CHALLENGE 4 – ATTACK!!

(RETAKE EARTH)

BY JAMES BROWN (@JBCOMPVM)

Contents

Executive Summary

We have secured the human race on Mars and the Moon. It is now time to take back our home planet. Teams of robots and humans are being sent back to Earth to secure locations across the globe. Infrastructure will need to be setup. No IT personnel will be sent on this mission. The IT team has been tasked with creating a test scenario. When the team lands on Earth there will be mixed equipment and OS software available. We need to come up with multiple scenarios.

The Web server must run in a container. Two different operating systems, orchestration tools, and webservers need to be evaluated.

No IT personnel will be sent on this mission. All instructions need to be documented and provided to the invasion team.

If you accept this challenge, the Human race may be able to retake their home planet.

Requirements

1. Webserver must display “Welcome Back to Earth!”
2. Two different orchestrations tools
3. Two different OS
4. Two different Webservers
5. Web server must run inside a container
6. Code will be deploy by non-technical personnel

Constraints

1. Power and cooling could be hard acquire
2. The design must incorporate use of unknown or Frankenstein hardware

Assumptions

1. Internet will be available through a satellite connection
2. There will be access to a computer console (monitor, keyboard, mouse)
3. Communication link between Earth Team and IT Team needs to be established

Risks

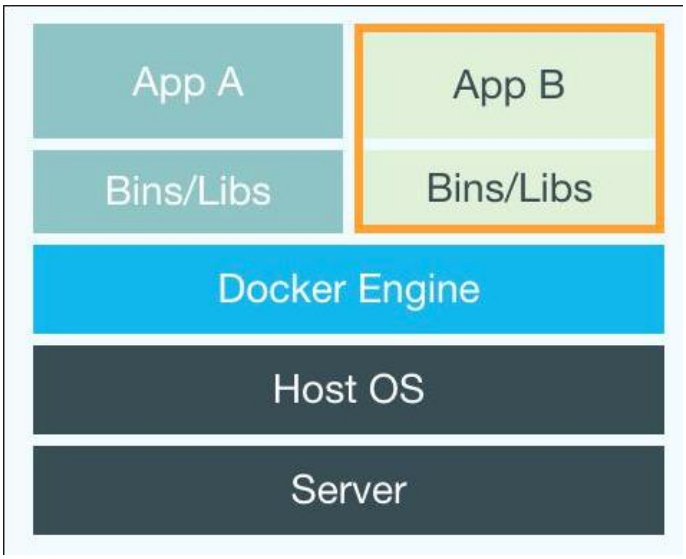
1. Docker Hub could be offline
2. Electricity
3. Hardware left behind
4. Earth Team will not have to knowledge
5. Zombies could capture the teams
6. Aliens could aid the zombies
7. Zombies have devoured all computer systems

Conceptual Design

Automation

The IT Team been given the task to create two webserver that can be deployed on any operating system.

Docker is widely available on Windows, Linux, and UNIX systems. Docker can run/install on top of any operating system base. Including a virtual guess OS.



Architecture Design

Physical Design

All hardware specifications are unknown until the HAZ (Human Against Zombies) team have landed and secured the location to bring the web server online.

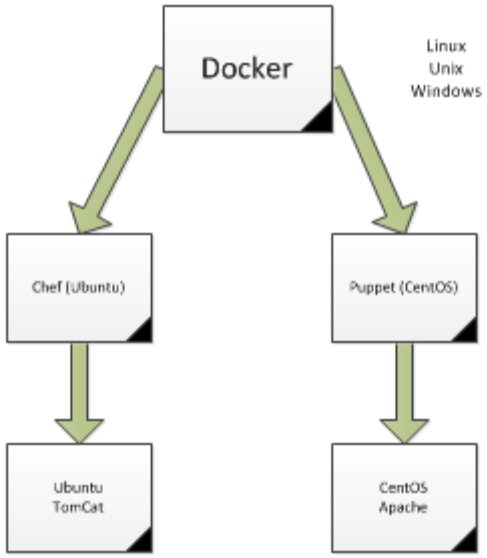
When the HAZ team land on Earth, it will have been one year sense the human race had fled. With coal, nuclear, and hydro power station offline, because of lack of supervision, electricity will be hard to come by. HAZ will need to find small generators to power small racks of hardware.

