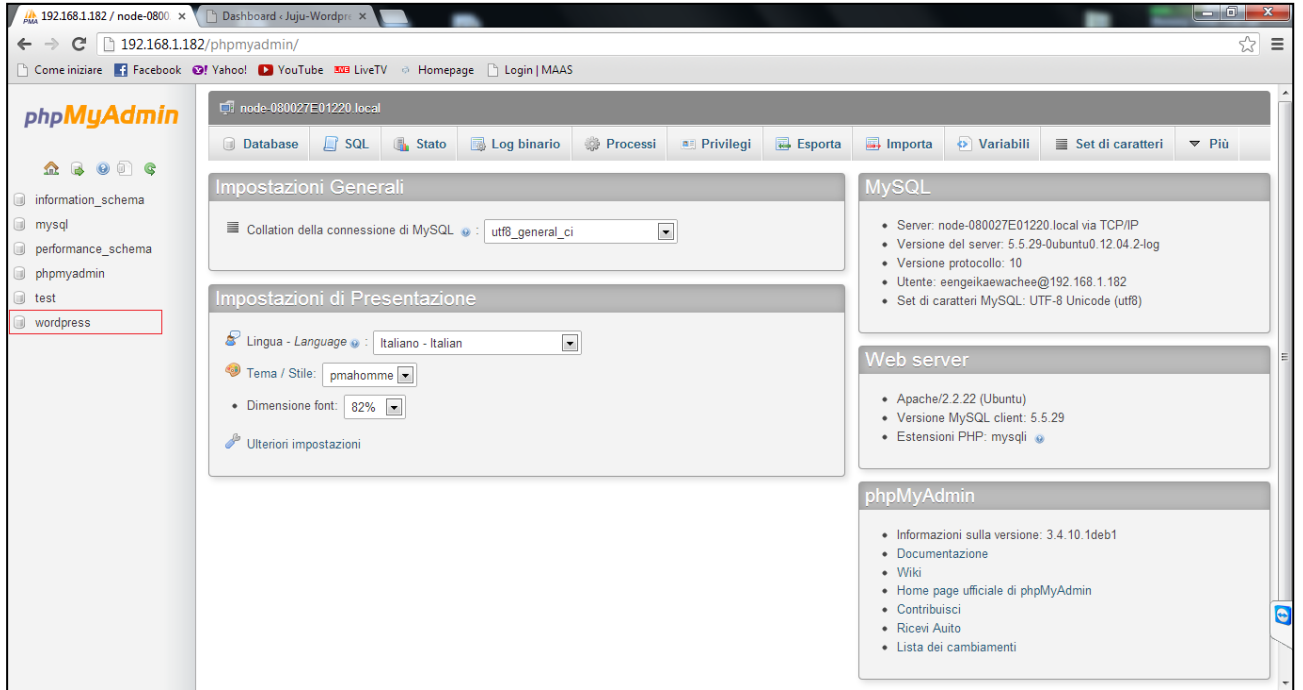
```
phpmyadmin:
  charm: cs:precise/phpmyadmin-3
  exposed: true
  relations:
    db-admin:
      - mysql
  units:
    phpmyadmin/0:
      agent-state: started
      machine: 2
      open-ports:
        - 80/tcp
      public-address: node-0800276EE841.local
wordpress:
  charm: cs:precise/wordpress-10
  exposed: true
  relations:
    db:
      - mysql
    loadbalancer:
      - wordpress
  units:
    wordpress/0:
      agent-state: started
      machine: 3
      open-ports:
        - 80/tcp
      public-address: node-08002745DA9A.local
2013-03-29 12:18:34,482 INFO 'status' command finished successfully
giuseppe@ubuntu-maas:~$
```

Wordpress è stato Deployato ed è attualmente attivo.

