



PDD

Project Management Report 3



Table of Contents

Table of Contents	1
Executive Summary	2
Intro	3
Statement of Community Partner Problem	
Statement of Business Context	
Members	
Scope and Requirements	4
Project Scope	
Project Requirements	
Requirements Matrix	7
Definitions	8
Environment: PROXMOX Tutorial	10
Environment: Chef Overview	14
Bootstrapping to Chef Using Chef DK	16
LXC Configuration for Cloning in ProxMox	21
Windows 7 KVM Unattended Bootstrap	26
Daily Summary	30
Schedule of Events	50
Username and Password List	51



Executive Summary

This report summarizes the work completed by Poly Development and Deployment in the 2017 winter quarter for the Boeing Forensics senior project. The report includes the tasks completed by the group, the scope, the contract with the client, requirements matrix and various tutorials for the different programs used.

Successes: As of the end of the quarter, the group has developed both KVM and LXC templates which will be used for cloning multiple virtual machines. In addition, a Chef server has been set up which will manage the bootstrapped nodes and will store future cookbooks. We have also developed the bash script used to bootstrap the LXC container and the powershell commands used to bootstrap the KVM.

Challenges: We did not have access to Proxmox until week and the internet was unstable after we received access. The firewall configuration for the VM in the initial attempt at knife bootstrapping interfered with the bootstrapping process. In addition, the learning curve for this project was high due to having to learn multiple new technologies such as Chef, Ruby, Powershell and bash scripting. The Windows bootstrapping also proved to be difficult due to the additional authentication required for Windows. After writing the knife code to bootstrap the nodes, we realized that the hostname would conflict if not changed on the LXC prior to running the bootstrap code. The SID and hostname would need to be changed on the Windows to avoid conflict. The random hostname unintentionally begins with a prefix of Juan followed by a random string of letters and numbers.

Recommendations and Solutions: This project required more research and trial and error than expected due to using many new technologies that we were tasked to learn in order to complete the objectives. We were able to solve many of the issues that occurred due to the hours of research and trial and error. Our project provides the foundation for future projects because of the functionality and resources that Chef offers.



Intro

The purpose of this report is to provide a dynamic narrative of what our group has and will complete for this project. This report also provides readers with a step by step account of what we did to complete this project. In addition to descriptions of various resources we used, this report contains descriptions on how we used our time to get to this stage of the project.

Statement of Community Partner Problem

The Boeing forensics class is run at Cal Poly Pomona to teach Boeing employees the fundamentals of Computer Forensics. The main issue with the project is the time it takes to build it. The project takes many hours to setup, configure, and deploy. Typically, the class is only ran during spring quarter, therefore it gets destroyed or forgotten after a year has passed. This causes the project to be constantly reimagined and recreated year after year. Automating the project will standardize the way the class is run; and hence preventing this endless loop of inefficiency.

Statement of Business Context

The Boeing Forensics class is a course that happens once a year and requires downloading and installing many programs.. Automating this process would save time and resources that are wasted yearly, setting up this class. This project was previously proposed and attempted in the past. The team that attempted to complete the project left various sections of the project incomplete. Our team will look at the pieces that the previous team did not accomplish and attempt to modify and edit their parts to complete the automation of the project. In addition to modifying the other team's files, we need to bootstrap various templates we develop for the virtual machines to accommodate Chef.

Members

Alvin Wadkins

Penuel Chow

Ameen Chowdhary

Qing Yue Wang

Joe Lagana

Veronica Roswell

Kyle Valino



Scope and Requirements

Project Scope

This project will consist of creating Chef node templates (LXC and KVM) that are ready to be deployed. The nodes will be stored in our Chef Server and the client should be able to access them to deploy virtual machines. The project will be completed by March 5, 2017.

Our primary goals are KVM and LXC templates that can be activated by an admin. The templates are to make virtual machines that are both connected to the Chef Server and have unique hostnames.

Project Requirements

1. Set up Chef Server which enables uploading of cookbook
2. Create Nodes, install Chef Client
3. Set up workstations containing Chef DK
4. Develop LXC Template
5. Develop KVM Template
6. Develop script to change hostname of LXC
7. Develop script to change hostname of KVM



Basic Contract and Terms

Definitions:

First Party: **Juan Ortega** representing Cal Poly Pomona

Second Party: Poly Development and Deployment Team consisting of

**Penuel Chow, Ameen Chowdhary, Joe Lagana, Veronica Roswell,
Kyle Valino, Alvin Wadkins, and Qing Yue Wang**

This contract is entered into and between the First Party and the Second Party. The term of this Agreement will become effective on 1/18/2017 and shall continue until 3/9/2017.

The goal to be assigned to the Second Party as defined by this Agreement, is the automation of the Boeing computer forensic class environment.

The specific tasks of this Agreement are as follows:

1. Bootstrap Virtual Machine with Chef Client
 - LXC template to be developed
 - KVM template to be developed
2. Review, evaluate, and modify pieces of incomplete previous project provided by the First Party
3. Install programs provided by First Party, including activation of Windows 7 and Microsoft Office

In consideration of the agreement detailed above, the Second Party agrees that it shall meet often and communicate on a regular basis, learn and divide the tasks needed to accomplish the goal, talk with the First Party often to explain how the project is coming along, and do everything to the best of their ability to accomplish the goal before the end date of the Agreement.

In consideration of the agreement detailed above, the First Party agrees that it shall provide the Second Party with as much assistance as they request to accomplish the goal, and be available to answer questions or to point to resources to answer questions that the Second Party may have with regards to accomplishing the goal before the end date of the Agreement.

This contract cannot be modified in any way unless such modifications are made in writing and signed by both Parties. This document constitutes the entire agreement between the Parties.



Basic Contract and Terms

This Contract is legally binding upon the Parties, their successors, and heirs, and will be enforced according to the laws of California.

Additional Comments or Clarifications:

It is agreed. By signing below, the Parties agree to be bound by the terms of this Agreement.

[Signature of First Party]

[Signatures of Second Party]

Ameen Chaudhary

-QING YUE WANG

-Penuel Chow

Veronica Rosenthal

Joe Lagana

Joe Lagana

Kyle Valino

ALVIN WADKINS

Date: Jun 18, 2017



Requirements Matrix

#	Requirement Name	Description	Test Case
1	Set up Chef server	A chef server is where we upload our cookbooks. The server also allows us to see the nodes that we are currently managing. In our case, we are using a hosted server by Chef. Setup Chef accounts for management.	Once we can login and connect to the server, set up has been completed.
2.	Set up Chef Workstations	A Chef workstation is a computer (physical or virtual) where we make and upload our cookbooks. In this case, we will use our own computers.	Downloading and creating Chef DK on our computers and connecting to the chef server with knife config.
3.	Prep Linux Container	Install chef-client coupled with checking firewall options in order to connect to the Chef server.	Verify installation of Chef-client
4.	Create startup script for Linux Container	Create a bash script to change the hostname to a random string to differentiate the virtual machines.	Hostname changes
5.	Prep Windows Template	Install chef-client coupled with checking firewall options in order to connect to the Chef server.	Verify installation of Chef-client
6.	Create startup script for Windows Template	Create a PowerShell script to change the hostname to a random string to differentiate the virtual machines.	Hostname changes
7.	Set up Chef nodes	A chef node is any computer (physical or virtual) that is managed by the Chef Server. These computers have the Chef client downloaded on them. In this case, we will use the virtual machines set up on our Proxmox server.	Logging onto Chef.io and being able to see and manage the nodes.



Definitions

- **Bootstrap:** Bootstrap is a process that installs the chef-client on a target system so that it can run as a chef-client and communicate with a Chef server. After the bootstrap process, the node should appear on the chef server.
- **Knife bootstrap vs unattended bootstrap:**
 - Knife bootstrap: The knife bootstrap subcommand is used to run a bootstrap operation that installs the chef-client on the target node. The knife command is used to SSH into the target machine and do what is needed to allow the chef-client to run on the node. The chef-client is installed, keys are generated and the node is registered with the Chef server.
 - Unattended bootstrap: Unattended bootstrap is when the chef-client is manually installed on the prospective node instead of the knife command installing the program. A command is still needed to bootstrap the machine and make it appear on the Chef server.
- **KVM hypervisor:** A KVM hypervisor is a Windows virtual machine that is turned into a template. The template can then be cloned to make additional virtual machines. In order to clone a KVM, the template must be sysprepped and the powershell command must change the hostname so the SID and hostname will not have conflict with the original after cloning.
- **SID:** The Security Identifier is a unique identifier of the machine used for Windows.
- **LXC container:** A LXC container is a Linux virtual machine that is turned into a template. The template can then be cloned to make additional virtual machines. To clone an LXC container, the hostname must be changed in order to avoid conflict. To change the hostname, a bash script can be used.
- **Sysprep:** Sysprep is a program preinstalled on Windows that can be run to facilitate changing the SID.
- **Audit mode vs out of box:**
 - Audit mode: Audit mode allows the user to install applications, add device drivers, run scripts and test a Windows installation. Audit mode launches the virtual machine straight to the desktop.
 - OOBE: Out of the box experience forces the virtual machine to the Windows welcome screen where the user can customize Windows.

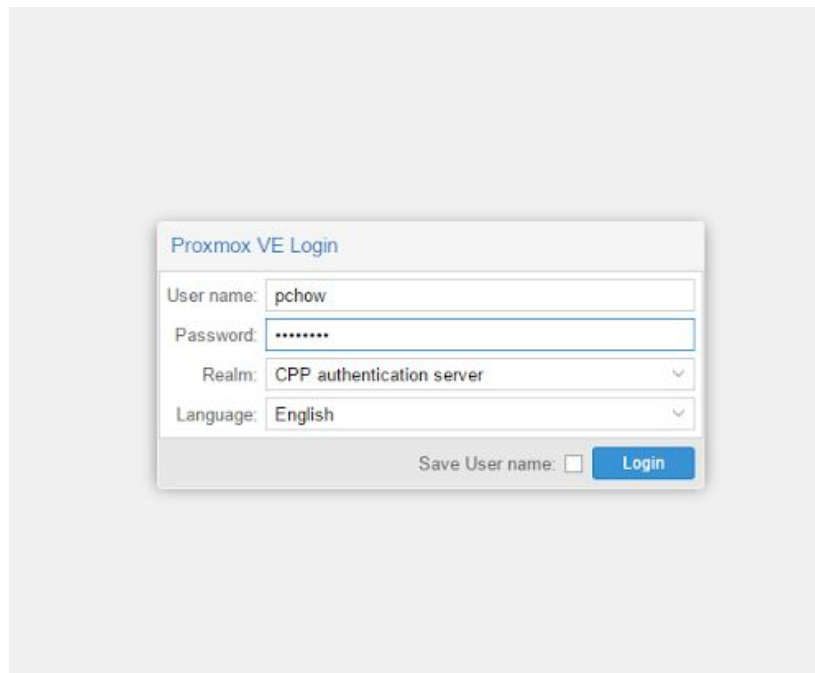


- **Bash script:** A bash script is a text file that contains a series of commands such as changing the hostname.
- **Powershell:** Powershell is a task automation and configuration management framework in Microsoft.
- **Ruby:** Ruby is an object-oriented programming language commonly used in conjunction with Chef.