These data centers that were left on Earth were heavily designed with compute, storage, and networking. If these server were not maintained, all system will be offline. Systems of this size require the correct boot order. A communication link between the HAZ team and the IT Team will need to be established. If possible video and remote control software need to be in place to provide the utmost support for this team.

Logical Design

Docker will be used as the base/starting point for this design. The onsite personnel will need to install Docker on a server that has been acquired. This can be on Linux (CentOS and Ubuntu) and/or Windows. If a virtual infrastructure is still online you can create a guest VM to install Docker.

Server	vCPU	RAM (GB)	OS Disk (GB)	Data Disk (GB)	Quantity
Docker	2	8	80	200	1



Application Design

Docker

Docker is the key piece for this deployment. All server that will be deployed will run within a container. Docker will be installed on any physical hardware system that is still running. (CentOS, Ubuntu, or Windows)

CentOS 7 will be used for the purpose of this design. Bootable USB images of CentOS 6 and 7, Ubuntu 14.04 and 12.04, and Windows 2012 R2 will be label and brought to earth.

If for some reason, the Docker Hub is offline skip down to the **Backup Image Registry** section.

Linux Install (CentOS)

Requirments:

Docker requires a 64-bit installation regardless of your CentOS version. The kernel must be 3.10 or above.

To check your current kernel version, open a terminal and use “uname –r” to display your kernel version.

Install:

1. Log into your machine as a user with sudo or root privileges.
2. Make sure your existing yum packages are up-to-date.
    a. $ sudo yum update
3. Run the Docker installation script.
    a. $ curl -sSL https://get.docker.com/ | sh
    b. This script adds the `docker.repo` repository and installs Docker.
4. Start the Docker daemon.
    a. $ sudo service Docker start

Linux Install (Ubuntu)

Requirements:

Docker requires a 64-bit installation regardless of your CentOS version. Also, your kernel must be 3.10 at minimum.

To check your current kernel version, open a terminal and use “uname –r” to display your kernel version

Trusty 14.04

There are no prerequisites for this version.

For Precise 12.04 (LTS)

For Ubuntu Precise, Docker requires the 3.13 kernel version. If your kernel version is older than 3.13, you must upgrade it. Refer to this table to see which packages are required for your environment:

linux-image-generic-lts-trusty	Generic Linux kernel image. This kernel has AUFS built in. This is required to run Docker.
linux-headers-generic-lts-trusty	Allows packages such as ZFS and Virtual Box guest additions which depend on them. If you didn't install the headers for your existing kernel, then you can skip these headers for the "trusty" kernel. If you're unsure, you should include this package for safety.
xserver-xorg-lts-trusty	Optional in non-graphical environments without Unity/Xorg. *Required* when running Docker on machine with a graphical environment. To learn more about the reasons for these packages, read the installation instructions for backported kernels, specifically the LTS Enablement Stack.
libgl1-mesa-glx-lts-trusty	

To upgrade your kernel and install the additional packages, do the following:

1. Open a terminal on your Ubuntu host.
2. Update your package manager.
    a. $ sudo apt-get update
3. Install both the required and optional packages.
    a. $ sudo apt-get install linux-image-generic-lts-trusty

    Depending on your environment, you may install more as described in the preceding table.
4. Reboot your host.
    a. $ sudo reboot

For Saucy 13.10 (64 bit)

Docker uses AUFS as the default storage backend. If you don't have this prerequisite installed, Docker's installation process adds it.

Install:

1. Log into your Ubuntu installation as a user with sudo privileges.
2. Verify that you have wget installed.
    a. $ which wget

