DEVELOPMENT OF A MULTI-TENANT CLOUD PLATFORM BASED ON OS CONTAINERS

OpenHuaca Developer's Guide

Annex of a Degree Thesis

Submitted to the Faculty of the

Escola Tècnica d'Enginyeria de Telecomunicació de Barcelona

Universitat Politècnica de Catalunya

by

Rafael Genés Durán

In partial fulfilment

of the requirements for the degree in

NETWORK ENGINEERING

Table of contents:

❖ Introduction

In this document there is all the information useful for the user of the software OpenHuaca. The paper is divided in four sections in order to introduce the reader to the project in a simple and organized way. Moreover, the information is described to be used as a Developer's Guide for people who would be incorporated to work in the project someday.

Apart from that, the document is going to be updated so that the new improvements and functionalities can be added periodically.

The sections:

1. Project description
   The first describes the bases of the project, the principal content of the product and gives some brief information about how it was the creation of it.

2. Technical description
   The second describes in detail how the main functionalities were developed from a technical point of view.

3. Commands
   The third introduce all the commands developed, including a list of the principal parameters that can be used for every one of them, also describes the main functionality and the principle way of use.

4. Installation files
   The fourth includes the information related with the installation package, mainly where the files are installed once inside the computer in order to give to the user a quick guide to find easily all the needed.

5. Guides
   The fifth give to the user some examples of practical utilization of the commands and the project in general terms.

6. Environments

   The sixth includes a complete guide about the initialization of a new environment. From the beginning, the installation of the OpenHuaca software in the host to the last step that is giving access to the environment users.

7. Work in progress

   The seventh includes the description of the work that is being done nowadays and explains the possible improvements that can appear in further updates of the document.

8. Summary

   The last section summarizes the State of the project and its principal functionalities. Besides, the section includes a table with all the commands of the programme.

1. Project description

OpenHuaca is a cloud platform developed to let the users of the product the possibility of managing OS containers, so the user can create different virtual machines that work in isolation on the same kernel of a physical machine. OpenHuaca allows the user to manage all the machines through domains, so it is not necessary to have any knowledge of the IP or passwords, because it uses the dnsmasq and ssh technologies so that the client only needs to know his Username, the name of the machine and in which domain it is connected. In addition, the user will receive its own certificate in order to be able to access in his container easily.

A part from that, OpenHuaca also takes care of all the details in order to facilitate the job of the network administrator. This program is distributed as a Debian installable package, so with a simple command OpenHuaca can be installed in any Linux distribution with Debian support. Besides, the product has a precise monitoring of resources at the same time that allows to dynamically personalizing the services destined to each container.

The best way to show the potential of the project is with the help of an example based in a real situation: a group of teaching laboratories. Imagine the situation of having a big lab shared by students of three different subjects; in this case, each subject would be a domain and in each domain it will be created the whole of machines needed.

There will be a user “Teacher” who will have access to all the containers of each domain, so if that user has any doubts can access any machine with a single command to see what the other user are doing (students).

On the other hand, when a student arrives has to indicate the name and the subject, so that it does not matter where person is sitting or if there was another subject before in that student post. At that moment the container of the student will be loaded so all the files will be placed how where left by the student in the last connection to the laboratories. Finally, the administrator in this case can be the manager of the laboratory; can monitor with a single command the consumption of resources of all machines in real time, so if there is need to modify the resources of a container can do it in a very simple way.

2. Technical description

In this section there are explained how the OpenHuaca functionalities where developed from a technical point of view. First, there is a brief description about the entire life process of the domains. After, the ssh credentials creation are described. At the end, there is an explanation about the functionality of the natting in the project.

2.1. Domain

In order to understand clearly all the information below, it is necessary to know what a domain is. As it is explained previously, a domain is not something specific, in each environment is defined differently. In OpenHuaca, a domain is a virtual switched network that groups several containers and isolates them from the rest of the virtual networks. It is possible to create a tree-like structure from the host to group the containers. Besides, aliases are used, so to access another container it is not needed to know any IP, only the name of the destination container and its domain, "container.domain".

Finally, the machines of a domain can be seen directly between them but to access other domains have to pass the traffic through the host, so that the traffic can be blocked or not depending on the configuration of the desired environment.

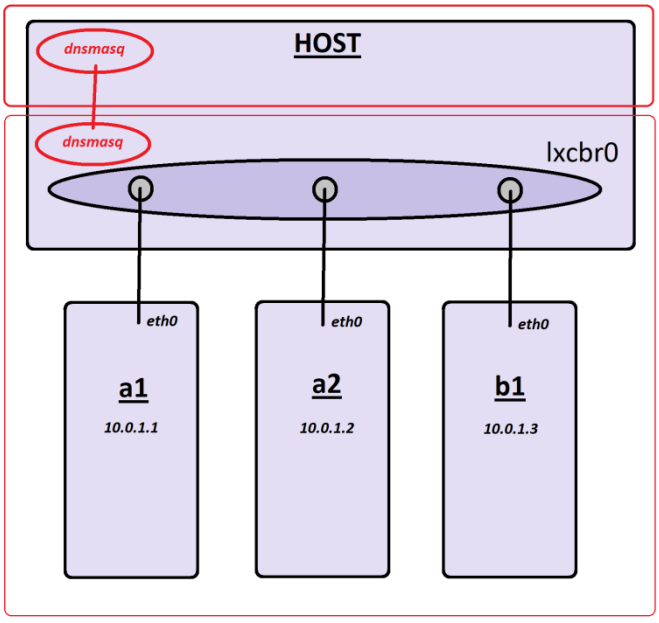


Figure 1: LXC structure

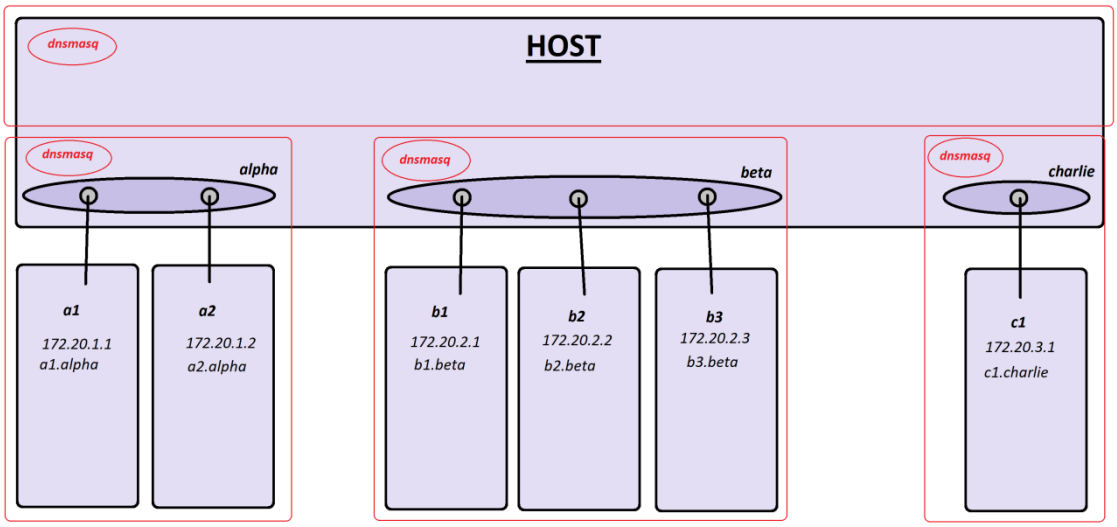


Figure 2: Huaca structure

Once it is clear what is a domain, it can be explained technically how have been realized in the project. When a domain is created, a configuration file is generated, where the free IP range is assigned to the host. In addition, a random MAC address is also generated. Also, it has been created a service that initializes the domains created every time that the OS host starts. However, the domains can be initializes through the command line too.

When a domain is initialized, the early step is creating a bridge with the first IP of the assigned range. Once created and running it is configured a dnsmasq service and the host is advised that the handling of dns and dhcp of that range will be responsible for this new service.

