Building a virtualisation appliance with FreeBSD/bhyve/OpenZFS

Jason Tubnor

ICT Senior Security Lead

Latrobe Community Health Service

Introduction

- Building an virtualisation appliance for use within a NGO/NFP Australian Health Sector
  - About Me
  - Latrobe Community Health Service (LCHS)
- Background
- Problem
- Concept
- Production
- Reiteration

About Me

- 26 years of IT experience
- Introduced to Open Source in the mid 90’s
- Discovered OpenBSD in 2000
- A user and advocate of OpenBSD and FreeBSD
- Life outside of computers:
    - Ultra endurance gravel cycling

Latrobe Community Health Service

Latrobe Community Health Service (LCHS)

- Originally a Gippsland based NFP/NGO health service
- ICT manages 900+ users
- Servicing 51 sites across Victoria, Australia
- Covering ~230,000km^2
  - Roughly the size of Laos in Aisa or Minnesota in USA
- “Better health, Better lifestyles, Stronger communities”

Background

- First half of 2016 awarded contract to provide NDIS services
- Mid 2016 – deployment of initial infrastructure
  - MPLS connection
  - L3 switch gear
  - ESXi host running a Windows Server 2016 for printing services

Background – cont.

- Staff number grew
- We hit capacity constraints on the managed MPLS network
- An offloading guest was added to the ESXi host
- VPN traffic could be offloaded from the main network
    - Using cheaply available ISP internet connection

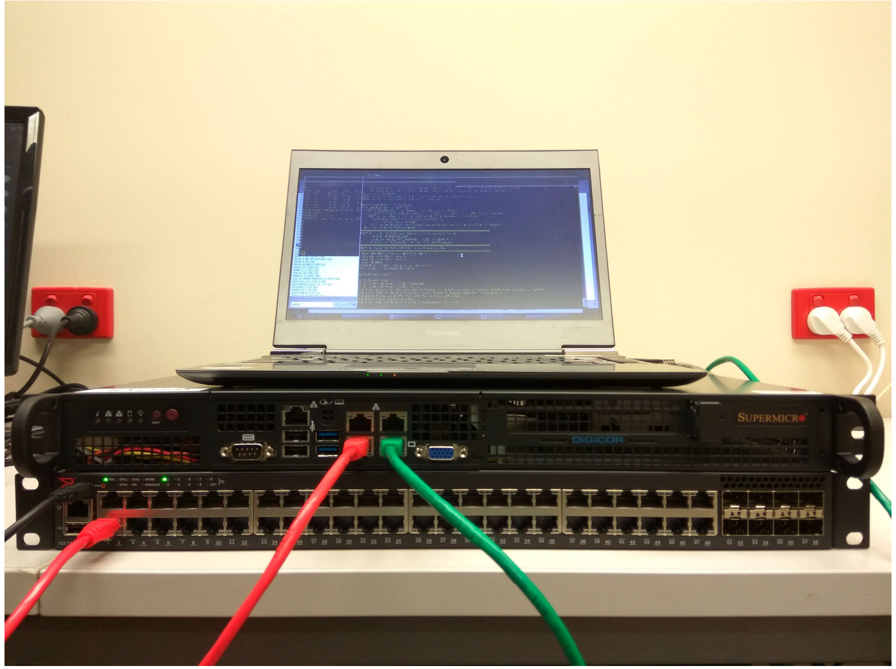
Problem

- Taking stock of the lessons learned in the first phase
- We needed to come up with a reproducible device
- Device required to be durable in harsh conditions
- Budget constraints/cost savings
- Licensing model
- Phase 2 was already being negotiated so a solution was required quickly

Concept

- bhyve [FreeBSD] was working extremely well in testing
  - Excellent hardware support
  - Liberally licensed
  - OpenZFS
  - Simplistic
  - Small footprint for a type 2 hypervisor
- Hardware discovery phase
  - FreeBSD
  - Required virtualisation components in CPU

Concept – cont.



Concept – cont.

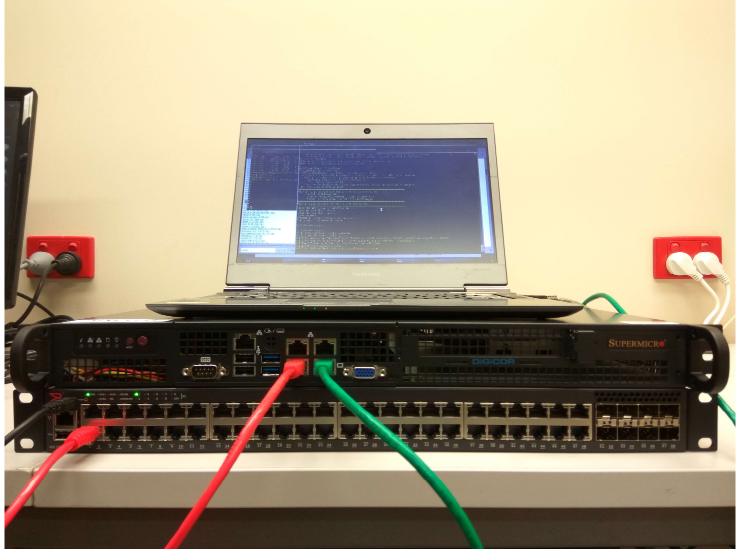
- SuperMicro SuperServer 5019A-FTN4 was chosen
  - 4 x 1Gb Ethernet ports
  - Low powered
  - Ran cool without relying on moving fans
- Storage (internal)
  - 2 x 240GB Intel Enterprise SSDs
  - OpenZFS used to mirror drives

Concept – cont.