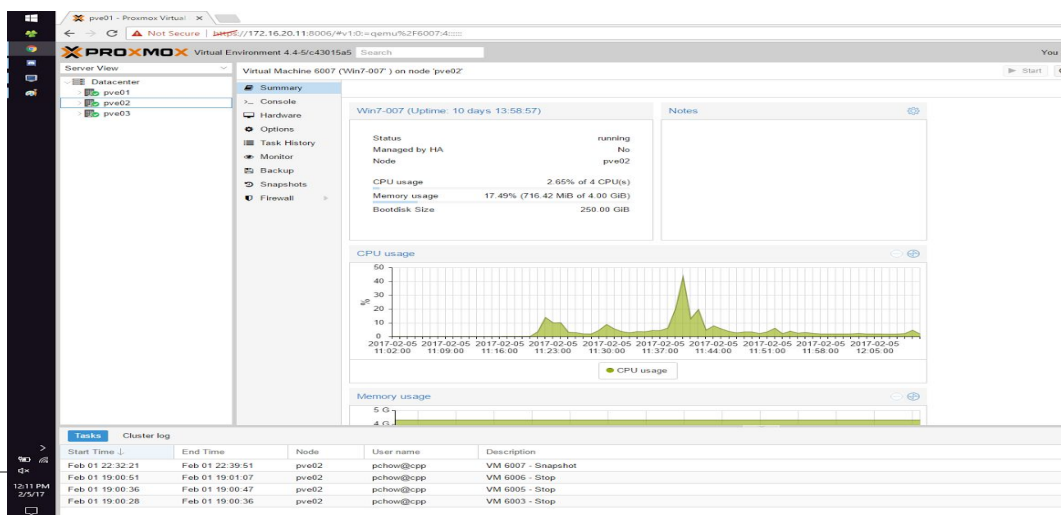
Environment: PROXMOX Tutorial

Our servers are setup through the open-source environment ProxMox which lets us manage virtual servers. The following is a small tutorial on how to navigate the ProxMox environment and to access pre-made virtual machines.

- 1.) ProxMox allows for authentication and users with different accesses. The client gave us personal accounts allowing us to log on to his server and the ProxMox environment.

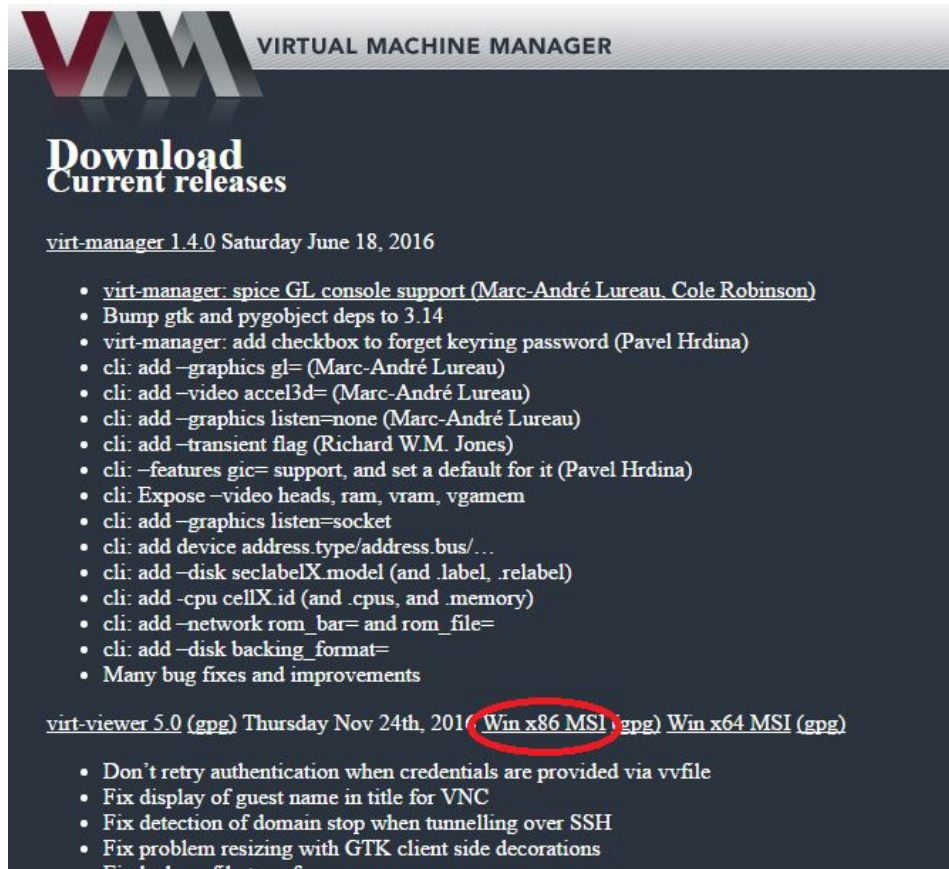


- 2.) Once logged in, the user can manage and check the different servers being managed on the left hand side.

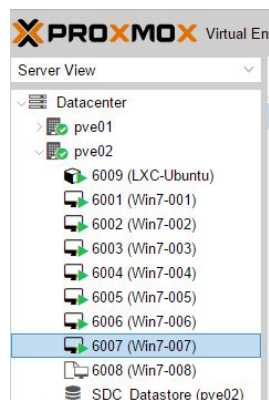




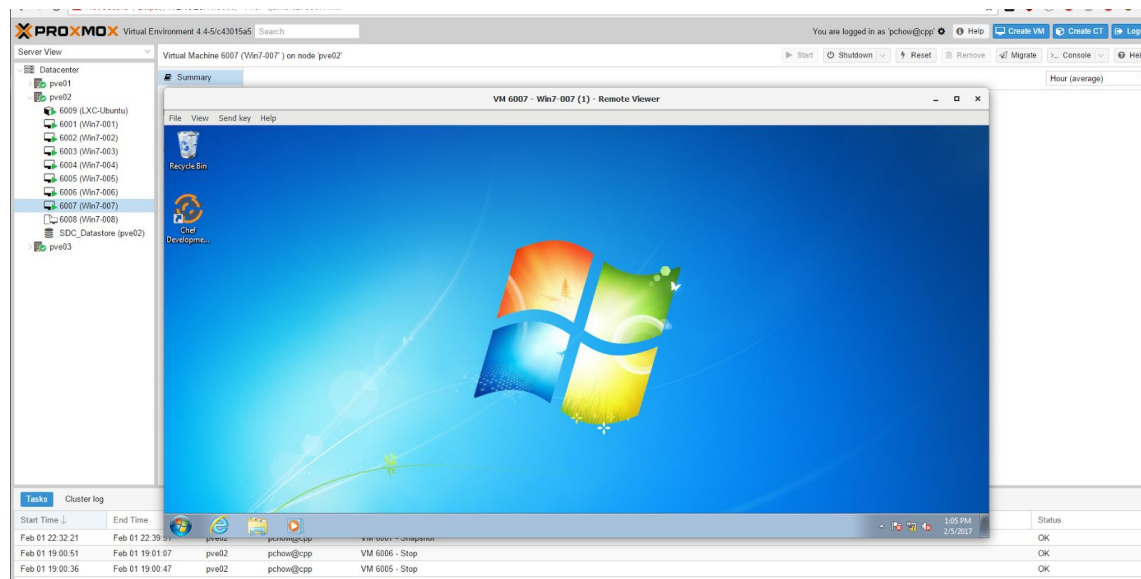
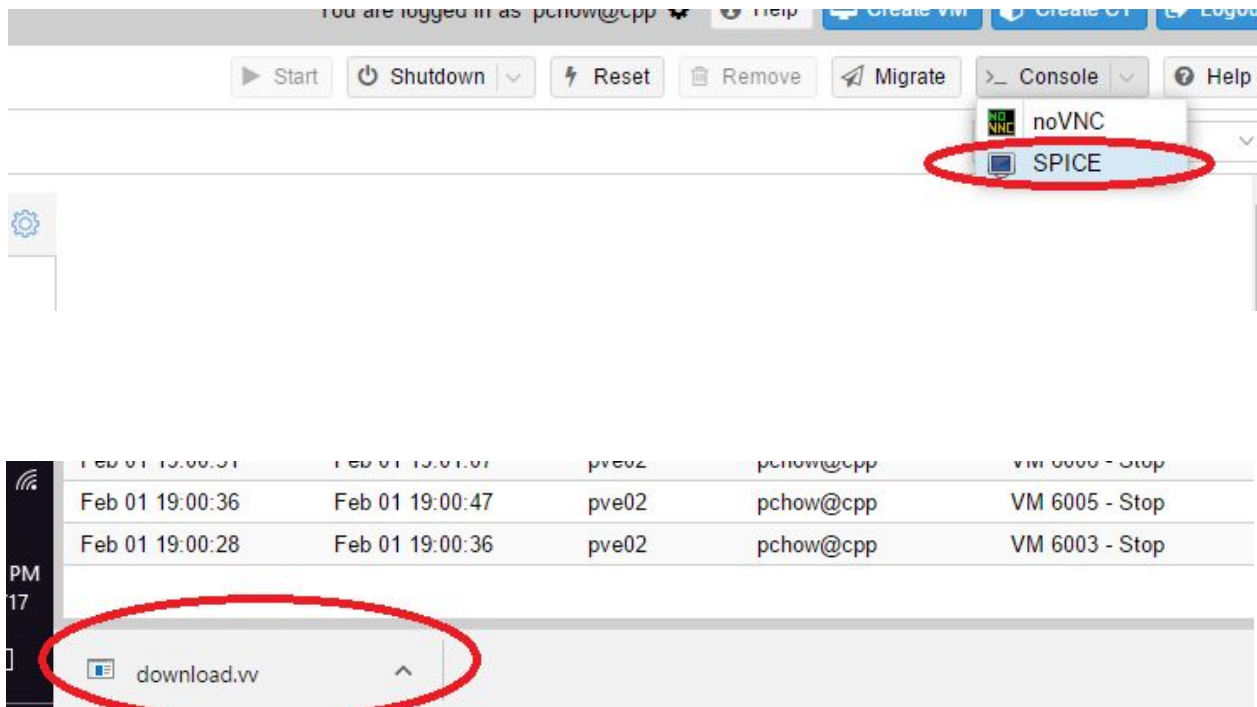
3.) The client set-up virtual machines in which we could use to test the environments. ProxMox has its own console which allows us to view and work with the virtual machine. Due to input lag, the program Virt-Viewer allows us to access another console that reduces the lag. Virt-viewer can be downloaded from <https://virt-manager.org/download/>.



4.) To start up a virtual machine, highlight it on the left hand side under the server node. Our server node is pve02 and in this example we will be using virtual machine 6007.



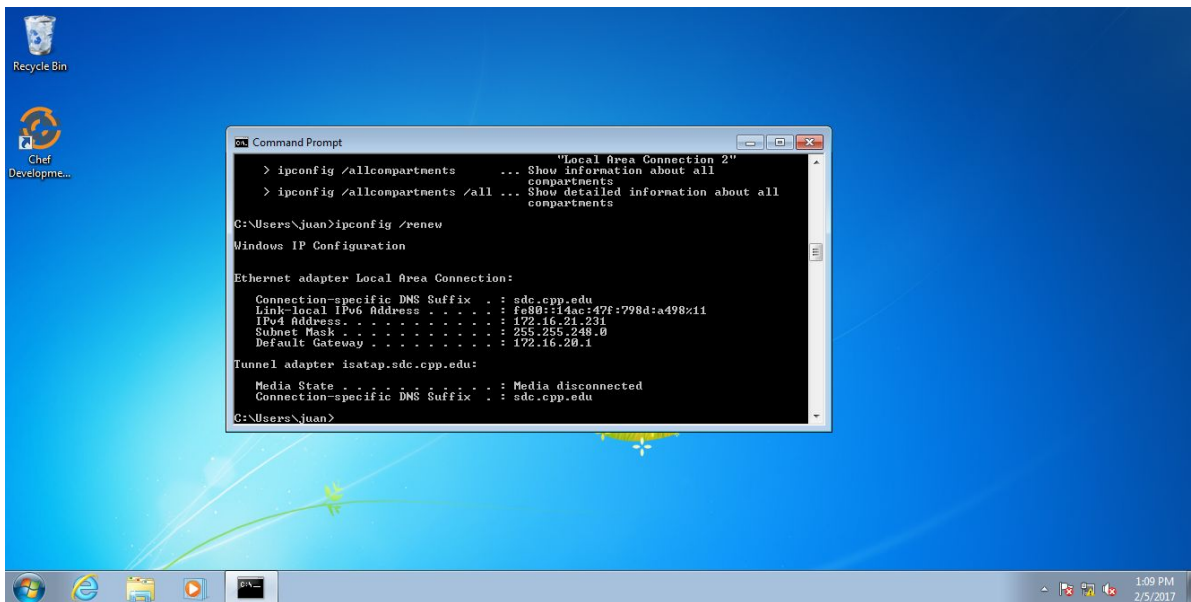
5.) Once highlighted, on the right hand side, select the arrow next to console and select “SPICE”. This sets up a temporary “download.vv” file which opens up the virt-viewer program. The download file is only temporary and does not last once opening. This is for security reasons and every time you want to access the virtual machine once you close it, the same process is needed to open it up again. The “noVNC” option is also available, however this is ProxMox’s console and as explained earlier, this console is laggy and is not advisable.