After that a pair of entries in the iptables are created in order to redirect the dns traffic and add the domain alias to the host dns. In this way the domain is mounted as a bridge that autogestionates its range of IPs and it has access from the host with its alias "domain". Finally, to connect a container to the bridge we use virtual Ethernet technology, VETH. So, in the bridge it is generated an interface type veth with a random name "vethXXXX" that connects to the container like eth0 assigning an IP of the range directly to the container.

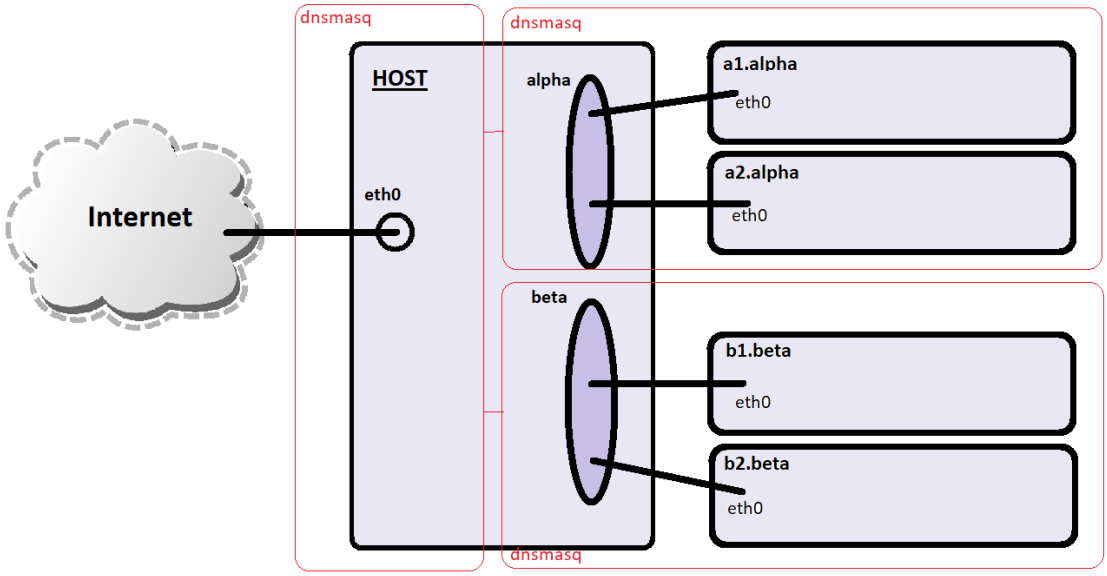


Figure 3: Bridge + VETH

On the contrary, if the desired is to turn off one or several domains what is done before anything is to stop all the containers in the domain, once there aren't containers in the bridge, the domain, the dnsmasq and that alias of the host are deleted. This leaves only the configuration file. Note that the range of IPs is reserved for that domain even if it is stopped. Finally, in order to delete a domain completely it is necessary to delete the configuration file too, which also frees the IP range.

2.2. SSH Credentials

An ssh certificate is generated by the administrator of the environment to users so that they can access containers. It is necessary to create a certification authority for each domain. Creating the certificate generates three files: the public key, the private key and the certificate. With these three files, that are delivered to the users, it will be possible to accede during the specified weekly to all the containers of a domain that contain the specified user.

The system generation needs to design a key sharing system. For this, a certification authority has been generated for each domain, the containers have been configured to accept the CA only from their domain and finally the certificates must be generated from a CA, a user and the validity weeks.

The next step is the creation of a CA, a pair of asymmetric keys of type RSA are generated, so a public key and another private one from a passphrase. It is important to underline that all the passwords of the CA are only known by the administrator or the responsible for generating the certificates. Once the two keys are specified, the public key has to be distributed to all the containers. A part from sending it to the containers it is necessary to edit the file "/etc/ssh/sshd_config" in order to consider the public key as a trusted CA.

After that the configuration of the CA of the domains is finished, so the user can start emitting certificates. In the generation of a certificate it is necessary to pass the domain, the user and the validity weeks as parameters. Evidently, the user must know the password of the selected CA. The service has been configured to add an ID of 8 hexadecimal characters to each certificate. This identifier is useful to be able issuing several certificates for a single user. Having different certificates is useful to grant access with different validity dates.

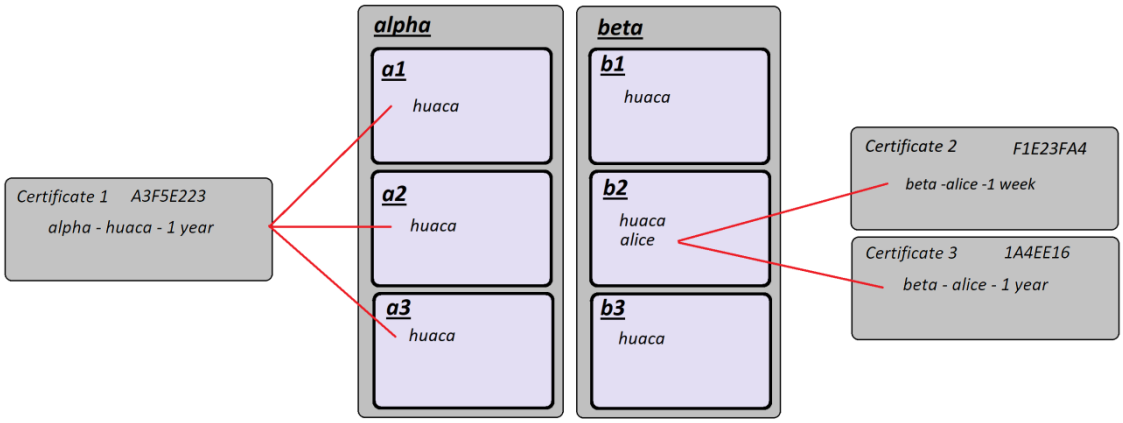


Figure 4: Certification examples

2.3. NAT

A command has been created to manage the "natting" of the containers. In this way, the user can redirect the internet traffic to the Port of a container. For example, if a user wants to offer a web service, it can generate a container where the nat is created and with the help of the huaca-nat command, which configures the iptables automatically, the user can redirect all the traffic from the host to the port X and send it to a container to port Y. So, the user can have connection directly from the internet to a container, without having the intermediate step of the host.

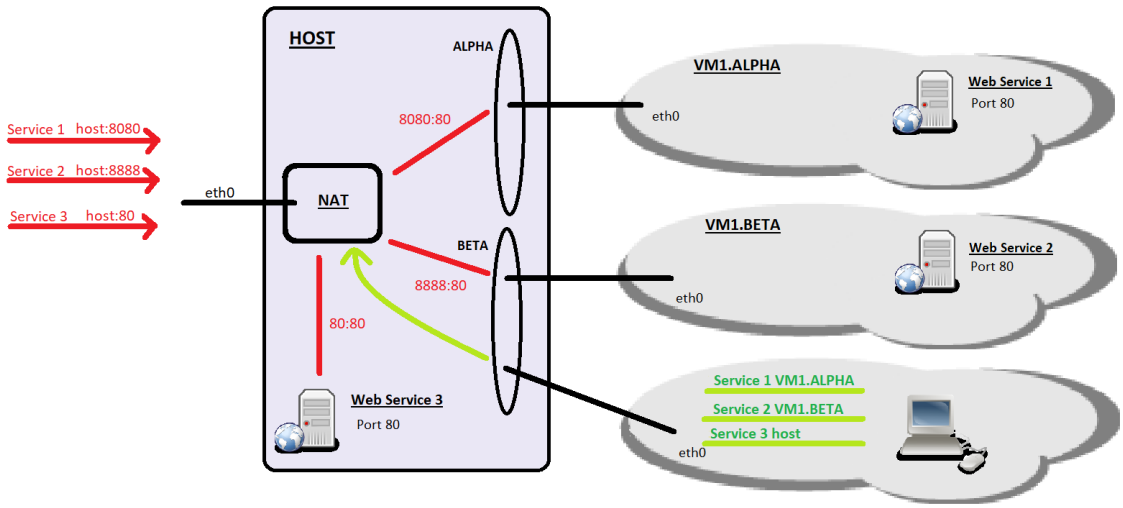


Figure 5: Huaca natting

3. Commands

In this section there is a brief description about every command and it is specified the parameters needed by all of them. Finally, there is included at least one typical example of the functionality. Nowadays, OpenHuaca functionality focuses on CLI, so all the interactions between the user and the product are implemented with commands.