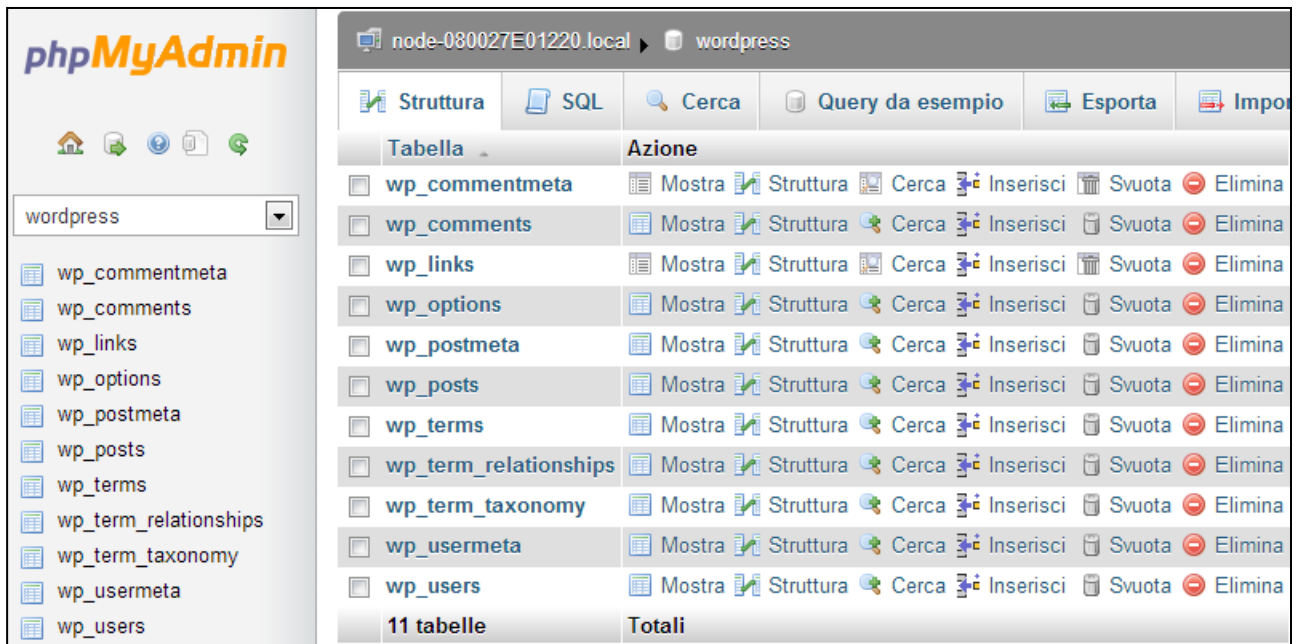
Anche in questo caso, un *ping* tornerà utile per scoprire l'indirizzo ip assegnato alla macchina che lo contiene. Dopo di che con procedimento analogo al precedente, dal browser ottengo la visualizzazione del front-end di Wordpress. Ovviamente prima di questa schermata, una precedente mi chiedeva di inserire una nuova utenza di riferimento, di fatti in alto a sinistra risulta loggato all'interno del sistema.



A questo punto tornando su PhpMyAdmin, e considerato che il database MySQL è stato aggiornato per aggiungere la relazione con Wordpress, avrò ottenuto la creazione in automatico di una nuova base di dati con intestazione *“wordpress”* (evidenziata in rosso sulla sinistra).



Al suo interno, tutte le tavole necessarie al funzionamento dell'applicativo e la possibilità, come bene noto, di crearne di nuove e di avere a disposizione tutte le funzionalità dei 3 famosi ambienti di sviluppo Deployati.



Terminare un servizio

Nel caso in cui ci fosse la necessità di distruggere un servizio, terminare una macchina, ri-deployare una funzionalità, possono tornare utili i seguenti comandi, che tuttavia vanno adoperati conoscendo approfonditamente l'ambiente di lavoro, onde evitare danni al sistema MAAS.