Troubleshooting:

Locked out- When the virtual machine is booted up, often times the administrator account is locked. However, if you simply switch user and log back in, the account should be able to be accessed.

Internet on the VM does not work- If the internet does not work on the virtual machine, start the command prompt and type in “ipconfig /renew”. This renews all of the network adapters.





Environment: Chef Overview

Chef is a management tool written in Ruby that allows for Chef recipes to be uploaded and deployed by a server. These recipes are essentially scripts that lets you turn infrastructure and operations into deployable code. Chef is used as a means to achieve efficiency, as the forensics classes would need a multitude of different programs for use in the class. Installing each program on each individual machine would be tiresome and the Chef client would allow each virtual machine to be deployed quickly and with the required software.

The system used by Chef comprises of three different parts: the workstation, the Chef server, and the nodes. The workstation is the computer where code for recipes are written and tested. You can also administrate the server from the workstation. This is done through the use of the Chef Development Kit (ChefDK) and the user can interact with the chef server using various tools. Recipes are written in Ruby and then can be authored on the workstation through ChefDK. These recipes can be organized into cookbooks, which provide some structure to recipes. The Chef server is where the recipes and cookbooks are saved. This server also allows us to interact with the nodes. The nodes are any machines (physical or virtual) that are under management by the Chef server. These nodes perform the configuration tasks onto the machine. Chef servers can be hosted by Chef themselves, on a self-run server, or through cloud services such as AWS or Azure. In our scenario, we are using the Chef server through Chef's hosting. This option is free up to five nodes. The following is a picture of our server hosted on Chef along with the current nodes it manages.

Another major aspect of the Chef environment is the Chef supermarket. The Chef

CHEF MANAGE						
Nodes Reports Policy Administration Feedback On						
> Nodes Delete Manage Tags Reset Key Edit Run List Edit Attributes	Showing All Nodes					
	Node Name	Platform	FQDN	IP Address	Uptime	Last Check-In
	node1-windows	windows	WIN-B35JQ000QKT	192.168.103.131	14 days	3 days ago

supermarket is an online repository that houses cookbooks written by the general public and community. Cookbooks on the supermarket can be used by any and in general comprise of very general templates.



The recipes or cookbooks are deployable scripts written in the ruby programming language. The recipes are simple, consisting of a few lines of codes that run with the chef-client. These recipes and cookbooks can be easily modified to suit the needs of the organization.

```
unless node['platform_family'] == 'windows'
  return "Chocolatey install not supported on #{node['platform_family']}"
end

Chef::Resource.send(:include, Chocolatey::Helpers)

install_ps1 = File.join(Chef::Config['file_cache_path'], 'install.ps1')

template install_ps1 do
  action :create
  backup false
  source 'InstallChocolatey.ps1.erb'
  variables :download_url => node['chocolatey']['install_vars']['chocolateyDownloadUrl']
end

powershell_script 'Install Chocolatey' do
  environment node['chocolatey']['install_vars']
  cwd Chef::Config['file_cache_path']
  code install_ps1
  not_if { chocolatey_installed? && (node['chocolatey']['upgrade'] == false) }
end

chocolatey 'wireshark' do
  version '2.2.1'
  action :install
end
```




Bootstrapping to Chef Using Chef DK

Setting up Chef guide:

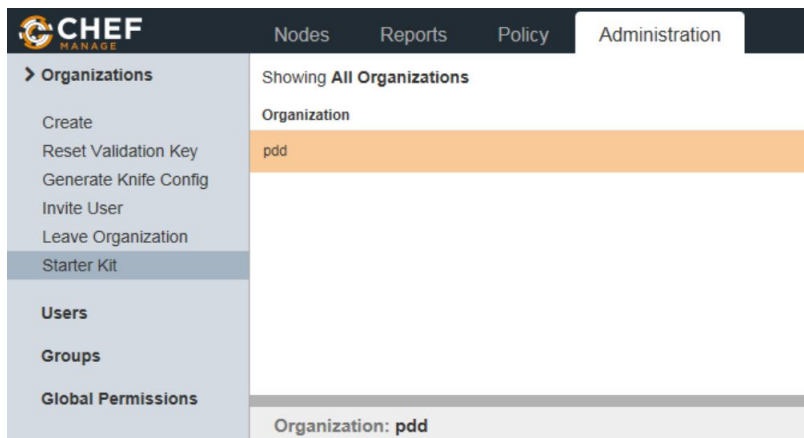
1. Install Chef Development Kit on the workstation. This is where you will create and maintain your recipes and cookbooks.
 - a. <https://downloads.chef.io/chef-dk/>
2. Create a hosted Chef account at <https://manage.chef.io/signup>

Start your free trial of Hosted Chef

You're one step away from access to all the power and flexibility of Chef. Get ready to automate your infrastructure, accelerate your time to market, manage scale and complexity, and safeguard your systems. Just complete the form to get started.

Full Name	<input type="text"/>
Company	<input type="text"/>
Email	<input type="text"/>
Username	<input type="text"/>
<input type="checkbox"/> I agree to the Terms of Service and the Master License and Services Agreement .	
<input type="button" value="Get Started"/>	

3. On your workstation, log in to your hosted chef server and create a Starter Kit, which downloads a folder titled Chef-Repo containing all necessary files you will need (knife.rb, organization-validator.pem, user.pem). Add this folder to a directory you'd like to work in.





Bootstrap to Chef server guide using ChefDK (Windows only):

1. On the Windows node, make sure you have admin access and a password set for the user.
2. Open CMD on the node and type the line below to set up Windows Remote Manager:

```
$ winrm quickconfig -q
```

3. Update WinRM settings:

To update these settings, run the following commands:

```
$ winrm set winrm/config/winrs '@{MaxMemoryPerShellMB="300"}'
```

```
<
```

and:

```
$ winrm set winrm/config '@{MaxTimeoutms="1800000"}'
```

```
<
```

and:

```
$ winrm set winrm/config/service '@{AllowUnencrypted="true"}'
```

```
<
```

and then:

```
$ winrm set winrm/config/service/auth '@{Basic="true"}'
```

4. Type command “winrm get winrm/config/service/auth”. Options should be set like below:

```
Administrator: Command Prompt

ProviderFault
WSManFault
Message = The WS-Management service cannot process the configuration request. The XML contains an unknown URI: http://schemas.microsoft.com/wbem/wsman/1/config/services/auth.

Error number: -2144108485 0x8033803B
The WS-Management service cannot process the request. The resource URI is missing or it has an incorrect format. Check the documentation or use the following command for information on how to construct a resource URI: "winrm help uris".

C:\Windows\system32>winrm get winrm/config/service/auth
Auth
  Basic = true
  Kerberos = true
  Negotiate = true
  Certificate = false
  CredSSP = false
  CbtHardeningLevel = Relaxed

C:\Windows\system32>
```



5. Go to your workstation's working directory, where the Chef-Repo folder is, and type "knife bootstrap windows winrm <IP address of node> -x <Node username> -P <Node password> -N <name of node>" in the console. Answer "Yes" to the prompts following the command. A successful run should look like below:

```
Administrator: ChefDK (juan)
PS C:\Users\juan\chef-repo> knife wsman test 172.16.21.25 --manual-list
Connected successfully to 172.16.21.25 at http://172.16.21.25:5985/wsman.
PS C:\Users\juan\chef-repo> knife bootstrap windows winrm 172.16.21.25 -x Admini
strator -P Ieanpdd17 -N Win7-001
Node Win7-001 exists, overwrite it? (Y/N) y
Client Win7-001 exists, overwrite it? (Y/N) y
Creating new client for Win7-001
Creating new node for Win7-001

Waiting for remote response before bootstrap.172.16.21.25 .
172.16.21.25 Response received.
Remote node responded after 0.03 minutes.
Bootstrapping Chef on 172.16.21.25
172.16.21.25 Rendering "C:\Users\ADMINI~1\AppData\Local\Temp\bootstrap-3164-1486
695073.bat" chunk 1
172.16.21.25 Rendering "C:\Users\ADMINI~1\AppData\Local\Temp\bootstrap-3164-1486
695073.bat" chunk 2
172.16.21.25 Rendering "C:\Users\ADMINI~1\AppData\Local\Temp\bootstrap-3164-1486
695073.bat" chunk 3
172.16.21.25 Rendering "C:\Users\ADMINI~1\AppData\Local\Temp\bootstrap-3164-1486
695073.bat" chunk 4
172.16.21.25 Rendering "C:\Users\ADMINI~1\AppData\Local\Temp\bootstrap-3164-1486
695073.bat" chunk 5
172.16.21.25 Rendering "C:\Users\ADMINI~1\AppData\Local\Temp\bootstrap-3164-1486
695073.bat" chunk 6
```

```
Administrator: ChefDK (juan)
172.16.21.25 echo.node_name "Win7-001"
172.16.21.25 echo.log_level :info
172.16.21.25 echo.log_location STDOUT
172.16.21.25 > 1>C:\chef\client.rb
172.16.21.25 C:\Users\Administrator>(echo.<"run_list":[]>) 1>C:\chef\first-boot.
json
172.16.21.25 Starting chef to bootstrap the node...
172.16.21.25 C:\Users\Administrator>SET "PATH=C:\Windows\system32;C:\Windows;C:\
Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\opscode\che
fdk\bin\;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\ruby\bin\;C:\opscode\chef
\bin\;C:\opscode\chef\embedded\bin"
172.16.21.25 C:\Users\Administrator>chef-client -c c:/chef/client.rb -j c:/chef/
first-boot.json
172.16.21.25 [2017-02-09T18:56:28-08:00] INFO: *** Chef 12.18.31 ***
172.16.21.25 [2017-02-09T18:56:28-08:00] INFO: Platform: i386-mingw32
172.16.21.25 [2017-02-09T18:56:28-08:00] INFO: Chef-client pid: 1636
172.16.21.25 [2017-02-09T18:56:52-08:00] INFO: Setting the run_list to [] from C
LI options
172.16.21.25 [2017-02-09T18:56:52-08:00] INFO: Run List is []
172.16.21.25 [2017-02-09T18:56:52-08:00] INFO: Run List expands to []
172.16.21.25 [2017-02-09T18:56:52-08:00] INFO: Starting Chef Run for Win7-001
172.16.21.25 [2017-02-09T18:56:52-08:00] INFO: Running start handlers
172.16.21.25 [2017-02-09T18:56:52-08:00] INFO: Start handlers complete.
172.16.21.25 [2017-02-09T18:56:53-08:00] INFO: HTTP Request Returned 404 Not Fou
nd:
172.16.21.25 [2017-02-09T18:56:53-08:00] INFO: Error while reporting run start t
o Data Collector. URL: https://api.chef.io/organizations/pdd/data-collector Exce
ption: 404 -- 404 "Not Found" (This is normal if you do not have Chef Automate)
172.16.21.25 [2017-02-09T18:56:54-08:00] INFO: Loading cookbooks []
172.16.21.25 [2017-02-09T18:56:54-08:00] WARN: Node Win7-001 has an empty run li
st.
172.16.21.25 [2017-02-09T18:56:54-08:00] INFO: Chef Run complete in 2.109375 sec
onds
172.16.21.25 [2017-02-09T18:56:54-08:00] INFO: Running report handlers
172.16.21.25 [2017-02-09T18:56:54-08:00] INFO: Report handlers complete
172.16.21.25 [2017-02-09T18:56:54-08:00] INFO: Sending resource update report (r
un-id: 35dc92a5-88b3-4ee9-9053-37eaf00672c6)
PS C:\Users\juan\chef-repo>
```