After this small introduction there is the complete list of commands in alphabetical order:

- **huaca-attach :**

Let the user interact with the container wanted.
It has the option of connecting to any active container with the help of the namespaces. Also, it can execute a command in a remote way that means not connecting to the container directly. That tool is very useful in order to scripting; the script will allow the user the possibility of starting a container and executing a command as privileged user into the container.

```
Usage: huaca-attach --domain=H_DOMAIN --name=H_NAME [-- COMMAND]

Execute the specified COMMAND - enter the container H_NAME

Options :
  -d, --domain=H_DOMAIN  H_DOMAIN of the container
  -n, --name=H_NAME   H_NAME of the container
  -e, --elevated-privileges=PRIVILEGES
                      Use elevated privileges instead of those of the
                      container. If you don't specify privileges to be
                      elevated as OR'd list: CAP, CGROUP and LSM (capabilities,
                      cgroup and restrictions, respectively) then all of them
                      will be elevated.
                      WARNING: This may leak privileges into the container.
                      Use with care.
  -a, --arch=H_ARCH   Use H_ARCH for program instead of container's own
                      architecture.
  -s, --namespaces=FLAGS
                      Don't attach to all the namespaces of the container
                      but just to the following OR'd list of flags:
                      H_MOUNT, H_PID, UTSH_NAME, IPC, USER or NETWORK.
                      WARNING: Using -s implies -e with all privileges
                      elevated, it may therefore leak privileges into the
                      container. Use with care.
  -R, --remount-sys-proc
                      Remount /sys and /proc if not attaching to the
                      mount namespace when using -s in order to properly
                      reflect the correct namespace context. See the
                      lxc-attach(1) manual page for details.
      --clear-env     Clear all environment variables before attaching.
                      The attached shell/program will start with only
                      container=lxc set.
      --keep-env      Keep all current environment variables. This
                      is the current default behaviour, but is likely to
                      change in the future.
  -L, --pty-log=FILE
                      Log pty output to FILE
  -v, --set-var       Set an additional variable that is seen by the
                      attached program in the container. May be specified
                      multiple times.
      --keep-var      Keep an additional environment variable. Only
                      applicable if --clear-env is specified. May be used
                      multiple times.
  -f, --rcfile=FILE
                      Load configuration file FILE

Common options :
  -o, --logfile=FILE              Output log to FILE instead of stderr
  -l, --logpriority=LEVEL         Set log priority to LEVEL
  -q, --quiet                     Don't produce any output
  -?, --help                      Give this help list
      --usage                     Give a short usage message
      --version                   Print the version number

Mandatory or optional arguments to long options are also mandatory or optional
for any corresponding short options.

*** See the huaca-attach man page for further information.
```

FIGURE 6:

huaca-attach –help

Typical uses

- Connect to a container with namespaces. Change current tty into the containers' prompt.

```
root@rgenes_jose:~# huaca-attach -d alpha -n a1
root@a1:~#
```

- Execute a command into a container.

```
root@rgenes_jose:~# huaca-attach -d alpha -n a1 -- ls
```

- Execute a command into a container as a privileged user.

```
root@rgenes_jose:~# huaca-attach -d alpha -n a1 -e ls
```

❖ **huaca-autostart**

LXC has the functionality to activate containers while the OS host is starting. This command allows the user to manage the state of a determinate container and also the order of the execution can be modified by priorities.

```
Usage: huaca-autostart
huaca-autostart managed auto-started containers

Options:
  -d, --domain=H_DOMAIN auto-start H_DOMAIN containers
  -k, --kill         kill the containers instead of starting them
  -L, --list         list all affected containers and wait delay
  -r, --reboot       reboot the containers instead of starting them
  -s, --shutdown     shutdown the containers instead of starting them

  -a, --all          list all auto-started containers (ignore groups)
  -A, --ignore-auto ignore lxc.start.auto and select all matching containers
  -g, --groups       list of groups (comma separated) to select
  -t, --timeout=T    wait T seconds before hard-stopping

Common options :
  -o, --logfile=FILE               Output log to FILE instead of stderr
  -l, --logpriority=LEVEL          Set log priority to LEVEL
  -q, --quiet                      Don't produce any output
  -?, --help                       Give this help list
      --usage                      Give a short usage message
      --version                    Print the version number

Mandatory or optional arguments to long options are also mandatory or optional
for any corresponding short options.

*** See the huaca-autostart man page for further information.
```

Figure 7: huaca-autostart –help

- Start all the containers in a specific domain 'alpha' with autostart flag enabled:

```
root@rgenes_jose:~# huaca-autostart -d alpha
```

- Kill all the containers in specific domain ‘alpha’ with autostart flag enabled:

```
root@rgenes_jose:~# huaca-autostart -d alpha -k
```

❖ **huaca-backup**

This command has been made by the j3o team of the network department. It makes backups of the filesystems. It can be configured in order to make the backups from some containers or from all of them. Also, the user can decide if the backups have to be done periodically, in a fixed date or in the same moment.

- List the backup of a container:

```
root@rgenes_jose:~# huaca-backup ls -d alpha -n a1
```

❖ **huaca-bases**

This command has been made by the j3o team of the network department. It lists the templates created. It means that, the user can create a basic container and from it making copies.

- List the bases:

```
root@rgenes_jose:~# huaca-bases
```

❖ **huaca-build**

This command complements the one over, huaca-bases, let the user list the templates and also create new containers from the existent bases. It is useful to

```
huaca-build -b base -n container [-p #sshport]

        -b | --base          container base
        -n | --name          container name
        -p | --ssh-port      external ssh port
        -s | --snapshot      create rootfs as a snapshot

        Builds a container based on "base" container template.
        If ssh-port option is selected, then a dnat command on host
        will be added to forward ssh-external port to container ssh
        port (22)
```

create containers with software or with an environment preconfigured.

Figure 8: huaca-build –help

- Build a container from a base:

```
root@rgenes_jose:~# huaca-build -b base -d alpha -n a1
```

❖ huaca-certification

Let the user manage the authorities of certification and the certificates. Once the certification authority is created, it is configured in all the containers of the domain. On the other hand, when a certificate is created, a public and a private key are generated too. These three files are kept in a directory from the host in order to be distributed easily.

- List the current CAs:

```
root@rgenes_jose:~# huaca-certification ca list
alpha
charlie
delta
```

- Create a new CA:

```
root@rgenes_jose:~# huaca-certification ca create -d beta
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/huaca/certification/CAs/beta.
Your public key has been saved in /etc/huaca/certification/CAs/beta.pub.
The key fingerprint is:
SHA256:i/KS3myDiC5RSM8K7olNXHdfujluROJTFjsqQq+9WaQ beta CA
The key's randomart image is:
+---[RSA 4096]----+
|                 |
| .         .     |
|..o         o    |
|o .oo . o * .    |
|oo.o o oSB +     |
|.oo . o++.+      |
|o=...*E.oo o     |
|+oo ==+o  =      |
|o. ..o*o o..     |
+----[SHA256]-----+
```

- Delete a CA:

```
root@rgenes_jose:~# huaca-certification ca delete -d beta
```

- Create a new certificate:

```
root@rgenes_jose:~# huaca-certification cert create -d alpha -u user -w 52
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/huaca/certification/certs/alpha_user-2a34c501.
Your public key has been saved in /etc/huaca/certification/certs/alpha_user-2a34c501.pub.
The key fingerprint is:
SHA256:OZeP4td+Bv6vw3sFMXRAjXu3xtguuYEW7pqAZfLEt9g alpha_user-2a34c501 CA
The key's randomart image is:
+---[RSA 4096]----+
|              o++.|
|               +..|
|                + |
|       .  . .  o o|
|      . =S.o.  +oo|
|       B +ooo+. =.|
|      . +.E.*.++ .|
|        ...= oo*..|
|         .+.o.=**.|
+----[SHA256]-----+
Enter passphrase:
Signed user key /etc/huaca/certification/certs/alpha_user-2a34c501-cert.pub: id "2a34c501"
```

