Acorn System Administration Documentation

Release 1.0

Pavan Rikhi

May 30, 2020

CONTENTS:

1 IT Architecture **3**
- 1.1 Servers . . . 3
- 1.2 Buildings . . . 4
- 1.3 Networking . . . 7
- 1.4 VM Cluster . . . 8

2 Hardware **11**
- 2.1 Console . . . 11
- 2.2 Switches . . . 11
- 2.3 Cluster Nodes . . . 15

3 Administration & Maintenance Guides **19**
- 3.1 Server Rack . . . 20
- 3.2 Networking . . . 21
- 3.3 Website/VPS . . . 24
- 3.4 Arch Linux . . . 25
- 3.5 CUPS Print Server . . . 26
- 3.6 Slackware . . . 27
- 3.7 Windows . . . 29

4 Slackware Server Administration **33**

5 Debian Workstation Automated Setup **35**
- 5.1 Quickstart . . . 35
- 5.2 Automated Installs . . . 36
- 5.3 Ansible Setup . . . 36
- 5.4 Automated Maintenance with Fabric . . . 38
- 5.5 To Do . . . 39

6 Acorn VM Cluster **41**
- 6.1 Setup & Configuration . . . 41
- 6.2 Cluster Maintenance . . . 43
- 6.3 Cluster Initialization . . . 50
- 6.4 Appendix: Notes From Setup Trials . . . 57

7 Indices and tables **63**

This document outlines the general architecture of Acorn's Networking & Servers, along with documentation on maintenance & our automation.

CHAPTER

ONE

IT ARCHITECTURE

Our local network is run on a 10Gb/s switch that splits out to 1Gb ethernet connections. Additional buildings are connected via ENH202 wifi bridges.

Our internet connection runs at a dedicated 3Mb/s with a burst of up to 10Mb/s.

Our router is called `Cerberus` - it runs FreeBSD and runs the Packet Filter firewall, NAT, DHCP, BIND DNS, Samba WINS, & a Squid Web Proxy.

Our servers runs Slackware Linux - we have `Vishnu`, our Database server, `Aphrodite`, our general server, & `Adonis`, our backup server.

Vishnu contains our Business files & is backed up hourly, daily, monthly, & yearly to both Adonis & Aphrodite.

Aphrodite holds our Community & Personal files & is backed up daily, monthly, & yearly to Adonis.

We currently have one public Linux workstation, `SewingMachine`, that runs Debian & KDE - but the setup has been automated to make it easier to expand.

1.1 Servers

1.1.1 Cerberus

Cerberus is our router that runs FreeBSD, serves `.acorn` DNS requests, provides DHCP & WINS, & caches HTTP requests.

There's a guide available from the terminal, SSH into cerberus, then run `cerberus_help` for a long guide & `cerberus_quick` for a quick reference of config files & useful commands.

TODO: Move that documentation over here! Configs, Services, Commands, CLI Guide

There are a couple of bandwidth graphs:

- Live Usage graph: http://cerberus.acorn:667
- Longer per-host graph: http://cerberus.acorn/bandwidth
- Live User per-host graph: run `sudo iftop`

These links might be helpful:

- FreeBSD Handbook: http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/
- PF Docs: http://www.openbsd.org/faq/pf/index.html
- Firewalling with PF: http://home.nuug.no/~peter/pf/en/index.html
- Newbie Guide for PF on OpenBSD: http://www.thedeepsky.com/howto/newbie_pf_guide.php

- TCP/IP Primer: http://www.ipprimer.com/

TODO: Buy a server w/ a lotta ram, ssd, & a 10Gb nic(for squid) and upgrade cerberus!

1.1.2 Aphrodite

Aphrodite is a general-purpose Slackware server that runs the following services:

- **cups** - Print Server running at http://printers.acorn
- **http** - Apache webserver serving redmine to http://projects.acorn & minecraft to http://minecraft.acorn
- **samba** - Personal, Community, & Backup Windows Shares
- **minecraft** - Minecraft 1.7.2 Server at `minecraft.acorn`
- **murmur** - Chat/VoIP server at `chat.acorn`, port `64738`.
- **moinmoin** - Wiki server running at http://wiki.acorn
- **AcornAccounting** - Custom accounting software running at http://accounting.acorn
- **zabbix** - Network/System Monitoring at http://monitor.acorn

1.1.3 Adonis

Adonis is our Slackware backup server, that hosts daily, monthly, & yearly backups of the Business, Community, & Personal shares.

It uses hs-backup to make backups using the configuration file at `/etc/hs-backup.yaml`.

1.2 Buildings

1.2.1 Seed Office

The seed office is where our backbone switch lives & where the WAN line comes in.

The office's ethernet jacks terminate in patch panels(labelled `A` & `B`), and are connected to 2 Quanta LB4Ms(`LB4M-1` && `LB4M-2`, `manual`). These LB4Ms connect to a Quanta LB6M(`LB6M-1-PUBLIC`, `manual`) which is used as our public LAN's backbone.

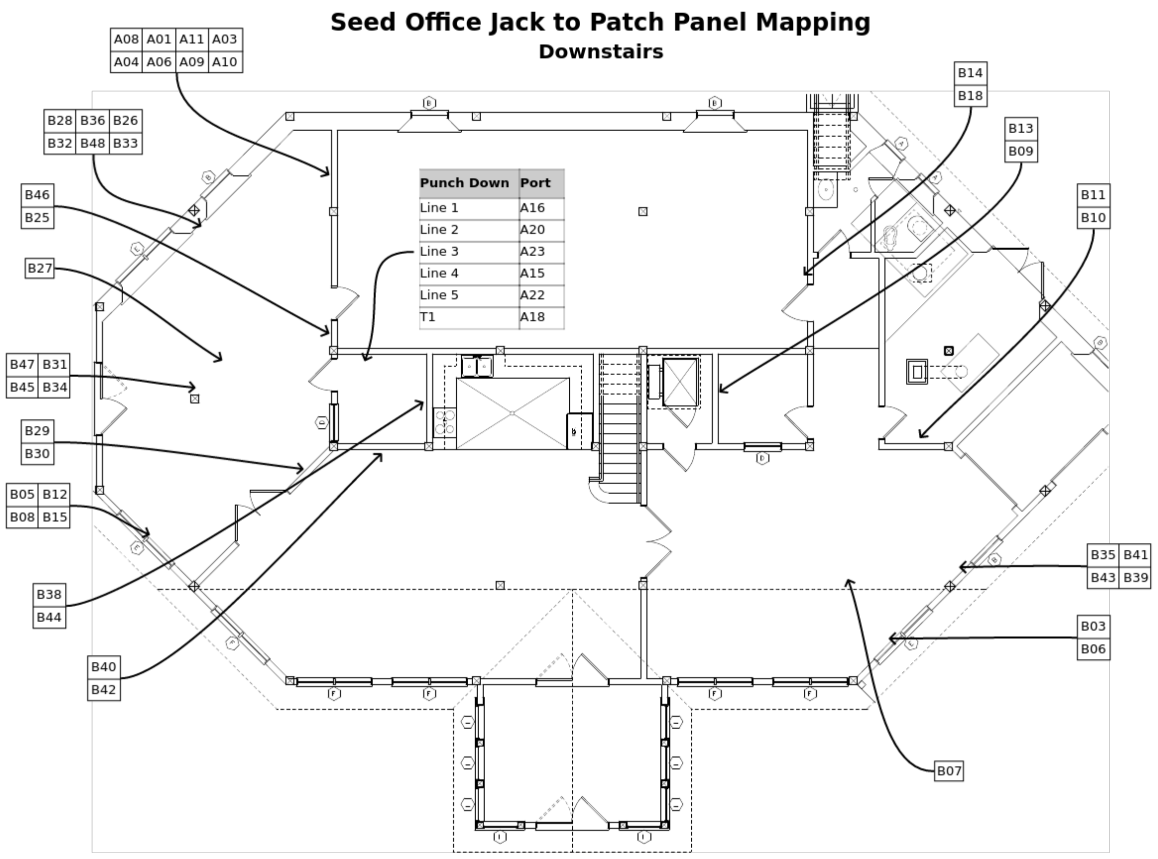
`LB6M-1-PUBLIC` also connects our public LAN to the VM Cluster. See the *Networking* section for more information & the *Switch Hardware* section for port layouts/assignments of the switches.

The following maps show the Patch Panel Ports for each Wall Jack in the building:

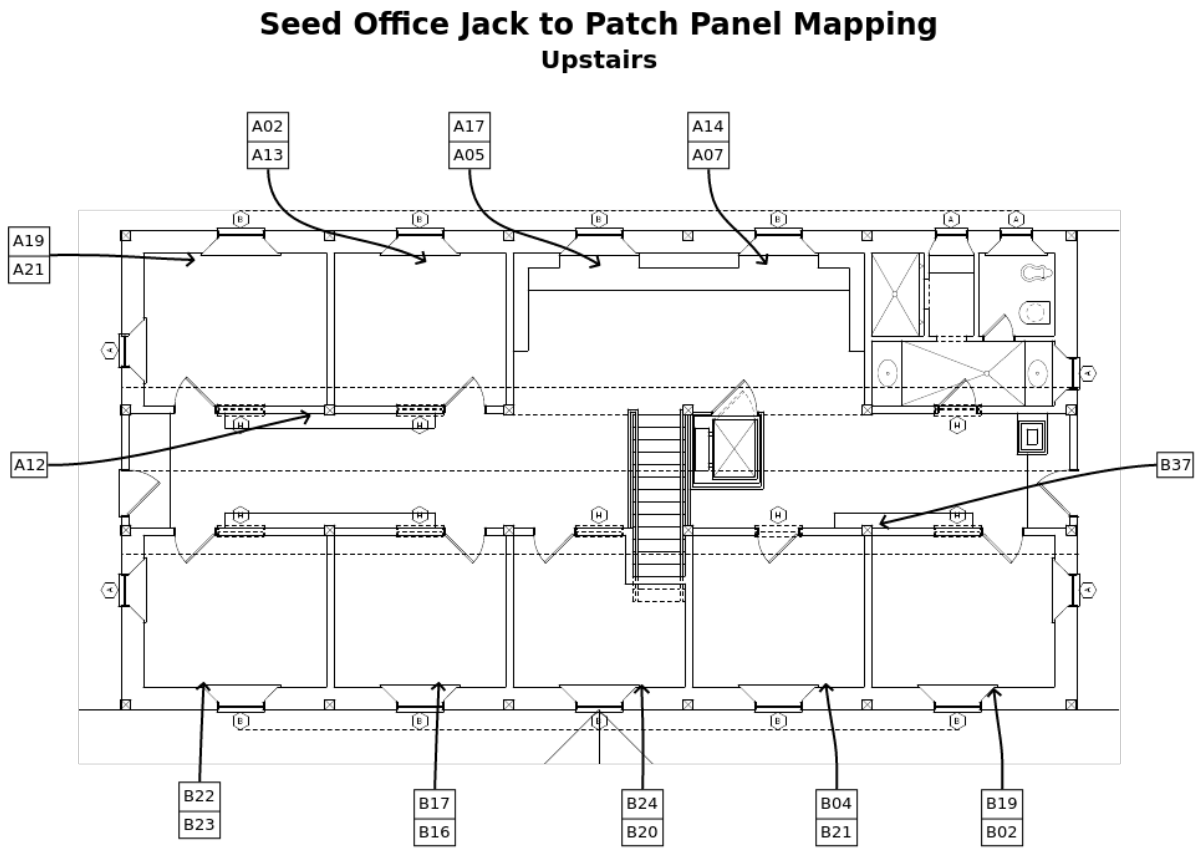
1.2.2 Heartwood

Heartwood is connected to the Seed Office via a pair of ENH202 wifi points. The wifi line enters from the dining room & is switched to an AP and the switch in the living room. The upstairs switch feeds to workstations, 2 other switches that also feed to workstations, & 2 ENH202s(one for the BarnYard wifi, one for the Trailer connection).

The closet office computer connects to the AP.



Punch Down	Port
Line 1	A16
Line 2	A20
Line 3	A23
Line 4	A15
Line 5	A22
T1	A18


Seed Office Jack to Patch Panel Mapping
Upstairs
A02
A13
A17
A05
A14
A07
A19
A21
A12
B37
B22
B23
B17
B16
B24
B20
B04
B21
B19
B02

1.2.3 Farmhouse

The Farmhouse is connected to the Seed Office via an ENH202 wifi point, which goes to a switch that has an AP and runs to workstations.

1.2.4 Trailer

The Trailer get's it internet access from Heartwood via an ENH202 wifi point.

1.3 Networking

We have 6 networks:

Network	IP CIDR
Public LAN	192.168.1.0/24
VM LAN	10.0.1.0/24
Cluster Management	10.2.1.0/24
Cluster Overlay	10.3.1.0/24
Cluster Storage	10.4.1.0/24
Cluster Sync	10.5.1.0/24

Hosted across 3 LB4M(`manual`) & 2 LB6M(`manual`) switches:

- *LB4M-1*
- *LB4M-2*
- *LB4M-3-MGMT*
- *LB6M-1-PUBLIC*
- *LB6M-2-STORAGE*

Cerberus provides DHCP to the Public LAN & all addressing of cluster nodes is done manually, using static IPs.

We use the following color-coding for ethernet cabling:

RED	Phone Lines
YELLOW	Power over Ethernet
BLACK	WAN Line
GREEN	Router Link
BLUE	Public LAN
ORANGE	Cluster Management
WHITE	Cluster Overlay
PURPLE	Cluster Provider
GREY	Cluster Storage

All the Fiber cables are 50/125 OM3, which are aqua colored. We use Juniper Networks EX-SFP-10GE-SR fiber transceivers.

The Public LAN is what our workstations connect to. It is routed to the internet and the Cluster Management network by *Cerberus*. Only HTTP & SSH connections to the Management's controller nodes are allowed. It is hosted by *LB4M-1*, *LB4M-2*, & *LB6M-1-PUBLIC*.

The VM LAN is a virtual network hosted by OpenStack, it's the network that all running VMs connect to. OpenStack maps addresses on this network to a range of addresses on the Public LAN when you assign a VM a Floating IP.

The Cluster Management network is used for cluster nodes to talk to each other & the WAN(via *Cerberus*). The Cluster Overlay network is used for internal communication between VMs. These two networks reside on the same hardware, *LB4M-3-MGMT*.

The Cluster Storage network provides nodes with access to the distributed storage cluster. The Cluster Sync network is used for syncing the Storage nodes. Both the Storage & Sync networks reside on *LB6M-2-STORAGE*.

See also:

Cluster Nodes for the interfaces & ip ranges each node type uses for each Network.

Switches for the Network allocation & port connections for each switch.

1.4 VM Cluster

Currently, we use 3 Controllers, 3 Computes, & 3 Storage nodes in a High Availability configuration. Neutron is setup to support self-service networks.

TODO: Explain a little about how openstack works.

1.4.1 High Availability

See the High Availability Guide for reference.

For setup directions, see the *High Availability Initialization* section and the *Cluster Expansion* section.

Storage nodes use Ceph for distributed & high availability image & block storage. An odd number of 3 or more storage nodes is recommended.

Ceph administration is done with `ceph` and `ceph-deploy` on controller nodes. Each controller node runs a monitoring daemon and each storage node runs one storage daemon per drive.

Controller nodes are have various services setup in distributed & failover configurations. Pacemaker is used to share a virtual IP address that is shared between all the Controller nodes. When a node goes down, another node adopts the virtual IP.

OpenStack services & endpoints are distributed using HAProxy. HAProxy takes requests to the virtual IP address and distributes them across all available controller nodes.

RabbitMQ, Memcached, & MySQL are all clustered as well. RabbitMQ & Memcached use other nodes as failovers, while MySQL uses Galera for replication & HAProxy for handling failovers.

TODO: Do memcached urls for openstack service auth & horizon need configuration?

TODO: Add stuff about Open vSwitch distributed networking

Warning: Compute nodes are not setup for high availability, there is currently no automated relaunching of VMs on failed Compute nodes.

1.4.2 Node Services

TODO: Split into sections & describe what each service is for.

The controller nodes run the following services:

- ceph-mon
- cinder-api
- cinder-scheduler
- cinder-volume
- tgt
- glance-api
- neutron-dhcp-agent
- neutron-l3-agent
- neutron-linuxbridge-agent
- neutron-metadata-agent
- neutron-server
- nova-api
- nova-conductor
- nova-consoleauth
- nova-novncproxy
- nova-scheduler

The compute nodes run the following services:

- neutron-linuxbridge-agent
- nova-compute

The storage nodes run the following services:

- ceph-osd

TODO: Update for our new DVR Open vSwitch configuration

1.4.3 Network Addressing

IP addressing of nodes is done manually in `/etc/network/interfaces`.

See also:

Cluster Nodes for the specific interface to network mappings of each node.

Networking for information on each Network.

Management Network

`10.2.1.0/24`

- `5` is reserved for Cerberus.
- `10` is reserved for the Master Controller's Virtual IP.

- `11` to `40` reserved for Controller nodes.
- `41` to `70` reserved for Compute nodes.
- `71` to `100` reserved for Storage nodes.

Overlay Network

`10.3.1.0/24`

- `11` to `40` reserved for Controller nodes.
- `41` to `70` reserved for Compute nodes.

Storage Network

`10.4.1.0/24`

- `11` to `40` for Controller nodes.
- `41` to `70` for Compute nodes.
- `71` to `100` for Storage nodes.

Storage Sync Network

`10.5.1.0/24`

- `71` to `100` for OSD nodes.

CHAPTER

TWO

HARDWARE

The server rack in the *Seed Office* contains our network backbone as well as a 9-node High-Availability OpenStack VM cluster.

2.1 Console

We use a Dell PowerEdge 2160AS 16-Port KVM Console Switch(`manual`) w/ a PowerEdge 15FP Console KMM(`install guide`, `user guide`).

We have a combination of USB & PS/2 KVM Ethernet Cables, the Computes use USB cables while the Controller & Storage nodes use PS/2.

See also:

How to Switch Between Hosts on the KVM

2.2 Switches

We've got 3 LB4M(`manual`) & 2 LB6M(`manual`) switches.

The following figures show our tentative plans on subdividing the switch ports between networks. Tables follow, showing which nodes are currently hooked up to which ports.

See also:

Cluster Nodes for information on each node's NIC, Network, & IP Range.

2.2.1 LB4M-1

2.2.2 LB4M-2

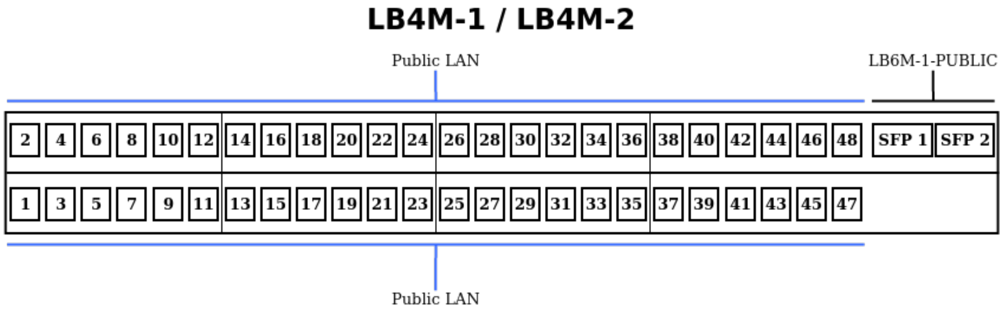


Fig. 1: See the *Seed Office* section for a map of Patch Panel Ports to Wall Jacks.

Port	Host
SFP 1	*LB6M-1-PUBLIC* Port 1
SFP 2	*LB6M-1-PUBLIC* Port 2

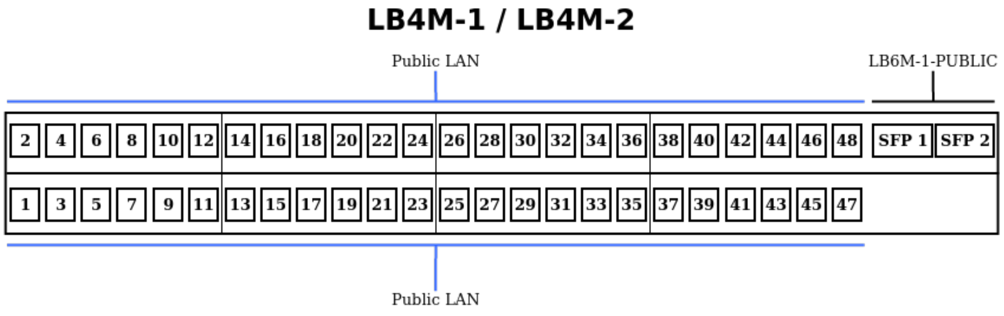
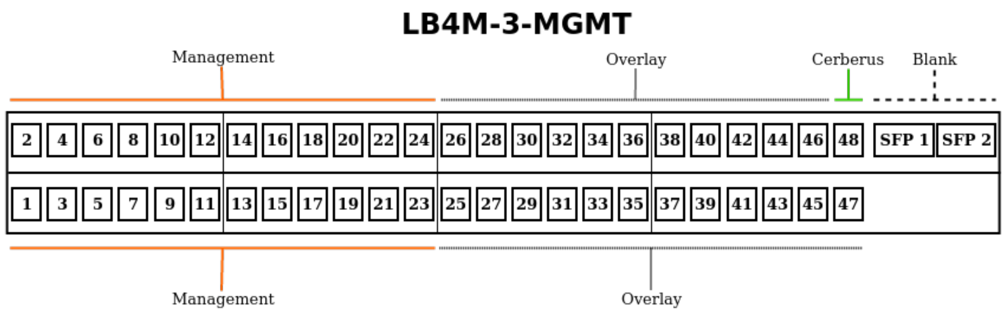


Fig. 2: See the *Seed Office* section for a map of Patch Panel Ports to Wall Jacks.

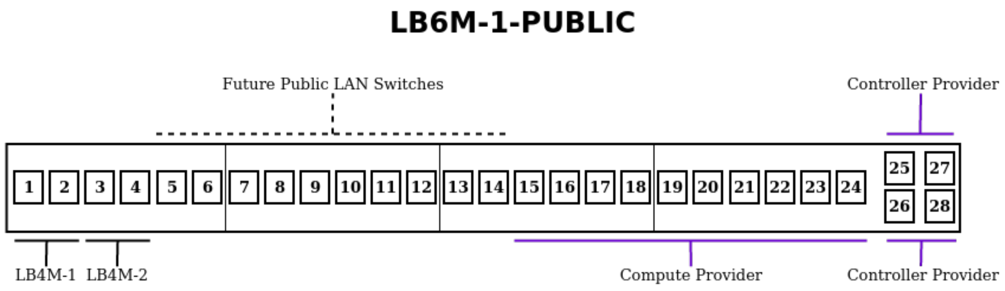
Port	Host
SFP 1	*LB6M-1-PUBLIC* Port 3
SFP 2	*LB6M-1-PUBLIC* Port 4