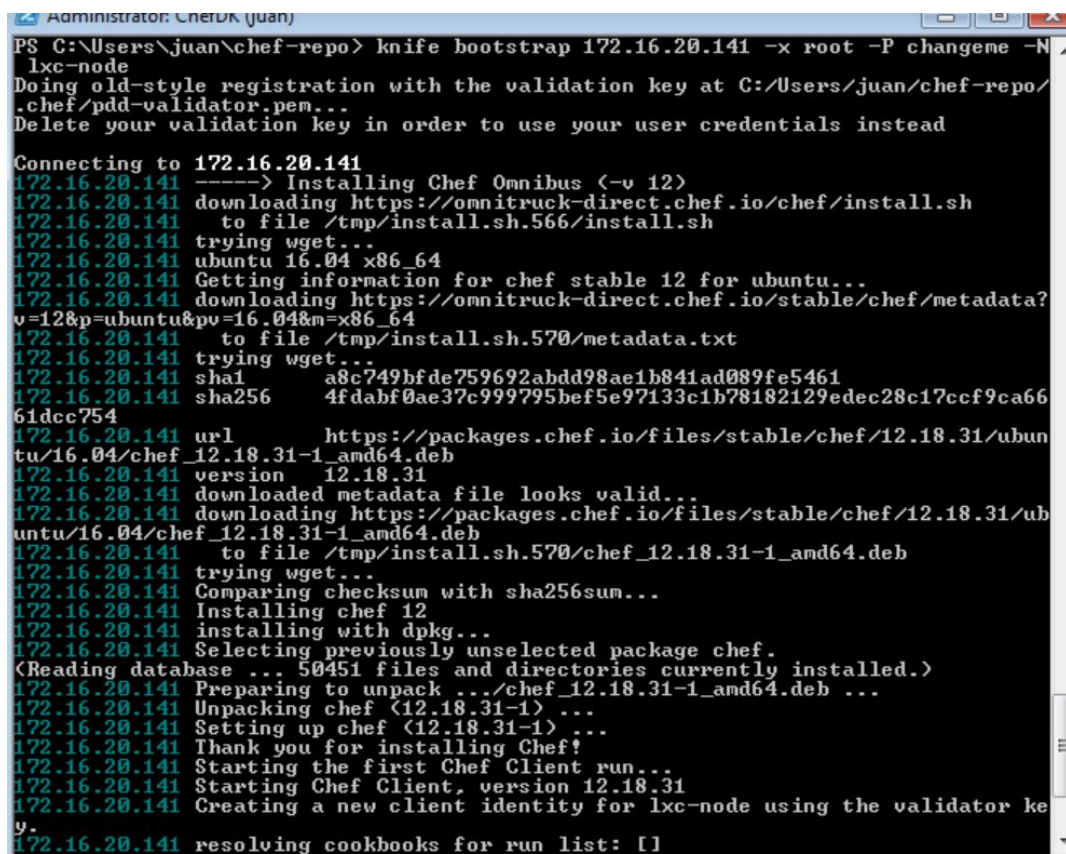
Bootstrap to Chef server guide using ChefDK (LXC only):

1. On the LXC node, first, make sure ports 5985 and 5986 for WinRM and port 22 for SSH are open. If not type the command below, adding the port number you want open after "dport":

If you want to open an incoming TCP port, type the following:

```
iptables -I INPUT -p tcp --dport 12345 --syn -j ACCEPT
```

2. After making sure all necessary ports are open, run command "knife bootstrap <ip address> -x <username> -P <password> -N <node name>" from your working directory on your workstation, where the Chef-Repo folder is. A successful run should look like below:



```
Administrator: ChefDK (juan)
PS C:\Users\juan\chef-repo> knife bootstrap 172.16.20.141 -x root -P changeme -N lxc-node
Doing old-style registration with the validation key at C:/Users/juan/chef-repo/.chef/pdd-validator.pem...
Delete your validation key in order to use your user credentials instead

Connecting to 172.16.20.141
172.16.20.141 -----> Installing Chef Omnibus (-v 12)
172.16.20.141 downloading https://omnitruck-direct.chef.io/chef/install.sh
172.16.20.141 to file /tmp/install.sh.566/install.sh
172.16.20.141 trying wget...
172.16.20.141 ubuntu 16.04 x86_64
172.16.20.141 Getting information for chef stable 12 for ubuntu...
172.16.20.141 downloading https://omnitruck-direct.chef.io/stable/chef/metadata?v=12&p=ubuntu&pv=16.04&m=x86_64
172.16.20.141 to file /tmp/install.sh.570/metadata.txt
172.16.20.141 trying wget...
172.16.20.141 sha1 a8c749bfde759692abdd98ae1b841ad089fe5461
172.16.20.141 sha256 4fdabf0ae37c999795bef5e97133c1b78182129edec28c17ccf9ca6661dccc754
172.16.20.141 url https://packages.chef.io/files/stable/chef/12.18.31/ubuntu/16.04/chef_12.18.31-1_amd64.deb
172.16.20.141 version 12.18.31
172.16.20.141 downloaded metadata file looks valid...
172.16.20.141 downloading https://packages.chef.io/files/stable/chef/12.18.31/ubuntu/16.04/chef_12.18.31-1_amd64.deb
172.16.20.141 to file /tmp/install.sh.570/chef_12.18.31-1_amd64.deb
172.16.20.141 trying wget...
172.16.20.141 Comparing checksum with sha256sum...
172.16.20.141 Installing chef 12
172.16.20.141 installing with dpkg...
172.16.20.141 Selecting previously unselected package chef.
(Reading database ... 50451 files and directories currently installed.)
172.16.20.141 Preparing to unpack .../chef_12.18.31-1_amd64.deb ...
172.16.20.141 Unpacking chef (12.18.31-1) ...
172.16.20.141 Setting up chef (12.18.31-1) ...
172.16.20.141 Thank you for installing Chef!
172.16.20.141 Starting the first Chef Client run...
172.16.20.141 Starting Chef Client, version 12.18.31
172.16.20.141 Creating a new client identity for lxc-node using the validator key.
172.16.20.141 resolving cookbooks for run list: []
```




```
172.16.20.141 ubuntu 16.04 x86_64
172.16.20.141 Getting information for chef stable 12 for ubuntu...
172.16.20.141 downloading https://omnitruck-direct.chef.io/stable/chef/metadata?v=12&p=ubuntu&pv=16.04&m=x86_64
172.16.20.141 to file /tmp/install.sh.570/metadata.txt
172.16.20.141 trying wget...
172.16.20.141 sha1 a8c749bfde759692abdd98ae1b841ad089fe5461
172.16.20.141 sha256 4fdabf0ae37c999795bef5e97133c1b78182129edec28c17ccf9ca6661dcc754
172.16.20.141 url https://packages.chef.io/files/stable/chef/12.18.31/ubuntu/16.04/chef_12.18.31-1_amd64.deb
172.16.20.141 version 12.18.31
172.16.20.141 downloaded metadata file looks valid...
172.16.20.141 downloading https://packages.chef.io/files/stable/chef/12.18.31/ubuntu/16.04/chef_12.18.31-1_amd64.deb
172.16.20.141 to file /tmp/install.sh.570/chef_12.18.31-1_amd64.deb
172.16.20.141 trying wget...
172.16.20.141 Comparing checksum with sha256sum...
172.16.20.141 Installing chef 12
172.16.20.141 installing with dpkg...
172.16.20.141 Selecting previously unselected package chef.
(Reading database ... 50451 files and directories currently installed.)
172.16.20.141 Preparing to unpack .../chef_12.18.31-1_amd64.deb ...
172.16.20.141 Unpacking chef (12.18.31-1) ...
172.16.20.141 Setting up chef (12.18.31-1) ...
172.16.20.141 Thank you for installing Chef!
172.16.20.141 Starting the first Chef Client run...
172.16.20.141 Starting Chef Client, version 12.18.31
172.16.20.141 Creating a new client identity for lxc-node using the validator key.
172.16.20.141 resolving cookbooks for run list: []
172.16.20.141 Synchronizing Cookbooks:
172.16.20.141 Installing Cookbook Gems:
172.16.20.141 Compiling Cookbooks...
172.16.20.141 [2017-02-16T01:37:35+00:00] WARN: Node lxc-node has an empty run list.
172.16.20.141 Converging 0 resources
172.16.20.141 Running handlers:
172.16.20.141 Running handlers complete
172.16.20.141 Chef Client finished, 0/0 resources updated in 33 seconds
PS C:\Users\juan\chef-repo>
```



LXC Configuration for Cloning in ProxMox

1. Copy /etc/hosts file and move it to /usr/local/etc. Rename the file “clonehosts” and change the hostname after 127.0.1.1 to “nh”. The file should look like below:

```
root@LXC-Ubuntu:/usr/local/etc# ls
clonehosts
root@LXC-Ubuntu:/usr/local/etc# cat clonehosts
127.0.0.1      localhost
::1           localhost ip6-localhost ip6-loopback
ff02::1       ip6-allnodes
ff02::2       ip6-allrouters
# --- BEGIN PVE ---
127.0.1.1 nh
# --- END PVE ---
root@LXC-Ubuntu:/usr/local/etc#
```

2. Create a file named “cloneconf.sh” and put it in /usr/local/sbin. This will be the configuration file that runs on startup. The contents of the file should look like this:

```
File View Send key Help
root@LXC-Ubuntu:/usr/local/sbin#
root@LXC-Ubuntu:/usr/local/sbin# cat cloneconf.sh
#!/bin/sh

echo "Guest Configuration After Cloning"
echo

rhex='cat /dev/urandom | tr -cd 'a-f0-9' | head -c 8'
nh="Ubuntu-$rhex"

if [ "$1" = "" ]; then
    hostname $nh
    echo -n "The hostname is now $nh"
else
    nh="$1"
fi

echo
echo "This script will configure the host with the following settings:"
echo Hostname: $nh
echo
echo /etc/hostname:
echo $nh
echo
echo /etc/hosts:
cat /usr/local/etc/clonehosts | sed -e "s/nh/$nh/g"
echo
echo $nh > /etc/hostname
cat /usr/local/etc/clonehosts | sed -e "s/nh/$nh/g" > /etc/hosts
echo
echo "Bootstrapping to Chef server..."

if test -f "/etc/chef/client.pem"; then
    rm /etc/chef/client.pem
fi
/usr/bin/chef-client

echo
echo "Removing 'cloneconf.sh' file..."

if grep -Fxq 'sh /usr/local/sbin/cloneconf.sh' /etc/rc.local; then
    sed -i 's/sh \.usr\/local\/sbin\/cloneconf.sh/d' /etc/rc.local
fi
rm -- "$0"
echo
echo "The clone configuration has completed. Hostname changed and bootstrapped to Chef server."
echo
root@LXC-Ubuntu:/usr/local/sbin#
```

The script will, first, generate a random series of letters and numbers and configure it as



the hostname. It will then replace the system's old hostname with this new hostname in /etc/hosts and /etc/hostname files. After this, the system will be added to Chef server by running the "chef-client" command. Finally, the configuration file will remove itself and delete the command in /etc/rc.local so that the hostname is not changed again on successive reboots.

