



Software Appliances, Virtualization, and the Zimbra™ Collaboration Suite

Abstract

Appliances are typically created by specializing and repackaging general purpose technologies to make a particular system easier and cheaper than the “build your own” alternative. When we think “appliance”, we tend to think of a hardware gadget, the vast majority of which also include software. Herein we argue that the on-going commoditization of computation and storage has progressed to the point that more server software is going to be packaged as *software* appliances—software that is ready to simply “drop” onto the “raw iron” of an unconfigured server machine or the abstracted computing platform of today’s virtualized blades and network-attached storage. Indeed, software appliances will prove crucial to realizing the benefits of virtualized computing. In addition to making the general case for software appliances, we will discuss the initial release of the Zimbra software appliance based on [rPath Linux](#) (think Linux distribution for software appliances) and supporting the [VMware virtualization platform](#). We will also point out some limitations of software appliances.

Introduction

From its first inception, the Zimbra¹ Collaboration Suite (ZCS) was envisioned to be *like a software appliance* in that we prepackaged leading open source infrastructure components—a relational database (MySQL), a web/Java application server (Apache Tomcat and now Jetty), a directory (OpenLDAP), a Mail Transfer Agent/MTA (Postfix), a Java Virtual Machine (the Sun JVM), and so on—within ZCS in such a manner that routine configuration and management (that is, modulo advanced troubleshooting) would not require a Zimbra administrator to interface with these underlying technologies. So by *embedding* these components (as well as numerous other open source libraries) so that they were hidden by the Zimbra installation, configuration and management tools, Zimbra administrators were spared the cost of downloading, installing, configuring, monitoring, managing, securing, patching, and upgrading versions of all these open source subcomponents (a complete list can be found [here](#)). And moreover, the Zimbra team was spared the myriad of customer support, quality assurance, and packaging complexities associated with testing and supporting each new ZCS software release across multiple versions of databases, application servers, directories, virtual machines, and so on.

Of course, Zimbra's source code developers remain free to make their own open source ZCS distributions leveraging whatever underlying technologies they so choose. But most Zimbra administrators would rather get a binary release (either for the Open Source or Network Edition) that embeds the essential infrastructure components so that they generally do not have to procure, install, configure, and manage those technologies, but still derive all of the benefits of Zimbra's modular architecture and the hardening investment that has gone into each of those open source subcomponents.²

Software appliances

By *software appliance*, then, we mean an alternative packaging for a collection of software (typically, an application plus essential systems subcomponents) that has been integrated and tuned for a particular need (typically that of the application) in order to lower complexity and total cost of ownership. Wikipedia says:

A **software appliance** combines a software application and a streamlined version of system software (operating system, file system, application server, etc.) that readily installs on industry standard hardware (typically a server). The customer receives all service and maintenance from the application vendor,

¹ Zimbra is a trademark of Zimbra, Inc. All other trademarks belong to their respective owners.

² Zimbra certainly would not exist in anything close to its current form without the hard work of so many talented open source communities; Zimbra, as Newton would have put it, has the luxury of “standing on the shoulders of giants.”

eliminating the requirement to manage multiple maintenance streams, licenses, and service contracts.

In a software appliance, subcomponents are necessarily *embedded* rather than *exposed*, so that during normal operations (again, outside of advanced troubleshooting) the end-user and administrator can remain blissfully unaware that these subcomponents are seamlessly cooperating on his or her behalf. The goal, then, in building a software appliance is to ensure that:

- Installation and configuration of the included software components should be correct the first time - There should be no need to work through esoteric configuration issues with vendor product support;
- Operations, administration, and management are accomplished within a single unified set of tools and utilities - One of the big appeals of Linux-based software appliances is that not only does a Windows administrator not need to learn Linux, but he or she need not worry about administering the server operating system (OS) whatsoever. Any “covers” that are required, such as IP address configuration, are simply surfaced in the appliance’s administrative environment;
- There is “one throat to choke” when problems arise - It is the software appliance vendor’s responsibility to isolate and resolve problems that may occur within the application or any underlying systems subcomponents included in the software appliance; and
- There is a much reduced burden of change management - The software appliance vendor is responsible for aggregating and integrating defect and security patches for all of the included subcomponents, often delivering them via an administrator-controlled automated update utility. You can even make the argument that software appliances are the “on premises” analog to Software as a Service (SaaS) in that much of the complexity of managing the collection of individually changing software components is assumed by the software appliance provider.

