

# VirtShell

## Framework para aprovisionamiento de soluciones virtuales

**Carlos Alberto Llano R.**

Escuela de Ingeniería de Sistemas y Computación  
Maestría en Ingeniería con énfasis en Ingeniería de Sistemas y Computación

Director:  
**John Alexander Sanabria**

August 26, 2016



# Contents

- ① Propósitos alcanzados
- ② Flujo de aprovisionamiento
- ③ Experiencia y Evaluación
- ④ Conclusiones
- ⑤ Recomendaciones



# Objetivos

## General

Diseñar un framework web, que permita el aprovisionamiento de software automático, para ambientes virtualizados.

# Objetivos

## General

Diseñar un framework web, que permita el aprovisionamiento de software automático, para ambientes virtualizados.

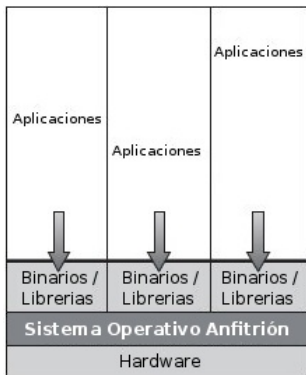
## Específicos

- 1 Evaluar diferentes técnicas y soluciones de aprovisionamiento que se utilizan en la actualidad.
- 2 Evaluar diferentes mecanismos de aprovisionamiento.
- 3 Realizar una ejemplificación del framework.

●○○  
○○○○○○○○  
○○○○

Técnicas y soluciones de aprovisionamiento actuales (Objetivo #1)

# Técnicas de Virtualización trabajadas

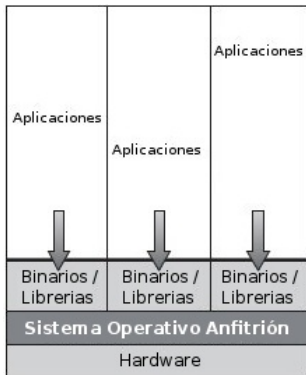


Contenedores

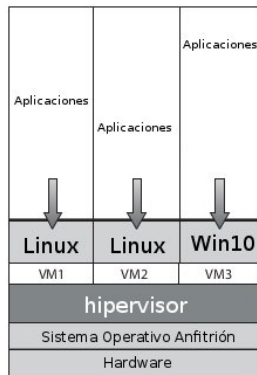


Técnicas y soluciones de aprovisionamiento actuales (Objetivo #1)

# Técnicas de Virtualización trabajadas



Contenedores



Máquinas Virtuales



Técnicas y soluciones de aprovisionamiento actuales (Objetivo #1)

# Soluciones de aprovisionamiento evaluadas

- Fabric
- Chef
- Puppet
- Juju
- CFEngine
- Bcfg2
- Ansible
- Cobbler
- SmartFrog
- Amazon EC2
- Docker composer
- SaltStack
- Vagrant



Técnicas y soluciones de aprovisionamiento actuales (Objetivo #1)

# Soluciones de aprovisionamiento evaluadas

Solución	Sop.Nubes	Curv.Apren	Crea	Aprov	API REST	Mult.SO
Chef	Todas	Alta	✓	✓	✓	✓
Juju	OpenStack MAAS	Baja	✓	✓	✗	✗
Puppet	La mayoría	Media	✗	✓	✗	✓
Ansible	Amazon OpenStack	Baja	✗	✓	✗	✓
Amazon EC2	Amazon	Media	✓	✓	✓	✓
Docker	✗	Media	✓	✓	✓	✓
Vagrant	✗	Baja	✓	✓	✗	✓



○○○  
●○○○○○○○  
○○○○

Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

# Selección del mecanismo de aprovisionamiento

Se evaluaron diferentes formas de escritura de scripts de aprovisionamiento

- json
- xml
- yaml
- archivos texto
- lenguajes de programación (ruby, python, etc.)
- bash



Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

# Selección del mecanismo de aprovisionamiento





Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

# Selección del mecanismo de aprovisionamiento



```
#!/bin/bash
```



Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

# Selección del mecanismo de aprovisionamiento



Bash coming to Windows



Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

## Selección del mecanismo de aprovisionamiento

Se evaluaron diferentes formas de guardar los scripts en el sistema

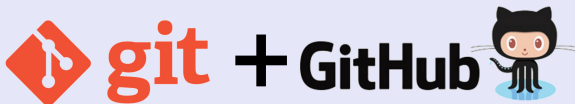
- repositorio central propio?
- enviarlo en el momento del aprovisionamiento?