3. Check if Chef-Client is installed on the system.
 - a. Type "chef-client --version" in console
 - b. If it is not, go to <https://downloads.chef.io/chef> and download the appropriate version.
4. Make sure the following files are in /etc/chef directory (You may need to create this directory yourself and add these files manually):

```
root@LXC-Ubuntu:/etc/chef# ls
client.rb  pdd-validator.pem
root@LXC-Ubuntu:/etc/chef#
```

- a. client.rb file:

```
client.rb
1 # See http://docs.chef.io/config_rb_knife.html for more informatio
2
3 current_dir = File.dirname(__FILE__)
4 log_level   :info
5 log_location STDOUT
6 validation_key "/etc/chef/pdd-validator.pem"
7 validation_client_name "pdd-validator"
8 chef_server_url "https://api.chef.io/organizations/pdd"
9
```

- b. Example of pdd-validator.pem (go to manage.chef.io → click on your organization → reset key to download)

```
pdd-validator.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEowBAQAQEAwb0jv1/DdHiXQ0eDYNQvYvD3hXX8J1xq0WYj1DqCvNRBcRA0
xYhL/meW07Xx8Rm1I0dPMf0pgM1XL+UisIPUwJNe/X0BEfoWvX0Vc06hYUHM6kZ
GogVnwEiHdLmbScjmjmhJQMIhPz/1JCs4VWdiwLq+oSGVVCvF0UVEyS6K9+avV0
KIDwKUP+zKKBuv1xGHVBFx6G3o63gNXT9NuXi1tWidZEBZ12X25kdVpj1NawypOT
37o53+9+RpbTL+YlFAn11GAq+00hE0Jf9U7D27IE5MUPoZa+nDn9BMGeIBSgW6Z0
qE5GncJtCAR5K16cLLpqf6vJ8i+MaBmosdAbwIDAQABAQIBAF3Iz2i5Qj07QeIP
bbHNKmBaUU3CEWmXyNqjIjNYXQJ2NnU6poJZhyCFIVgWMMNkbk03GuYR1LXPPjKH
YDY4YWFHdhpEq7uZ24e1KayKA2KAqYJt3F6amBw56i41kaESfT7UzRS1exo2BU
6A9194Bg//Xpgmz+UrboL6sEFjDaJ6tFxfge7EqJLqUjWTkq0Fcs1RL3ypIhJkN8
NluqN02ksM4ih3G3VLjh/S2BNuoN/R5rG554fADLIuY6XgV6aesz+mldc6lxsK
28TnuYpGtRAN0nxJa4sAQReis3GqjhaEAo276TLg+7owXjlxEcmpYfnyz0x90fN
XirFAKcGYEA55DdwIz3wjRqpljh9EK3R/g+AtkiBJ3Uxo26E6iaHtnzgrFEirLL
c+4tJoz3K8SrmX707k1IINwYis9zu5JKyI37/Krk2TVxwNYZ+J12D3W0Rc1A9Jr0
YgXs1Ed0RoB04MXy46T0SuoG0TN1VfBS3x/5Es6PucnluUbcXp1yUCgYEA11P5
X0khXtSGiMNXGnFX8CKGMVtyBD7tPisF9jIGljFquBawbJsEdow2tEcsuva8b32
rb6f47njftafBVFeagnIIHivKHsSL/ajsv0X6uy2f3uPdMTYwSPEGwLuzo5S3Ix
C9sEC+rLq0wDcWysRukuEf7c2ygZDc5LhRlTBscCgYEA26GCZCNCB/1N3fL0jzq+
8iYXHAarT05Kgb/Qf43+20jb4k5m56jHceS00XE2MyV+I/zFC2N6GB54s0RARDDU
Ux0UbUKiPeyA2cJfuR8e+8+TFDntWaYtGE1tiqrjJE/p6fszxVBBtL/0z6pzMScL
Z1Tzq7SUPlCvm8pEWp7eBI0CgYBLOHXHh9vGA0eFMQDu0/Os8a0rwTzUJ8sTJerl
sYu3bAajtyeKNM+Lcc1nbd8j5n+RPuzif7yQd0K50r73qU0vT+b2n0LzM3n+GfAa
pJoh5SE909qtMvDzbk2G2j1AtkSer8+0t2zejWyD7196xuS9yPxsFb2UN6EH/wd
M+kbTWKbgCqNm/7QZwcq/SeVLMZ031hc7/Zer4e/Ee7LzkdQEDVgAy9ZGn3up
UFh+3uim9wPhhSYik2S3JGfOfd99CB5xLpK3RDYh0Buev75xzz4e0V07Gm75SeR+
+MqLFXj157cFRSymVstMuAXKLVODfLVnvwkHsQNgIzsrbWDPHMCJ
-----END RSA PRIVATE KEY-----
```



5. Type “nano /etc/rc.local” as root to edit and add the command “sh /path/to/file.sh” to the end of the file:

```
GNU nano 2.5.3 File: /etc/rc.local

#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

sh /usr/local/sbin/cloneconf.sh

exit 0
```

6. After verifying the command above is in /etc/rc.local, reboot the system. On reboot, the /etc/rc.local file will run the cloneconf.sh script to change the system hostname and bootstrap to Chef server. (This script runs only once, on first startup or reboot of the system. After this, the cloneconf.sh file will be deleted so the system’s hostname is permanent).

A successful run of this script will look like below:

```
root@LXC-Ubuntu:/usr/local/sbin# sh /etc/rc.local
Guest Configuration After Cloning

The hostname is now Ubuntu-c41ff3e9
This script will configure the host with the following settings:
Hostname: Ubuntu-c41ff3e9

/etc/hostname:
Ubuntu-c41ff3e9

/etc/hosts:
127.0.0.1    localhost
::1         localhost ip6-localhost ip6-loopback
ff02::1     ip6-allnodes
ff02::2     ip6-allrouters
# --- BEGIN PVE ---
127.0.1.1 Ubuntu-c41ff3e9
# --- END PVE ---

Bootstrapping to Chef server...
[2017-03-04T09:29:59+00:00] INFO: Forking chef instance to converge...
Starting Chef Client, version 12.18.31
[2017-03-04T09:29:59+00:00] INFO: *** Chef 12.18.31 ***
[2017-03-04T09:29:59+00:00] INFO: Platform: x86_64-linux
[2017-03-04T09:29:59+00:00] INFO: Chef-client pid: 8213
```




```
Bootstrapping to Chef server...
[2017-03-04T09:29:59+00:00] INFO: Forking chef instance to converge...
Starting Chef Client, version 12.18.31
[2017-03-04T09:29:59+00:00] INFO: *** Chef 12.18.31 ***
[2017-03-04T09:29:59+00:00] INFO: Platform: x86_64-linux
[2017-03-04T09:29:59+00:00] INFO: Chef-client pid: 8213
Creating a new client identity for Ubuntu-c41ff3e9 using the validator key.
[2017-03-04T09:30:30+00:00] INFO: Client key /etc/chef/client.pem is not present - registering
[2017-03-04T09:30:31+00:00] INFO: HTTP Request Returned 404 Object Not Found: error
[2017-03-04T09:30:32+00:00] INFO: Run List is []
[2017-03-04T09:30:32+00:00] INFO: Run List expands to []
[2017-03-04T09:30:32+00:00] INFO: Starting Chef Run for Ubuntu-c41ff3e9
[2017-03-04T09:30:32+00:00] INFO: Running start handlers
[2017-03-04T09:30:32+00:00] INFO: Start handlers complete.
[2017-03-04T09:30:33+00:00] INFO: HTTP Request Returned 404 Not Found:
[2017-03-04T09:30:33+00:00] INFO: Error while reporting run start to Data Collector. URL: https://api.chef.io/organizations/pdd/data-collector Exception: 404 -- 404 "Not Found" (This is normal if you do not have Chef Automate)
resolving cookbooks for run list: []
[2017-03-04T09:30:33+00:00] INFO: Loading cookbooks []
Synchronizing Cookbooks:
Installing Cookbook Gems:
Compiling Cookbooks...
[2017-03-04T09:30:33+00:00] WARN: Node Ubuntu-c41ff3e9 has an empty run list.
Converging 0 resources
[2017-03-04T09:30:33+00:00] INFO: Chef Run complete in 1.5518513 seconds

Running handlers:
[2017-03-04T09:30:33+00:00] INFO: Running report handlers
Running handlers complete
[2017-03-04T09:30:33+00:00] INFO: Report handlers complete
Chef Client finished, 0/0 resources updated in 34 seconds
[2017-03-04T09:30:33+00:00] INFO: Sending resource update report (run-id: c3368a52-a916-4b90-9323-fcc755c9b8f5)

Removing 'cloneconf.sh' file...

The clone configuration has completed. Hostname changed and bootstrapped to Chef server.

root@LXC-Ubuntu:/usr/local/sbin#
```

- After the script runs successfully, a new node should appear in Chef server with the same hostname that was created in the previous script:

Nodes	Reports	Policy	Administration	Feedback Organization pdd Signed In as Joseph Lagana		
Showing All Nodes				<input type="text" value="Search Nodes..."/>		
Node Name	Platform	FQDN	IP Address	Uptime	Last Check-In	Environment
Win7-001	windows	pc01.sdc.cpp.edu	172.16.20.32	8 minutes	17 days ago	_default
lxc-node	ubuntu	LXC-Ubuntu	172.16.20.141	a day	3 days ago	_default
Ubuntu-c41ff3e9	ubuntu	Ubuntu-c41ff3e9	172.16.20.141	4 days	3 minutes ago	_default
test123	windows	WIN7-UMGSQHT7Z2...	172.16.20.71	2 minutes	a day ago	_default

- Typing the command “hostname” should show the newly configured hostname:

```
root@LXC-Ubuntu:/usr/local/sbin#
root@LXC-Ubuntu:/usr/local/sbin# hostname
Ubuntu-c41ff3e9
root@LXC-Ubuntu:/usr/local/sbin#
```

- In the /etc/chef folder, a new file should've been added, titled “client.pem”:



```

root@LXC-Ubuntu:/etc/init# cd /etc/chef
root@LXC-Ubuntu:/etc/chef# ls
client.pem client.rb pdd-validator.pem
root@LXC-Ubuntu:/etc/chef# █