Zimbra software appliance

In the introduction above, we characterized the original Zimbra Collaboration Suite release to be *like a* software appliance, but presumably at that point not yet a full software appliance. The obvious gap, and one we think essential to gaining entry into the category of software appliances, is embedding the requisite operating system components within the distribution along with the other essential systems software.

The idea started quite simply enough: Numerous small businesses (SMBs) we talked to were interested in deploying Zimbra, but lacked the necessary Unix/Linux skills because they were “Windows shops.” What these SMBs thought they wanted was a Windows port of the Zimbra server. While Zimbra already provided the necessary integration with their existing Microsoft

infrastructure (including support for Active Directory, Outlook, IE, coexistence with Exchange, mailbox migration automation, and so on), these SMBs were not comfortable considering Zimbra if they had to first teach their administrators Unix/Linux (even though this is getting easier by the day with ever easier Linux distributions).

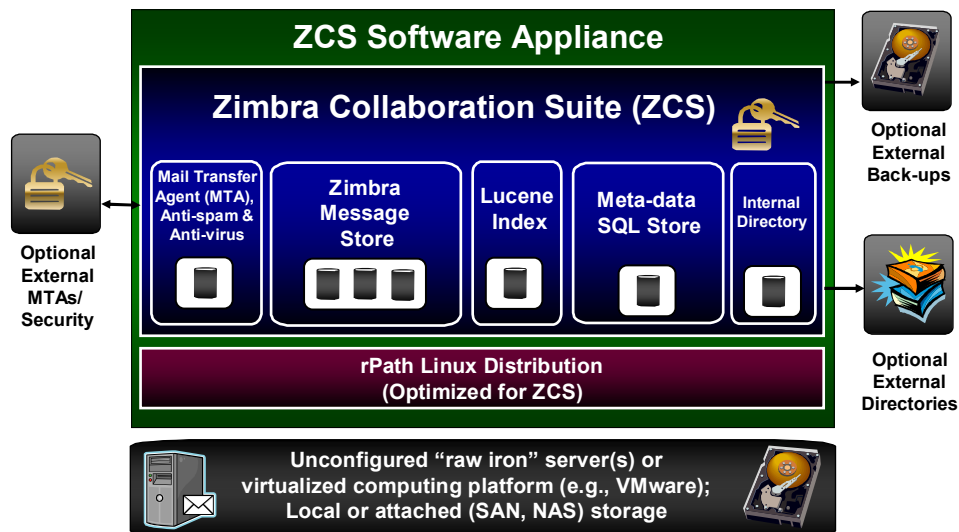


Figure 1: Software Appliance for the Zimbra Collaboration Suite

However, when offered the choice of a Windows port of ZCS or a software appliance in the form of a bootable disk that only requires the user to install, configure, and administer the ZCS application, such SMBs have so far overwhelmingly voted for a software appliance over a Windows port. After all, who wants another Windows server (or any OS for that matter) to manage and secure? (And even if you do still want to run ZCS on Windows, why not leverage virtualization technology to make that easier to manage?—See below.)

Other obvious benefits include the delivery of a “live” ISO image: If you want to simply play with a server like ZCS, why should you have to dedicate a machine to the task and then have to go through an uninstall when you are finished (especially if that machine is your own laptop)? Instead, you can simply mount a live ISO image or, equivalently, use a virtualization solution like the free VMware Player to experiment without installation.

Software appliances and virtualization

Like many, we chose the software appliance model as a means to provide lower total cost of ownership (TCO) on generally stand-alone server(s) that are typically associated with smaller-scale deployments (install via boot onto “raw iron”). What was less obvious to us initially was the broader fit of the software appliance model for virtualization architectures. The key insight is that when the Host and Guest OS are decoupled by virtualization, you realize that the Guest OS can be pared down to the minimal functionality required for the application and systems technologies it is running. After all, the responsibility for “multiplexing” the underlying hardware falls on the Host OS/virtualization platform itself. All the Guest OS is responsible for are the needs of that particular application.

Taken together, then, the application, its other infrastructure components, and the Guest OS *is* a software appliance, albeit one targeted at virtualized deployment. By excluding unnecessary services and device drivers, the OS embedded within the appliance is easier to install, secure (it has a smaller “surface area” that could be attacked), manage, and keep up to date. For example, the ZCS software appliance VMware image based on rPath Linux is 320Mb and the ISO image is 475Mb. A comparable image that includes a complete Linux distribution would be well over 2Gb.³ As the CEO, Billy Marshall, of rPath puts it, “There are less things to break and less things to break into.” Moreover, with a software appliance, the application is tested and packaged with the appropriate release of the embedded/Guest OS, which means that with virtualization technology, multiple software appliances with their own underlying OS versions/patch levels can seamlessly coexist and interoperate on the same servers and networks.