Il seguente comando distrugge un servizio dopo aver revocato il suo stato di esposizione al Web e termina la macchina contenente il servizio de-allocando la risorsa e rendendo la macchina nuovamente disponibile per il Deploy di un nuovo servizio (il nodo regredisce dallo stato “*allocated*” (blu) allo stato “*ready*” (verde)):

```
$ juju unexpose wordpress  
$ juju destroy-service wordpress  
$ juju terminate-machine 3
```

Questo comando invece distrugge *l’environment*, rilasciando completamente tutte le risorse e riportando il sistema a zero, nella fase in cui l’unico servizio attivo era il Server MAAS stesso:

```
$ juju destroy-environment
```

Troubleshooting

In questa sezione riporto gli errori e le problematiche più comuni riscontrate durante il processo di installazione e configurazione di MAAS. Per quanto mi è stato possibile, alcune delle problematiche sono state volta per volta risolte grazie all'aiuto dei forum online e in generale di un'enorme quantità di ricerca in rete. Tuttavia, laddove il problema si ripresentava o non c'era verso di risolverlo, l'unica soluzione è stata quella di stoppare tutto e riprendere l'installazione dall'inizio, dato che per quanto si possa fare attenzione o altro, errori casuali durante la configurazione è sempre possibile averli.

Problema di perdita delle informazioni di rete

E' un problema a cui già ho accennato in precedenza. In pratica, nel momento in cui si lancia l'aggiornamento del sistema con le operazioni di *update* e *upgrade* è possibile che gli indirizzi ip e pertanto, tutte le configurazioni, cambino spontaneamente. E' sempre, quindi, necessario monitorare lo stato del proprio network lanciando il comando:

```
$ ifconfig
```

Qualora effettivamente, le configurazioni subiscono un cambiamento, occorre procedere a un ri-settaggio delle stesse con i parametri corretti, mediante i seguenti comandi:

```
$ sudo dpkg-reconfigure maas
$ sudo dpkg-reconfigure maas-dhcp
$ sudo dpkg-reconfigure maas-provision
```

Problema di inoltro delle chiavi

In fase di inoltro delle chiavi, si potrebbe verificare che ci siano errori di comunicazione tra i vari ambienti di MAAS dovuti al fatto che gli Host da contattare sono sconosciuti al sistema. Il messaggio che si potrebbe verificare è il seguente:

```
ERROR SSH forwarding error: @@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
```

In questo caso bisogna forzare a mano la lettura degli Host da parte del Server MAAS per permettere l'inoltro delle chiavi:

```
$ ssh-keygen -f "~/ssh/known_hosts" -R <IP_NODE>
```

Username e password del superuser MAAS dimenticati

E' necessario un semplice ri-settaggio dei parametri:

```
$ sudo maas changepassword root
```