```

- a. Example of client.pem file (will be different every time it is generated during chef-client run):

```

root@Ubuntu-b690a021: ~
root@Ubuntu-b690a021:~# cat /etc/chef/client.pem
-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEA5GudWSv8WzIj5cRBxtIf6n5A10x0Afyh1nS5j8D/RcWPKjz
qjIt9ItL2g+c6ZhZ+VZXU2uWkSMYmNIoAmuDBhgmZV38b4AbFoQ10C1yJbpxbl8B
DckMubZ5nh1IoM8sZ+e7QfLcuHVVHYLxLYz10SHT/rICN2gc0U9gna5TqN0sms6z
n1FOP5YH4HU+b4N+apwcv8ahaLDNWIFbEt25FSce4GTbH7H+jPW2Gn5KoJQNs12J
aYVhsTuyqxzhjtwX0l823vdiHn+HSTm1K81xI7WBRDJyBQnA3laChfz3WKJU1pFn
SLdvBrAUF2ZKkvE1D5blq1XJpYMAx+Ajl7gMawIDAQABoIBABzS76InCngd17L2
6LEq4jUp78A5a4DC3/nwfcqMPSacCoE9GLJrbp0jTskPm09m6E0sf4yJURCzixEN
w/q6Mmh8d6vm9knjsode25Bh07xDELlKx8LDi5WLz0ECLP2L5wadmWwz392jf4Af
2/hxNyikeinzsI/pVVZuC9UaAHlYY1a4E107U+uIkPCEKiLo5A4uLPrg+qXmnLw9
FCir9DczE4f1n8tlvLm8EjFc4LPLR0pekuJg9WHXhasu45Acq1e5VBZPzaDGY8p
mZf5G1caqm68aGSDcmnxkYK3ma0iar8L/HcegrhCxd7wSFwFUCBovUFU50EY0/9p
oe0WoHECgYEA8vGJN2vYYycwgIaxf07sgTQoLuJcInvmM94RfNmTfQTVxve00NqQ
ZZFLW41VDYF9v9uUINE1siCnltKetjHDLPTyaNIHGc8VaZshnvBUPCg1Hmo/ZA6c
jCCrpbk9vy0G02ECKoFzcZpWceL+aojVm/IayyqV1hgGneu3erCBbo0CgYEA8LJE
PtqMb1GmCVQ7aKcGrd9Gkxu86De0KnJfQUXLBKu4us2/+yJkGhsKKkXRnJCnm1n
+mvz42HlubyxhFkK2bip09TkaMI/tH4bbcd+i1n0BiY4HAFy2NXLsabTJ187mLX
AcAFevgy0Cka+Uh2bb149NvwaL/RD76TuRzyBNCgYEA298P8dGwAGTqd5fvsINI
FtN0fs33q3/JWzPiwS+yIHdb91YTshojNvjnek9aJ6QHXAItcZoGf9Uf9mTAz+IP
o3N4pLDY59nMpkxBxHPduJrEweckwfbZBL5Zj5VJNSo2rq8TB3PCANHU0cg02j1
8oJWD4PWIfKSnuF/qBFwsw0CgYEAx/DNKA1rrthUPFukHf2Qe96FGZ5pbd1/4+KY
y1F835w5VZzeUamYwERokD4L/3basA/h17vLFHU2B1wfefvQkZs8N6MZ0Yx1ruQy
LLFEwS0v7rd8JrPnnFQhDniLeNE5/lxMvj9sTgHu88uxhHIVLoBrYbtxsRHL8FY3
EAZjZHECgYAO/u0boo/QqIub9Y85kChxCuqPCIrqbgebsYSVUIFYQjZ/SA04Gex
vDn9TH8AZx9BHLzQaDQq04GPD4c0unrZUG1ILrhqWdkFZH10Wx97Us5fEr2FjYEV
moUgIEbfLzj87U4Jz42oLAmjAcdbjc1Z4Amn4L/jbAqr95yXmIw2XA==
-----END RSA PRIVATE KEY-----
root@Ubuntu-b690a021:~# █

```

10. ProxMox reverts all changes made to hostname files upon reboot. In order to prevent ProxMox from making changes to your files, you must create a file titled “.pve-ignore.<file you want ignored>”.

We had to create these files for both “/etc/hosts” and “/etc/hostname” files. These were the commands we used to do this:

- a. touch /etc/.pve-ignore.hosts
- b. touch /etc/.pve-ignore.hostname

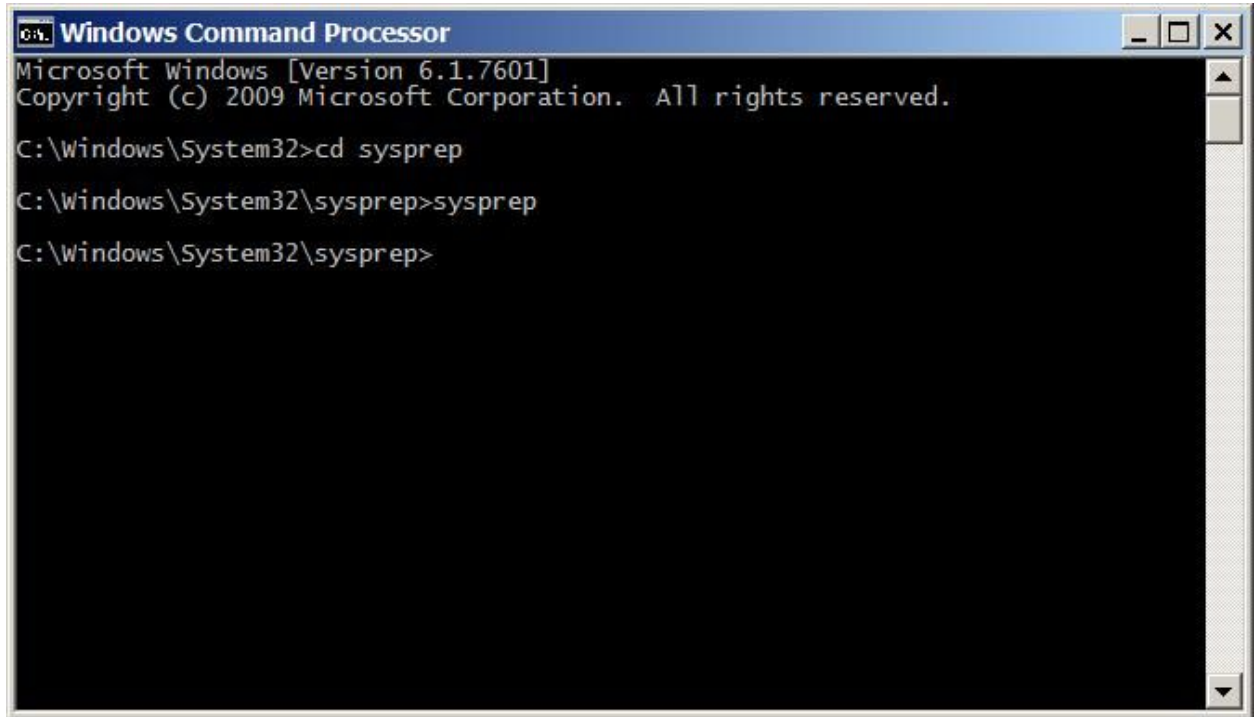
Some more information about this:

https://pve.proxmox.com/wiki/Linux_Container#_guest_operating_system_configuration

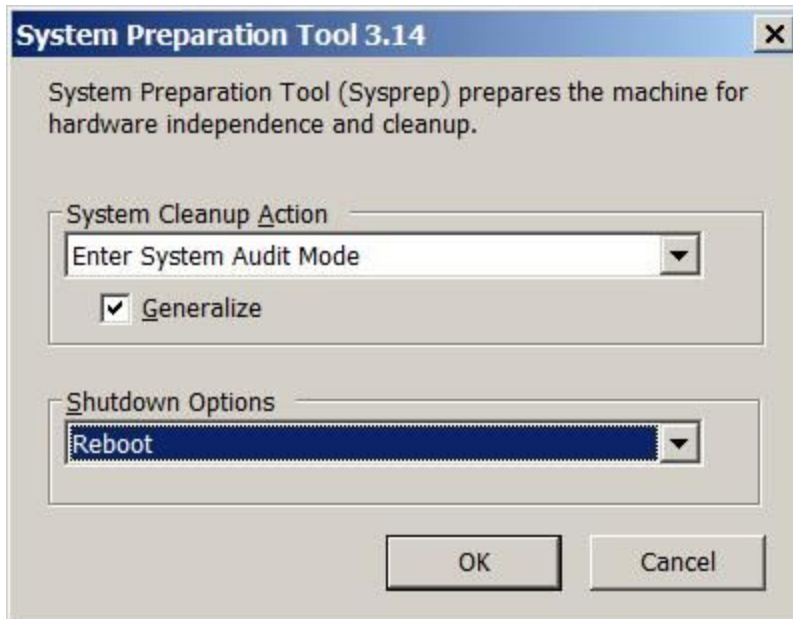


Windows 7 KVM Unattended Bootstrap

1. Clone or create a fresh installation of Windows 7 and launch the VM.
2. Open up command prompt and navigate to the sysprep directory.
(C:\windows\system32\sysprep)



3. Run sysprep with Audit mode selected, Generalize checked, and Reboot selected.



4. When the machine restarts, download and install chef-client (Windows 7 x86_64: <https://downloads.chef.io/chef>)



Windows 7

License Information

Architecture: **x86_64**

SHA256:

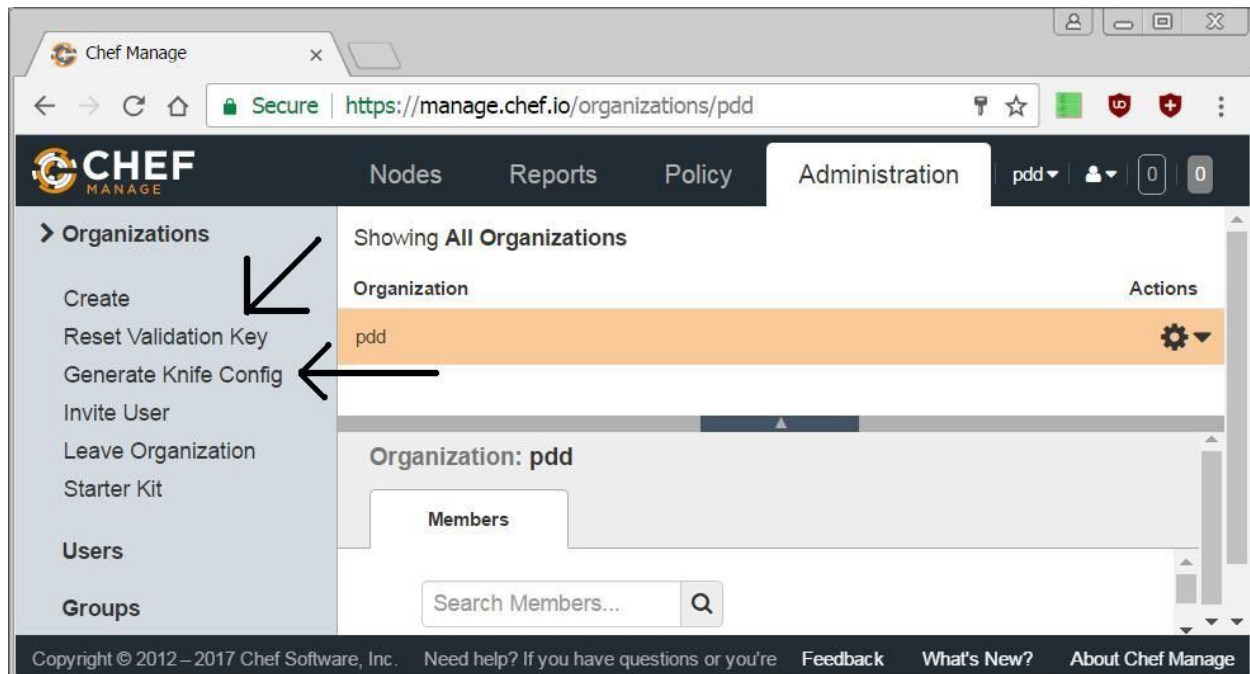
1baed41a777d298a08fc0a34dd1eaaa76143bde222fd22c31aa709
c7911dec48

URL:

