# Containers in production since 2004

Peter Tribble

# Who am I?

- Theoretical Astrophysicist
- Systems Administrator
- OpenSolaris participant
  - OGB Vice Chair
- illumos developer
  - Tribblix distro maintainer

# (what's illumos?)

- 2005 Sun open source Solaris
- I get involved
- 2010 along came Oracle…
- OpenSolaris forked as illumos
- Wide variety of distributions follow

# Solaris 10

- Released 2005 with big ticket features
- Zones
- ZFS
- DTrace
- SMF

# Zones

- Shared kernel
- Process isolation
- Filesystem isolation
- Separate network and port space
- Resource controls
- Hard security boundary
- Zero performance penalty

# Zones vs Containers

- Fundamentals are similar
  - namespaces
  - cgroups
- Allow multiple manifestations
  - LXC, LXD, Docker, ...
  - Traditional zones like LXC, full system container
- Technology vs Implementation blurred

# Early days

- Available to beta testers 2003/2004
- Put in production by accident
  - A web server had a power supply blow
  - Create zone with same IP address
  - Restore backup
  - Back in service in minutes

# Zones vs Containers

- Zones presented as a finished item
  - Not a kit of parts to use
- Solaris Zones tied to OS packaging
  - Shared installer, fully integrated
  - In hindsight, a bad decision
- Docker build and run are distinct
  - No comparable zones concept

# Building zones

- Not image-based (then)
- Build in layers
  - OS
  - Application stack
  - Configuration + data
- Never manage the OS in a zone
  - Just the application

# Example Configuration

```
zonename: illumos-build
zonepath: /export/zones/illumos-build
brand: whole-root
autoboot: true
limitpriv:
scheduling-class:
ip-type: shared
hostid:
fs-allowed:
fs:
dir: /export/packages
        special: /export/packages
        raw not specified
        type: lofs
        options: []
net:
        address: 192.168.0.212/24
        allowed-address not specified
        physical: e1000g0
        defrouter not specified
admin:
        user: ptribble
        auths: manage
```

# Zone variants

- Sparse-root
  - Mount OS from host readonly
- Whole-root
  - Copy OS from host
- ipkg (OpenSolaris derivatives)
  - Install minimal OS from network each time
- illumos has a much wider variety

# Converting doubters

- Project delayed by lack of hardware
  - Stressful meetings!
- Offer to build a zone instead
  - Users doubtful, but willing to give it a try
- Next day “Err, could we have another one”
- Benefit of transparent and “just works”

# Basic principles

- Delivery now independent of hardware
  - Make as many as you need
- Zone delivers one unit of functionality
  - Function, not process, not service
- Don't mix zones with non-zones
  - It does your head in
- Everything zoned, even if 1 zone per system
  - It's a portable abstraction that can be moved

# Simplification

- Applications use defaults
  - eg web always port 80, mysql 3306
  - logfiles always where you expect
- All instances look the same
  - Much easier for ops to handle
- Eliminates port mapping, redirectors, load balancers and all that junk

# Architecture

- Combine principles with simplification
- Deliver application the way it wants
- Deliver application in the unit that makes sense
- You can build your overall architecture to match the optimal needs of you application
- NOT mangling your application to fit your architecture!

# Scaling

- Sparse-root shares OS binaries
  - Zone footprint ~5M
- Whole-root zone has own fs
  - Zone footprint ~50M
- Shared network – 8192 zones max
- 1000 zones per system achievable
  - Other scaling limits come into play

# (aside - brands)

- In Linux, the syscall is the stable ABI
- In Solaris, libc is the stable ABI
  - So libc **must** match the kernel
- Originally, zone software was version locked
- That's restrictive, so we have **brands** ...
- … a **shim** layer to map incompatible ABIs between kernel and userland

# Later evolution

- BrandZ – Linux emulation
  - 2.4 kernels only
  - Never evolved and withered away
- “Containers”
  - Sun marketing…
  - Solaris 8/9 emulation layer (brand)
- Fully virtual network – project crossbow

# Zones vs Containers

- Zones do not have:
  - An API
  - uid mapping
  - pid mapping
  - overlay filesystems
  - nesting
  - port mapping

# Zones vs Containers

- Zones do have:
  - Persistent storage
  - Native networking
  - Full OS integration
    - Everything is zone-aware
  - First-class system status

# Zones in the open

- Moribund BrandZ resurrected as LX
  - Syscall shim, current Linux kernels
  - Backed by native kernel functionality
  - In SmartOS, OmniOS, Tribblix
  - Run a Docker image in a zone!
  - Productized by Joyent as Triton
    - Docker API, K8S, etc...

# Thank You!

Questions?