Problema del trasferimento di chiavi

Questo tipo di problema è per lo più casuale e riguarda le chiavi SSH. Durante il Bootstrap si suppone che Juju trasferisca la chiave pubblica dal server MAAS al nodo. Ma talvolta ciò non accade e compare questo messaggio nel momento in cui si interroga il Cluster:

```
$ juju -v status
2012-05-05 08:38:05,582 DEBUG Initializing Juju status runtime
2012-05-05 08:38:05,590 INFO Connecting to environment...
2012-05-05 08:38:05,751 DEBUG Connecting to environment using node-01...
2012-05-05 08:38:05,764 DEBUG Spawning SSH process with remote_user="Ubuntu"
remote_host="node-01" remote_port="2181" local_port="45720".
2012-05-05 08:38:06,039 ERROR Invalid SSH key
2012-05-05 08:38:06,337:19176(0x7fb457ef6700):ZOO_INFO@log_env@658: Client
environment:zookeeper.version=zookeeper C client 3.3.5
2012-05-05 08:38:06,338:19176(0x7fb457ef6700):ZOO_INFO@log_env@662: Client
environment:Host.name=Ubuntu
2012-05-05 08:38:06,338:19176(0x7fb457ef6700):ZOO_INFO@log_env@669: Client
environment:os.name=Linux
2012-05-05 08:38:06,338:19176(0x7fb457ef6700):ZOO_INFO@log_env@670: Client
environment:os.arch=3.2.0-23-generic
2012-05-05 08:38:06,338:19176(0x7fb457ef6700):ZOO_INFO@log_env@671: Client
environment:os.version=#36-Ubuntu SMP Tue Apr 10 20:39:51 UTC 2012
2012-05-05 08:38:06,339:19176(0x7fb457ef6700):ZOO_INFO@log_env@679: Client
environment:user.name=leseb
2012-05-05 08:38:06,339:19176(0x7fb457ef6700):ZOO_INFO@log_env@687: Client
environment:user.home=/home/leseb
2012-05-05 08:38:06,339:19176(0x7fb457ef6700):ZOO_INFO@log_env@699: Client
environment:user.dir=/var/log/MAAS/rsyslog
2012-05-05 08:38:06,340:19176(0x7fb457ef6700):ZOO_INFO@zookeeper_init@727:
Initiating client connection, Host=localhost:45720 sessionTimeout=10000
watcher=0x7fb455edb6b0 sessionId=0 sessionPasswd=<null> context=0x2f44df0 flags=0
2012-05-05
08:38:06,340:19176(0x7fb452cb7700):ZOO_ERROR@handle_socket_error_msg@1579: Socket
[127.0.0.1:45720] zk retcode=-4, errno=111(Connection refused): server refused to
accept the client
```

Talvolta, distruggendo il contesto attuale e rifacendone il Bootstrap, il problema si risolve:

```
$ juju destroy-environment
$ juju bootstrap
```

Ottimizzazioni di MySQL

Per quanto riguarda MySQL, una volta Deployato, è possibile aggiungere un'autenticazione SSH al servizio e accedere alla console di come utente root:

```
$ juju ssh <unit>
$ mysql -u root -p
# enter root password - /var/lib/juju/mysql.passwd
```

Inoltre si può ottimizzare il servizio con una delle molte opzioni a disposizione:

- *max-connections* - Numero massimo di connessioni consentite al server. '-1' per default.
- *preferred-storage-engine* - Un elenco separato da virgole di "storage engine". Il primo della lista è contrassegnato come motore predefinito di archiviazione. 'InnoDB' ad esempio.
- *tuning-level* - Le possibilità per scegliere la sicurezza di transazione sono 'safest', 'fast' o 'unsafe'.
- *dataset-size* - Dimensione di allocazione di memoria per la cache. Valore Suffisso con 'K', 'M', 'G' o 'T' per indicare unità di kilobyte, megabyte, gigabyte o terabyte, rispettivamente. Valore Suffisso con '%' per utilizzare percentuale della memoria totale della macchina.
- *query-cache-type* - Per attivare la cache sulle query si può impostare il valore su 'ON', 'DEMAND' o 'OFF'.
- *query-cache-size* - Dimensione della cache per le query. In predefinito è '-1' è utilizza il 20% di allocazione di memoria.

La sintassi standard è:

```
$ juju set <service> <option>=<value>
```

Si riporta un esempio di utilizzo di queste opzioni:

```
$ juju set mysql preferred-storage-engine=InnoDB
$ juju set mysql dataset-size=50%
$ juju set mysql query-cache-type=ON
$ juju set mysql query-cache-size=-1
```

Infine MySQL supporta la possibilità di replicare i database su nuove istanze. Ciò consente, ad esempio, di bilanciare il carico delle query e distribuirlo su più unità oppure utilizzare un'unità separata per eseguire il backup, il tutto mentre non vengono diminuite le prestazioni dell'unità centrale.

```
# deploy second service
$ juju deploy mysql mysql-slave

# add master to slave relation
$ juju add-relation mysql:master mysql-slave:slave

# add further slaves
$ juju add-unit mysql-slave
```