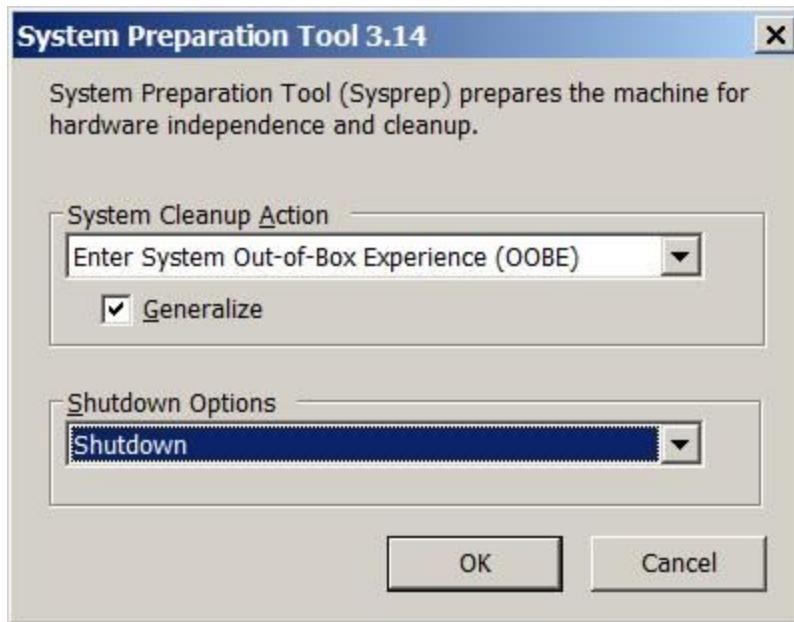
<https://packages.chef.io/files/stable/chef/12.19.36/windows/2008r2/chef-client-12.19.36-1-x64.msi>

Download

5. Install any necessary drivers or programs and remove the temporary account used in the initial installation.
 - a. There were issues where the VM would hang at “checking video performance” after installing system updates. To avoid this issue, we recommend installing any necessary updates after the VM is cloned.
6. Generate the client.rb and validation.pem files from your Chef server (or ones previously supplied).
 - a. Be sure to remove the node name line from the client.rb file in order to have the node name match the FQDN. Otherwise the node name will be the same every time a new VM bootstraps itself.
 - b. If you must reset the validation key, you will have to rebuild any templates that you have previously made and use the new private key generated.



7. Download and place the scripts in the root folder of the file system (C: drive) **make sure to unblock these files** by going to the file properties and clicking unblock.
8. The files that should be in place are:
 - a. Two powershell scripts in the root of the C: drive (startup_script.ps1 and bootstrap.ps1)
 - b. Two chef files (client.rb and validation.pem) in the chef folder (C:\chef)
 - i. Current validator for our server located at
<https://drive.google.com/open?id=0B0jXyCbOOwcdX0pzUVBsLVRfMVE>
 - c. A cmd file (C:\windows\setup\scripts\SetupComplete.cmd)
 - d. An answers file (C:\windows\system32\sysprep\unattend.xml)
 - i. You can generate your own at
http://windowsafg.no-ip.org/win7x86_x64.html or use the included one.
9. Open up command prompt and navigate to the sysprep directory again.
(C:\windows\system32\sysprep)
10. Run sysprep with OOBE mode selected, Generalize checked, and shutdown selected.



11. Snapshot the VM when it is shutdown. Create a clone from the snapshot then convert it to a template. You should now be able to create linked clones from it.
12. Start the cloned VM and it will go through an initial set up (a lengthy process) and then it will generate a new computer/host name. Afterwards, it will restart itself for the changes to take effect and upon starting the second time, it will bootstrap itself to the Chef server.



Daily Summary

January 12

Original meeting with Juan discussing general premises with explanations of good items to research and review. The project requirements are explained and a blueprint of a contract for Juan to review is drafted. The previous group's Chef implementation is included in the review in order to understand what has been previously developed. As an extension to this, we received the original project write up given to the team last quarter.

The Chef software is a management tool written in Ruby that allows for recipes to be written to be managed by server applications. These recipes are essentially scripts that lets you turn infrastructure and operations into deployable code. The original project used Chef as a way to install different programs on a new virtual machine. The Chef client is used as a means to achieve efficiency as the forensics classes would need a multitude of different programs for use in the class. Installing each program on each individual machine becomes inept and mundane. The Chef client allows each virtual machine to be deployed quickly and with the required software.

January 14

Logo and business cards are created along with the first write-up of the contract. The first version of Gantt chart and time cards are put in place to track progress. Because the environment and servers we needed to work on were not fully set up, the Gantt chart revolved around what we envisioned completing. The group's personal work consisted of research in order to familiarize ourselves with the different concepts we would be working with such as Chef and Python.

January 18

Juan rejects the original contract as most of the agreements revolve around the previous group's work. After working with Juan in order to get a better explanation, we drafted another contract and Juan agrees to the new one. The main requirement stipulated within the contract is to bootstrap a Linux container and KVM hypervisor with virtual machines. Once these are made, the Chef client and the previous group's scripts can be made to deploy virtual machines ready for the Boeing forensics class.

Linux Containers (LXC) is a virtualization method that allows one to run multiple Linux systems hosted on a single Linux kernel. An LXC provides a virtual environment that has its own CPU and memory which allows resources to be freed. CPU usage with LXC drops dramatically when running.

A hypervisor is a program that allows multiple operating systems, in this case Windows 7, to share hardware. A KVM hypervisor would allow multiple instances of virtual machines running windows to run with the allocation of a single hardware host.



January 23

Meeting with Juan cancelled due to unforeseen circumstances. Actual documentation of the previous group is agreed to be given to us on January 24 with an official meeting on January 25.

January 25

Juan and his team has set up a virtual machine in their Proxmox virtual environment. Access to the server has been granted to each member of the team, including access to individual virtual machines to conduct work with. Other topics included programs such as virt-viewer. In-depth explanations were given on how to connect to the server including OS X and connecting to the Cal Poly wifi. We were also able to access the previous groups' Chef account in order to see their recipes called "Chocolatey".

Proxmox Virtual Environment is an open-source environment that allows users to manage virtual servers. The management solution is based on KVM and container virtualization. Promox is a graphical interface frontend in which we can manage the status of servers and has the functionality to deploy virtual machines coupled with the ability to revoke and give privileges to created users. Proxmox is the environment used to create and run the virtual machines needed to work on the project.

Virt-viewer is an application used in conjunction with the Proxmox server to view and work on a virtual machine. Due to browser limitations, the client that proxmox deploys has lag and sync problems. Using virt-viewer allows us to download a temporary session to give us access to the virtual machine without any input lag.

January 30

Groups continue to do research on the various topics that are associated with our project. However, as we continue to do individual research on the pieces, we realize no one knows how to put the pieces together, nor how we sincerely start this project. We decide to meet to further clarify how we go about the project.

February 1

We meet to review the previous group's documentation and further understand how to start our project. We realize we don't fully understand what Juan is asking from us. We go through the previous documentation. Certain parts we know how they did, and we are confused whether or not we are supposed to do what they have done so far. We decided to compile a list of questions we need to ask, continue going through Chef.io tutorials and documentation, and make a meeting with Juan to ask all our questions at once.



February 3

We meet with Juan and further get clarification on how to go through the other group's work. We have to set up a workstation and connect it to their server to get access to their scripts from their project. After getting a better understanding of how to access the work, we decide we need to prioritize our knowledge of Chef node management, and agree to meet on Sunday to set up the workstation. Anyone that doesn't fully understand needs to complete at least section 2 of the Chef tutorials in order for us to properly set up the work environment.

February 5

We set up the management server hosted by Chef. In addition to the server, we also managed to configure one of the workstation VMs as a Chef Development Kit workstation. We attempted to bootstrap one of the VMs to the Chef server, but had no success. There were issues affecting the connectivity of the VMs (DNS issue) and most, if not all of them could not access the internet. We did figure out that one of the reasons why the node was not bootstrapping was that there was no cookbook on the Chef server.

February 6

There are still issues in connectivity with the virtual machines. The group writes documentation on how to connect to the work environment and the workflow. We refer to the previous group's work as well as Chef documentation for more information on how to upload a cookbook to the server. Later on, we manage to upload a cookbook to the server. We also add every member of the group as users on the Chef server. We resume attempting to bootstrap a node to the Chef server, with difficulties in authentication with WinRM. We also begin a redesign of the logo.

February 8

We updated everyone's privileges on the Chef server to admin. We work on the second progress report and continue to attempt to bootstrap a Windows node to the Chef server. We also work on the presentation for the next day with a fresh new logo chosen among a few.

February 9

We present our current progress on the Chef implementation to the class and professor. Later on in the night, we managed to bootstrap the Windows KVM node to the Chef server. The issue with WinRM authentication was resolved when a password was added to the virtual machine.

February 15

We met, discussed issues we had, demonstrated bootstrapping a node in Chef server, and



consolidated questions to ask Juan. We also managed to bootstrap an LXC container via our main workstation in ProxMox to our Chef server. We first had to enable root login through SSH and allow access through SSH, HTTP, and HTTPS ports in order to resolve the errors we were getting.

February 16

We had a meeting with Juan to discuss our current progress, any potential problems with the project, and discussed possible solutions. We also discussed any future goals for the project.

February 17

We did some testing and research on bash and PowerShell scripting as well as some more research on Chef node management. We were able to confirm that we had finished the original requirements for the project. This is primarily outlined by bootstrapping an LXC container and KVM Template. After some discussion, Juan suggested that we could move onto a bonus part of the project if we had completed the original. This was outlined by changing the hostname for the different templates.

February 20

We relayed to the other members the outlines on the Friday meeting. We were able to convene and began researching what scripts we needed to complete the remainder of the additional assignment. After discussion, we began trying to figure how to implement batch and bash scripts in order to change the hostnames.

February 22

We met again and came up with a blueprint of a solution on how to change the different names. It revolved around creating a script that would run on the startup for either the Linux or Windows virtual machine. The script would run and change the hostname to a random name. Discussion and research was done to figure out how the names would always be different, and to make sure that they would not repeat.

February 24

We tried contacting Juan to clear up some questions we had about how he would clone the VMs. He couldn't meet with us today, nor this weekend. We scheduled to meet on Monday.

February 26

Started an initial draft of an LXC startup script. The script worked for changing the hostname in a test environment on a home workstation. We also attempted to do an unattended bootstrap of the node as well, to no avail. No testing was done on the live environment.



February 27

We met up to make progress on the LXC startup script. We made significant progress as we were able to randomize the hostname as well as get the script to run on startup. However, we ran into a roadblock of what we assume is Proxmox reverting any changes made to the hostname or any files related to it. We attempted to block write access to the files, but as stated above, the container would revert whatever changes we made.

February 28

We met with Juan in the afternoon to notify him of our progress on the scripts. He shifted the priority on the Windows startup script, so we halted progress on the LXC startup script. He then joined our development chat channel for a direct line to him for quick communication.

March 1

We met up and conducted extensive work on researching and working on the Windows script. We developed an initial draft ready for testing for use on the live machines. We also learned of some issues that needed resolving in order to have a production ready template.

March 2

We met up again in the afternoon to work on the Windows startup script. We managed to resolve some issues and produced a nearly complete script, with only fine tuning needed for implementation into the live machines.

March 3

We resumed work on the LXC startup script. There are still issues on the script with startup and hostname. We plan to meet later in the weekend to resolve those issues.



Agenda 1

1/18/2017

- ☐ Look over Contract
- ☐ Final Clarifications
- ☐ Sign Contract
- ☐ Discuss Tasks to accomplish
 - ☐ Section it into subtasks
- ☐ Start Filling Gantt Chart
- ☐ Confirm meeting on Saturday for Domain Controllers

Subjects that needed to be review:

Windows Admin
Powershell
SysPrep
Batch
Ruby
Chef
Bootstrapping
Guacamole

Overall Group Questions:

How do we work on the project/ access it?
What do we need to prepare before Saturday?

Meeting Memos

1. Confirmed requirements with Juan.
2. Contract revised and signed by team and client
3. Gantt chart made and estimation of individual tasks planned.
4. Discussion about which subject we should study on in order to finish each task.