- List the current certifications:

```
root@rgenes_jose:~# huaca-certification cert list
Certificate                     Domain  Name    Created     Expire      Serial  Key ID
alpha_huaca-99bbf489-cert.pub   alpha   huaca   2017-06-20  2017-06-27  0       99bbf489
alpha_rgenes-7ab63bf0-cert.pub  alpha   rgenes  2017-06-08  2017-06-15  0       7ab63bf0
alpha_ubuntu-33d5ab75-cert.pub  alpha   ubuntu  2017-06-08  2018-06-07  0       33d5ab75
alpha_ubuntu-8771bd43-cert.pub  alpha   ubuntu  2017-06-08  2017-06-15  0       8771bd43
alpha_user-2a34c501-cert.pub    alpha   user    2017-06-29  2018-06-28  0       2a34c501
beta_ubuntu-9e7f7671-cert.pub   beta    ubuntu  2017-06-08  2018-06-07  0       9e7f7671
delta_jose-2c6b0ce8-cert.pub    delta   jose    2017-06-08  2017-07-06  0       2c6b0ce8
delta_rgenes-8168a685-cert.pub  delta   rgenes  2017-06-08  2017-07-06  0       8168a685
delta_ubuntu-9143de60-cert.pub  delta   ubuntu  2017-06-08  2017-08-17  0       9143de60
```

- **huaca-cgroup**

  This command has been wrapped from LXC. It restricts the resources offered to every container. For example, the user can define a memory limit or a bound for the CPU available in each machine.

- **huaca-checkconfig**

  This command has been wrapped from LXC. It shows the whole information related with the environment configuration, it is inherited from LXC so it displays the information corresponding to this program.

- **huaca-checkpoint**

  This command has been wrapped from LXC. It allows serializing containers on the disk so that they do not consume resources and let the possibility of returning to the original state in case of necessity. Moreover, it let the user to control the state of

the containers when are serialized, that means stopping or restarting them on time of recovering.

- **huaca-config**

  This command has been wrapped from LXC. It let the user to modify the predefined variables such as the directory where the containers are stored, the memory assigned by default or the place where the certifications are kept.

- **huaca-console**

  It allows the user to login in a container. It changes the established terminal or another for the tty of the container specified. It is so useful in order to access to the user own machines with the help of the namespaces.

- **huaca-copy** It creates a copy of a container. It is useful in order to create a template, a base, and from it, coping various containers. Very used in the creation of the command build and backup.

- **huaca-create**

  This command has been wrapped from LXC. It creates new containers, it has several options like defining the domain where have to be added or giving it a name. Apart from that the user can configure the type of the container. Nowadays, the program dispose of the official LXC templates repository, but in following updates of the project the user will be able to add KVM containers too. Finally, the filesystem needed can be also defined.

```
Usage: huaca-create -d H_DOMAIN -n CONTAINER -t H_TEMPLATE [OPTIONS...]

huaca-create creates a container in a domain

Options :
  -d, --domain=H_DOMAIN                  H_DOMAIN of the container
  -n, --name=H_NAME                 H_NAME of the container
  -f, --config=H_CONFIG             Initial configuration file
  -t, --template=H_TEMPLATE         Template to use to setup container
  -B, --bdev=H_BDEV                 Backing store type to use
      --dir=H_DIR                   Place rootfs directory under H_DIR

  H_BDEV options for LVM (with -B/--bdev lvm):
      --lvname=H_LVH_NAME             Use LVM lv name H_LVH_NAME
                                   (Default: container name)
      --vgname=VG                  Use LVM vg called VG
                                   (Default: lxc)
      --thinpool=TP                Use LVM thin pool called TP
                                   (Default: lxc)

  H_BDEV options for Ceph RBD (with -B/--bdev rbd) :
      --rbdname=RBDH_NAME           Use Ceph RBD name RBDH_NAME
                                   (Default: container name)
      --rbdpool=POOL               Use Ceph RBD pool name POOL
                                   (Default: lxc)

  H_BDEV option for ZFS (with -B/--bdev zfs) :
      --zfsroot=PATH               Create zfs under given zfsroot
                                   (Default: tank/lxc)

  H_BDEV options for LVM or Loop (with -B/--bdev lvm/loop) :
      --fstype=TYPE                Create fstype TYPE
                                   (Default: ext3)
      --fssize=SIZE[U]             Create filesystem of
                                   size SIZE * unit U (bBkKmMgGtT)
                                   (Default: 1G, default unit: M)

Common options :
  -o, --logfile=FILE                Output log to FILE instead of stderr
  -l, --logpriority=LEVEL           Set log priority to LEVEL
  -q, --quiet                       Don't produce any output
  -?, --help                        Give this help list
      --usage                       Give a short usage message
      --version                     Print the version number

Change path location are not available in huaca.

*** See the huaca-create man page for further information.
```

Figure 9: huaca-create –help

- Create a basic Ubuntu container

```
root@rgenes_jose:~# huaca-create -d alpha -n a0 -t ubuntu
Checking cache download in /var/cache/lxc/xenial/rootfs-amd64 ...
Copy /var/cache/lxc/xenial/rootfs-amd64 to /var/lib/huaca/alpha/a0/rootfs ...
Copying rootfs to /var/lib/huaca/alpha/a0/rootfs ...
Generating locales (this might take a while)...
  en_US.UTF-8... done
Generation complete.
Creating SSH2 RSA key; this may take some time ...
2048 SHA256:Cc7OXv1JekCKFsVgH+W8zEQ2KxHx4spqUMZ3iph0D/o root@rgenes_jose (RSA)
Creating SSH2 DSA key; this may take some time ...
1024 SHA256:jqo3INfudtMA64oanHnRs+T8ehX+oX6Yn0R0JRhDM04 root@rgenes_jose (DSA)
Creating SSH2 ECDSA key; this may take some time ...
256 SHA256:YL+MaptC9ggCVp5EcgF1L81mt1ifG8dymJvprqwBAVo root@rgenes_jose (ECDSA)
Creating SSH2 ED25519 key; this may take some time ...
256 SHA256:zVpGeqTrlkvUIjUcPQVBKFEVqdTpTfd2ialNJSDyy7o root@rgenes_jose (ED25519)
invoke-rc.d: could not determine current runlevel
invoke-rc.d: policy-rc.d denied execution of start.

Current default time zone: 'Etc/UTC'
Local time is now:      Thu Jun 29 00:08:30 UTC 2017.
Universal Time is now:  Thu Jun 29 00:08:30 UTC 2017.


##
# The default user is 'ubuntu' with password 'ubuntu'!
# Use the 'sudo' command to run tasks as root in the container.
##
```

- **huaca-destroy**

  It destroys one or various containers from a domain. It is important to have the container stopped, if not the command is not going to work.

```
Usage: huaca-destroy --name=H_NAME [-f]

huaca-destroy destroys a container with the identifier H_NAME

Options :
  -d, --domain=H_DOMAIN  H_DOMAIN of the container
  -n, --name=H_NAME   H_NAME of the container
  -s, --snapshots   destroy including all snapshots
  -f, --force       wait for the container to shut down
  --rcfile=FILE     Load configuration file FILE

Common options :
  -o, --logfile=FILE              Output log to FILE instead of stderr
  -l, --logpriority=LEVEL         Set log priority to LEVEL
  -q, --quiet                     Don't produce any output
  -?, --help                      Give this help list
      --usage                     Give a short usage message
      --version                   Print the version number

Mandatory or optional arguments to long options are also mandatory or optional
for any corresponding short options.

*** See the huaca-destroy man page for further information.
```

Figure 10: huaca-destroy –help

- Destroy a container:

```
root@rgenes_jose:~# huaca-destroy -d alpha -n a0
Destroyed container a0
```

❖ **huaca-device**

It adds a device to a container like a USB, a HD or a disc reader. That let the user to append hardware to the filesystem wanted.

