Narrative – Motivation for Docker

Slide 1 – Introduction

- General introduction
- Note I'm a JEE guy
- Work for American Electric Power
- Will be approaching this subject from an enterprise perspective

Slide 2 – Segregation of duties

- Different groups have different responsibilities in maintaining all the parts comprising a runtime image
  - Use AEP as an example
    - Open systems
      - OS services
        - bash
        - openSSL
    - Middleware
      - Application servers
      - Enterprise integration/messaging services
    - Application delivery
      - this is the actual application

Slide 3 – Segration of duties

- We want all the groups to only have to worry about their part of the image
- Images then need to be layered somehow
- Pull it all together to form a single, runnable image

Slide 4 – Application isolation

- Gotta keep 'em separated
- Isolation increases application uptime, reduces business disruptions
  - Other misbehaving apps can't wreak havoc on your application
  - “It works fine when it's not running on the farm”
- Separate servers are too expensive
  - In our environment a typical web server had 15% utilization
  - That's a lot of money in power and cooling costs
    - Not to mention software licenses, ongoing hardware maintenance, etc.
    - O&M budgets ballooning
- VMWare not really a good solution either

- Was never intended to isolate applications
  - It's an infrastructure play, not an application play
- Takes too long to provision
  - Same process and procedures for installing, configuring and security scanning a physical server
  - Takes 4-6 weeks to get allocated
  - And you're still charged an O&M tail
    - $2,000 per year at AEP
    - Storage charged separately
- VMWare is good for Line of Business isolation
  - Create virtual servers for each line of business
    - Guaranteed resources
      - memory
      - processor
    - Each line of business can manage their application pool
      - This is where Docker comes in

Slide 5 – Application migration

- Everybody is using abCI these days – we can repeatably build our software
- Deployment is a little trickier
  - Development
  - Test
  - QA
  - Production
  - DR
- The build artifacts are the same
  - But what about the configuration?
  - Ever have production bust because a property setting had the wrong value?
  - Ever have production bust because an environment variable either had the wrong value or wasn't set at all?
  - Things get even more complex when talking about database configurations, message services and so forth
- Turns out ensuring the artifacts are the same does NOT ensure your application will run successfully in production
- If you have to back out a deployment how easily could you redeploy the previous version?
  - How will you ensure it's environment was properly configured?
  - What if the previous version could only be run on an older version of middleware software?

- What a nightmare!
- I was trying to figure out how we could solve all these problems and I saw VMWare wasn't getting it. I couldn't create and manage images the way I wanted.

Slide 6 – Netflix architecture

- Attended Netflix architecture precompiler at CodeMash a couple of years ago
- Netflix doesn't build and deploy applications, they build and deploy images
- They use a proces they call baking

Slide 7 – Netflix baking

- Segregation of duties
  - Applications group
    - Fix problems
    - Implement features
    - Check in code when complete
    - Developers can work in relative isolation
  - Systems group
    - Patches underlying OS
    - Maintains application platform
- Nightly build process
  - Check code out of source control management system
  - Unit test
  - “Bake” with latest stuff from systems group
  - System test that result
    - Slow-rollout deploy
      - Deploy image (not application)
      - Watch for errors
        - They're also big users of ChaosMonkey
      - Backout with previous image if necessary
    - Fast-rollout deploy
      - If deployed to enough servers without issues, deploy to all servers
    - Everyday new stuff is going to production
- I was interested in the “baking”
  - Producing an image
  - Deploying an image
  - Rolling back an image
  - The image had everything you needed – self contained
- In-house built technology
  - He wasn't talking about how they built it

    - That was their “secret sauce”
- I wanted a process like Netflix had, but I didn't have the developer resources to create it
- But I also knew I couldn't be the only one wanting this technology
    - Revolutionizes application development, deployment and migration
    - Fits quite well with Agile development methodologies
        - Continual development and deployment

Slide 8 – Columbus Code Camp

- Had time to kill
- Attended some session on virtualization
- It was about Docker
- Demo didn't work very well – he had lots of problems
- Three takeaways
    - Containers are not virtual machines
        - much more lightweight
        - all containers running in the same kernel space
    - Containers can be started very quickly
        - about the same speed as launching an applications
    - Containers can be layered
        - this is the holy grail I'd been looking for!

Slide 9 – Docker all the things

- Everyone else seems to think Docker is the holy grail too
- Which is probably why you're all here – to find out what Docker is about
- Docker will impact our lives in IT as much as the internet
    - Applications will be delivered as Docker images
    - Ecosystem of container management tools are popping up everywhere

Slide 10 – Gartner hype cycle

- This is the gartner hype cycle
- Almost all technologies follow this cycle
- Where do you think Docker is?
- 'Peak of inflated expectations'
- Sometime in late 2015 or into 2016 we can expect Docker to hit the trough of disillusionment
- I'm trying to get you to plan for the plateu of productivity

Slide 11 – Only Tux does Docker

- Windows doesn't
- OS X doesn't
- BSD doesn't
- Microsoft has announced plans for Docker on Windows
  - No details
  - It might be a pure Windows play
  - It might allow Linux containers to run
  - Who knows?
- Containers are being added to BSD
  - Seems like you're asking for trouble
    - And this comes from a guy who likes BSD!
- Apple hasn't said anything about Docker
  - Docker is more of a developer and IT thing
  - Especially on servers
  - Neither developers nor servers are much in the forefront at Apple
    - My prediction: not a big play
- Oh, and this is why the precompiler prerequisites asked that you have VirtualBox installed on your system
  - You need to have Linux
  - There's an application called boot2docker you may have heard of
    - Really only good for running images
      - Not as good for creating images
    - Still requires VirtualBox to be installed on your system
      - There's no getting away from VirtualBox on Windows or Apple laptops