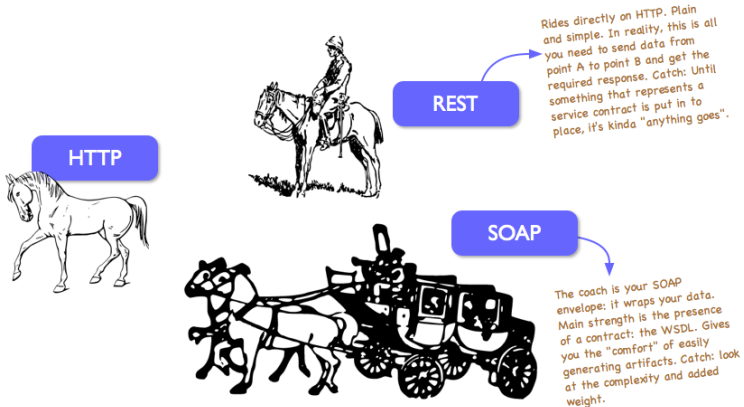
Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

# Selección del mecanismo de aprovisionamiento

## Solución



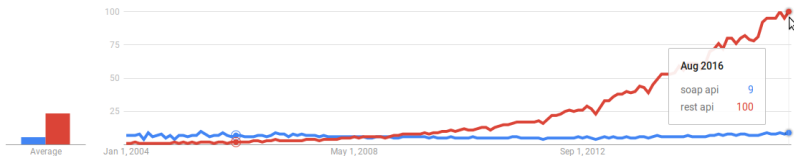
# Evaluación y selección de un estilo arquitectural





Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

# Evaluación y selección de un estilo arquitectural



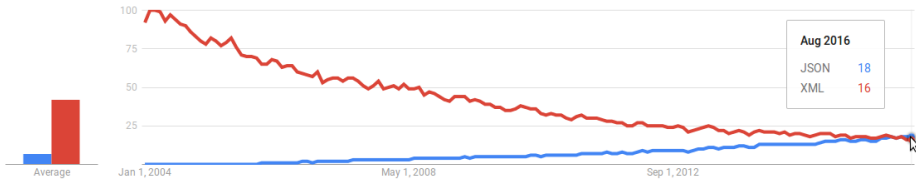
Fuente: Google Trends





Evaluar diferentes mecanismos de aprovisionamiento. (Objetivo #2)

# Evaluación y selección de un estilo arquitectural

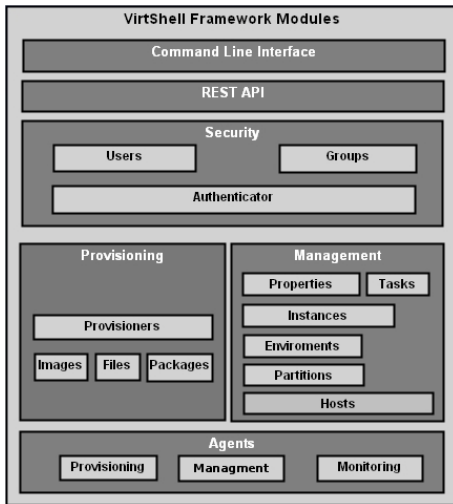


Fuente: Google Trends



Realizar una ejemplificación del framework. (Objetivo #3)

# Framework





Realizar una ejemplificación del framework. (Objetivo #3)

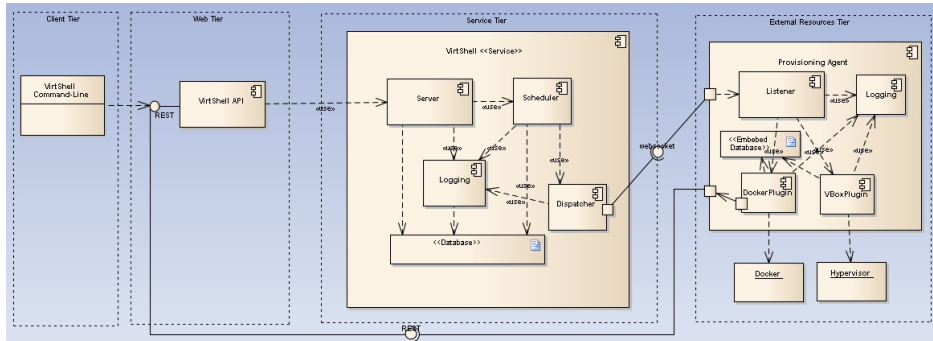
# Características del Framework

- Programable
- Repetible
- Modular
- Seguro
- Extensible
- Inyección de dependencias virtuales
- Interoperable



