InfoWorld

JULY 30, 2014

GET TECHNOLOGY RIGHT®

Review: OpenShift shines for developers and ops

From the pain-free install and easy app deployment to gear idling and automatic scaling, OpenShift fulfills the promise of platform as a service

THE TWO MAJOR OPEN SOURCE PAAS (PLATFORM as a service) offerings are Red Hat's OpenShift and Pivotal's Cloud Foundry. Like Cloud Foundry, OpenShift was designed to provide rapid self-service deployment of common languages, databases, Web frameworks, and applications. One of OpenShift's current differentiators is that continuous integration (using Jenkins) is a standard part of its workflow.

OpenShift is useful for both developers and operators, as well as for streamlining the process from development through testing and production deployment. Like Cloud Foundry, it can run in a public or private cloud, or on-premise.

OpenShift comes in three versions: Online (no version number, but monthly updates), Enterprise (version 2.1), and Origin (version 3.0), all produced from the same upstream code. The Origin version is the bleeding edge, community supported, free open source edition. It receives daily updates and runs on your hardware using Fedora as the underlying operating system. You can download a preconfigured virtual machine, configure it yourself with Puppet or Vagrant, or even build it yourself from source code. Origin is not really intended for production environments, but can provide a good, fast, free development environment that runs on a laptop or desktop.

The OpenShift team takes a three-week-sprint "cut" from the Origin version for the Online version on a monthly basis. OpenShift Online runs on Red Hat Enterprise Linux (RHEL) hosted on Amazon and has seen roughly 1.8 million total apps deployed. Any bugs that surface under this heavy use are fixed in the Origin version. The Online version is appealing to developers and testers and to provide anyone interested in trying it out a quick, installation-free PaaS.

Two to three times a year, a stable cut from the Online version is released to the Enterprise channel. If you look at the cartridge versions ("cartridge" is the OpenShift term for a deployment package) you'll notice that the language, Web framework, database, and application versions are newest in Origin, older and more stable in Online, and most stable in Enterprise. The Enterprise version most often is deployed on-premise, but it can also be deployed onto any IaaS cloud that can run RHEL.