- FreeBSD 11.0
  - Easy to maintain and report bugs
  - Patch support and delivery provided by the FreeBSD project
  - UEFI support for Windows Server 2016
  - 5 year Long Term Support (LTS)
- Guest Management
  - chyves (a fork of iohyve)

Concept – cont.

- Guests
  - OpenBSD 6.1 using grub-bhyve
  - Windows Server 2016 using UEFI
- Networking
  - Best security – VLAN on host
  - Main igb0 port a parent of multiple VLANs
  - Secondary port bridged to OpenBSD guest for offloading and/or VPN activites



Concept – cont.

- OpenZFS
  - Each guest had it own *zvol* for storage
  - Snapshots provide a fail-safe way to rollback in the event of a bad guest upgrade
- Ports/Packages installed:
  - openssh-portable
  - openntpd
  - grub2-bhyve
  - chyves
  - smartmontools
  - aria2
  - zfsnap2
  - zxfer

Concept – cont.

- Configuration:
  - /etc/rc.conf VLAN setup for bridging VLANs to guests:

```
ifconfig_igb0="up“
ifconfig_igb1="up“
vlans_igb0="vlan10 vlan11 vlan12 vlan13 vlan14“
create_args_vlan10="vlan 10 up“
create_args_vlan11="vlan 11 up“
create_args_vlan12="vlan 12 up“
create_args_vlan13="vlan 13 up“
create_args_vlan14="vlan 14 up“
ifconfig_vlan10="inet 10.1.1.20 netmask 255.255.255.0“
defaultrouter="10.1.1.1"
```

Concept – cont.

- Guest installation
  - OpenBSD was installed individually, not from a master image
  - Windows Server 2016 was installed from a maintained master image
    - 21GB in size

    -
      ```
      fetch -o - https://mirror.in.lchsict.com/pub/ndia/Win2k16-Server-20190121.zvol | zfs recv -Fv tank/vm/windowshost/disk0
      ```
    - Installation would take about 4 minutes

Concept – cont.

- Problems
  - chyves
    - Couldn’t handle boot priority when different boot methods were used
    - Required hacking the chyves library scripts depending on the OpenBSD install
    - Used a complex dataset layout
  - Boot method
    - Having two methods for starting guests was overly complex
    - Console access for the OpenBSD guest was difficult for a non-UNIX admin
    - The UEFI bootloader in ports at the time brought in compilers and other non-essential tools that should not exist on the host

Concept – cont.

- Problems – cont.
  - FreeBSD
    - Issues with network interfaces (required -txcsum -tso6 -tso4 -lro in /etc/rc.conf file) 11.0
    - hw.vmm.topology.cores_per_package="8“ and hw.vmm.topology.threads_per_core="1“ were required in /etc/loader.conf for guests with CPU licensing issues.

Production

- Problems were not a show stopper
- In its current state the concept device provided:
  - 90% usability
  - 100% functionality
- Project Point.5 had management commitment
- Went ahead and purchased inventory for V1.0 rollout
- Re-assess and refactor tooling as appliance matures to improve usability

Production – cont.

Version 1.0

- Supermicro SuperServer 5019A-FTN4
- 25 units
- FreeBSD 11.0
- chyves
  - grub-bhyve - OpenBSD
  - UEFI – Windows Server 2016