❖ **huaca-domain**

This command has been created in order to manage the domains. It lets the user to create, delete, list, start, stop or restart domains. Apart from that, OpenHuaca offers to the users a service that starts the domains at the initialization of the OS host.

➢ Create a domain:

```
root@rgenes_jose:~# huaca-domain create domain
Creating domain domain...
Finding a valid IP range...
Valid IP: 172.20.5.0/24
```

➢ Destroy a domain:

```
root@rgenes_jose:~# huaca-domain destroy domain
```

➢ List domains with their containers:

```
root@rgenes_jose:~# huaca-domain list
- [RUNNING] alpha
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
a1   RUNNING 0         -      172.20.1.68  -    alpha
a11  RUNNING 0         -      172.20.1.212 -    alpha
a2   RUNNING 0         -      172.20.1.207 -    alpha
a3   STOPPED 0         -      -            -    alpha
a4   STOPPED 0         -      -            -    alpha
a5   STOPPED 0         -      -            -    alpha
c2   STOPPED 0         -      -            -    alpha

- [RUNNING] beta
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
b1   RUNNING 0         -      172.20.2.180 -    beta
b2   STOPPED 0         -      -            -    beta
b3   STOPPED 0         -      -            -    beta

- [RUNNING] charlie
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
c1   RUNNING 0         -      172.20.3.83 -     charlie
c2   RUNNING 0         -      172.20.3.85 -     charlie

- [RUNNING] delta
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
d1   RUNNING 0         -      172.20.4.227 -    delta

- [RUNNING] domain2
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
```

➢ Show the status of each domain:

```
root@rgenes_jose:~# huaca-domain status
 Status:
 [RUNNING] alpha          CA=active      IP=172.20.1.0/24
 [RUNNING] beta           CA=none        IP=172.20.2.0/24
 [RUNNING] charlie        CA=active      IP=172.20.3.0/24
 [RUNNING] delta          CA=none        IP=172.20.4.0/24
 [RUNNING] domain2        CA=none        IP=172.20.6.0/24
```

- Start/stop domains:

```
root@rgenes_jose:~# huaca-domain start alpha beta charlie delta
```

- Start/ stop all domains:

```
root@rgenes_jose:~# huaca-domain stop-all
```

- **huaca-execute**

  It combines the functionality of the commands create and attach. It let the user to test if a container exist, if not it creates the new container and executes the commands.

- **huaca-freeze :**

  It “freezes” a container; it means that it blocks the cpu to all the processes of that container. In this way, we can leave the container in a state that does not consume resources and restore it when we think that it is convenient.

- **huaca-info**

  It is a help command. It gives to the user all kinds of relevant information about a container. For example, the name, the state of the machine, the cpu and the memory in use, the Tx and Rx consumed, etc.

  - Show the basic information of a container:

```
root@rgenes_jose:~# huaca-info -d alpha -n a1
Name:           a1
State:          RUNNING
PID:            25209
IP:             172.20.1.68
CPU use:        0.39 seconds
BlkIO use:      5.93 MiB
Memory use:     16.80 MiB
KMem use:       0 bytes
Link:           veth5COW2G
 TX bytes:      1.16 KiB
 RX bytes:      1.59 KiB
 Total bytes:   2.75 KiB
```

- **huaca-ls**

  It is another help command, allows the user to view in a friendly way all the containers created, being able to filter them by domains, states, network data, etc. It is very useful to know relevant information of several containers at the same time.

  - Show a fancy list of containers:

```
root@rgenes_jose:~# huaca-ls -f
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6      DOMAIN
a1   RUNNING 0         -      172.20.1.68 -     alpha
a11  STOPPED 0         -      -           -     alpha
a2   STOPPED 0         -      -           -     alpha
a3   STOPPED 0         -      -           -     alpha
a4   STOPPED 0         -      -           -     alpha
a5   STOPPED 0         -      -           -     alpha
c2   STOPPED 0         -      -           -     alpha
b1   RUNNING 0         -      172.20.2.180 -    beta
b2   STOPPED 0         -      -            -    beta
b3   STOPPED 0         -      -            -    beta
c1   RUNNING 0         -      172.20.3.83 -     charlie
c2   STOPPED 0         -      -           -     charlie
d1   RUNNING 0         -      172.20.4.227 -    delta
```

- **huaca-monitor**

  This command has been wrapped from LXC. It let the user to monitor the state of a container. So, the user will be able to create a register of states of every machine.

- **huaca-nat**

  This command has been made by the j3o team of the network department. It manages the nat between the containers and the host. So, the user can redistribute the traffic from the host to one of the containers. For example, if arrives a packet to

the host requesting a webpage in the Port 80, the user can have a register in the nat in order to send it to the container “web” and that will be the manager of returning the content to the webpage.

- **huaca-rebase**

  This command has been created by Jorge, a Peru project mate. It was developed by the need to update a container that was created from a database and has been updated. This command generates a new container from the new updated database but preserving the data and software installed in the desired container.

- **huaca-snapshot**

  This command has been wrapped from LXC. It manages the snapshots created, the user can copy, restore o delete old captures.

- **huaca-sshconf**

  This command has been made by the j3o team of the network department. It is a complementary command of the huaca-nat that configures the connections of the containers created.
  This command is deprecated, so it will be eliminated in following updates of the project.

- **huaca-start**

  This command has been wrapped from LXC. It let the user to create a container from a LXC template. It generates the containers in a directory specified in the configuration of the system, and it separates them in domains. Also, it configures the containers to accept the corresponding CAs.

- Start a container:

```
root@rgenes jose:~# huaca-start -d alpha -n a2
```

❖ **huaca-stop**

This command has been wrapped from LXC. It function is the contrary of huaca-start, so it stops the containers selected. This command does not delete the containers, it just stop them.

- Stop a container:

```
root@rgenes jose:~# huaca-stop -d alpha -n a2
```

❖ **huaca-top**

This command allows the user to list all the containers according to the consumption they have in real time. It can also filter them to show just a single domain or several.

- Show a top command about a domain:

Container Name	CPU Used	CPU Sys	CPU User	BlkIO Total	Mem Used
a1	0.40	0.25	0.08	5.93 MB	16.55 MB
a2	0.37	0.25	0.09	0.00	13.08 MB
a3	0.24	0.15	0.04	13.73 MB	16.85 MB
TOTAL 3 of 3	1.00	0.65	0.21	19.66 MB	46.48 MB

❖ **huaca-unfreeze**

It function is the contrary of huaca-freeze, so it is responsible for reassigning cpu times to the desired container by returning it to the exact state where it was previously.

❖ **huaca-unshare**

This command has been wrapped from LXC. Nowadays it is deprecated, so it will be eliminated in following updates of the project.

- **huaca-user**

  It manages the creation, the deletion and the list of the users from a container. It is possible to add a password and to establish permissions of administration in the container.

  - Create a privileged user into a container:

```
root@rgenes_jose:~# huaca-user create -d alpha -n a2 -u huaca -p huaca123 -s
```

  - List the users of a container:

```
root@rgenes_jose:~# huaca-user list -d alpha -n a2
ubuntu
rafa
huaca
```

  - Delete a user from a container:

```
root@rgenes_jose:~# huaca-user delete -d alpha -n a2 -u huaca
userdel: huaca mail spool (/var/mail/huaca) not found
```

- **huaca-usernsexc**

  This command has been wrapped from LXC. Nowadays it is deprecated, so it will be eliminated in following updates of the project.

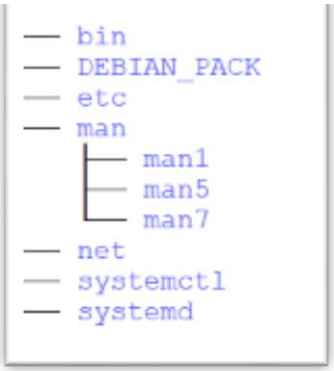
- **huaca-wait**

  It controls the state of a specific container and it does not execute any command before the container has arrived to the established state.