Realizar una ejemplificación del framework. (Objetivo #3)

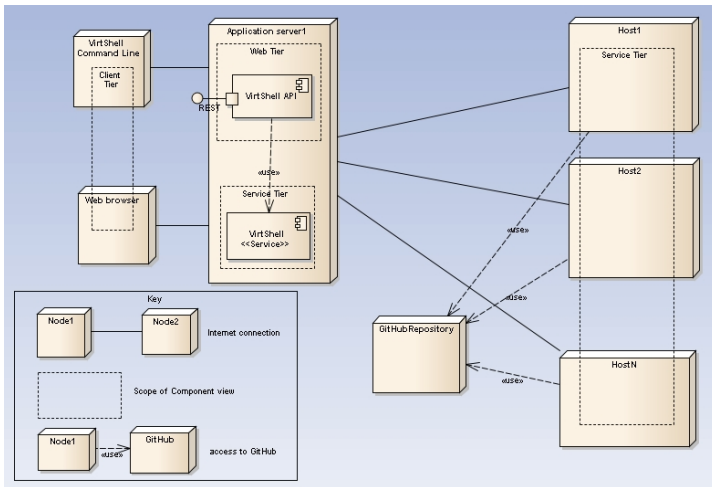
# Vista de componentes





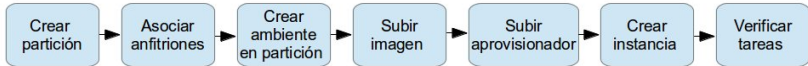
Realizar una ejemplificación del framework. (Objetivo #3)

## Vista de deployment



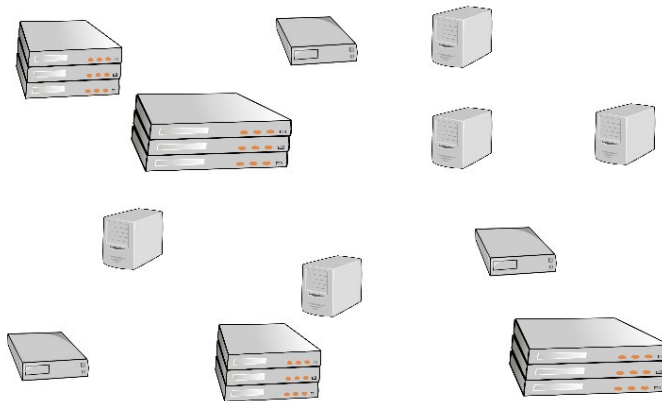
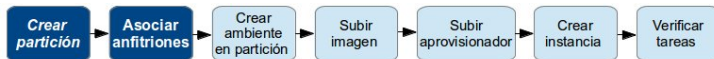
ooo  
ooooooooo  
oooo

# Flujo de aprovisionamiento



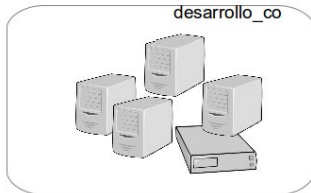
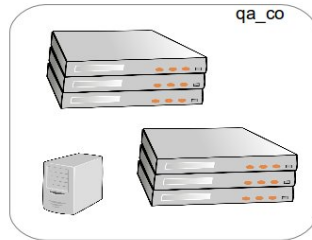
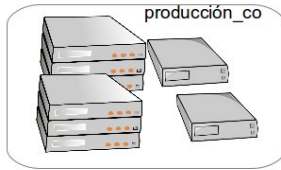
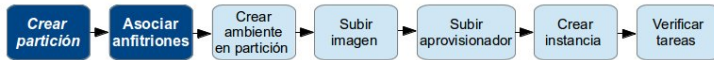
ooo  
ooooooooo  
oooo

# Particiones y Anfitriones



ooo  
ooooooooo  
oooo

# Particiones y Anfitriones





ooo  
ooooooooo  
oooo

## Creación de una partición (Ejemplo)

```
1 curl -X POST http://virtshellsrv:80/partitions/  
   -d "{\"name\":\"development_co\",  
3     \"description\":\"Collection of servers oriented to  
       development team in Colombia.\"}"  
   -H "accept:application/json" | jq .
```

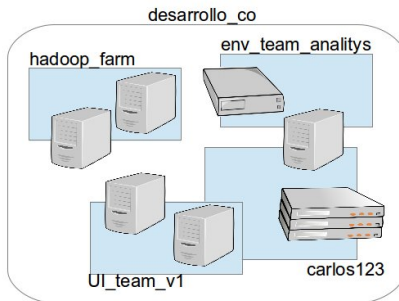
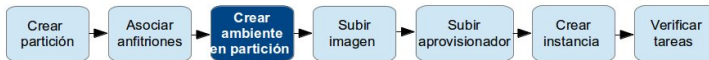
ooo  
ooooo  
ooooo  
oooo

## Asociación de un anfitrión a una partición (Ejemplo)

```
curl -X POST http://virtshellsrv:80/hosts/  
-d "{\"name\": \"host-server-01\",  
    \"os\": \"ubuntu-14.04.4-amd64\",  
    \"memory\": \"2GB\",  
    \"partition\": \"development_co\",  
    \"type\": \"GeneralPurpose\",  
    \"local_ipv4\": \"192.168.56.101\",  
    \"drivers\": [\"docker\", \"lxc\"]}"  
-H "accept: application/json" | jq .
```

ooo  
ooooooooo  
oooo

# Ambientes de trabajo



ooo  
ooooooooo  
oooo

## Ambientes de trabajo (Ejemplo)

```
1 curl -X POST http://virtshellsrv:80/enviroments/  
   -d "{\"name\": \"development\",  
3     \"description\": \"Development enviroment\",  
     \"partition\": \"development_co\",  
5     \"users\": [{\"login\": \"development_user\",  
                  {\"login\": \"guest\"}}]}\"  
7 -H "accept: application/json" | jq .
```

ooo  
ooooo  
ooooo

# Imágenes



## De dos Tipos

- ISO
- Templates

ooo  
ooooooooo  
oooo

# Imágenes (ISO)

`https://github.com/CALLanoR/ubuntu-unattended`

ooo  
ooooooooo  
oooo

# Imágenes (ISO)

```
1 curl -sv -X PUT \  
    -H 'accept: application/json' \  
3    -H "Content-Type: text/plain" \  
    -H 'X-VirtShell-Authorization: UserId:Signature' \  
5    -d '{"name": "ubuntu_server_14.04.2_amd64",  
        "type": "iso",  
7        "os": "ubuntu",  
        "timezone": "America/Bogota",  
9        "key": "/home/callanor/.ssh/id_rsa.pub",  
        "preseed_url": "https://<host>:<port>/api/virtshell/v1/  
        files/seeds/seed_ubuntu14-04.txt",  
11       "download_url": "http://releases.ubuntu.com/raring/ubuntu  
        -14.04-server-amd64.iso"}' \  
    'http://virtshellsrv:8080/api/virtshell/v1/image' | jq .
```

```

ooo
ooooo
ooooo
ooo

```

# Imágenes (Template)

```

FROM ubuntu:14.04
MAINTAINER Carlos Llano <carlos_llano@hotmail.com>
|
RUN locale-gen en_US.UTF-8
RUN dpkg-reconfigure locales

RUN sed 's/##ModLoad imudp/$ModLoad imudp/' -i /etc/rsyslog.conf
RUN sed 's/##$UDPServerRun 514/$UDPServerRun 514/' -i /etc/rsyslog.conf
RUN sed 's/##ModLoad imtcp/$ModLoad imtcp/' -i /etc/rsyslog.conf
RUN sed 's/##$InputTCPServerRun 514/$InputTCPServerRun 514/' -i /etc/rsyslog.conf
EXPOSE 514/tcp 514/udp
CMD ["/usr/sbin/rsyslogd", "-dn", "-f", "/etc/rsyslog.conf"]

RUN apt-get update -y
RUN apt-get install -y openssh-server git

RUN mkdir /var/run/ssh
RUN useradd -s /bin/bash -m virtshell
RUN echo "virtshell:virtshell" | chpasswd
RUN echo "virtshell ALL=(ALL) NOPASSWD: ALL" >> /etc/sudoers

RUN echo "root:virtshell" | chpasswd
RUN sed -i 's/PermitRootLogin without-password/PermitRootLogin yes/' /etc/ssh/sshd_config

RUN sed 's@session\s*required\s*pam_loginuid.so@session optional pam_loginuid.so@g' -i /etc/pam.d/ssh

ENV NOTVISIBLE "in users profile"
RUN echo "export VISIBLE=now" >> /etc/profile

EXPOSE 22
CMD ["/usr/sbin/sshd", "-D"]

RUN apt-get install -y python3-setuptools python3-pip
RUN pip3 install virtshell_commands

RUN echo 'export LC_ALL=C' >> ~/.bashrc
CMD ["source .bashrc"]

RUN apt-get install -y supervisor
RUN mkdir -p /var/log/supervisor
RUN printf '[supervisord]\nno-daemon=true\n\n' >> /etc/supervisor/conf.d/supervisord.conf
RUN printf '[program:sshd]\nautostart=true\ncommand=/usr/sbin/sshd -D' >> /etc/supervisor/conf.d/supervisord.conf

CMD /usr/bin/supervisord -c /etc/supervisor/conf.d/supervisord.conf