Agenda 2

1/23/2017

- ☐ Discuss additional reports to be made
- ☐ Have everyone go over the learn.chef.io on their own before next meeting
- ☐ Discuss Domain Controllers with Juan
- ☐ Ask Juan to view other group's work
 - ☐ Ask to see other group's paperwork and reports as well if possible (maybe talk to Pike)
- ☐ Photos to be used in the final report

Overall Group Questions:

How do we work on the project/ access it?

Where are the workstations that are going to be connected to Chef?

How do we access the server?

Outcome of meeting:

Juan asked to move the meeting to Wednesday because something came up and he was busy. He agreed to go over Domain Controllers with us. He also said he would provide us with the Documentation for the previous group's project. On Wednesday, he said he would give us access to the server that the previous group was working on.



Agenda 3

1/25/2017

- ☐ Domain Controllers with Juan
- ☐ Access other group's server
 - ☐ Ask to see other group's documentation
- ☐ Photos to be used in the final report
- ☐ Discuss additional reports to be made
 - ☐ Chef Reports
 - ☐ Node Management?
 - ☐ Web App Cookbook?
 - ☐ Develop and Test Locally Report?
 - ☐ Chef Automate / Collaborative Development?
 - ☐ Custom Resources?
 - ☐ What is an LXM Template
 - ☐ What is a KVM Template
 - ☐ Diagnostics of what was left to us from the previous group

Outcome of Meeting

Juan showed us the Proxmox virtual environment that we would be allowed to work with. He gave us all access to the environment. In addition, he showed us the Chef scripts from the previous groups project. Advised us to start with looking up bootstrapping a vm in Chef. All of the group needs to complete the Chef Tutorials. We can begin working on the LXC templates Juan asked us to make.



Agenda 4

2/1/2017

- ☐ Discuss Chef Tutorials
- ☐ Split up workloads
- ☐ What was left to us from the other group?
- ☐ Node Management in Chef

Outcome of Meeting

We divided into research groups to further try to understand what the work we exactly needed to do was. We had a Chef Group, an LXC group and a KVM group. After more discussion, we realized we needed to further clarify what we are supposed to be doing. We further looked through the other group's documentation and realized they did a lot of what we need to do, and are now have questions for how to begin the actual work, rather than just accessing workstations.



Agenda 5

2/3/2017

- ☐ Chef Scripting
- ☐ Meet with Juan to clarify deliverables and what we need to do
- ☐ Identify Workstations
- ☐ Assign sections for progress report
- ☐ Previous Group's Documentation

Outcome of Meeting

One more member of group was given access to Proxmox. We need to do everything that the previous group did with their project, so we also need to set up a Chef Server account and make a Chef Workstation and Nodes.

Questions:

Do we rebuild the other group's environment?

Are we supposed to install Chef_DK on our own machines and use the VMs provided as the nodes for this project?

Are chef clients supposed to be installed on the virtual machines?

What part are we supposed to be automating?



Agenda 6

2/5/2017

- ☐ Catch up members who are not caught up.
- ☐ Set up Chef management server
- ☐ Get members to have Chef accounts
- ☐ Connect Chef accounts to Chef management server
- ☐ Start writing reports for Second Project Report
 - ☐ Proxmox sign in tutorial
 - ☐ Chef
 - ☐ Cookbook descriptions
- ☐ Begin Powerpoint Presentation
- ☐ Research Cookbooks
- ☐ Redesign Logo?

Outcome of Meeting

Configure Chef workstation. Set up Chef Server. Create Chef accounts. Using each other as a resource, we got a better understanding of how to use Chef Installed Chef Development Kit into workstation 7. Internet connectivity issues prompted us to abandon Win7-007 and install the Development Kit in Win7-001. We begun trying to bootstrap a node but realized we need recipes to be uploaded to the Chef server. Also having issues with Node User information. Began testing. Uploaded a cookbook to our server as a trial.

Questions:

Uploading Cookbooks to Chef Server?

Get a node to bootstrap?

Node Configuration



Agenda 7

2/6/2017

- ☐ Catch up members who are not caught up.
- ☐ Finish connecting Chef accounts to Chef management server
- ☐ Continue writing reports for 2nd Management Report
 - ☐ Proxmox sign in tutorial
 - ☐ Environment: Chef
 - ☐ Cookbook descriptions
 - ☐ Project Requirements
 - ☐ Requirements Matrix
- ☐ Research Cookbooks and recipes
- ☐ Redesign Logo?

Outcome of Meeting

Attempted to use workstation but issues with internet connectivity on the virtual machine made us rethink how to do this. Chef group continued to try to send a cookbook to server. Group began writing documentation on how to access the work environment and how we set it up. Reported status updates on research for Cookbooks and how the previous group did their work.



Agenda 8

2/15/2017

- Where do we go from here
- We can bootstrap a node
- We have a windows machine connected to the Chef server, ready to be copied.
- Does he want to install specific recipes in it?
- Decided show Juan the status of our project and ask for advice on how to proceed.
- **Got LXC to Bootstrap *******
- Continue researching cookbooks that might be useful to install
 - LXC - Test this cookbook before applying to existing nodes. Lots of updates have been applied, and some tooling has been replaced. By Chrisroberts

Dependant cookbooks

<code>iptables-ng >= 0.0.0</code>
<code>polipo >= 0.0.0</code>
<code>dpkg_autostart >= 0.1.10</code>

- KVM - Install KVM Hypervisor on Ubuntu, Debian, CentOS or RHEL by guilhemfr

Dependant cookbooks

<code>cpu >= 0.0.0</code>
<code>sysfs >= 0.0.0</code>
<code>modules >= 0.0.0</code>
<code>sysctl >= 0.0.0</code>
<code>ntp >= 1.2.0</code>

Questions:

KVM and LXC containers:

Why do we set up KVM and LXC containers? So Juan can have VMs running off of it and using its resources?



What is the idea behind setting up two virtual machines, one running KVM, and the other running LXC?

How does Juan want to use the containers? Implement a script or use a template of what we made?

Bootstrapped node:

What does this mean? Like keep an image of a VM that we have that can take chef commands? What does it mean to bootstrap KVM and LXC? Connect the node to get resources?

Cookbooks:

Do we just use the ones the old group used?



Agenda 9

2/20/2017

- LXC and KVM can be made.
- Issue: VM clones need unique hostname
- General Research
- Brainstorming session for how to approach problem
 - Bash or Bashrx/startup
 - Bash script in Chef?
- How do we implement it?
- Chef resources to execute scripts
 - Bash is a Chef resource that executes scripts using bash interpreter
- Posting in Chef Forum for assistance
 - awaiting Response
- reddit.com/r/bash
 - Awaiting response



Agenda 10

2/22/2017

- LXC and KVM can be made.
- Will attempt to create clone vm's with unique hostnames. One of the group members has admin authority gained from our client to try to do this.
- Changing the hostname via chef doesn't work because Chef doesn't recognize it as multiple nodes.
- How does Juan clone the VMs? Walkthrough the process perhaps?
- Attempting to change the LXC hostname via advice from r/bash subreddit
 - Issues with admin rights
- <https://communities.vmware.com/thread/198551?start=0&tstart=0>
- Continue researching and looking for a solution to changing hostnames.
- Will try to meet with Juan on Friday to either approve our solution, or provide a solution of his own perhaps.
- Bash Research
- Batch Research
- Cookbooks possibly?
- Change hostnames on startup?



Agenda 11

2/27/2017

- Attempted again to meet Juan today
 - Meeting postponed by Juan due to unforeseen circumstances.
 - Rescheduled meeting to 2/28/2017
- Working on Bash Scripting and Batch Scripting
- Continue researching how to change hostnames for VMs
- Begin compiling documentation for 3rd project report
- Begin Powerpoint presentation
- Hostname change experimentation.
 - Group members try to change Hostnames through various methods
 - Linux and Ubuntu
 - Developing a script to change hostname on startup and delete itself
 - attempt to push script to Chef to change Node name
 - Script didn't change hostname. Must try again.
 - Script didn't run. RC local?



Agenda 12

2/28/2017

- Meeting with Juan today
 - Juan gave admin rights to one member
 - Agreed to meet and be present for presentation
- Script changes hostname but has issues with proxmox
- KVM needs to be worked on.
- LXC is almost complete
- Added Juan to Slack communication to allow more frequent communication with him in this crunch time.



Agenda 13

3/1/2017

- Issues to be addressed with duplicating virtual machines
 - Same Hostname
 - Same SID
- The only issue we've discussed with Juan is hostname changing.
- Script for KVM hostname change is being written.
 - Attempting to convert Bash Script to Batch Script.
- Script for LXC made but having issues.
 - Works for other virtual machines designed on personal system. Does not work for Proxmox environment for some reason.
 - New VM doesn't show up as new node in Chef Manager.
 - New script made that changes hostname in Ubuntu VM.
 - Will try in Proxmox later



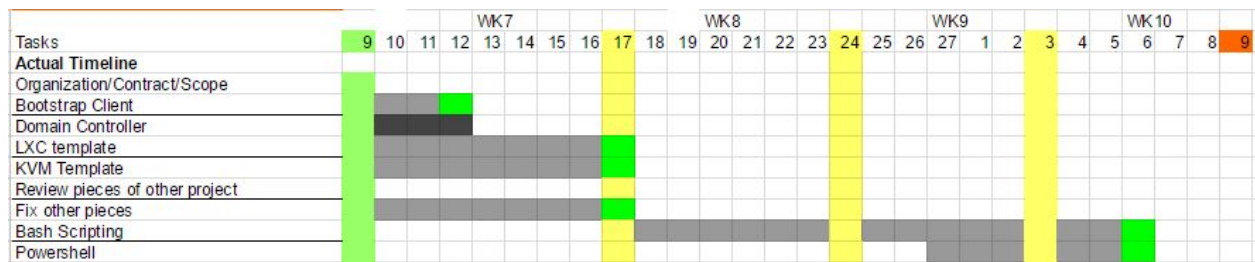
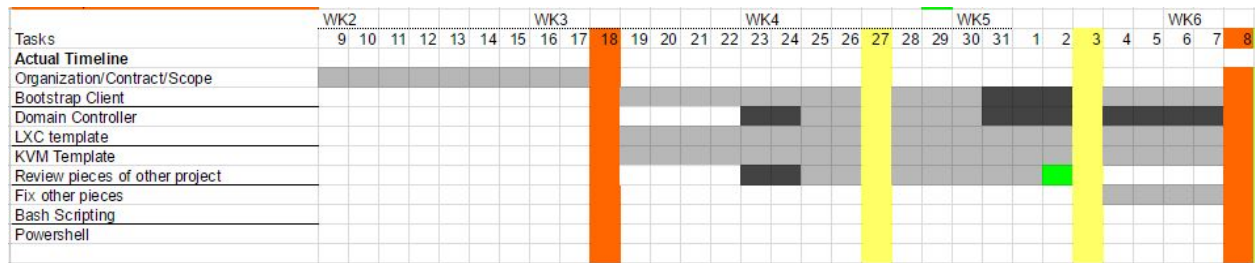
Agenda 14

3/2/2017

- Issues to be addressed with duplicating virtual machines
 - Same Hostname
 - Same SID
- Researching how to change SID.
- Hostname changed.
- Sysprep?
- Bootstrap client?
- Needs to add a new node to the Chef server when it boots.
- Trying to connect the Server to Chef on the new VM.
- Linux managed to change Hostname
- Attempting to change Hostname using a script.
 - Accidentally used old script.
 - Tried again with new script.
- Organization Key?
- Validation Key?
- Managed Script to update Validation Key and be seen on Chef



Schedule of Events





Username and Password List

Account	Username	Password
manage.chef.io	cpppddgroup@gmail.com	cis466pike
Gmail	cpppddgroup@gmail.com	cis466pike
GitHub	cpppddgroup	cis466pike