4. Installation files:

OpenHuaca is distributed in a Debian Package in order to let the user use the program in any Linux compatible distribution. This chapter contributes to the correct understanding about what the Package contains and where have to be installed. Moreover, there is all the information needed in order to modify and personalize the environment created without generating conflicts with the software.

First of all, in the following paragraphs there is the description about how is the Package installed. OpenHuaca starts from some directories where the project has been developed:



Once the initialization have begun, the user has to execute the script ***DEBIAN_PACK/pre-script.sh***. that let the user know the directories where every file is attached.

The following list contains the directories of the installation Package:

- systemd -> /etc/systemd/system/
- systemctl -> /lib/systemd/system/.
- etc -> /etc/huaca/.
- bin -> /usr/local/bin/.
- Man -> /usr/share/man/.
- net -> /usr/lib/x86_64-linux-gnu/huaca/.

Finally, it is important to know that in the etc. Directory there is the file huaca.conf. This archive lists the variables with the addresses of the files and contents of the program. So, thanks to it the user will be able to modify many parameters of the program without creating any conflicts.

5. Guides:

This section will focus on creating small guides on specific functionalities. We will begin with detailing the management of the domains. We will follow the administration of the containers and the visualization of the whole environment. Finally we will make an introduction to the control and monitoring of resources.

5.1 Domains:

First of all to correctly understand the project it is necessary to define what a domain is. A domain is a set of containers separated from the rest. A practical example could be the following: a user of a school has two classes with 5 computers in each. So you could define two domains, one per class and create 5 containers in each. Another possibility to define domains could be in the same class with several computers. We can create as many domains as classes are taught in this class and in each one put as many containers as computers have.

Once it is clear what the domains are, it is important to learn how to handle them. The first thing of all will be to know the available tools to manage them. There is the huaca-certification command that allows the user to manage the certification authorities for each domain. It also includes a huaca-net service that will raise the domains to the start of the host OS. Finally, it has the command huaca-domain, specific to manage the rest of functionalities.

To begin, the two possible ways of displaying the status of the containers are shown below:

1. Status parameter
2. List parameter

The status parameter let the user know the state of every domain, if it has an active authority certification and the IP range assigned to each one.

```
root@rgenes_jose:~# huaca-domain status
 Status:
 [RUNNING] alpha          CA=active      IP=172.20.1.0/24
 [RUNNING] beta           CA=none        IP=172.20.2.0/24
 [RUNNING] charlie        CA=active      IP=172.20.3.0/24
 [RUNNING] delta          CA=active      IP=172.20.4.0/24
```

The second option, list parameter, let the user list all the containers that have the domains together with the useful information.

```
root@rgenes_jose:~# huaca-domain list
- [RUNNING] alpha
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
a1   RUNNING 0         -      172.20.1.68  -    alpha
a2   RUNNING 0         -      172.20.1.207 -    alpha
a3   STOPPED 0         -      -            -    alpha
a4   STOPPED 0         -      -            -    alpha
a5   STOPPED 0         -      -            -    alpha
c2   STOPPED 0         -      -            -    alpha

- [RUNNING] beta
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
b1   RUNNING 0         -      172.20.2.180 -    beta
b2   STOPPED 0         -      -            -    beta
b3   STOPPED 0         -      -            -    beta

- [RUNNING] charlie
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
c1   RUNNING 0         -      172.20.3.83 -     charlie
c2   RUNNING 0         -      172.20.3.85 -     charlie

- [RUNNING] delta
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
d1   RUNNING 0         -      172.20.4.227 -    delta
```

Once the user has visualized the domains, it can proceed to create new once. That it is possible with the command create that allow the user to create a new domain with the desired name.

```
root@rgenes_jose:~# huaca-domain create domain1 domain2
Creating domain domain1...
Finding a valid IP range...
Valid IP: 172.20.5.0/24
Creating domain domain2...
Finding a valid IP range...
Valid IP: 172.20.6.0/24
```

After that, the parameter “start” will start the domain.

If the user uses the parameter status at this moment will observe a list of the domains with all the IP assigned and the status will appear RUNNING.

```
root@rgenes_jose:~# huaca-domain start domain1
root@rgenes_jose:~# huaca-domain status
 Status:
 [RUNNING] alpha          CA=active       IP=172.20.1.0/24
 [RUNNING] beta           CA=none         IP=172.20.2.0/24
 [RUNNING] charlie        CA=active       IP=172.20.3.0/24
 [RUNNING] delta          CA=active       IP=172.20.4.0/24
 [RUNNING] domain1        CA=none         IP=172.20.5.0/24
 [STOPPED] domain2        CA=none
root@rgenes_jose:~#
```

At the same time, the user can execute the following parameters: stop to unset a domain or destroy to delete it definitely.

```
root@rgenes_jose:~# huaca-domain stop domain1
root@rgenes_jose:~# huaca-domain destroy domain1
root@rgenes_jose:~# huaca-domain status
 Status:
 [RUNNING] alpha          CA=active       IP=172.20.1.0/24
 [RUNNING] beta           CA=none         IP=172.20.2.0/24
 [RUNNING] charlie        CA=active       IP=172.20.3.0/24
 [RUNNING] delta          CA=active       IP=172.20.4.0/24
 [STOPPED] domain2        CA=none
```

Finally, the user can create a certification authority for each domain, so that certificates can be issued for their containers. It is possible to create, delete and list the certificates with the parameter certification.

```
root@rgenes_jose:~# huaca-certification ca create -d domain2
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/huaca/certification/CAs/domain2.
Your public key has been saved in /etc/huaca/certification/CAs/domain2.pub.
The key fingerprint is:
SHA256:pWda2hqJsSTCU3AQ8N5L/9lOIhgkMf49mAKgpvTgPlw domain2 CA
The key's randomart image is:
+---[RSA 4096]----+
|o.*o.            |
|oo =             |
|o++ o     .      |
|=+o* +   o       |
|..*EO = S +      |
|o .= B = O       |
| +  o = * o      |
|  .    o B       |
|        +.o      |
+----[SHA256]-----+
ls: cannot access '/var/lib/huaca/domain2': No such file or directory
root@rgenes_jose:~# huaca-certification ca list
alpha
charlie
delta
domain2
root@rgenes_jose:~# huaca-certification ca delete -d domain2
ls: cannot access '/var/lib/huaca/domain2': No such file or directory
root@rgenes_jose:~# huaca-certification ca list
alpha
charlie
delta
```

All the information about domains and its management have been explained in that section.

5.2 Containers:

Once the domains are defined, the user is ready to create containers inside them. A container is an operating system isolated from the host, so it accesses directly to the kernel and does not make all the requests through the host, in this way we gain in efficiency. OpenHuaca focuses the majority of its commands in having a wide and comfortable management of these.

In order to start with this second guide it is necessary to create a container. This can be done in two ways; the first is to create it from a template of the official LXC repository.

```
root@rgenes_jose:~# huaca-create -t ubuntu -d alpha -n a11
Checking cache download in /var/cache/lxc/xenial/rootfs-amd64 ...
Copy /var/cache/lxc/xenial/rootfs-amd64 to /var/lib/huaca/alpha/a11/rootfs ...
Copying rootfs to /var/lib/huaca/alpha/a11/rootfs ...
Generating locales (this might take a while)...
  en_US.UTF-8... done
Generation complete.
Creating SSH2 RSA key; this may take some time ...
2048 SHA256:cM63qQX7u3eEDugLN9rEK911QpCFuFqxL2oj98JU678 root@rgenes_jose (RSA)
Creating SSH2 DSA key; this may take some time ...
1024 SHA256:ad2Wo0hKRMJmsO0p1ZwxDp+03yXC0/jQiDrfzArvXxo root@rgenes_jose (DSA)
Creating SSH2 ECDSA key; this may take some time ...
256 SHA256:dPSxa5mRtrChWyET5eWOcOa//HoJEsIVd+dHmZJrHdY root@rgenes_jose (ECDSA)
Creating SSH2 ED25519 key; this may take some time ...
256 SHA256:O6QZ/LwvYu4uhgxMJ63ruKqKuECzJZz9hI/z4gUnjP0 root@rgenes_jose (ED25519)
invoke-rc.d: could not determine current runlevel
invoke-rc.d: policy-rc.d denied execution of start.

Current default time zone: 'Etc/UTC'
Local time is now:      Mon Jun 19 23:49:40 UTC 2017.
Universal Time is now:  Mon Jun 19 23:49:40 UTC 2017.


##
# The default user is 'ubuntu' with password 'ubuntu'!
# Use the 'sudo' command to run tasks as root in the container.
##
```

The other option consists on creating a proper template; that can be listed with the command huaca-bases and generated with huaca-build.

Once the container is created, it can be run with huaca-start or stop with huaca-stop.

```
root@rgenes_jose:~# huaca-start -d alpha -n a11
root@rgenes_jose:~# huaca-stop -d alpha -n a11
```

The visualization of the containers can be carried out with two commands, huaca-ls and huaca-info. The first one, huaca-ls, allows the user to list all the containers of each domain in a very comfortable way to parse it.

```
root@rgenes_jose:~# huaca-ls
- alpha:
a1   a11  a2   a3   a4   a5   c2
- beta:
b1   b2   b3
- charlie:
c1   c2
- delta:
d1
- domain1:
root@rgenes_jose:~# huaca-ls -f
NAME     STATE   AUTOSTART GROUPS IPV4 IPV6     DOMAIN
a1   RUNNING 0         -      172.20.1.68  -    alpha
a11  STOPPED 0         -      -            -    alpha
a2   RUNNING 0         -      172.20.1.207 -    alpha
a3   STOPPED 0         -      -            -    alpha
a4   STOPPED 0         -      -            -    alpha
a5   STOPPED 0         -      -            -    alpha
c2   STOPPED 0         -      -            -    alpha
b1   RUNNING 0         -      172.20.2.180 -    beta
b2   STOPPED 0         -      -            -    beta
b3   STOPPED 0         -      -            -    beta
c1   RUNNING 0         -      172.20.3.83  -    charlie
c2   RUNNING 0         -      172.20.3.85  -    charlie
d1   RUNNING 0         -      172.20.4.227 -    delta
```

On the other hand, OpenHuaca has the command huaca-info with which the user can obtain more detailed information of a container, such as the state, the IP, the network traffic consumed or even the resources allocated.

```
root@rgenes_jose:~# huaca-info -d alpha -n a11
Name:            a11
State:           STOPPED
root@rgenes_jose:~# huaca-info -d alpha -n a1
Name:            a1
State:           RUNNING
PID:             14649
IP:              172.20.1.68
CPU use:         31.02 seconds
BlkIO use:       173.15 MiB
Memory use:      17.55 MiB
KMem use:        0 bytes
Link:            vethXB89K4
 TX bytes:       293.68 KiB
 RX bytes:       299.12 KiB
 Total bytes:    592.80 KiB
```

After managing the state of the containers and their visualization, there are two more commands that must be explained to manage the users of each container and finally the certificates of the container.

The command huaca-user gives the user the possibility of creating, deleting and listing the users of one or various domains.

```
root@rgenes_jose:~# huaca-user list -d alpha -n a11
ubuntu
root@rgenes_jose:~# huaca-user create -d alpha -n a11 -u huaca -p huaca123
root@rgenes_jose:~# huaca-user create -d alpha -n a11 -u sudo_huaca -p sudo_huaca123 -s
root@rgenes_jose:~# huaca-user list -d alpha -n a11
ubuntu
huaca
sudo_huaca
```

In the capture bellow, there is the creation of a huaca user and another sudo_huaca, the latter with the -s parameter so that it has management permissions. The utility of this permissions are explained in a following section.

Now, the user can delete the user Ubuntu with the help of the parameter delete:

```
root@rgenes_jose:~# huaca-user delete -d alpha -n a11 -u ubuntu
userdel: ubuntu mail spool (/var/mail/ubuntu) not found
root@rgenes_jose:~# huaca-user list -d alpha -n a11
huaca
sudo_huaca
```

Once the control of the users is explained the next command related to the container management, more specifically with the administration of certificates is the huaca-certification; it let the user to can list the certificates, update them and create new ones.

```
root@rgenes_jose:~# huaca-certification cert create -d alpha -u huaca -w 1
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /etc/huaca/certification/certs/alpha_huaca-99bbf489.
Your public key has been saved in /etc/huaca/certification/certs/alpha_huaca-99bbf489.pub.
The key fingerprint is:
SHA256:wH2p98MHZ9YOFj9PTEi2gcXUjV1UQbHkT6wUuyeNfm0 alpha_huaca-99bbf489 CA
The key's randomart image is:
+---[RSA 4096]----+
|             =OXX|
|     . .   ..o=B=|
|      o . o   *o+|
|       . o   . %.|
|        S . . @.O|
|         . o B *+|
|            + o E|
|             o o |
|                 |
+----[SHA256]-----+
Enter passphrase:
Signed user key /etc/huaca/certification/certs/alpha_huaca-99bbf489-cert.pub: id "99bbf489" serial 0 for huaca v
alid from 2017-06-20T00:11:00 to 2017-06-27T00:12:56
root@rgenes_jose:~# huaca-certification cert list -u huaca
Certificate                    Domain  Name    Created         Expire          Serial  Key ID
alpha_huaca-99bbf489-cert.pub  alpha   huaca   2017-06-20      2017-06-27      0       99bbf489
```

In the above capture it is observed that knowing the passphrase of the domain can generate the desired certificate indicating the domain, the user and the weeks of validity.

Once created the necessary certificates the users can access the containers in different ways.

The System administrator can access through the namespaces, only changing its terminal by one inside the container without needing any type of authentication.

```
root@rgenes_jose:~# huaca-attach -d alpha -n all
root@all:/#
```

The other option is the access enabled to the users of the platform through ssh connections. The generated certificate is entered in the connection and the user can access with the requested user.

```
root@rgenes_jose:/etc/huaca/certification/certs# ssh sudo_huaca@all.alpha -i alpha_sudo_huaca-478e8b5d
The authenticity of host 'all.alpha (172.20.1.212)' can't be established.
ECDSA key fingerprint is SHA256:dPSxa5mRtrChWyET5eWOcOa//HoJEsIVd+dHmZJrHdY.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'all.alpha,172.20.1.212' (ECDSA) to the list of known hosts.
Welcome to Ubuntu 16.04.2 LTS (GNU/Linux 4.4.0-66-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

The user will be able to access without the need of a password, just with the certification.

5.3 Resources:

Finally, the user can monitor and edit the resources assigned to a container. For the monitoring OpenHuaca has the top Debian command applied to the containers, which creates a table with the consumption of cpu in real time.

Container Name	CPU Used	CPU Sys	CPU User	BlkIO Total	Mem Used
a1	31.15	16.16	11.25	178.90 MB	19.69 MB
a11	0.58	0.38	0.14	25.00 KB	13.92 MB
a2	29.79	15.07	11.42	188.14 MB	16.61 MB
TOTAL 3 of 3	61.52	31.61	22.81	367.06 MB	50.21 MB

All of this part is extended from lxc. For more information, see the LXC manuals.

In order to end the resource section, highlight huaca-cgroups with which the user can modify the resources assigned to a container, such as cpu cycles, available memory or the size of the filesystem.

All this part has been inherited by the commando lxc-cgroups, so all the necessary information can be found on its version of LXC.

6 Environments:

This section explains in detail the configuration of the predefined environments, the first one will be a basic environment in order to familiarize the user to the OpenHuaca commands. The next subsection it will detail the creation of a real simulation, based on a group of teaching laboratories.

6.1 Basic environment

This first environment is a basic simulation; so the following information describes the creation of a domain and a unique container. Then, it will be added some users, one with root privileges. Finally, the certificates will be created to allow access to the users. Once checked the correct working of the environment it will be detailed the way to destroy it and leave the host installation clear.

First of all it is necessary to install the Debian package. It can be done with different installations; all of them are into the official Ubuntu repositories like **dpkg** or **gdebi**.