Where can I find the ZCS software appliance?

Please check the appropriate downloads page for the latest software appliance builds for the Zimbra Collaboration Suite [Open Source Edition](#) and the [Network Edition](#). Different binary images have been posted for different deployment targets—for example, a “bootable” ISO image for installation, and a VMware image for VMware Players or Servers (no install required). Please also keep in mind that an image designed for experimentation (on a laptop) rather than installation (on a server) will be preconfigured with particular assumptions (e.g. disk volume sizes) that are unlikely to be appropriate for eventual production deployment (of course, you can always change those settings later).

³ No doubt image size is not an ideal measure of complexity, but at the same time, why spend disk, memory, and cycles to support unused code. “Make everything as simple as possible, but not simpler” - Albert Einstein.

How does using the ZCS software appliance differ from the non-appliance distributions of Zimbra?

There is, of course, no change in the experience for ZCS endusers (sending mail, updating appointments, etc.), and ZCS administrators continue to use the same Zimbra administrative console for managing their Zimbra deployments. The difference is primarily in the installation and initial configuration of the ZCS appliance by the Zimbra administrator. The most notable change is the switch from the command line install process you may be familiar with to a web-based interactive form for the ZCS appliance. This web-based interface adds configuration options for the rPath Linux distribution embedded within the ZCS appliance, including time zone, root password and network settings. In order to make deploying the appliance as simple as possible, we reduced the number of ZCS configuration questions to the bare minimum (admin password and domain name) and pre-populated the rest of the configuration settings with defaults. Administrators can then customize the install (change these defaults) after the appliance is up and running via the Zimbra administration console.

Why did Zimbra choose rPath for its initial software appliance distribution?

The work necessary to support a software appliance model for the other ZCS dependencies (MySQL, Apache, JVM, OpenLDAP, ...) was straightforward; these open source technologies were generally designed to be easily embedded within other applications. It would have been a great deal more work to make an optimal Linux distribution for ZCS, and cover/hide the necessary installation, configuration, and management thereof. We also would have had to build the necessary (and nontrivial) infrastructure for automating updates.

So the Zimbra team ultimately made a “buy versus build” decision to choose rPath. rPath provides Zimbra with a richer software appliance platform than we could have afforded to build ourselves. Choosing rPath allowed Zimbra to continue to focus on our core competency—striving to remake the messaging and collaboration experience through innovation. rPath provides substantial versatility—allowing us to easily manufacture software appliances tuned to different platforms: “bootable ISO” (boot to install on “raw” iron), “live ISO” (mount and run without installation), VMware, and other virtualization platforms. And rPath like ZCS is open source, thereby protecting Zimbra’s and Zimbra’s users investment. (More on our experiences with rPath technology below.)

What are the differences between software appliances and Software as a Service (SaaS)?

Some consider the software appliance model to be included within the SaaS category, because appliances like SaaS reduce the complexity and TCO of software deployments. Moreover, automatic update, offsite data backup &

restore, and other services that may be available for software appliances typically use SaaS.

We, however, would argue that software appliances simply represent yet another packaging model for software. Going forward, an enduser can choose software delivered via traditional packaging, a software appliance, or SaaS. Software appliances can be run “on-premises” in the enduser’s own data center or “on-demand” by outsourcing the operations to a SaaS provider, who has independently chosen a software appliance to lower *their* TCO (and hopefully pass some of the cost savings along to the enduser).

In weighing this choice, traditional software deployments tend to be the most flexible, but also the most costly in terms of management overhead. SaaS is often the least flexible, but then also the least costly in terms of IT operations.

Software appliances generally provide a “middle ground” in terms of both flexibility and administrative overhead: Like traditional software, an on-premises software appliance must be managed by your own operations staff, and requires the provisioning of computing and storage (whether or not that is virtualized). On-premises deployments (whether traditional or via software appliances) provide potential advantages over SaaS in (1) performance – no WAN crossings; (2) availability – no loss of service if the WAN is down; (3) customization; and (4) integration with other internal systems (such as employee directories, Voice over IP telephony systems, enterprise applications, and so on). At the same time, the administrative overhead for a software appliance will be less than that of traditional software, particular with respect to managing change.

Will software appliances replace more traditional distributions of Zimbra targeted at a particular OS release?

The large majority of Zimbra downloads today (both for the Open Source and Network Editions) are for installation on an existing Unix release, including leading Linux distributions (Red Hat Enterprise Linux/RHEL, Fedora, SuSE, Ubuntu, Debian, Mandriva, etc.), