Windows Image Build and Management

Implementation Roadmap

Date: May 2018
Authors: Kendall Libby
Reviewers:

- Travis Blow [r?]
- Chris Cooper [r?]

Motivation

The two primary motivating factors are to have a well-understood and automated method to generate Windows base AMIs, and to replace the OpenCloudConfig configuration management tool.

The current Windows base AMI was built through a series of manual steps and hand configuration that is not well known beyond the original implementor. There is also the belief that a long running issue with graphics stutter on AWS GPU instances might be traced back to driver installation or configuration, that might be solved through reviewing what goes into the base AMI.

OpenCloudConfig (OCC) is a set of PowerShell scripts using Desired State Configuration (DSC), that was created out of the need for a lightweight configuration management tool on Windows. Originally we attempted to use Puppet, as we do for POSIX systems, but found that it did not work well on Windows at the time. Because we were already using DSC to install Puppet, expanding its use was the easiest path forward. Over time, however, it has become unwieldy. It runs on multiple versions of Windows, with varying versions of PowerShell; it must manage both AWS instances and data-center hardware; and there have been problems of varying degrees with how OCC interacts with different parts of TaskCluster.

As both of these pieces are fundamental to how we build and maintain Windows systems, replacing them with new systems using a CI process will ultimately allow us to test, verify, and deploy OS updates more quickly. Eventually, it should allow us to define Windows worker types in-tree.

Requirements

1. Automated process to generate and quickly deploy Windows images.
2. Minimizing changes made on-the-fly and during boot up.
3. An easy-to-use, lightweight process for system configuration management.
4. No running background processes while tasks are running.
5. Transparency and visibility in changes, to help sheriffs associate issues with deployments.
6. A well defined process for submitting, reviewing, testing, and verifying changes.
    a. Per-PR TaskCluster worker-types
7. Store and track SHA256 hashes of generated artifacts

Nice to haves

- Easy availability of debugging tools for developers to use on loaners.
- Use a more widely accessible languages and tools.

Proposal

Our design consists of four major pieces: creating Windows images in a CI environment, testing and verification of images in CI, fast hardware re-imaging, and replacing OCC with a new configuration management tool.

Windows image generation will use CI to run a PowerShell conversion process to convert ISOs to virtual hard drives (VHDs), which can then be provisioned by Packer and a configuration management tool to generate both AMIs and hardware images. Initially we propose to use OCC, as it's a known state and allows for a faster deployment of this stage, but intend to replace it.

Testing and verification will initially rely on using CI to re-run known good tasks, e.g. latest green mozilla-central build, to show that our generated images behave correctly. This will require setting up new TaskCluster worker types and defining a process for image promotion, as well as rollback.

Fast hardware re-imaging is part of our goal to make hardware systems less permanent ("cattle vs pets"), and allow us to quickly and easily replace the OS on a blade. Our goal is to use systems similar to what Packet and HPE do, using a Remote Imaging Server (RIS) and a storage service to quickly copy OS images to disk. We will use two Moonshot blades for the RIS and storage, and test installation to blades on the same chassis and adjacent chassis (vendors suggest having a RIS and storage blade per-chassis, but initially that seems to be an expensive proposition). We will also need to create our own PXEboot service, as we currently rely on IT's systems for this; this will allow us to more easily modify our configurations, as well as providing more separation between security domains. Lastly, we'll look at adding support in Roller for generating the PXEboot configs and initiating re-installs.

The final piece is to replace OCC with a new configuration management tool. We intend to evaluate masterless Puppet 5 as we believe their support for Windows has improved significantly. We will need to identify and remediate any gaps between what we currently do in OCC and what Puppet 5 can provide, and then convert the DSC configurations and OCC manifests to Puppet manifests. These changes can then be tested via the new image generation and testing environments.

Implementation

1. Windows image generation
    a. Conversion of ISO to virtual hard drive (VHD) in CI
        i. Slipstream changes limited to what packer requires to function
        ii. Generate and track SHA256 hashes of each
    b. Use packer in CI to provision an AMI ready for deployment
        i. Use OCC initially, but design for easy change to masterless puppet
        ii. Bake as much as possible into the image; on-boot changes should be as small as possible to facilitate fast start up
        iii. Configuration management should NOT run again after boot up

        iv. Generate and track SHA256 hashes of generated images
        v. Create gecko-image-builder worker type for building and distributing AMIs
2. Testing and verification
    a. Run tests in CI to verify viability of generated AMI
        i. Create new custom TaskCluster worker-types, per-PR
        ii. Determine what tests to run and how to run them (e.g. rerun latest green m-c build with a try push to PR-specific worker-type)
        iii. Add our own smoke tests
    b. Determine processes for promoting AMI to production status and for rollback
        i. Include management of resources that we create as part of image generation and testing (i.e. volumes, snapshots, AMIs for per-PR worker-types)
        ii. Promotion should not require rebuilding an approved image
3. Fast hardware re-imaging
    a. Establish target time-to-install
    b. Select tool for system imaging
        i. [Need selection criteria]
        ii. Determine how to automate imaging
        iii. Investigate using 2ndary NICs for install network
    c. Establish PXEboot service
    d. Testing phase
        i. Compare install times on local vs remote blade (i.e. across chassis) vs target time
        ii. Determine resources (i.e. blades) required for imaging service
    e. Add API/methods to Roller
        i. Generate configurations on PXEboot server
        ii. Initiate PXEboot, reboot system, verify running state
        iii. Track which OS systems are running
4. Evaluate and implement new configuration management tool
    a. Compare Puppet 5 capabilities to what we currently do in OCC and identify any gaps
    b. Create Puppet configuration and test
        i. Determine configuration layout (i.e. learn from difficulties with PuppetAgain)
        ii. Convert DSC configurations and OCC manifests to Puppet manifests
        iii. Replace OCC with Puppet in image generation
        iv. Compare generated AMIs and CI test results