```
gdebi -n huaca*.deb

dpkg -i huaca*.deb
```

In order to know if the installation have been done correctly, the user can check it executing any command with the parameter version that shows you the current version of LXC and OpenHuaca.

```
huaca-create –version
```

Once checked, the next step is creating the first domain, it will be called ***alpha***.

```
huaca-domain create alpha
```

The output of the command shows the network assigned, for example in this case: 172.20.1.0/24. The parameter status let the user to consult the states of the domains.

```
huaca-domain status
```

After executing the command below, the user can observe a list with the whole domains generated, in this case just one, and can observe that is stopped and it does not have any certificate authorization, CA. This means that the user is not able to make certificates to this domain.

First of all, the user has to start the domain. This action allows the creation of containers and CAs. A domain with an active CA can generate certificates to let the user access in the containers. In this case, the name of the domain, alpha, it is set as the CA password.

```
Huaca-domain start alpha
```

```
Huaca-certification ca create –d alpha
```

If the parameter status is executed, it is observed that the domain has the previous assigned network, the CA is active and the state is RUNNING.

```
Huaca-domain status
```

Once the domain is created, the user can generate the container, in this example it is used a LXC template called Ubuntu 16.04, with the following command just indicating the name and the domain:

```
Huaca-create –t Ubuntu –d alpha –n a1
```

The user must have to wait until the template is completely downloaded, after that the base is kept in the cache and creates the container from it. This LXC template is stored in the computer, so if the user wants to create a second container with it, the time of processing will be considerably reduced. In order to start to work with the container, it has to be started with the following command:

```
Huaca-start –d alpha –n a1
```

After that, the dnsmasq has to assign a direction to the container, this process can take a time, and finally configures the container in the background.

```
Huaca-domain list

Huaca-ls –f
```

The administrator has access to any of the machines as root with the help of the command below that uses namespaces:

```
Huaca-attach –d alpha –n a1
```

In order to enable access to the users of the environment created, it is necessary to generate a list of users. By default the system has created the user "Ubuntu". So, a pair of new users is incorporated to the environment with the commands below, one without administrator permissions and the other with.

```
Huaca-user create –d alpha –n a1 –u huaca –p huaca123

Huaca-user create –d alpha –n a1 –u superhuaca –p superhuaca123 –s
```

The parameter list it can be used to see if the generation of the users have been done correctly:

```
Huaca-user list –d alpha –n a1
```

The next step is creating a certificate per user with the parameter –w the user can specify the period of validation, expressed in weeks.

```
Huaca-certificate cert create –d alpha –u huaca –w 1

Huaca-certificate cert create –d alpha –u superhuaca –w 52
```

Against the user can validate the task done before with the parameter list:

```
Huaca-certificate cert list –d alpha
```

Once the environment is completely configured, the last step is giving the certificates to the users in order to let them connect to the virtual machines. All certificates are stored in the "/ etc / huaca / certification / cert /" directory. It is very important not to delete them from this directory in order to keep a control of the issued certificates and to be able to send them again in case the user loses his proper.

To confirm that the user has access, the best way is to test the ssh connection.

```
Ssh huaca@a1.alpha –i /etc/huaca/certification/cert/alpha-huaca*
```

Observe, the superhuaca user is in the sudo group and therefore has administration permissions, while not huaca.

Finally, in order to destroy the current environment the user has to execute the following commands.

```
Huaca-user delete –d alpha –n a1 –u huaca

Huaca-user delete –d alpha –n a1 –u superhuaca

Huaca-stop –d alpha –n a1

Huaca-destroy –d alpha –n a1

Huaca-certification ca delete –d alpha

Huaca-domain stop alpha

Huaca-domain destroy alpha
```

After that the container has been deleted together with the CAs and the domain. The best option to prove that is using the two commands below:

```
Huaca-ls –f

Huaca-domain status
```

6.2 Real simulation – Teaching laboratories

This section simulates the creation of a real environment, specifically a teaching laboratory with three classes, in each class there will be 2 computers and in each container the user "teacher" will be established and the corresponding student too.

First of all, the creation of the domains:

```
Huaca-domain create class1 class2 class3
Huaca-domain start class1 class2 class3
Huaca-certification ca create –d class1
Huaca-certification ca create –d class2
Huaca-certification ca create –d class3
```

In order to check if the domains are well configured use the following command:

```
Huaca-domain status
```

Now, it is possible to create and initialize the containers of each domain:

```
Huaca-create –t Ubuntu –d class1 –n pc1 && huaca-start –d class1 –n pc1
Huaca-create –t Ubuntu –d class1 –n pc2 && huaca-start –d class1 –n pc2
Huaca-create –t Ubuntu –d class2 –n pc1 && huaca-start –d class2 –n pc1
Huaca-create –t Ubuntu –d class2 –n pc2 && huaca-start –d class2 –n pc2
Huaca-create –t Ubuntu –d class3 –n pc1 && huaca-start –d class3 –n pc1
Huaca-create –t Ubuntu –d class3 –n pc2 && huaca-start –d class3 –n pc2
```

The checking of the creation can be done with just one command:

```
Huaca-ls –f
```

Once checked, the user can create the new users, differentiating the teacher from the students because the teacher will have permissions.

```
Huaca-user create –d class1 –n pc1 –u teacher –p teacher -s
```

```
Huaca-user create –d class1 –n pc2 –u teacher –p teacher -s
Huaca-user create –d class2 –n pc1 –u teacher –p teacher -s
Huaca-user create –d class2 –n pc2 –u teacher –p teacher -s
Huaca-user create –d class3 –n pc1 –u teacher –p teacher -s
Huaca-user create –d class3 –n pc2 –u teacher –p teacher -s

Huaca-user create –d class1 –n pc1 –u student –p student
Huaca-user create –d class1 –n pc2 –u student –p student
Huaca-user create –d class2 –n pc1 –u student –p student
Huaca-user create –d class2 –n pc2 –u student –p student
Huaca-user create –d class3 –n pc1 –u student –p student
Huaca-user create –d class3 –n pc2 –u student –p student
```

At least, in the following box there is the creation of the specific certificates for every user with a validation of one year:

```
Huaca-certification cert create –d class1 –u teacher –w 52
Huaca-certification cert create –d class2 –u teacher –w 52
Huaca-certification cert create –d class3 –u teacher –w 52
Huaca-certification cert create –d class1 –u student –w 52
Huaca-certification cert create –d class2 –u student –w 52
Huaca-certification cert create –d class3 –u student –w 52
```

After this process, the user has generated three domains with two computers each; every one contains a “teacher” user with administration permissions and one “student” user without permissions. The certifications needed to access to the containers have been created using ssh connections like:

```
Ssh $user@$container.$domain –i $cert
```

7 Work in Progress:

In this part of the document the updates and future developments of the project are described and planned.

- **In Progress:**

→ Gerard Implementing the web interface.

→ Jorge Implementing Terraform plugin

→ Rafa Purging and revoking the functionalities of huaca-certification.

- **Tasks to do:**

  Implement KVM.

  Create some bases.

  Update the manuals.

  Rewrite project commands in Python.

It should be noted that this first version of the document, in order to be presented in Rafa's TFG thesis, only includes the work done by him. The next versions of the document will include the work done by all project partners.

8 Summary

Description	Command
Create domain	huaca-domain create domain
Destroy domain	Huaca-domain delete domain
List domains	Huaca-domain status
Create CA	Huaca-certification ca create –d domain
Delete CA	Huaca-certification ca delete –d domain
Create container	Huaca-create –t template –d domain –n name
Delete container	Huaca-destroy –d domain –n name
Start container	Huaca-start –d domain –n name
Stop container	Huaca-stop –d domain –n name
List containers	Huaca-ls –f
Create user	Huaca-user create –d domain –n name –u user –p pass
Create super user	Huaca-user create –d domain –n name –u user –p pass -s
Delete user	Huaca-user delete –d domain –n name –u user
List users	Huaca-user list –d domain –n name
Create cert	Huaca-certification cert create –d domain –u user –w 52
List cert	Huaca-certification cert list
Attach to a container	Huaca-attach –d domain –n name