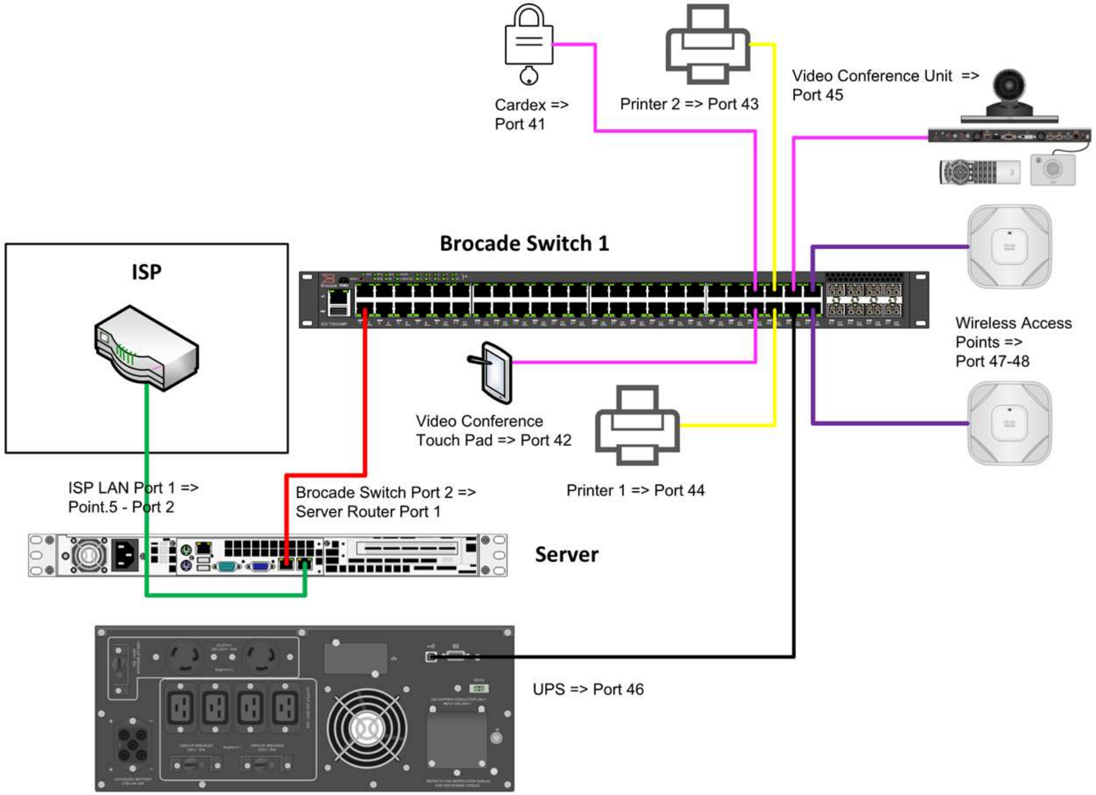
Production – cont.

- Appliances were spun up and shipped for install
- No issues on deployment
- freebsd-update fetch/install around guests wasn’t an issue
- VMWare ESXi host was even swapped out because of hypervisor support issues

Production – cont.

Installation

- Offload and full IKEv2 VPN editions cabled the same
- FTTP NTD, VDSL or ADSL modems attached to igb1
- All traffic VLAN trunked between appliance and switch


Cardex => Port 41
Printer 2 => Port 43
Video Conference Unit => Port 45
Brocade Switch 1
ISP
Wireless Access Points => Port 47-48
Video Conference Touch Pad => Port 42
Printer 1 => Port 44
ISP LAN Port 1 => Point.5 - Port 2
Brocade Switch Port 2 => Server Router Port 1
Server
UPS => Port 46

Reiteration

- Faster hardware required where environmental conditionals allowed
- All UEFI – no multiple boot loaders
- Simplistic management for all Administrators
- Address VNC console issues with bhyve/UEFI/OpenBSD
- Continue using other tools and workflows as per the original concept

Reiteration – cont.

Version 2.0

- Supermicro SuperServer 5019S-ML
- 11 units
- FreeBSD 11.1 and 11.2
- vm-bhyve
- OpenBSD and Windows Server 2016 both use UEFI
- Two different versions – thin guest and volume storage

FAQ

- Even if there were support issues with ESXi why chose bhyve?
  - VMWare ESXi would cause random crashing on OpenBSD guests usually when OpenBSD was under heavy IKEv2/ipcomp load or the ingestion of a large route table. bhyve never exhibits these issues with some units having very long uptimes.
- Why was vm-bhyve used?
  - Out of the box, vm-bhyve has worked faultlessly. Where there were gaps of missing features, they have been quickly addressed. The next ports release of vm-bhyve should see the introduction in detection of the media invoked by the installer – needed for OpenBSD.
- Are you planning to uplift the appliance to FreeBSD 12?
  - No. Currently FreeBSD does not have a LTS release outside of the 11.x branch. There was also sufficient breakage in the 12.0-RELEASE when testing which has also contributed.

Conclusion

- While it meets the business need and solved our problem, it exceeded expections
- Technically it is termed a type 2 hypervisor, however, we consider the appliance to be a type 1. Small footprint only guests and essential tasks running on the host
- Rock solid reliability
- Compatible with a wide range of guests (as long as UEFI is supported)
- Fast and flexible
- … on the horizon

A Special Thanks

- FreeBSD Project
- Michael Dexter
- Peter Grehan
- Rodney Grimes
- ….. and all those that work tirelessly on open source software

Latrobe
Community
Health Service

Donate

- You too can help:
  - FreeBSD Foundation https://www.freebsdfoundation.org/

Q & A

Thank You

- Jason Tubnor
  - Email: jason.tubnor@lchs.com.au
  - Email: jason@tubnor.net
  - Twitter: @Tubsta