Lezione 3

Infrastructure as Code

Infrastructure as code (IaC) is the process of managing and provisioning computer data centers through machine-readable definition files, rather than physical hardware configuration or interactive configuration tools.

- Nella scorsa lezione abbiamo visto come automatizzare il deployment della nostra applicazione di esempio su una o più macchine virtuali EC2. Come automatizzare il provisioning dell'infrastruttura?
- Varie soluzioni:
  - Moduli Ansible per AWS
    - unico strumento per infrastruttura e configurazione software
  - AWS CloudFormation
    - supporto per molti servizi AWS
  - Terraform
    - stessa configurazione utilizzabile con diversi provider Cloud
  - ...
- Oggi useremo Ansible per creare istanze EC2 e configurarle per eseguire la nostra applicazione.

Ansible + EC2

Requisiti:

- bisogna aver configurato *boto3* (v. lezione precedente)
- è necessario installare anche la versione 2 di boto

Inventory dinamico

- Inventory non più statico, ma *dinamico*:
  - il set degli host da utilizzare non è definito staticamente
  - vogliamo che Ansible recuperi le informazioni automaticamente da EC2
- Servono due file per la gestione dell'inventory dinamico tramite EC2, da scaricare nella cartella del playbook:

```
$ wget https://raw.githubusercontent.com/ansible/ansible/devel/contrib/inventory/ec2.py
$ wget https://raw.githubusercontent.com/ansible/ansible/devel/contrib/inventory/ec2.ini
$ chmod +x ec2.py
$ ./ec2.py --list    # just a test
```

Nota: lo script `ec2.py` fornito da Ansible al momento richiede la versione 2 di boto, per questo è necessario che sia installata sul proprio pc.

- L'inventory dinamico consente di eseguire dei playbook con un set di target host determinato a run-time secondo un criterio specificato, es:
  - una regione: e.g., `us-west-1`
  - una availability zone: e.g., `us-west-1a`
  - un tag: e.g., `tag_XXX_YYY` (XXX=Key, YYY=Value)
  - un security group: e.g., `security_group_XXX`

Per approfondire: https://aws.amazon.com/it/blogs/apn/getting-started-with-ansible-and-dynamic-amazon-ec2-inventory-management/

Creazione di istanze EC2

- Creiamo un playbook `create_instance.yaml`.
- Lo avviamo con il comando:

```
ansible-playbook -vvv create_instance.yaml
```

- Verifichiamo che una nuova istanza EC2 e' stata avviata nella regione specificata.
- Per effettuare il deployment della nostra applicazione, possiamo usare il playbook scritto durante la lezione precedente.
- Per usare l'inventory dinamico dobbiamo:
  1. specificare come insieme di host target `tag_app_photogallery`
  2. avviarlo specificando come inventory il file `ec2.py`
  3. specificare il path della chiave privata in `group_vars/tag_app_photogallery`

```
ansible-playbook -i ec2.py deploy_gallery.yaml
```

Per approfondire: Ansible permette di fare molto altro (es. rendere i playbook riusabili tramite l'uso di variabili e roles). La documentazione ufficiale e' un'ottima risorsa per chi volesse approfondire.

PhotoGallery: introduciamo Amazon DynamoDB

- Estendiamo l'applicazione web su cui abbiamo lavorato nella scorsa lezione
- Le immagini sono salvate in un bucket di Amazon S3
- Vogliamo che l'utente possa inserire delle informazioni aggiuntive oltre alle immagini (es. titolo, tag)
- Utilizziamo **Amazon DynamoDB**, un database di tipo chiave-valore
  - Basato su: Tables, Items, Attributes
  - Ogni tabella ha una ed una sola chiave primaria
  - Offre due modelli di **consistenza** per le operazioni di lettura, con garanzie e costi differenti
  - (Altri dettagli su DynamoDB e altre soluzioni simili durante il corso)
- Creiamo una tabella `sdccgallery` che abbia come primary key `imageid`
- **ESERCIZIO**: a partire da `photogallery_v3` integrare Amazon DynamoDB nell'applicazione web di esempio per fare in modo che gli utenti possano inserire un titolo e dei tag per ogni immagine caricata; queste informazioni vengono visualizzate insieme alle immagini.

PhotoGallery: introduciamo Amazon CloudFront

- CoudFront e' il servizio di Content Delivery Network di AWS
- Per incrementare la scalabilita' della nostra applicazione, vogliamo che le immagini siano servite da CloudFront
- Creiamo una *distribution*
- Come *Origin Domain Name* possiamo specificare il nome di dominio associato al nostro bucket S3
- CloudFront permette di configurare diversi aspetti del servizio:
  - caching
  - tipo di richieste HTTP da accettare
  - aree geografiche in cui distribuire i contenuti
  - ...
- Una volta creata la distribuzione, tutto cio' che bisogna fare e' aggiornare l'URL con cui gli utenti accedono ai contenuti della nostra applicazione