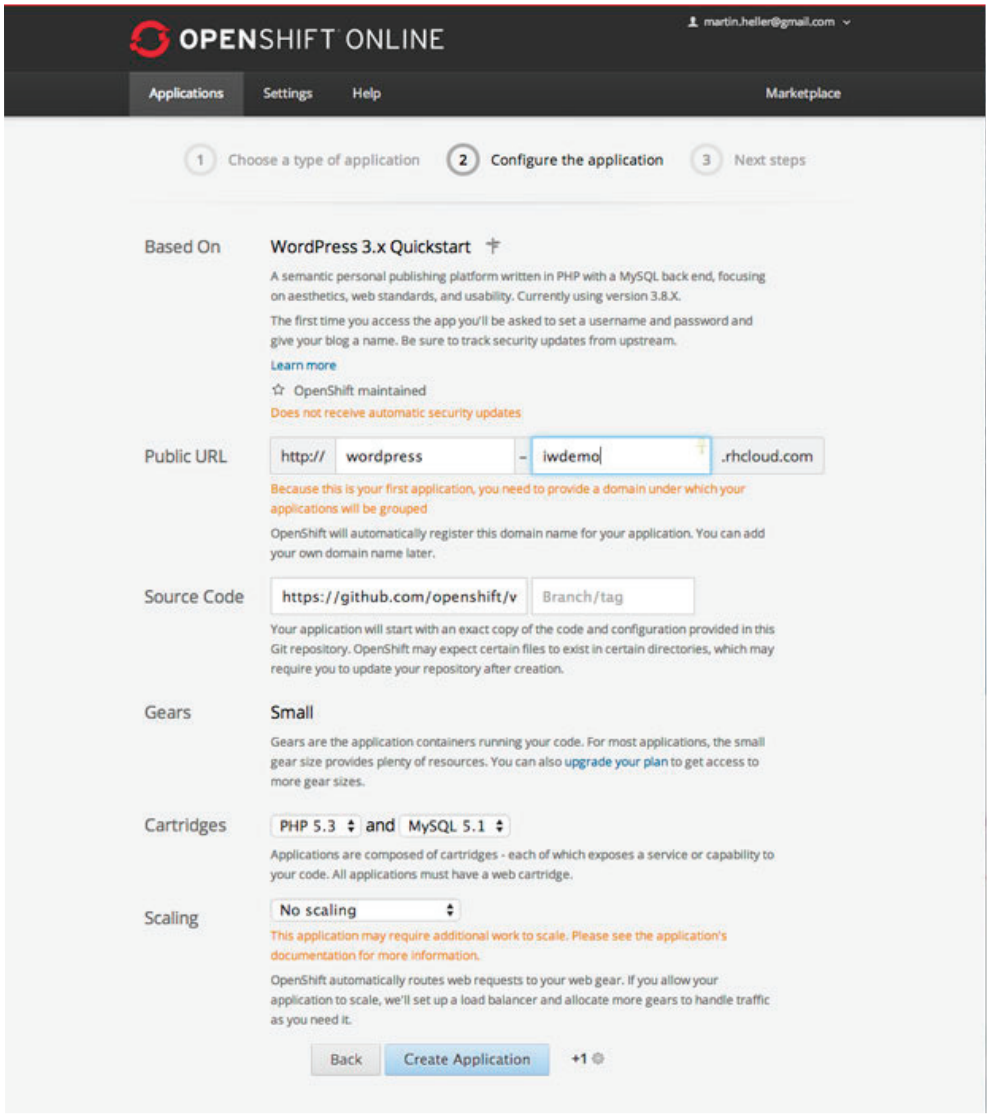
Test Center Scorecard

	Ease of Use	Management	Breadth of support	Documentation	Intstallation and setup	Value	Overall Score
	20%	20%	20%	15%	15%	10%	
Red Hat OpenShift Enterprise 2.1	9	9	8	8	9	9	8.7 Very Good

OpenShift architecture and features

OpenShift uses two kinds of servers, referred to as "brokers" and "nodes." Brokers are responsible for managing user logins, DNS, application state, and general orchestration of the applications. Users communicate with brokers using the Web console, CLI tools, or the JBoss Tools/Eclipse IDE.

Brokers, which are Rails applications, communicate with nodes over a message queue.



OpenShift QuickStarts combine application code with one or more cartridges, making it easy to install whole applications. This QuickStart uses PHP 5.3 and MySQL 5.1 cartridges to deploy a WordPress instance in OpenShift Online.

Nodes host the cartridges that will be made available to users and the "gears" where user applications will actually be stored and served. A gear represents the slice of the node's CPU, RAM, and base storage. Gears combine the partitioning capabilities of control groups with the security features of SELinux.

Cartridges represent pluggable components that can be combined within a single application. These include programming languages, database engines, and various management tools. The built-in cartridges are different for the three versions of OpenShift, but the lists are all extensive, albeit limited to items that run on RHEL. (There is a version of OpenShift that runs on Windows Server, but we have not reviewed it.)

If your company has LDAP, OpenShift can use it for creating teams. If not, you can still create teams, using OpenShift's own identities.

OpenShift supports the cloning of applications and uses multiple subdomains. When you combine these, you can set up OpenShift to easily do promotion of builds from development to QA to staging to production. You would give the owner of each level read access to the next lower level; for example, QA could pull a clone to its domain when the developers say they have a build ready for testing.

OpenShift applications can be versioned. You can define how many versions are retained in your cloud at any time in the application lifecycle. If you discover a bug after you deploy a new version, you can revert to a stored version.

The OpenShift Watchman feature automatically stops and restarts misbehaving applications, which helps avoid downtime. Automatic scaling adds gears and even nodes when an application becomes heavily used. It's built into OpenShift and doesn't require a front-end scaling service. Both features reduce the amount of monitoring and operations work you have to do to run an application on OpenShift.

Along the same lines, OpenShift automatically detects applications that are not getting any HTTP traffic, and eventually idles the applications' gears without any intervention needed by the developer or operations. At such time as the application is requested again, the environment will automatically load it back into memory and handle the HTTP requests. Gear idling allows OpenShift to support a very high density of applications.

OpenShift uses SELinux for gear isolation. This keeps gears from being able to intrude on one another's memory space. In the future, OpenShift will be changing its container model and cartridge specification to use Docker. A Red Hat proposal for Docker container integration with OpenShift cartridges is posted on GitHub.

When you absolutely have to work offline — say, on a plane trip without Wi-Fi — you can use OpenShift Enterprise's offline mode to take an image of the PaaS with you. When you reconnect, you can sync your offline changes to the live PaaS.

OpenShift installation and use

I did full installations of the Red Hat Cloud command-line, rhc, and an OpenShift Origin virtual machine on a Mac, then installed the WordPress QuickStart in both OpenShift Online and OpenShift Origin. With the small exception of incompatibility with one of my virtual machine managers (namely, Parallels Desktop), the whole installation process was quick and easy.

The rhc tool comes as a Ruby gem. Once you install it, it tells you to run rhc setup. You don't have to do this right away. If it's not set up, rhc defaults to connecting to OpenShift Online, and lets you enter your credentials manually.

Downloading the 2GB Zip file for the OpenShift Origin VM took all of 10 minutes. I tried to use Parallels on my iMac to run the virtual machine even though the OpenShift docu-

mentation didn't mention it. Alas, Parallels could not convert the VM to its own format, although it was able to mount the disk image for Mac OS X.

I could have used VirtualBox, which is free from Oracle and supported by Red Hat for this purpose, but I didn't feel like putting another virtual machine manager on my iMac at the time. (I later installed VirtualBox to run Vagrant, a tool to automate the management and provisioning of development environments.) I also could have used dd or hdiutil or Apple's Disk Utility to make an ISO from the mounted image. Instead, I copied the Zip archive across my LAN to my MacBook, where VMware Fusion was able to load and run the OpenShift Origin VM with no problem.