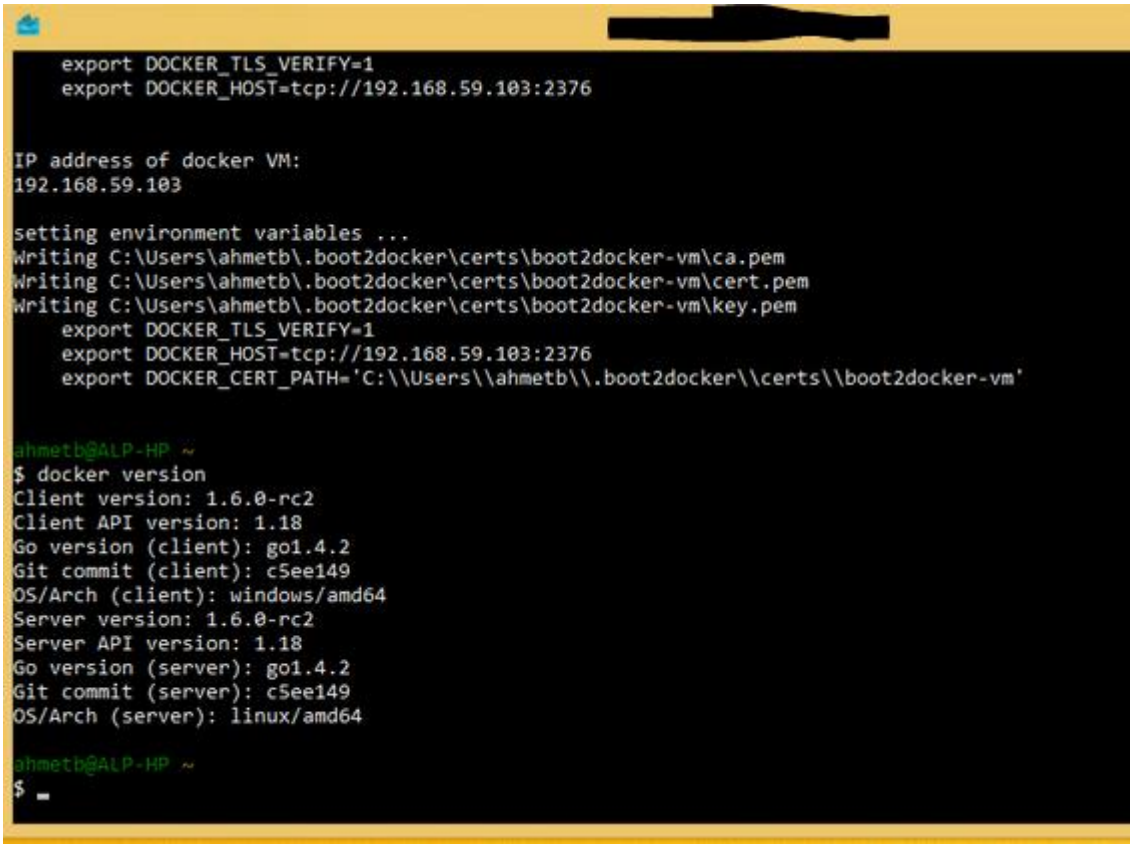
    If wget isn't installed, install it after updating your manager:
    $ sudo apt-get update
    $ sudo apt-get install wget
3. Get the latest Docker package.
    a. $ wget -qO- https://get.docker.com/ | sh

Windows Install

1. Download the latest release of the Docker for Windows Installer.

2. Run the installer, which will install Docker Client for Windows, Virtual Box, Git for Windows (MSYS-git), the boot2docker Linux ISO, and the Boot2Docker management tool.

3. Run the Boot2Docker Start shortcut from your Desktop or "Program Files → Boot2Docker for Windows". The Start script will ask you to enter an ssh key passphrase - the simplest (but least secure) is to just hit [Enter].
4. The Boot2Docker Start will start a UNIX shell already configured to manage Docker running inside the virtual machine. Run Docker version to see if it is working correctly:

5. 

Running Docker

Note: if you are using a remote Docker daemon, such as Boot2Docker, then do not type the sudo before the Docker commands shown in the documentation's examples.

Boot2Docker Start will automatically start a shell with environment variables correctly set so you can start using Docker right away:

Let's try the hello-world example image. Run

$ Docker run hello-world

This should download the very small hello-world image and print a Hello from Docker.

You can find these instructions at https://docs.docker.com/installation/ .

Puppet Master Container

The first container will run Puppet. Puppet will push a Docker image that includes CentOS, Apache, PHP, etc. Puppet will also setup an index.php website that displays "Welcome Back to Earth!"