2.2.3 LB4M-3-MGMT



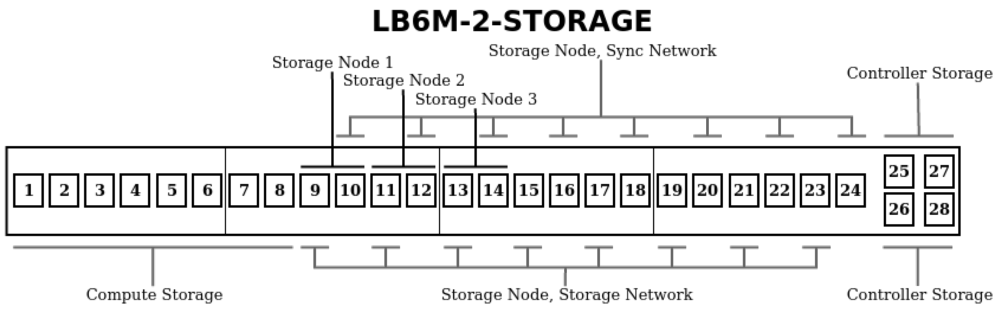
Port	Host	Port	Host
1	stack-controller-1	25	stack-controller-1
2	stack-controller-2	26	stack-controller-2
3	stack-controller-3	27	stack-controller-3
4	stack-compute-1	28	stack-compute-1
5	stack-compute-2	29	stack-compute-2
6	stack-compute-3	30	stack-compute-3
7	stack-storage-1	31	
8	stack-storage-2	32	
9	stack-storage-3	33	
10		34	
11		35	
12		36	
13		37	
14		38	
15		39	
16		40	
17		41	
18		42	
19		43	
20		44	
21		45	
22		46	
23		47	
24		48	*Cerberus*
_	_	SFP 1	*LB6M-1-PUBLIC* Port 1
_	_	SFP 2	*LB6M-1-PUBLIC* Port 2

2.2.4 LB6M-1-PUBLIC



Port	Host	Port	Host
1	*LB4M-1* SFP 1	13	
2	*LB4M-1* SFP 2	14	
3	*LB4M-2* SFP 1	15	stack-compute-1
4	*LB4M-2* SFP 2	16	stack-compute-2
5		17	stack-compute-3
6		18	
7		19	
8		20	
9		21	
10		22	
11		23	
12		24	
_	_	25	stack-controller-1
_	_	26	stack-controller-2
_	_	27	stack-controller-3
_	_	28	

2.2.5 LB6M-2-STORAGE



Port	Host	Port	Host
1	stack-compute-1	13	stack-storage-3 (storage)
2	stack-compute-2	14	stack-storage-3 (sync)
3	stack-compute-3	15	
4		16	
5		17	
6		18	
7		19	
8		20	
9	stack-storage-1 (storage)	21	
10	stack-storage-1 (sync)	22	
11	stack-storage-2 (storage)	23	
12	stack-storage-2 (sync)	24	
_	_	25	stack-controller-1
_	_	26	stack-controller-2
_	_	27	stack-controller-3
_	_	28	

2.3 Cluster Nodes

Acorn runs a 9-node OpenStack cluster with 3 controller, 3 compute, & 3 storage nodes.

2.3.1 Controller Nodes

CPU	Intel Xeon L5640 6-Core 2.2GHz
Chassis	1U Supermicro XC815TQ-560B
HDD	250GB OS
Motherboard	Supermicro X8DTU-F
NIC	2x1GB Onboard & 2x1GB via Supermicro AOC-PG-12+
PSU	1x560w
RAID Controller	LSI 9211-4i
RAM	32GB

The chassis' top cover has a locking screw on it's front-right side.

The OS drive is in the leftmost bay.

NICs

rear panel location	interface	network	ip range
bottom-left	enp1s0f0	management	10.2.1.11-40
bottom-right	enp1s0f1	overlay	10.3.1.11-40
top-left	enp3s0f0	provider	n/a
top-right	enp3s0f1	storage	10.4.1.11-40

2.3.2 Compute Nodes

CPU	2x AMD Opteron 6172 12-Core 2.1GHz
Chassis	1U HP Proliant DL165 D7
HDD	1TB OS
NIC	4x1GB Onboard & 2x10GB via HP NC522SFP PCI-E
RAM	48GB

The chassis' top cover has no locking screw.

The OS drive is in the leftmost bay.

NICs

rear panel location	interface	network	ip range
ethernet "4" - top	enp3s0f1	management	10.2.1.41-70
ethernet "3" - bottom	enp3s0f0	overlay	10.3.1.41-70
ethernet "2" - left	enp4s0f1	not used	
ethernet "1" - right	enp4s0f0	not used	
fiber left	ens1f0	provider	n/a
fiber right	ens1f1	storage	10.4.1.41-70

Eventually, we might use the spare 2x 1GB NICs as failovers for the fiber links, or a fatter overlay pipe.

2.3.3 Storage Nodes

8x3TB + 2x6TB per node gives us a safe size of 24TB & risky size of 36TB, with the ability to add 1 more drives per node, or a journal drive.

http://florian.ca/ceph-calculator/

CPU	2x Intel Xeon E5645 6-Core 2.4Ghz
Chassis	2U Supermicro CSE-826TQ-R800LPB SuperChasis
HDD	250GB OS, 8x3TB + 2x6TB SAS Storage
Motherboard	Supermicro X8DTN+
NIC	2x1GB Onboard & 2x10GB via Supermicro AOC-STGN-i2S
PSU	2x800w
RAID Controller	8-Ports via Adaptec ASR-5805, 4-Ports via ASR-5405Z
RAM	48GB

The chassis' top cover has a locking screw on it's back-left side.

NICs

rear panel location	interface	network	ip range
ethernet left	enp10s0f0	management	10.2.1.71-100
ethernet right	enp10s0f1	not used	
fiber top	enp9s0f0	storage	10.4.1.71-100
fiber bottom	enp9s0f1	sync	10.5.1.71-100

Eventually, we might use the spare 1GB NIC as a failover for a fiber link.

HDDs

This is the order the OS sees the drives as being in:

_	left			right
top	11	02	05	08
middle	10	01	04	07
bottom	09	12	03	06

The OS drive lives in bay `1`, the 8x 3TB HDDs live in bays `2` to `9`, & the 2x 6TB HDDs live in bays `10` & `11`.

CHAPTER

THREE

ADMINISTRATION & MAINTENANCE GUIDES

- *Server Rack*
  - *Switching KVM Ports*
  - *Configuring LB4M/LB6M Switches*
- *Networking*
  - *Disabling the Squid Proxy*
  - *Add Static IPs*
  - *Killing A Host's Network Access*
  - *Add or Modify DNS Entries*
  - *Switch Internet from CVALink to Telnes*
- *Website/VPS*
  - *Updating Wordpress*
  - *Importing Newsletter Emails*
  - *Optimizing Images*
- *Arch Linux*
  - *LAN Package Cache*
- *CUPS Print Server*
  - *Install CUPS Server on Slackware*
  - *Configure CUPS Clients*
- *Slackware*
  - *14.0 Upgrade*
- *Windows*
  - *Fresh Install*

3.1 Server Rack

3.1.1 Switching KVM Ports

To switch between hosts on the KVM, press `PrintScreen` on the keyboard to pop up the selection window, then the up/down arrow keys to navigate the host list, and `Enter` to select a host.

3.1.2 Configuring LB4M/LB6M Switches

Serial Connection

For the initial configuration of an LBXM switch, you will need to do everything over a serial connection. We use a Prolific USB to Serial Adapter paired with a DB9 RS232 to RJ45 Adapter Cable:

1. Connect the two adapters so you have a USB to RJ45 cable.
2. Grab a Windows laptop.
3. Install the Drivers for the Prolific Adapter from the included CD or the Prolific Driver Site.
4. Download PuttySSH or KittySSH.
5. Plug the USB end of the cable into the laptop and the RJ45 end into the `Console` port of the switch.
6. Launch Putty or Kitty.
7. Select the `Serial` radio option.
8. Enter the COM port of the adapter into the address input box. Usually this is `COM3` or `COM4`. Check the COM section of Device Manager if you are unsure.
9. Leave the speed at the default value, or try `115200` if you have issues.
10. Hit the `Ok` button.
11. If a window pops up but there is no text try hitting the `Enter` key to bring up the login prompt.
12. Authenticate using the `admin` user and a blank password.

You can now do the initial configuration of the switch. Afterwards, you will be able to connect via SSH instead of a Serial connection.

Initial Setup

The initial setup for every switch should set a unique hostname & static IP, and enable the ports & SSH server:

```
enable
ip ssh
network parms 192.168.1.211 255.255.255.0
network gateway 192.168.1.254
configure
no shutdown all
hostname LB6M-1-Public
exit
write memory
```

You can now connect to the switch via SSH:

```
$ ssh admin@192.168.1.211
```

Link Aggregation

We use LAG/LACP to aggregate multiple 10G fiber links between the LB4M switches and the LB6M switches.

You will need to group the ports together. Start by SSHing into a switch and then configure the interfaces on the LB4M:

```
enable
configure
interface 0/49,0/50
addport 3/1
lacp actor admin state passive
exit
interface 3/1
description 'LACP to LB6M-1-Public'
no port-channel static
port-channel load-balance 6
exit
exit
write memory
exit
```

On the LB6M switch, add matching entries for each LB4M:

```
enable
configure
interface 0/1,0/2
addport 1/1
lacp actor admin state passive
exit
interface 1/1
description 'LACP to LB4M-1-Public'
no port-channel static
port-channel load-balance 6
exit
exit
write memory
exit
```

3.2 Networking

3.2.1 Disabling the Squid Proxy

To save bandwidth, we use a transparent Squid proxy as a network-wide web cache. Sometime some software doesn't like to play nice with this & you might need to temporarily disable the re-routing of HTTP requests to the proxy and just send them straight out of the WAN.

You can do this from Cerberus.

Start by commenting the following line in `/etc/pf.conf` (by adding a `#` to the front of it):

```
rdr pass on $int_if inet proto tcp from any to any port www -> 192.168.1.254 port 3128
```

That line is responsible for redirecting all HTTP traffic to the proxy. With it commented out, you can refresh PF by running `pf_reconfig` & the proxy should be bypassed.

You can verify this by looking at the Squid Proxy Screen on Zabbix.

To re-enable the Squid proxy, simply remove the `#` from the line in `/etc/pf.conf` & re-run `pf_reconfig`.

3.2.2 Add Static IPs

You can assign a host a static IP address from Cerberus. You'll need the MAC address of the host's network interface. The available static IP range is `150` to `189`.

Start by editing `/usr/local/etc/dhcpd.conf`:

```
sudo nano /usr/local/etc/dhcpd.conf
```

In the `group { }` section, add a new `host { }` section(ensuring the IP addresses are in ascending order):

```
host ComputerHostname {
    fixed-address 192.168.1.183
    hardware ethernet AA:BB:CC:11:22:33;
}
```

Check your config by running `dhcpd -t`. If there are no errors, restart the `isc-dhcpd` service:

```
sudo service isc-dhcpd restart
```

3.2.3 Killing A Host's Network Access

If a computer is hogging the internet & you don't know whose it is, you might just want to kill their network access. You can do this from Cerberus & you need either their hostname(`MyCarKeys.acorn`), or their IP address.

If you need to figure out who is hogging the internet, try running `sudo iftop` or check http://cerberus.acorn/bandwidth/.

If you only have their hostname, figure out their IP addresses using `dig`:

```
$ dig MyCarKeys.acorn | grep 192.168.1
MyCarKeys.acorn.    3600    IN      A       192.168.1.36
cerberus.acorn.             86400   IN      A       192.168.1.254
```

Now open up `/etc/pf.conf` and add the following between the `block drop log on $ext_if all` line and the `antispoof for $ext_if int` line:

```
block log quick from <HOSTS_IP> to any
# For Example: block log quick from 192.168.1.36 to any
```

Now run the `pf_reconfig` command(which is just an alias for `sudo pfctl -f /etc/pf.conf`) to refresh PF.

This will only block new connections from the Host, you also need to use `pfctl` to kill all their current connections:

```
$ sudo pfctl -k MyCarKeys.acorn
# Or use their IP
$ sudo pfctl -k 192.168.1.36
```

You might need to run this two or three times to kill all the connections.

To unblock their network access, simply remove or comment out the line you added to `/etc/pf.conf` and re-run `pf_reconfig`.

3.2.4 Add or Modify DNS Entries

To add, edit, or delete the DNS entries for the `.acorn` domain, we use `nsupdate` to send commands to the BIND server on Cerberus.

You will need the keyname & secret for DNS updates from `/etc/namedb/named.conf`.

```
$ nsupdate -y KEYNAME:KEYSECRET
> update add <Domain> <TTL> <Type> <Destination>
> send

# Adding a CNAME record
> update add projects.acorn 86400 CNAME aphrodite.acorn
> send

# Adding new A & PTR records
> update add allium.outdoor.acorn 86400 A 192.168.1.246
> update add 246.1.168.192.in-addr.arpa 86400 PTR allium.outdoor.acorn
> send

# Deleting A, TXT, & PTR records
> update delete barn.outdoor.acorn A
> update delete barn.outdoor.acorn TXT
> update delete 245.1.168.192.in-addr.arpa PTR
> send
```

Note: The DHCP server will automatically create A, TXT, & PTR records for hosts, pointing `<hostname>.acorn` to their IP address. These records are tied to the hosts MAC address via the TXT record.

This means that the DNS records will not be updated if a host's MAC address changes. To fix this, you need to delete the A, TXT, & PTR records for the host and then renew the DHCP lease from the host(e.g., run `ipconfig /renew` on windows).

3.2.5 Switch Internet from CVALink to Telnes

When our main internet(from CVALink) is down, you can follow these directions to switch to our slower, backup internet(from Telnes).

- Go to back of server rack.
- Unplug black cable(labelled `WAN`) from port 10 in 2nd patch panel(not the switch) and plug it into the Ethernet `P0` port on the Telnes modem on shelf at top of rack.
- SSH into Cerberus, edit `/etc/rc.conf`: `sudo nano /etc/rc.conf`
- Comment out the following lines by adding a # sign in front of them:

```
ifconfig_em1="inet 104.245.228.34 netmask 255.255.255.248"
defaultrouter="104.245.228.33"
```

- Un-comment the Telnes lines by removing the leading # sign:

```
#ifconfig_em1="inet 208.46.125.98 netmask 255.255.255.248"
#defaultrouter="208.46.125.97"
```

- You should now have something like this:

```
#ifconfig_em1="inet 104.245.228.34 netmask 255.255.255.248"
#defaultrouter="104.245.228.33"

ifconfig_em1="inet 208.46.125.98 netmask 255.255.255.248"
defaultrouter="208.46.125.97"
```

- Save the file and exit.
- Restart the network interfaces & routing service:

```
sudo service netif restart; sudo service routing restart
```

You should now have a working connection, you can test it by pinging Google:

```
ping 8.8.8.8
```

If there are still internet problems after following this procedure, it's highly likely that the Telnes connection is down as well.

You should check the lights on the top of the modem, if they are all green there's a small chance the problem is with Cerberus.

Test that by plugging the modem into a laptop instead of Cerberus and setting it to connect with the above static IP. If that doesn't work, or the lights aren't all green, call Telnes support - they will probably ask you to plug the modem directly into the internet box on the side of the Seed Office warehouse.

3.3 Website/VPS

3.3.1 Updating Wordpress

Backup the files & database first:

```
cp -r ~acorn/htdocs ~/acorn_wp_backup
mysqldump -u acorn acorn > ~/acorn_wp.sql
```

Then Log-In, visit the Updates page, and hit `Update`.

Sometimes the `reCAPTCHA` plugin's API keys need to be re-entered. You can grab those from the reCAPTCHA Admin by logging in as `acorncommunity@gmail.com`.

3.3.2 Importing Newsletter Emails

You can use this procedure if you have a list of emails you want to add to our newsletter.

You'll need a text file containing the emails or a CSV file(without a header row) of `Name,Email`.

- Log in to our Sendy server.
- Click the `SESE Retail` brand.
- Click `View all lists` under `Lists & Subscribers` in the left menu.
- Click the `Garden Guide` list.
- Click the `Add Subscribers` button at the top of the page.
- Either select & upload your file, or paste it into the box & submit the form.

3.3.3 Optimizing Images

There is a cronjob that runs this monthly, but if you've done a bulk image upload and want to optimize them immediately, you can run these commands from the SESE VPS:

```
find ~/public_html/images -iname '*.png' -exec optipng -o7 -quiet -preserve -log ~/
↪optipng.log '{}' \;
find ~/public_html/images -iname '*.jpg' -exec jpegtran -copy none -optimize -
↪progressive -outfile '{}' '{}' \;
```

3.4 Arch Linux

3.4.1 LAN Package Cache

We have a shared Arch Linux package cache at `ssh://cache@aphrodite.acorn:/mnt/DataShare/Misc/Cache/pacman/pkg`.

You can follow these steps to link your Arch Linux workstation up to the shared cache:

```
# become the root user
sudo -i
# create ssh key, copy to aphrodite.acorn
ssh-keygen -t ecdsa
ssh-copy-id cache@aphrodite.acorn
# add mountpoint to fstab
echo 'cache@aphrodite.acorn:/mnt/DataShare/Misc/Cache/pacman/pkg  /var/cache/pacman/
↪pkg   fuse.sshfs  defaults,_netdev,allow_other    0    0' >> /etc/fstab
```

Clearing pacman's cache will delete all packages except those that are currently installed. In a shared cache where computers may have different packages installed, clearing the cache will remove packages other computers have installed.

You can fix this by changing the `CleanMethod` option in `/etc/pacman.conf` to `KeepCurrent`.

See also:

https://wiki.archlinux.org/index.php/Custom_local_repository_with_ABS_and_gensync#Network_shared_pacman_cache

https://wiki.archlinux.org/index.php/SSHFS

3.5 CUPS Print Server

3.5.1 Install CUPS Server on Slackware

Install CUPS & the various printer drivers:

```
slackpkg install cups hplip gutenprint ghostscript ghostscript-fonts lcms2 poppler
```

Enable running on boot:

```
chmod +x /etc/rc.d/rc.cups
```

Edit the config at `/etc/cups/cupsd.conf`:

```
Port 631
ServerName printers.acorn
ServerAlias *
Browsing On

<Location />
    Order allow,deny
    Allow from 127.0.0.1
    Allow from 192.168.1.*
</Location>
<Location /admin>
    AuthType Basic
    Order allow,deny
    Allow from 127.0.0.1
    Allow from 192.168.1.*
</Location>
<Location /admin/conf>
    AuthType Basic
    Order allow,deny
    Allow from 127.0.0.1
    Allow from 192.168.1.*
</Location>
```

Start the server:

```
/etc/rc.d/rc.cups start
```

Visit http://printers.acorn:631, click `Administration` & log in as `root`. Click `Find New Printers` & `Add Printer`.

For the HP LaserJet M601, use the JetDirect Connection Socket `socket://yourprinter:9100` with the HP LaserJet 600 M601 Postscript driver.

Add PDF Printer(optional)

Install the additional dependencies:

```
slackpkg install libmpc mpfr
```

Install `cups-pdf` via SlackBuilds:

```
mkdir ~/builds; cd ~/builds
wget http://slackbuilds.org/slackbuilds/14.0/office/cups-pdf.tar.gz
```

(continues on next page)

(continued from previous page)

```
tar xvfz cups-pdf.tar.gz
cd cups-pdf
wget http://www.cups-pdf.de/src/cups-pdf_3.0beta1.tar.gz
./cups-pdf.SlackBuild
installpkg /tmp/cups-pdf*_SBo.tgz
```

Add HTTP Proxy(optional)

This allows you to access http://printers.acorn for management, instead of http://printers.acorn:631.

Add the following Virtual Host to `/etc/httpd/extra/httpd-vhosts.conf`:

```
<VirtualHost *:80>
    ServerName printers.acorn
    ServerAlias www.printers.acorn
    ProxyRequests Off
    ProxyPass / http://localhost:631/
    <Proxy *>
        Order allow,deny
        Allow from all
    </Proxy>
    <Location />
        ProxyPassReverse http://localhost:631/
        ProxyHTMLEnable On
        ProxyHTMLURLMap / /
    </Location>
</VirtualHost>
```

3.5.2 Configure CUPS Clients

Arch Linux

```
# Install
pacman -S libcups

# Add Server
echo 'ServerName printers.acorn:631/version=1.1' > /etc/cups/client.conf
```

3.6 Slackware

3.6.1 14.0 Upgrade

Fully upgrade the current distribution:

```
slackpkg update
slackpkg upgrade-all
```

Run LILO & reboot if the kernel was upgraded:

```
lilo -C /etc/lilo.conf
reboot
```

Now insert the Slackware 14.0 DVD or mount the ISO:

```
mount /dev/sdg /mnt/cdrom
```

Switch into single-user mode:

```
telinit 1
```

Blacklist the kernel & 3rd party packages by adding the following to `/etc/slackpkg/blacklist`:

```
kernel-firmware
kernel-headers
kernel-source
kernel-generic
kernel-generic-smp
kernel-huge
kernel-huge-smp
kernel-modules
kernel-modules-smp
[0-9]+_SBo
[0-9]+alien
[0-9]+compat32
```

Navigate to the DVD mount point, install the new kernel & update slackpkg:

```
cd /mnt/cdrom/slackware64
installpkg a/kernel-huge-3.2.29-x86_64-1.txz
installpkg a/kernel-modules-3.2.29-*.txz
upgradepkg ap/slackpkg-2.82.0-noarch-8.tgz
```

Find & merge any new config files:

```
find /etc -name "*.new"
vimdiff /etc/slackpkg/mirrors.new /etc/slackpkg/mirrors
vimdiff /etc/slackpkg/blacklist.new /etc/slackpkg/blacklist
```

Upgrade the package utilities & tools:

```
upgradepkg a/pkgtools-*.tgz
upgradepkg a/tar-*.tgz
upgradepkg a/xz-*.tgz
upgradepkg a/findutils
```

Update the package list:

```
slackpkg update
```

First upgrade the C libraries, then all packages:

```
slackpkg upgrade glibc-solibs
slackpkg upgrade-all
```

Remove any deprecated packages:

```
slackpkg clean-system
```

Install the new packages:

```
slackpkg install kmod
slackpkg install-new
```

After upgrading, use the slackpkg menu or vimdiff to go through the configuration files and merge/remove .new files:

```
find /etc -name "*.new"
vimdiff /etc/mdadm.conf.new /etc/mdadm.conf
# Or run
slackpkg new-config
```

Edit `/etc/lilo.conf` to include an entry to the old kernel:

```
image = /boot/vmlinuz-huge-2.6.37.6
    root = <same as above entry>
    label = "2.6.37.6"
    read-only
```

Reconfigure lilo, switch out of single-user mode and reboot the computer:

```
lilo -C /etc/lilo.conf
telinit 3
reboot
```

If the computer booted successfuly, edit `/boot/lilo.conf` and remove the entry to the old kernel. Also remove the kernel lines from `/etc/slackpkg/blacklist`.

Check for new kernel upgrades:

```
slackpkg update
slackpkg upgrade-all
```

Reconfigure lilo and reboot if a new kernel was installed:

```
lilo -C /etc/lilo.conf
reboot
```

Finally, rebuild all custom SlackBuilds and remove the filters from the /etc/slackpkg/blacklist file.

3.7 Windows

3.7.1 Fresh Install

This is what we do to our Windows workstations after a clean install.

Configuration

Users

Create an `SESE` user as well as an `Admin` administrator account.

Networking

Open up the IPv4 settings for the network connection & set the `WINS` server to `192.168.1.254`.

Misc

Create links in the Windows Explorer Favorites menu to `//Aphrodite/Community`, `//Aphrodite/Personal`, & `//Vishnu/Business`.

Applications

There is a folder that contains the setup files for commonly installed applications at `//Aphrodite/Community/Applications/5 Fresh Windows Install`.

Internet Explorer

Updating to Windows 7 Service Pack 1 & Internet Explorer 11 is required for computers that will be used with `StoneEdge`.

The default version of Internet Explorer(and therefore MS Access & StoneEdge) uses **only** insecure SSL versions & ciphers, which are all disabled on the SESE website.

If you skip this step, the computer will not be able to import orders from the website.

Mumble

- Follow or cancel the Audio Wizard.
- Follow the Certificate Wizard.
- **Add a new favorite server:**
    - Name: Acorn Chat Server
    - Address: chat.acorn
    - Port: 64738
    - Username: <hostname of new computer>
    - Password: <blank>
- **Set the following options under `Configure -> Settings`:**
    - User Interface -> Enable `Hide in Tray`
    - User Interface -> Disable `Use selected item as the chat bar target`
    - Network -> Enable both settings under `Connection`
    - Overlay -> Disable the Overlay

In the Start Menu, copy the Mumble application to the `Startup` folder.

Firefox/Chrome

Add the following bookmarks:

- Acorn Accounting
- Acorn Project Tracker
- Acorn Wiki

Add the following addons/extensions:

- HTTPSeverywhere
- uBlock Origin
- Disconnect

Zabbix Monitoring Agent

Grab the agent archive from `\\Aphrodite\Community\Applications\5 Fresh Windows Install\zabbix_agents.win.zip` or from the Downloads Page.

Extract it to `C:\zabbix\` and edit the `conf/zabbix_agentd.win.conf` file with notepad, changing the following settings:

```
Server=monitor.acorn
ServerActive=monitor.acorn
Hostname=<workstations_hostname>
```

Save the file to `C:\zabbix_agentd.conf`. Hit `Win+R` and enter `cmd` to open a terminal. `cd` to the exracted `bin\win32` or `bin\win64` directory and run `zabbix_agentd.exe -i` then `zabbix_agentd.exe -s`.

Open up Windows firewall and manually add entries allowing the `zabbix_agentd.exe` through.

Now head to Acorn's Zabbix Server and log in. At the `Configuration -> Hosts` Page, click the `Create host` button.

Set the following options:

- Hostname - the same Hostname defined in the workstation's config file.
- Groups - Windows workstations
- Agent interface - Connect to DNS. The DNS name should be "<hostname>.acorn"
- Templates - OS Windows Workstation. Be sure to click add before clicking save!
- Inventory - Set to manual or automatic and add any relevant details that you know.

Save the new host.

After a short while, the host's Z icon should turn blue, this means the host is being monitored correctly. You can check the latest data by selecting `Monitoring -> Latest Data` and selecting the new workstation from the dropdown menus.

Tweaks

Unfragmented Paging File

Windows normally increases the size of the paging file as needed. When the disk starts to fill up this can cause the paging file to become fragmented.

This can be circumvented by allocating a single size to the paging file instead of using the default range, immediately after installing Windows.

Windows 7

- Right-click `Computer` in Start Menu.
- Click `Properties`.
- Click `Advanced system settings` link.
- Click `Performance Settings...` in `Advanced Tab`.
- Click `Change...` in `Virtual memory` box in `Advanced Tab`.
- Uncheck `Automatically manage paging file size for all drives`
- Click `Custom Size:` radio button.
- Enter the desired size (size of RAM + 300MB allows for a full core dump).

- Click `Set`.
- Click `OK` for all dialogs.
- Restart Computer.

CHAPTER

FOUR

SLACKWARE SERVER ADMINISTRATION

Fabric is used to automate package installation and upgrades on Acorn's Slackware servers. The `slackware_servers` module includes a `fabfile.py` that defines the possible commands.

Install Fabric using `pip`:

```
pip install fabric
```

List possible commands:

```
cd slackware_servers
fab -l
```

Update all packages on all hosts:

```
cd slackware_servers
fab upgrade_all_packages
```

CHAPTER

FIVE

DEBIAN WORKSTATION AUTOMATED SETUP

The office_workstation folder contains files used for automated installation, configuration and maintenance of Acorn's Linux Workstations, which run Debian Linux.

5.1 Quickstart

1. Download the Debian Stretch Netinstall Image.
2. Copy the ISO to a USB Stick:

```
dd if=debian-stretch.iso of=/dev/sdg
```

3. Boot the new workstation from the USB stick. When the installer menu pops up, hit escape and enter the following:

```
auto hostname=NewWorkstation url=http://lucy.acorn/workstation-preseed.cfg
```

4. After installation is complete, jump to your workstation and install ansible:

```
pip install ansible
```

5. Add the new workstation to the workstations file:

```
echo 'NewWorkstation.acorn' >> playbook/workstations
```

6. Copy your SSH key over to the new workstation:

```
ssh-copy-id seseadmin@NewWorkstation.acorn
```

7. Run the playbook:

```
cd playbook; ansible-playbook acorn.yml
```

8. Make some coffee…
9. Once the playbook finishes, you should be logged in as the Public User and Mumble should have popped up.
10. Go through Mumble's Audio Wizard, complete the Certificate Wizard. To get our Mumble server to show up in the favorites, you will have to rerun the playbook with the mumble tag:

```
ansible-playbook acorn.yml -t mumble
```

11. Right-click the Desktop & hit Unlock Widget, then Configure Desktop. Change the Wallpaper tab's Layout option to Folder View.

12. You might need to do some our layout tweaks, like rearranging the Desktop Icons, or increasing the height of the Task Bar(`Right-Click Task Bar -> Panel Options -> Panel Settings`). Afterards, right-click the Desktop again and choose `Lock Widgets`.
13. Open PlayOnLinux and hit `Run a Local Script`. Choose the `PlayOnLinux_msoffice.sh` file in the Home directory.
14. Cleanup by removing the MS Office ISO and the PlayOnLinux script and shortcut from the Public User's home folder.
15. Reboot the workstation.

5.2 Automated Installs

The `preseed.cfg` file is configuration file that can be used with the Debian Automated Installer. It is based off of the Automated Install documentation and the example pre-seed file .

Simply boot up using a netinstall image. When the graphical menu appears, press `<ESC>` and enter `auto hostname=<workstation_hostname> url=<preseed_url>`. For example, if you wanted the new workstation to be named `HelloWorld` and your preseed was hosted at http://lucy.acorn/~prikhi/preseed.cfg, you would type:

```
auto hostname=NewWorkstation url=http://lucy.acorn/workstation-preseed.cfg
```

This will automatically partition the drives and setup an SSH server along with an `seseadmin` admin user.

The Ansible playbook may then be used for further configuration.

Note: You can use the *mkpasswd* command to generate crypted passwords for the pre-seed file:

```
printf "someSecurePassword" | mkpasswd -s -m sha-512
```

5.3 Ansible Setup

While the Debian Automated Install gets a minimal system up and running with just an SSH server, the Ansible playbook adds the GUI(KDE), desktop applications, and Acorn specific customizations. It will do the following actions:

- Install basic system utilities and the desktop environment(KDE)
- Install standard applications (LibreOffice, Firefox, Chromium, Flash)
- Configure for use with network services (samba, zabbix, cups)
- Create a public user
- Apply a standardized configuration to the public user (bookmarks, shortcuts)
- Prepare the workstation and public user for installing MS Office
- Create personal user accounts and apply specific configurations to them

Start by using `pip` to install Ansible:

```
pip install ansible
```

You can then run the entire playbook using `ansible-playbook`:

```
cd office_workstation/playbook
ansible-playbook acorn.yml
```

New hosts may be added to the `workstations` file. Plays will only be run if the host requires it.

You may run specific `tags` using the `-t` flag. The following command will only install and configure the Zabbix agent on hosts that do not have the agent installed or are improperly configured:

```
ansible-playbook acorn.yml -t zabbix
```

The following `tags` are available:

- `kde` - Install/Remove/Configure KDE packages.
- `apps` - Install/Remove available applications.
- `zabbix` - Install and configure the Zabbix agent.
- `samba` - Configure Samba and mount network shares on boot.
- `cups` - Install and configure the CUPS client(for printing).
- `users` - Create and configure accounts for all users.
- `public_user` - Create and configure a Public user account.
- `pavan` - Create and configure Pavan's user.

5.3.1 Playbook Overview

The playbook will first copy over the apt sources file. This ensures all workstations use a common mirror which allows caching via web proxy(we use squid). Then the new mirrors available packages are updated.

Next various applications are installed such as the desktop environment, web browsers, games, and educational applications. KDE applications are explicitly installed(instead of being implicity linked to the `kde-desktop` task).

The Zabbix agent is then installed and configured. We rely on Zabbix's auto-discovery features, monitoring only system resource usage.

Next we set up printing by installing and configuring the CUPS client, using a central print server instead of configuring printers on each machine.

A Public User is then created and application and DE customizations are copied over to it's home directory. Any additional users for specific people are then created and customized.

Samba is then setup to use a common workgroup and WINS server. Personal and Community samba shares are set to be automatically mounted on boot.

We then prepare the Public User's home directory for installing Microsoft Office 2007 using PlayOnLinux. This will mount the install ISO, copy over patch files and create a PlayOnLinux script in the Public User's home directory. The script must still be run manually.

Finally, we configure SDDM, the Display/Login Manager, to automatically login as the Public User.

5.3.2 Microsoft Office 2007

PlayOnLinux requires a GUI to install programs, so this playbook only prepares a workstation for the installation, the actual installation must be done by hand. The installation can be run by opening up PlayOnLinux, selecting `Tools -> Run a Local Script`, then choosing to run the `PlayOnLinux_msoffice.sh` script found in the Public User's home directory.

A network share containing the following files is required:

- An ISO of the Microsoft Office 2007 install disk
- The bin, lib and share folders for Wine 1.2.3(manually install Wine 1.2.3 using PlayOnLinux to get a copy of these)
- The wine-gecko install file
- The XP SP3 patch file

The Playbook will copy these files to the proper directories & mount the ISO.

5.3.3 Customization

The playbook can be modified for other networks by creating a replacement for the `acorn.yml` file. You can override any variables found in the `roles/common/vars/main.yml` file. This will allow you to customize various specifics like the CUPS or WINS servers and the name of the Public user account.

Variables can also be set in the `workstations` file. See the Ansible Documentation for more information.

5.3.4 Contributing

You should make sure any new features are properly abstracted from your specific implementation through the use of templates and variables.

The main issue tracker lives at http://bugs.sleepanarchy.com/projects/sysadmin, feel free to create a new issue(attach a patch file if you have one). Pull requests are also accepted from our github mirror at https://github.com/prikhi/sysadmintools.

5.4 Automated Maintenance with Fabric

A `fabfile.py` for Fabric is also included to help automate workstation maintenance. Currently it may be used to automatically install and upgrade packages.

First make sure you have Fabric installed:

```
pip install Fabric
```

To get a full list of commands, run `` fab` with the ``-l `` flag:

```
cd office_workstation
fab -l
```

To upgrade all packages, use the `update_and_upgrade` command:

```
fab update_and_upgrade
```

To upgrade all packages **and** install any new dependencies, use `full_upgrade`:

```
fab full_upgrade
```

5.5 To Do

- Abstract KDE specificities into a separate role
- Change some of the Public User's config files into templates or tasks, especially ones that have the `sese` user hardcoded in them.
- Add a role that uses a lightweight DE along with customizations for the Public User(for low-power comps or laptops).
- Refactor the "iommu=pt" grub option needed for SewingMachine into a `host_var` file.
- Address deprecation warnings.
- Update public user files for debian 9 & new KDE.
- Use Ansible Vault for password hashes.
- Pre-configure mumble so the audio wizard isn't required.
- Configure udevil to allow cifs mounting

CHAPTER

SIX

ACORN VM CLUSTER

Acorn's VM cluster runs OpenStack Stein on Ubuntu Server 18.04 LTS nodes.

TODO: Use apcupsd to shutdown VMs & nodes on power loss.

TODO: Document required variables by each role/group

TODO: Investigate Cinder Backup

6.1 Setup & Configuration

The initial OS install is automated using a Ubuntu pre-seed file. This will install a basic Ubuntu SSH server onto a node.

Then some manual setup is required to setup `sudo` access & the network interfaces.

The OpenStack configuration is automated using an Ansible playbook.

Our High Availability services require some manual setup when first initializing the cluster and when adding new nodes to a running cluster(see *Cluster Initialization* & *Cluster Expansion*).

6.1.1 Node Setup

Ubuntu Pre-Seed File

The `devstack-preseed.cfg` file is a pre-seed file for the Ubuntu Automated Installer. It sets up an ssh server, a `stack` user, and installs `python`, `git`, `htop`, & `vim`.

Start by booting up a Ubuntu Mini Image, when the menu appears, press `<TAB>` and add the following:

```
auto=true priority=critical interface=<management_interface> hostname=<desired_
↪hostname> url=<preseed_url>
```

See also:

If you don't know which interface on a node connects to the management network, reference the *Cluster Nodes* section.

Note: If you are using different hardware, leave out the `interface` option to enable the selection menu, then press `<CTRL>-<ALT>-<F2>` to get a shell and investigate which is plugged in, or just plug every network port in and specify `interface=auto`.

For example:

```
auto=true priority=critical interface=enp1s0f0 hostname=stack-controller-1.acorn␣
↪url=http://lucy.acorn/devstack-preseed.cfg
```

You will face issues if the installation media is on a USB stick and the installer sees it as `/dev/sda`. Grub will try to install to the MBR of `/dev/sda` and fail. To fix this, open a shell and run `grub-installer /target`, then choose to install grub to the proper drive(probably `/dev/sdb`).

Sudo Setup

The nodes should be setup to use `sudo` without a password:

```
visudo
# Add the following line:
# %sudo ALL=(ALL) NOPASSWD: ALL
```

Network Setup

The network connections & IP addresses must be setup by manually. Refer to the *Cluster Nodes* section for the interface & IP range to use for each node & network type. Configure the networks by editing `/etc/network/interfaces`:

```
# The Management Network Interface
auto enp0s8
iface enp0s8 inet static
    address 10.2.1.11
    gateway 10.2.1.5
    netmask 255.255.255.0
```

On controller & compute nodes, add the Provider & Overlay Network Interfaces:

```
# The Provider Network Interface
auto enp0s9
iface enp0s9 inet manual
up ip link set $IFACE up
down ip link set $IFACE down

# The Overlay Network Interface
auto enp0s10
iface enp0s10 inet static
    address 10.3.1.11
    netmask 255.255.255.0
```

On controller, compute, & storage nodes, add the Storage Network Interface:

```
# The Storage Network Interface
auto enp0s11
iface enp0s11 inet static
    address 10.4.1.11
    netmask 255.255.255.0
```

On storage nodes, add the Storage Sync Network:

```
# The Storage Sync Network Interface
auto enp0s12
```

(continues on next page)

(continued from previous page)

```
iface enp0s12 inet static
    address 10.5.1.71
    netmask 255.255.255.0
```

Then restart the networking service:

```
sudo systemctl restart networking
```

We also need to manually set the DNS resolver in `/etc/systemd/resolved.conf`:

```
DNS=10.2.1.5
```

6.1.2 Ansible Playbook

The Ansible playbook is a series of tasks(grouped into roles) that ensure OpenStack is installed & properly configured. The playbook currently has a `common` role for all nodes, as well as specific roles for `controller` and `compute` nodes.

The `cluster-servers` file specifies the address, name and node type of each of our OpenStack servers. Currently there are three controller nodes, three compute nodes, & three storage nodes.

You can run the playbook by installing ansible with pip and using the `ansible-playbook` command inside the `playbook` directory:

```
sudo pip install ansible
cd playbook
ansible-playbook acorn.yml
```

6.2 Cluster Maintenance

6.2.1 Cluster Status

There are various commands and log files that you can use to assess the cluster's health & status.

Pacemaker

The controller nodes run `pacemaker` to share the virtual IP address for the `stack-master-controller.acorn` domain. Only one controller node will be the master-controller at once. Pacemaker also ensures various services like the load balancer `haproxy` are running.

To check the status of the Pacemaker cluster, run `sudo pcs status` on any controller node. This will display the node with the master-controller virtual IP and the status of services on each controller node.

To restart the pacemaker service on a controller node, run `sudo systemctl restart pacemaker`. Log files for the service can be found at `/var/log/pcsd/` and you can run `sudo journalctl -u pacemaker` for additional service information.

MySQL

The MySQL database is replicated on each controller node. You can check a node's status in the cluster by running `sudo mysql` to open an SQL shell, and then run a `SHOW STATUS LIKE 'wsrep_local_state%';` query. A state comment of `Synced` means the node is synced with the rest of the cluster. You can check the logs by running `sudo journalctl -u mariadb`.

Ceph

The controller nodes are the monitors & manager for the Ceph storage cluster used by OpenStack.

To check the health of the storage cluster, run `ceph health`. For more detailed information, run `ceph status`. While the cluster is being re-balanced or repaired, you can run `ceph -w` to print the status and then monitor any health-related messages. Run `ceph iostat` to continuously print a table of the cluster's I/O activity.

The storage nodes have a Ceph OSD for each storage drive. To print the layout of storage nodes & OSDs, run `ceph osd tree`. For more detailed information like the used & available space per-OSD, run `ceph osd status`.

Ceph log files are found in `/var/log/ceph/` on the controller & storage nodes. You can view the ceph services running on each node by running `systemctl`. To restart all ceph services on a node, run `sudo systemctl restart ceph.target`. You can start a specific OSD by SSHing into the OSDs storage node and running something like `sudo systemctl start ceph-osd@2.service`.

OpenStack

You can check the status of openstack services & VMs by using the `openstack` command from any controller node. You will need to source one of the credential files on the node, e.g.: `. ~/admin-openrc.sh`.

Check the available services using `openstack service list`. Check the hypervisor services with `openstack compute service list` and the networking services with `openstack network agent list`. Check the hypervisor stats with `openstack hypervisor stats show` and the per-hypervisor resources with `openstack hypervisor list --long`.

Source the `~/acorn-openrc.sh` credential file to display details of the acorn project. `openstack server list` will show all VMs and their status. Similarly, run `openstack volume list` for all volumes, `image list` for all images, & `flavor list` for all VM flavors.

See `openstack --help` for all available commands and flags.

6.2.2 Automated Maintenance

There is a Fabric file that can be used to automatically update and upgrade the cluster servers:

```
fab upgrade
```

TODO: Fabric command to check & bootstrap inactive galera cluster?

6.2.3 Adding OS Images

You can download pre-made images or create your own image for Linux & Windows VMs using a virtualizer like KVM or VirtualBox.

Once you have an image file, you need to convert it to the `raw` format using `qemu-img`:

```
qemu-img convert -O raw my-src-image.qcow2 my-target-image.raw
```

Then you can add the image to the cluster via the Dashboard:

- Login to the Dashboard using the admin credentials.
- Under the `Admin` menu, select `Compute` and `Images`.
- Click `Create Image`, give it a name and description, select your raw image file, and change the format to `Raw`.
- Hit `Create Image` to upload the image file.
- Once complete, you should be able to switch to the `acorn` project & launch a VM using your new image.

6.2.4 Adding VM Flavors

Flavors let you set the available resources for a VM. You can customize the CPU count, RAM, swap, & OS drive size.

- Login to the Dashboard using the admin credentials.
- Under the `Admin` menu, select `Compute` and `Flavors`.
- Hit the `Create Flavor` button.
- Name the flavor and specify the resources sizs, then hit `Create Flavor`.

6.2.5 Adding / Replacing Storage Drives

When a storage node's OS drive fails, you need to replace the drive, create a new 1 volume RAID array using the Adapatec Configuration boot utility.

When a storage drive fails, you will need to shutdown the node, swap the drive out, & initialize the new JBOD drive:

- SSH into a controller node and run `ceph osd set noout` to prevent rebalancing when you take the storage node offline.
- Run `sudo poweroff` on the storage node to shut it off.
- Swap out the HDD with a replacement drive. We use 3TB SAS drives.
- Power the node back on.
- During the boot process, you will receive an error from the RAID card stating the drive configuration has changed. Press `Control-a` to enter the RAID setup.
- Select the RAID adapter that controls the new drive.
- Select the `Initialize Drives` option & select the new drive with `Space` and then press `Enter` to confirm.
- Press `Escape` to go back to the menu and select `Create JBOD`.
- Select the new drive and confirm the JBOD creation.
- Exit the menu to boot into the OS.

- Once the storage node has booted up, SSH into `stack-controller-1.acorn`.
- Enter the Ceph Deploy directory with `cd ceph-cluster` and deploy an OSD to the replacement drive by running `ceph-deploy osd create stack-storage-<node-number> --data /dev/<new-drive>`.
- Run `ceph osd unset noout` to enable data rebalancing on drive failure.
- Login to the Dashboard using the admin credentials.
- Ensure you are viewing the `admin` project. Under the `Identity` menu, navigate to the `Projects` page.
- Click the dropdown for the `acorn` project and select `Modify Quotas`.
- Click the `Volume` tab and enter the new `Total Size` of the cluster, using the `Safe Cluster Size` value from the Ceph Calculator.
- Hit the `Save` button.

6.2.6 Extending a Volume

If a VM's volume is running out of free space, you can extend the volume to increase it's total size:

- SSH into the VM & shut it down.
- Login to the Dashboard & navigate to the `Instances` page under the `Project > Compute` menu.
- Click the dropdown menu for the VM and click the `Detach Volume` option and select the desired volume.
- Navigate to the `Volumes` page under the `Volumes` menu.
- Use the dropdown menu for the volume to resize the volume.
- Once the volume is resized, re-attach it to the VM and boot up the VM.
- SSH into the VM.
- Unmount the volume by running `sudo umount <mount-point>` or `sudo umount /dev/<volume-disk>`. Run `mount` to see a list of all mounted drives on the system.
- Run `gdisk /dev/<volume-disk>` to partition the new drive.
- We need to delete the existing partition and create an identical one that fills up the entire disk partition. If there is only one partition, you can enter `d` to delete it, then `n` to create a new one and accept the defaults in the following prompts by hitting `<Enter>` to make it take the entire disk. Then enter `w` to write the new parition table.
- Run `sudo partprobe` to pickup the partition table changes.
- Run `sudo ressize2fs /dev/<volume-partition>` to extend the filesystem to the new partition size.

6.2.7 Shutting Down

To shutdown a cluster:

- Shutdown all the VMs using the web UI or with the `openstack server stop <server1> <server2> ...` command from a controller node.
- SSH into the Compute nodes and shut them down by running `sudo poweroff`.
- On a controller node, disable storage rebalancing so we can take the storage cluster offline by running `ceph osd set noout`.

- SSH into each Storage node and shut them down.
- On a controller node, put the pacemaker cluster into maintenance mode by running `pcs property set maintenance-mode=true`.
- Shut off the controller nodes in a staggered fashion from node 3 to node 1. E.g., shutdown `stack-controller-3`, wait a minute, shutdown 2, wait a minute, shutdown 1.

6.2.8 Starting Up

If you ever need to start a stopped cluster:

- Start up the Master Controller Node
- If this was the last node shutdown, run `sudo galera_new_cluster`.
- If you don't know which controller shutdown last, check `/var/lib/mysql/grastate.dat` on each controller for `safe_to_bootstrap:  1`. Run `sudo galera_new_cluster` on this controller.
- Once the first MySQL server starts, start mysql on the other nodes by running `systemctl start mysql`.
- Now start the Storage nodes. Verify all disks are up by running `ceph osd tree` on a controller node. Check the health of the storage cluster by running `ceph status`.
- On a controller node, re-enable drive re-balancing by running `ceph osd unset noout`.
- Start the Compute nodes.
- Once everything has booted up, you should be able to start the VMs from the dashboard.

6.2.9 Shutting Down

If you need to shutdown the cluster(e.g., in case of a power outage), do so in the following order:

- VMs
- Compute Nodes
- Storage Nodes
- Backup Controller Nodes
- Master Controller Node

6.2.10 Cluster Expansion

Adding additional controller, compute, or storage nodes to a cluster is fairly straightforward.

For every node, you should first follow the *Node Setup* section. Then add the host to a group in the `cluster-servers` file & add a config file in `host_vars/` (base it off of the configs for other hosts in that group).

Then run the full ansible playbook:

```
ansible-playbook acorn.yml
```

Controller

New controllers require some manual configuration due to the high availability setup.

MySQL

The new controller should automatically connect to the MySQL cluster. You can verify this by checking the cluster size:

```
echo "SHOW STATUS LIKE '%cluster_size';" | mysql -u root -p
```

RabbitMQ

The ansible playbook will have copied an erlang cookie to all the controller hosts. Restart the new node in clustering mode:

```
sudo rabbitmqctl stop_app
sudo rabbitmqctl join_cluster rabbit@stack-controller-1
sudo rabbitmqctl start_app
```

Pacemaker

You'll need to authenticate the new node from the master controller:

```
# On stack-controller-1
sudo pcs cluster auth -u hacluster stack-controller-4
```

Next, remove the default cluster from the new node:

```
# On stack-controller-4
sudo pcs cluster destroy
```

Add the new node using the master controller and start the service on the new node:

```
# On stack-controller-1
sudo pcs cluster node add stack-controller-4

# On stack-controller-4
sudo pcs cluster start
sudo pcs cluster enable
```

Ceph

Copy the SSH key from the master controller to the new controller:

```
# On stack-controller-1
ssh-copy-id stack-controller-4
```

Install & deploy Ceph on the new controller node:

```
# On stack-controller-1
cd ~/storage-cluster
ceph-deploy install --repo-url http://download.ceph.com/debian-luminous stack-
↪controller-4
ceph-deploy admin stack-controller-4
```

Setup the new controller as a Ceph monitor:

```
ceph-deploy mon add stack-controller-4
```

Copy the Glance Key to the new controller node:

```
# On stack-controller-1
ceph auth get-or-create client.glance | ssh stack-controller-4 sudo tee /etc/ceph/
↪ceph.client.glance.keyring
ssh stack-controller-4 sudo chown glance:glance /etc/ceph/ceph.client.glance.keyring
```

Extra Deploy Node

Copy the SSH key from each existing controller to the new controller:

```
ssh-copy-id stack-controller-4
```

Then initialize a key on the new server & copy it to the existing controller and storage nodes:

```
ssh-keygen -t ecdsa -b 521
ssh-copy-id stack-controller-1
ssh-copy-id stack-controller-2
ssh-copy-id stack-controller-3
ssh-copy-id stack-compute-1
ssh-copy-id stack-compute-2
ssh-copy-id stack-compute-3
ssh-copy-id stack-storage-1
ssh-copy-id stack-storage-2
ssh-copy-id stack-storage-3
```

And finally, copy over the `~/ceph-cluster` folder from an existing controller node to a new one:

```
# On stack-controller-1
rsync -avhz ~/ceph-cluster stack-controller-4:~/
```

Neutron

Add the new controller as a DHCP agent for the private network:

```
cd ~
. admin-openrc.sh
# Run this & find the ID of the `DHCP agent` on the new controller
openstack network agent list

# Then add the agent as a DHCP server
neutron dhcp-agent-network-add <dhcp-agent-id> private
```

Compute

The ansible playbook should handle all the required setup and OpenStack should pickup the additional compute node afterwards.

You can verify this by running `openstack compute service list` on a controller node. The list should include the new compute host.

Storage

Follow the *Setup & Configuration* instructions, then add the hostname to the `storage` group in the `cluster-servers` file and run the ansible playbook.

This will install Ceph and setup Cinder, but you'll need to manually add the new node and any new storage drives to our Ceph cluster.

Start by pushing the SSH key from the master controller to the new node:

```
# On stack-controller-1
ssh-copy-id stack-storage-4
```

Then use `ceph-deploy` on the master controller to install Ceph on the new node:

```
cd ~/storage-cluster
ceph-deploy install --repo-url http://download.ceph.com/debian-luminous stack-storage-
↪4
```

Note that we use `--repo-url` here instead of the `--release` flag, so that packages are downloaded through HTTP instead of HTTPS, which allows them to be cached by our web proxy.

Deploy an OSD to each new storage disk. It's recommended to split the journals out on a separate SSD with a partition for each OSD:

```
ceph-deploy disk list stack-storage-4
ceph-deploy osd create stack-storage-4:/dev/sdc:/dev/sdb1 stack-storage-4:/dev/sdd:/
↪dev/sdb2
```

You can monitor the rebalancing progress by running `ceph -w` on stack-controller-1.

6.3 Cluster Initialization

This details the process we went through the to initialize all cluster components for the first time. This requires some additional steps compared to starting an initialized cluster, or adding nodes to an existing cluster(see *Cluster Maintenance* for those topics).

6.3.1 Initial Cluster Setup

Setup the OS, sudo access, & networking for all of your Controller, Compute, & Storage Nodes according to the *Node Setup* section.

Add variables file for all of your nodes to the `host_vars` directory - look at hosts in the same group to see what variables need to be defined.

Next run the ansible playbook with the `initial` tag:

```
ansible-playbook acorn.yml -t initial
```

This will fail when mysql is restarted because there is no running cluster for the nodes to join - but that's OK because restarting MySQL is the last task in the `initial` tag.

Now run the playbook with the `ha` tag to install the High Availability dependencies:

```
ansible-playbook acorn.yml -t ha
```

Follow the instructions in the *High Availability Initialization* section to setup the MySQL Cluster, Master Controller Virtual IP Address, & HAProxy.

On your controllers, add the Open vSwitch Bridge:

```
sudo ovs-vsctl add-br br-provider
```

On your compute nodes, add the Open vSwitch Bridge & attach the provider interface:

```
sudo ovs-vsctl add-br br-provider
sudo ovs-vsctl add-port br-provider THE_NODES_PROVIDER_INTERFACE
```

Now run the entire playbook:

```
ansible-playbook acorn.yml
```

Once that's finished, follow the instructions in the *Ceph Initialization* section.

You should be set now, you can verify by running the following commands on the first controller node:

```
cd ~
. admin-openrc.sh

# Image Service
sudo apt-get install -y qemu-utils
wget http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-disk.img
qemu-img convert -f qcow2 -O raw cirros-0.3.5-x86_64-disk.img cirros.raw
openstack image create "cirros" --file cirros.raw --disk-format raw \
    --container-format bare --public
openstack image list

# Compute Service
openstack compute service list

# Networking Service
neutron ext-list
openstack network agent list

# Block Storage Service
openstack volume service list
```

(continues on next page)

(continued from previous page)

```
# Launch a VM
openstack flavor create --id 0 --vcpus 1 --ram 64 --disk 1 nano
. acorn-openrc.sh
openstack security group rule create --proto icmp default
openstack security group rule create --proto tcp --dst-port 22 default
openstack network list
PRIVATE_NETWORK_ID="$(openstack network list -f value -c ID -c Name | grep private |␣
↪cut -f1 -d' ')"
openstack server create --flavor nano --image cirros \
    --nic net-id=$PRIVATE_NETWORK_ID --security-group default test-instance
openstack server list
openstack floating ip create provider   # Check the created IP
FLOATING_IP="$(openstack floating ip list -c 'Floating IP Address' -f value)"
openstack server add floating ip test-instance $FLOATING_IP
echo $FLOATING_IP
# Should be able to ssh in as `cirros` w/ password `cubswin:)`
```

6.3.2 High Availability Initialization

Some manual setup is required for highly available controller nodes. All your Controller nodes should be online during this process & you should have already run the Ansible playbook with the `ha` tag.

MySQL

Stop the mysql server on the first controller node & start it as a cluster:

```
# On stack-controller-1
sudo systemctl stop mysql
sudo galera_new_cluster
```

Once that has finished, you can start mysql on the other controller nodes:

```
# On stack-controller-2, stack-controller-3
sudo systemctl start mysql
```

RabbitMQ

Join the backup controllers to the master controller:

```
# On stack-controller-2, stack-controller-3
sudo rabbitmqctl stop_app
sudo rabbitmqctl join_cluster rabbit@stack-controller-1
sudo rabbitmqctl start_app
```

Then, on any controller node, enable mirroring of all queues:

```
sudo rabbitmqctl cluster_status
sudo rabbitmqctl set_policy ha-all '^(?!amq\.).*' '{"ha-mode": "all"}'
```

Pacemaker

Ansible only installs the Pacemaker & HAProxy packages. You will need to create the cluster & Virtual IP address when first creating the OpenStack cluster.

Start by SSHing into stack-controller-1, removing the initial config file & authenticating the controller node:

```
# On stack-controller-1
sudo pcs cluster destroy
sudo pcs cluster auth stack-controller-1 stack-controller-2 stack-controller-3 \
    -u hacluster -p PASSWORD
```

Create, start, & enable the cluster:

```
sudo pcs cluster setup --start --enable --name acorn-controller-cluster \
    --force stack-controller-1 stack-controller-2 stack-controller-3
```

Set some basic properties:

```
sudo pcs property set pe-warn-series-max=1000 \
    pe-input-series-max=1000 \
    pe-error-series-max=1000 \
    cluster-recheck-interval=3min
```

Disable STONITH:

```
sudo pcs property set stonith-enabled=false
```

Create the Virtual IP Address:

```
sudo pcs resource create management-vip ocf:heartbeat:IPaddr2 \
    ip="10.2.1.10" cidr_netmask="24" op monitor interval="30s"
```

Add HAProxy to the cluster & only serve the VIP when HAProxy is running:

```
sudo pcs resource create lb-haproxy lsb:haproxy --clone
sudo pcs constraint order start management-vip then lb-haproxy-clone kind=Optional
sudo pcs constraint colocation add lb-haproxy-clone with management-vip
```

Add the Glance service to Pacemaker:

```
sudo pcs resource create glance-api lsb:glance-api --clone --force
```

Add the Cinder service to Pacemaker:

```
sudo pcs resource create cinder-api lsb:cinder-api --clone interleave=true --force
sudo pcs resource create cinder-scheduler lsb:cinder-scheduler --clone␣
↪interleave=true --force
```

6.3.3 Ceph Initialization

Ansible only installs the `ceph-deploy` tool on controller nodes, the Ceph storage cluster must be manually initialized.

Ceph Setup

Start by SSHing into the master controller, we'll make running repeated commands easier by setting some array variables:

```
# On stack-controller-1
CONTROLLERS=('stack-controller-1' 'stack-controller-2' 'stack-controller-3')
COMPUTE=('stack-compute-1' 'stack-compute-2' 'stack-compute-3')
STORAGE=('stack-storage-1' 'stack-storage-2' 'stack-storage-3')
```

Then generate an SSH key & copy it to the Controller & Storage nodes:

```
ssh-keygen -t ecdsa -b 521
for SRV in "${CONTROLLERS[@]}" "${COMPUTE[@]}" "${STORAGE[@]}"; do ssh-copy-id $SRV;␣
↪done
```

Now create a directory for the cluster configuration:

```
mkdir ~/ceph-cluster
cd ~/ceph-cluster
```

Deploy the initial cluster with the Controller nodes as monitors:

```
ceph-deploy new --public-network 10.4.1.0/24 ${CONTROLLERS[@]}
```

Open up the `ceph.conf` in `~/ceph-cluster/` and add the cluster network & nearfull ratio settings:

```
cluster_network = 10.5.1.0/24
mon_osd_nearfull_ratio = 0.67
```

A `nearfull ratio` of `0.67` is based off of allowing 1-node to fail in a 3-node ceph cluster.

