Deploying Hydra unconference notes

Notes from Deployment break-out session - Thursday October 2, 2014

What do people want to talk about/ think we are talking about? Puppet docker Capistrano Packer Maker vmware bash scripts Chef

This list displays a good breadth of ideas/interests/tools

Hydra did not have any installation documentation at all a year ago - we are still behind in documentation

It would be great to develop/document war stories about deployment

Most attendees manage their own infrastructure – maybe 40% have other teams to manage it for them

Most attendees came looking for solutions rather than offering them

Using vm’s mostly as an end run around differences in development environments – hardware/OS’s – for staff/developers

Can also use CI server to check code against a particular environment that matches production - one example is Travis triggered by check-ins on Github. Travis does require you to mock up http responses and other things - Travis can be slow – and it can mask production issues because it runs against hydra-jetty not production fedora/solr

Best practice: run full suite locally before pushing, smoke test on check-in, overnight full test suite that ran the longer stuff

Can also break up your test suite for optimization – pull out longer/slower stuff

Puppet is most useful in preparing for big changes (like Fedora4) – easy testing of the new environment. Also helpful for processes you don’t do very often, like building a whole new server once every six months. Puppet scripts preserve your present-day knowledge for reliable future use.

Why did people choose a particular solution (puppet, chef, ansible)? Mostly because sysops people chose it (or another institution started the bandwagon). Ansible made deployment “less spooky”. Allows you to revert to an older version if the latest deployment doesn’t work out.

Can use a single tool to bring up production servers and development vm’s. Chef uses .yml config files, puppet config was easier to handle.

Avalon installer is part of the plan to encourage outside adoption – uses Capistrano for web application, Puppet for everything else, also includes Vagrant so you can spin up a vm quite easily – this approach works, but it’s complicated and requires the right yml files in the right places.

Cleveland split their production systems: Web node with code/apache/passenger and ffmpeg, data node with tomcat/fedora/solr.

Aaron (institution?) has split production systems even further – solr and fedora separate. Not using puppet there. Chose this setup for performance improvements – with lots of ingests going on, fedora hogs memory on tomcat – if solr is running on the same container the web service slows down because solr has fewer resources available.

Ansible has the concept of roles so you can manage a distributed environment.

Performance monitoring is also very important. Have things in place so you know about problems before they get bad.

Resources – look at these repos for examples/ideas:

https://github.com/dpla/automation
https://github.com/jcftang/ansible-hydra
https://github.com/uclibs/scholar-puppet

Note-taker: Alicia Cozine, DCE