```





ooo  
ooooooooo  
oooo

## Primero el archivo (Ejemplo - Template)

```
curl -sv -X POST \  
-H 'accept: application/json' \  
-H "Content-Type: multipart/form-data" \  
-F "permissions=xrwxrwxr" \  
-F "file=@/home/callanor/Documents/Tesis/Repositories/VirtShell  
/virtshell_server/tests/files/dockerfile_ubuntu_server_14  
.04" \  
http://virtshellsrv:80/files/ | jq .
```

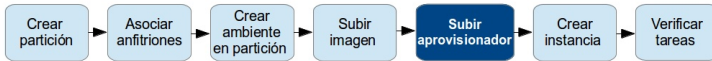
ooo  
ooooooooo  
oooo

## Luego la imagen (Ejemplo - Template)

```
curl -sv -X POST http://virtshellsrv:80/images/  
-d "{\"name\":\"ubuntu_server_14.04_amd64\",  
    \"type\":\"docker-container\",  
    \"container_resource\":\"http://virtshellsrv/file/  
    dockerfile_ubuntu_server_14.04\"}"  
-H 'accept: application/json' | jq .
```

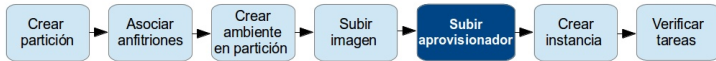
ooo  
ooooooooo  
oooo

# Aprovisionadores



ooo  
ooooooooo  
oooo

# Aprovisionadores

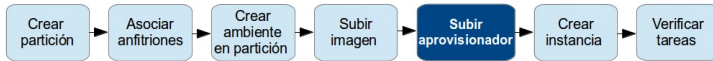


## BASH

```
https://github.com/CALLanoR/VirtShell_Provisioner_Simple_WebSite_Example
```

ooo  
ooooooooo  
oooo

# Aprovisionadores



## BASH

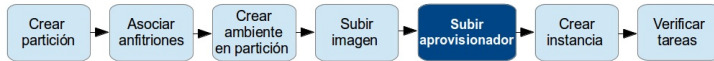
[https://github.com/CALLanoR/VirtShell\\_Provisioner\\_Simple\\_WebSite\\_Example](https://github.com/CALLanoR/VirtShell_Provisioner_Simple_WebSite_Example)

## BASH + VirtShell Commands

[https://pypi.python.org/pypi/virtshell\\_commands](https://pypi.python.org/pypi/virtshell_commands)

ooo  
ooooooooo  
oooo

# Aprovisionadores



## BASH

[https://github.com/CALLanoR/VirtShell\\_Provisioner\\_Simple\\_WebSite\\_Example](https://github.com/CALLanoR/VirtShell_Provisioner_Simple_WebSite_Example)