Install Ceph on the nodes(we specify the full repo URL instead of just using `--release mimic` to avoid HTTPS, allowing packages to be cached by our web proxy):

```
ceph-deploy install --release mimic --repo-url http://download.ceph.com/debian-mimic $
↪{CONTROLLERS[@]} ${STORAGE[@]}
```

Then create the initial monitors & start them on boot:

```
ceph-deploy mon create-initial
for SRV in "${CONTROLLERS[@]}"; do
    ssh $SRV sudo systemctl enable ceph-mon.target
done
```

Next, add the OSDs. You'll want an SSD with a journal partition for each OSD(`/dev/sdb#`), and an HDD for each OSD:

```
# Block Storage
ceph-deploy osd create stack-storage-1 --data /dev/sdc
ceph-deploy osd create stack-storage-1 --data /dev/sdd
```

(continues on next page)

(continued from previous page)

```
ceph-deploy osd create stack-storage-2 --data /dev/sdc
ceph-deploy osd create stack-storage-2 --data /dev/sdd
# etc.

# File Storage
ceph-deploy osd create stack-storage-1 --filestore --data /dev/sdc --journal /dev/sdb1
# etc.

# If your drive layout is identical on every storage server:
OSDS=('/dev/sdc' '/dev/sdd')
for SRV in "${STORAGE[@]}"; do
    for OSD in "${OSDS[@]}"; do
        ceph-deploy osd create $SRV --data $OSD
    done
done
```

Now copy the configuraton file & admin key to the controller nodes:

```
ceph-deploy admin ${CONTROLLERS[@]}
```

And set the correct permissions on the admin key:

```
for SRV in "${CONTROLLERS[@]}"; do
    ssh $SRV sudo chmod +r /etc/ceph/ceph.client.admin.keyring
done
```

Enable the manager daemon:

```
ceph-deploy mgr create ${CONTROLLERS[@]}
```

Check the health of the storage cluster with `ceph health` & watch syncing progress with `ceph -w`.

OpenStack Integration

Now we'll make OpenStack use the Ceph cluster for Image & Block storage. Start by creating some pools to use:

```
ceph osd pool create volumes 512 replicated replicated_rule 64
rbd pool init volumes
ceph osd pool create vms 128
rbd pool init vms
ceph osd pool create images 64
rbd pool init images
```

Create Ceph Users for the various OpenStack Services, and assign them the appropriate pool permissions:

```
ceph auth get-or-create client.glance mon 'allow r' osd 'allow class-read object_
↪prefix rbd_children, allow rwx pool=images'
ceph auth get-or-create client.cinder mon 'allow r' osd 'allow class-read object_
↪prefix rbd_children, allow rwx pool=volumes, allow rwx pool=vms, allow rwx␣
↪pool=images'
```

Then copy them to your nodes:

```
# Copy glance key to controllers
for SRV in ${CONTROLLERS[@]}; do
```

(continues on next page)

(continued from previous page)

```
    ceph auth get-or-create client.glance | ssh $SRV sudo tee /etc/ceph/ceph.client.
↪glance.keyring
    ssh $SRV sudo chown glance:glance /etc/ceph/ceph.client.glance.keyring
done

# Copy cinder key to controller & compute nodes
for SRV in "${CONTROLLERS[@]}" "${COMPUTE[@]}"; do
    ceph auth get-or-create client.cinder | ssh $SRV sudo tee /etc/ceph/ceph.client.
↪cinder.keyring
done

# Set the correct permissions on controller nodes
for SRV in "${CONTROLLERS[@]}"; do
    ssh $SRV sudo chown cinder:cinder /etc/ceph/ceph.client.cinder.keyring
done
```

Copy the `ceph.conf` to the Compute nodes(it should already be present on the other nodes):

```
for SRV in "${COMPUTE[@]}"; do
    ssh $SRV sudo tee /etc/ceph/ceph.conf < /etc/ceph/ceph.conf
done
```

Display the secret key for the `client.cinder` ceph user and add it to the ansible password vault as `vaulted_rbd_cinder_key`:

```
ceph auth get-key client.cinder
```

Generate a UUID to use for the `libvirt` secret using `uuidgen`. Add the UUID to the ansible password vault as `vaulted_rbd_cinder_uuid`. Make sure to re-run the ansible playbook for the compute nodes so the libvirt secret is added(`ansible-playbook acorn.yml -t compute`).

Finally, restart the OpenStack services:

```
# On Controller
for SRV in "${CONTROLLERS[@]}"; do
    ssh $SRV sudo systemctl restart glance-api
    ssh $SRV sudo systemctl restart cinder-volume
done

# On Compute
for SRV in "${COMPUTE[@]}"; do
    ssh $SRV sudo systemctl restart nova-compute
done
```

6.4 Appendix: Notes From Setup Trials

These are notes from various initial setup attempts we went through as we built our final configuration.

6.4.1 3/9/17 Test

This was done when setting up controller High Availability.

Started w/ just 1 controller node, no compute, no storage.

First playbook run failed at starting mysql, had to start new cluster:

```
sudo galera_new_cluster
```

Re-run, broke at rabbitmq user, fixed by re-ordering tasks & restarting rabbitmq before adding.

Re-run broke a bootstrapping identity service, needed to remove config options, fix name of config file.

Re-run broke at setting up projects. Need to do initial pacemaker config. Had to change `hacluster` user password manually.

Re-run finished all updated tasks(after Glance setup). Image service verified by image listing. Image creation does not work due to ceph not being setup.

Updated nova's tasks & config for controller HA.

Failed at nova re-run due to existing service but wrong endpoints.

Failed at nova addresses already bound. Fixed by setting `osapi_compute_listen`, `novncproxy_host`, & `metadata_listen_host` to management IP.

TODO: PR OpenStack HA Docs to Fix Required Nova Listen Options

Re-run finished all nova tasks. Nova service verified by compute service list.

Updated neutron's tasks & config.

Failed at neutron.wsgi unable to bind address. Fixed by setting `bind_host` in neutron.conf

TODO: PR OpenStack HA Docs to Fix Required Neutron Listen Options

Re-run finished all neutron tasks. Verified by service list.

Updated cinder's tasks & config.

Re-run finished all cinder tasks, verify by volume service list.

Updated horizon tasks.

Re-run finished all horizon tasks, verify by visitng site.

Re-run failed at creating router, not enough l3 agents available. Fixed by lowering min to `1`.

Re-run completed all controller tasks.

Add 1 Compute Node

Did minimal setup for new node & re-ran ansible playbook.

Verified by running `openstack compute service list`.

Add 2 Storage Nodes

Did minimal setup for new nodes & re-ran ansible playbook.

Followed initial ceph setup.

Verified by running `openstack volume service list`.

Test stack by adding image, & launching server by making image into volume.

Add Backup Controller Node

Did minimal setup for new nodes & re-ran ansible playbook.

Failed at restarting mysql. Issue was wrong list of ips for cluster setting. After fixing, it failed when trying to restart galera, since it brought all cluster servers down. Fixed by staggering restarts, backup controllers first, then the master controller.

Rerun of playbook passed. Followed instructions from "adding nodes".

Tested by shutting down controller 1 and provisioning a server. Failed at openstack auth, needed to copy fernet keys from master controller. Fixed by adding keys to vault.

Was then able to get token, failed at uploading image. Needed to setup ceph keys. After fixing & documenting, was able to create image, launch server, & SSH in. Then started master controller and shutdown backup, still able to SSH into server.

6.4.2 4/30/17 Test

Trial moving Ceph monitors to Controller. Started by wiping block storage servers, & purging ceph & data from controllers.

Ran ansible playbook.

SSH into controller, push ssh keys.

Deploy new node to controllers:

```
ceph-deploy new stack-controller-1 stack-controller-2
```

Install:

```
ceph-deploy new stack-controller-1 stack-controller-2 \
    stack-storage-1 stack-storage-2 stack-storage-3
```

From creating initial monitors onwards works the same. Verified by uploading image, creating volume, & launching instance.

6.4.3 5/1/17 Test 1

Testing setup of all nodes at once. Started with fresh install from preseed file on 2 controllers, 1 compute, & 3 storage nodes.

Ran playbook once, expected failure when restarting mysql for first time, since no cluster was initialized.

Setup master controller & then restarted mysql on backup:

```
# On stack-controller-1
sudo galera_new_cluster

# On stack-controller-2
sudo systemctl restart mysql
```

Then ran playbook again. Failed at retrieving openstack user list. Followed high availability setup instructions.

Then ran playbook again, finished fine. Followed with Ceph Initialization.

After Ceph finished, verified all services from master controller:

```
cd ~
. admin-openrc.sh

# Image Service
sudo apt-get install -y qemu-utils
wget http://download.cirros-cloud.net/0.3.5/cirros-0.3.5-x86_64-disk.img
qemu-img convert -f qcow2 -O raw cirros-0.3.5-x86_64-disk.img cirros.raw
openstack image create "cirros" --file cirros.raw --disk-format raw \
    --container-format bare --public
openstack image list

# Compute Service
openstack compute service list

# Networking Service
neutron ext-list
openstack network agent list

# Block Storage Service
openstack volume service list

# Launch a VM
openstack flavor create --id 0 --vcpus 1 --ram 64 --disk 1 m1.nano
. acorn-openrc.sh
openstack security group rule create --proto icmp default
openstack security group rule create --proto tcp --dst-port 22 default
openstack network list
PRIVATE_NETWORK_ID="$(openstack network list -f value -c ID -c Name | grep private |␣
↪cut -f1 -d' ')"
openstack server create --flavor m1.nano --image cirros \
    --nic net-id=$PRIVATE_NETWORK_ID --security-group default test-instance
openstack server list
openstack floating ip create provider   # Check the created IP
FLOATING_IP="$(openstack floating ip list -c 'Floating IP Address' -f value)"
openstack server add floating ip test-instance $FLOATING_IP
echo $FLOATING_IP
# Should be able to ssh in as `cirros` w/ password `cubswin:)`
```

6.4.4 5/1/17 Test 2

Rolled back to pre-ansible snapshots, ran playbook. Failed at mysql.

Initialized mysql cluster, then ran high availability playbook:

```
ansible-playbook acorn.yml -t ha
```

After completion, followed HA initialization setup. Re-ran full playbook. Controller 1 failed when trying to query networks. Had to modify playbook to flush handlers before setting up projects/networks. Rolled back to initial snapshot, re-tested & working OK now.

Ran Ceph initialization & verified cluster operation. Verification failed at compute service list, had to sync nova db & restart nova-compute on compute node. Failed again on volume service list due to unsync'd time, had to sync & restart:

```
sudo chronyc -a makestep
sudo systemctl cinder-volume restart
```

6.4.5 6/5/17 Additions

These changes been tested in a fresh install, but will be necessary next time we try.

On controllers:

```
sudo ovs-vsctl add-br br-provider
```

On computes:

```
sudo ovs-vsctl add-br br-provider
sudo ovs-vsctl add-port br-provider PROVIDER_INTERFACE
```

Verify distributed self-service networking: https://docs.openstack.org/newton/networking-guide/deploy-ovs-ha-dvr.html#verify-network-operation

6.4.6 6/6/17 Test

For testing DVR networking. Started w/ fresh preseed installs & all nodes running.

Ran playbook, controllers failed at mysql as expected. Initialized mysql cluster on controller-1. Started mysql on controller-2 afterwards.

Ran playbook. Failed at querying users for glance(since no VIP). Did HA setup.

Ran playbook. Failed at creating network. Did OVS setup & restarted `neutron-openvswitch-agent` & `neutron-metadata-agent` on controller & compute.

Ran playbook, everything passed. Did Ceph setup.

Verified everything, failed at assigning floating ip, had to restart `neutron-l3-agent` on compute nodes. Failed to ping from public LAN, tried some playbook tweaks & debugging but ended up rolling back to snapshot. Probably old config messing stuff up.

6.4.7 6/7/17 Test

Try to get DVR working again. . . .

Ran playbook, failed at mysql. Started cluster. Ran `ha` tags, setup pacemaker & OVS bridge.

Ran playbook, failed at creating neutron user. Re-ran playbook & it completed past that(maybe due to low resources?)

But failed at creating Router. Restarted neutron-metadata-agent on controllers & it completed(added restart to playbook).

Ran `pcs resource cleanup` to refresh pacemaker status.

Setup Ceph. Verified operation, can SSH into instance & ping internet.

6.4.8 8/4/17 Test

Test moving cinder-volume to controller nodes. 2 controllers, 1 compute, 2 storage.

Followed *Cluster Initialization*. Verified as working.

CHAPTER SEVEN

INDICES AND TABLES

- genindex
- modindex
- search