I was pleasantly surprised that the OpenShift VM pretty much configured itself, right down to using Bonjour for local DNS. OpenShift told me its admin URL, told me the full command line to use for configuring rhc setup correctly, generated SSH key pairs on my MacBook, uploaded the public key to the server VM, and asked me for a domain namespace name so that it could publish itself to the LAN. Finally, it offered to create and save an access token to allow me to avoid constant logins and told me the approximate token lifetime.

Here WordPress is installed in OpenShift Online. Note the options to add Jenkins and phpMyAdmin. In case you want to modify the application code in the future, OpenShift also offers to display the Git URL to the application source.

Deploying apps with cartridges and QuickStarts

All versions of OpenShift offer menus of cartridges — Web frameworks, databases, and JBoss services — and menus of QuickStarts. As I noted earlier, OpenShift Origin has the newest versions, OpenShift Online has older and more stable versions with bug fixes, and OpenShift Enterprise has the most stable versions. All the various cartridge types are plentiful and easy to install, as are the QuickStarts.

To get a feeling for what an application QuickStart installation involves, I installed the WordPress QuickStart both in OpenShift Online and in the Origin VM locally. I expected the local install to take longer because of download time, but I was wrong: The online install took two minutes, while the local install took 25 seconds. The online installation makes the application URL public in DNS, which is part of the reason for the additional time. The online installation can also be aliased to your own public DNS name, and a form lets you configure this easily.

The online installation asked if I'd be modifying the WordPress code and didn't supply the Git URL when I said no. The Origin installation didn't ask — it simply gave me the URL to git clone the code. I guess there's no harm in posting a URL to a private LAN endpoint.

QuickStarts combine code and one or more cartridges. They make it easy to install whole applications. While the OpenShift team doesn't maintain QuickStarts, anyone who is willing to be responsible for keeping one up-to-date with security issues is free to create and post one. Not surprisingly, WordPress, Drupal, and Ghost are among the top QuickStarts.

Building a QuickStart is simple. You start by creating a base OpenShift application, installing any necessary cartridges, and replacing the default OpenShift example pages with your QuickStart code. You'll want to save all of your code as a single Git commit.

Then you'll want to remove any unnecessary files — such as the application's log files and samples — from the repository. Then configure the .openshift/action_hooks (build and configuration scripts) to enable your application to run on OpenShift and to use OpenShift environment variables.

Review your QuickStart code and modify it as necessary to replace any static security values with values that are established on a per-instance basis, and use security libraries when available. Test the QuickStart extensively, and finally submit it for adoption.

Updating and scaling your application

When you want to update your application, you'll typically do a git push. When OpenShift sees the new code, it will rebuild your application if necessary, then restart it.

Note that if you want automatic application scaling, you just check a box when creating the application. You can then configure the traffic trigger points for adding and dropping gears. Using the HAProxy software load balancer, OpenShift will horizontally scale the application with increasing load. As OpenShift senses increased traffic, it creates additional capacity at the middle tier of the application by spinning up more gears. OpenShift scales applications across nodes for reliability.

I have to say that OpenShift can be a huge timesaver for developers compared to managing real servers or even compared to managing an IaaS cloud such as Amazon. I'm aware that my fellow InfoWorld contributor Andrew Oliver had issues with deploying a

Spring Java app to OpenShift in 2012. When Oliver looked at OpenShift, however, the PaaS was a developer preview; OpenShift Enterprise is now at version 2.1, and OpenShift Origin is at version 3.0. Among many other improvements, OpenShift now has a QuickStart for running the Spring Framework on JBoss EAP6.

OpenShift is outstandingly easy to use, manage, and install, and it presents little learning curve for developers familiar with Git and administrators familiar with Puppet. Enabling automatic horizontal scaling is as simple as checking a box in the application configuration. Automatic gear idling is enabled by default and allows for very high application density. And updating an application is as simple as doing a git push. For both developers and operators, OpenShift fulfills the promise of PaaS.

— Martin Heller

OpenShift at a glance

Pros	• Wide assortment of languages, Web frameworks, databases, and application stacks available and supported • Easy and fast self-service deployment • Automatic application scaling • Git integration at the source code level, with automatic deployment triggered by a git push • Gear idling allows OpenShift to support a very high density of applications • Runs on any hardware or cloud or virtual machine that supports Red Hat Enterprise Linux
Cons	• Largely limited to applications and services that run on Red Hat Linux, unless you use the Uhuru OpenShift.Net product, which we have not reviewed
Platforms	• OpenShift Enterprise: Red Hat Enterprise Linux. OpenShift Origin: KVM, VirtualBox, VMware Fusion/Player
Cost	• OpenShift Origin: free open source. OpenShift Online: first three gears and first gigabyte storage free, then a per-hour, per-gear charge for usage (ranging from 2 to 10 cents), plus a $1-per-gigabyte, per-month charge for additional storage; to use more than 16 gears online, there's an additional $20 per month service charge. OpenShift Enterprise: pricing starts at $4,000 per year depending on configuration (core/VM or socket pair).