## BASH + VirtShell Commands

[https://pypi.python.org/pypi/virtshell\\_commands](https://pypi.python.org/pypi/virtshell_commands)  
[https://github.com/CALLanoR/VirtShell\\_Generic\\_Provisioner\\_Simple\\_WebSite\\_Example](https://github.com/CALLanoR/VirtShell_Generic_Provisioner_Simple_WebSite_Example)

ooo  
ooooooooo  
oooo

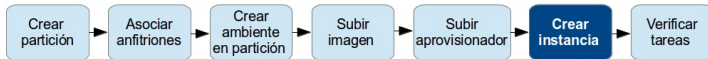
# Aprovisionadores

```
1 curl -sv -X POST http://virtshellsrv:80/provisioners/  
   -d "{\"name\": \"generic_simple_web_site\",  
3     \"launch\": 1,  
     \"memory\": 4,  
5     \"cpus\": 2,  
     \"hdspace\": 20,  
7     \"image\": \"ubuntu_server_14.04_amd64\",  
     \"builder\": \"https://github.com/janutechnology/  
       VirtShell_Generic_Provisioner_Simple_WebSite_Example.  
       git\",  
9     \"executor\": \"sh install.sh\",  
     \"tag\": \"backend\",  
11    \"permissions\": \"xwxwxrwxr\",  
     \"description\": \"Installs/Configures a generic simple  
       web site\"}"  
13 -H 'accept: application/json' | jq .
```



ooo  
ooooooooo  
oooo

# Tipos de instancias



## Tres Tipos

- VirtualBox
- Docker
- LXC
- Amazon (Under Construction...)



ooo  
ooooooooo  
oooo

## Creación de una instancia (Ejemplo)

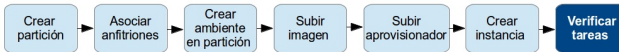
```
1 curl -X POST http://virtshellsrv:80/instances/  
   -d "{\"name\": \"website2\",  
3     \"memory\": 1024,  
     \"cpus\": 1,  
5     \"hdspace\": \"2GB\",  
     \"description\": \"WebServer\",  
7     \"environment\": \"development\",  
     \"provisioner\": \"generic_simple_web_site\",  
9     \"host_type\": \"GeneralPurpose\",  
     \"driver\": \"docker\"}"  
11  -H "accept:application/json" | jq .
```

```

ooo
ooooooooo
oooo

```

# Verificar tareas



```

{
  "tasks": [
    {
      "type": "create_instance",
      "date": "15:17:40.133812",
      "status_history": [
        [
          "pending",
          "15:16:52.321675",
          "received in tasks"
        ],
        [
          "dispatching",
          "15:17:02.342854",
          "creating message to agent"
        ],
        [
          "received",
          "15:17:02.500949",
          "by provisioning_agent"
        ],
        [
          "dispatched",
          "15:17:02.527353",
          "task dispatched to host: 192.168.56.101"
        ],
        [
          "setting up",
          "15:17:02.829316",
          "downloading dockerfile from http://192.168.56.103/file/dockerfile_ubuntu_server_14.04"
        ]
      ],
      [
        "creating",
        "15:17:03.365053",
        "docker_container_plugin creating docker-container website2... with ubuntu:14.04"
      ],
      [
        "created",
        "15:17:06.916909",
        "docker-container-plugin website2 created successfully, ipv4: 172.17.0.2.\n"
      ],
      [
        "provisioning",
        "15:17:12.322563",
        "docker_container_plugin provisioning docker-container website2, with ip 172.17.0.2"
      ]
    ],
    "description": "Create a new instance website2 using driver docker",
    "uuid": "df137dae-68b0-4b34-b321-a5c10299679e",
    "log": "Installing nginx..."
  ]
}

```

ooo  
ooooooooo  
oooo

## Consultar una tarea (Ejemplo)

```
1 curl -s http://192.168.56.103:80/tasks/b9bc6d72-cf78-4c92-bc34-c06809d4d52b | jq .
```

ooo  
ooooooooo  
oooo

# Demo



ooo  
ooooooooo  
oooo

# VirtShell Linea de Comandos

`https://github.com/CALLanoR/VirtShell`



# Experiencia y Evaluación

En las pruebas realizadas VirtShell demostró que parece ser una herramienta útil para aprovisionar software de manera sencilla y fiable.

La experiencia adquirida con la primera versión es la siguiente:

- *VirtShell funciona.*
- *Aprovisionar ambientes virtuales via web usando scripts escritos en el lenguaje que prefiera si es posible.*
- *El aprovisionamiento de máquinas virtuales o contenedores es prácticamente el mismo.*

# Experiencia y Evaluación

La primera versión de VirtShell fue desarrollada en el lenguaje Python (versión 3)

Se encuentra alojada en el repositorio git:  
<https://github.com/janutechnology/VirtShell>.