1. Log into the Docker Server
2. Download puppet image for Docker
    a. Docker pull macadmins/puppetmaster
3. Status: Downloaded newer image for macadmins/puppetmaster:latest
4. To preserve the Puppet certificates so that they're not lost if the Puppetmaster container is removed.
    a. Docker run -d --name puppet-data --entrypoint /bin/echo macadmins/puppetmaster Data-only container for puppetmaster
5. Start the container
    a. Docker run -d --name puppetmaster -h puppet -p 8140:8140 --volumes-from puppet-data macadmins/puppetmaster
6. We need to populate the Puppet configuration.
    a. Docker exec puppetmaster cp -Rf /etc/puppet /opt/
7. If Puppet has started we should see at least on Certification
    a. Docker exec puppetmaster puppet cert list –all

Web server with puppet

First add an IP address for this webserver on the Docker server. These container images below are available on the Docker Hub. https://registry.hub.docker.com/u/jbcompvm/vdm-centos/

- ip addr add 10.251.0.5/24 dev <Ethernet name>

Apply Docker image with Puppet

1. Go to the Docker system
2. Find the container ID for Puppet mast
    - Docker ps
        i. Locate Container ID for puppetmaster
    - Docker exec –I –t <Container ID> bash
3. Install puppet Docker module
    - Puppet module install garethr-docker
4. Create web.pp <nano web.pp>

```
include 'Docker'

Docker::image { 'jbcompvm\vdm-centos':

    version      => 'latest',

    image_tag => web1,

    tcp_bind     => 'tcp://10.253.0.5:80',
```

```
    manage_kernel => false,
}
```

5. Apply web.pp (type in puppet master window)
    - puppet apply web.pp

Chef Container

This container will run Chef Server. Chef will push a Docker image that includes Ubuntu, Tomcat, etc. Chef will also setup an index.html website that displays "Welcome Back to Earth!"

We need to first add an IP address for this webserver on the Docker server

- ip addr add 10.251.0.6/24 dev <Ethernet name>

Now we need to create a running image with that IP address.

- Docker run -t -d --name chefserver -p 10.251.0.6:80:80 ubuntu

1. Log into the Docker Server
2. Download puppet image for Docker
    a. Docker pull willoucom / chef-server-Docker
3. Status: Downloaded newer image for willoucom / chef-server-docker:latest
4. Start the container
    a. Docker run --privileged -dti --name chef_server -p 443:443 willoucom / chef-server-Docker

Web server with Chef

First add an IP address for this webserver on the Docker server. The container images below are available on the Docker Hub. https://registry.hub.docker.com/u/jbcompvm/vdm-ubuntu/

- ip addr add 10.251.0.6/24 dev <Ethernet name>

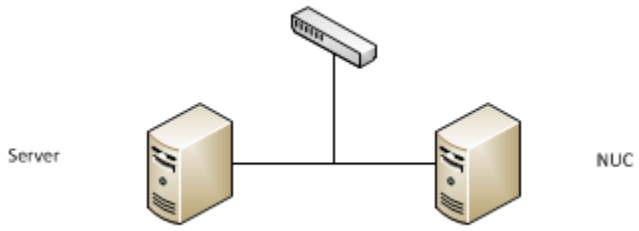
Apply the Docker image with Chef

1. Go to the Docker system
2. Find the container ID for Chef Server
    - Docker ps
3. Locate Container ID for chefserver
    - Docker exec –I –t <Container ID> bash
4. Install Chef Docker module
    - Chef gem install knife-container
5. Docker init
    - Container 'web2' --IP 10.253.0.6/24

Backup Image Registry

If you are reading this section nothing has gone according to plan. The IT Team has planned for the worst. In you backpack is an Intel NUC5i5RYH, 128GB USB stick and a netgear GS105NA 5-Port Gigabit Switch.

Boot the NUC with the 128GB USB stick installed. A local Docker Registry has been installed on this NUC.



The NUC will be running CentOS 7.

ISO images of each Operating system will be stored on a second 128GB USB disk.

To attach this USB drive and configure apt-get to search the ISO (Ubuntu):

- mkdir /mnt/dvd
- /mnt/storage/iSO/debian-i386-DVD-1.iso /mnt/dvd/ udf,iso9660 loop 0 0
- deb file:/mnt/dvd/ wheezy main contrib
- mount /mnt/dvd/
- apt-get update

To attach the USB and configure yum to search the ISO (CentOS):

- mkdir /mnt/cdrom
- `mount /dev/cdrom /mnt/cdrom/`
- `nano /etc/yum.repos.d/CentOS-Media.repo`

```
## Add the following lines ##

file:///mnt/cdrom/

gpgcheck=1

enabled=1

gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-6
```