Esta versión inicial aun no esta terminada y se encuentra en continuo desarrollo para lograr tener todas las funcionalidades funcionando.

# Conclusiones

- VirtShell framework permite administrar y aprovisionar servicios de infraestructura de TI, debido a que se apoya en las actuales tecnologías de virtualización



# Conclusiones

- VirtShell framework permite administrar y aprovisionar servicios de infraestructura de TI, debido a que se apoya en las actuales tecnologías de virtualización
- Por medio del API REST se puede controlar completamente los recursos físicos y virtuales.

# Conclusiones

- VirtShell framework permite administrar y aprovisionar servicios de infraestructura de TI, debido a que se apoya en las actuales tecnologías de virtualización
- Por medio del API REST se puede controlar completamente los recursos físicos y virtuales.
- Al adoptar el estilo arquitectural REST, se heredan nada menos que las propiedades del World Wide Web, las cuales ofrecen mayores ventajas sobre las demás soluciones.

# Conclusiones

- VirtShell framework permite administrar y aprovisionar servicios de infraestructura de TI, debido a que se apoya en las actuales tecnologías de virtualización
- Por medio del API REST se puede controlar completamente los recursos físicos y virtuales.
- Al adoptar el estilo arquitectural REST, se heredan nada menos que las propiedades del World Wide Web, las cuales ofrecen mayores ventajas sobre las demás soluciones.
- REST, le concede además capacidades de integración con diferentes plataformas de desarrollo

# Recomendaciones

## RoadMap

- Implementar una interfaz web que permita administrar los ambientes y máquinas virtuales.

# Recomendaciones

## RoadMap

- Implementar una interfaz web que permita administrar los ambientes y máquinas virtuales.
- Cambiar la forma de seleccionar un host para que tenga en cuenta las métricas e información del sistema de los anfitriones candidatos.

# Recomendaciones

## RoadMap

- Implementar una interfaz web que permita administrar los ambientes y máquinas virtuales.
- Cambiar la forma de seleccionar un host para que tenga en cuenta las métricas e información del sistema de los anfitriones candidatos.
- Implementar scripts que permitan el despliegue del servidor de VirtShell en uno o mas servidores con balanceadores de carga.

# Recomendaciones

## RoadMap

- Implementar una interfaz web que permita administrar los ambientes y máquinas virtuales.
- Cambiar la forma de seleccionar un host para que tenga en cuenta las métricas e información del sistema de los anfitriones candidatos.
- Implementar scripts que permitan el despliegue del servidor de VirtShell en uno o mas servidores con balanceadores de carga.
- Integrar VirtShell con diferentes nubes privadas como Amazon.

# Recomendaciones

## RoadMap

- Implementar una interfaz web que permita administrar los ambientes y máquinas virtuales.
- Cambiar la forma de seleccionar un host para que tenga en cuenta las métricas e información del sistema de los anfitriones candidatos.
- Implementar scripts que permitan el despliegue del servidor de VirtShell en uno o mas servidores con balanceadores de carga.
- Integrar VirtShell con diferentes nubes privadas como Amazon.
- Mejorar la separación de la base de datos en varios servidores, implementando una capa de abstracción que permita el ruteo dinámico de los datos.



# Recomendaciones

## RoadMap

- Implementar una interfaz web que permita administrar los ambientes y máquinas virtuales.
- Cambiar la forma de seleccionar un host para que tenga en cuenta las métricas e información del sistema de los anfitriones candidatos.
- Implementar scripts que permitan el despliegue del servidor de VirtShell en uno o mas servidores con balanceadores de carga.
- Integrar VirtShell con diferentes nubes privadas como Amazon.
- Mejorar la separación de la base de datos en varios servidores, implementando una capa de abstracción que permita el ruteo dinámico de los datos.
- Crear el servicio que proporcione monitorización para las instancias.

# Recomendaciones

## RoadMap

- Crear el servicio de auto scaling que permita escalar automáticamente las instancias en función de políticas definidas.

# Recomendaciones

## RoadMap

- Crear el servicio de auto scaling que permita escalar automaticamente las instancias en función de politicas definidas.
- Extender el servicio de creación de imágenes desatendidas para soportar diferentes distribuciones de sistemas operativos.

ooo  
oooooooo  
oooo

# Preguntas?

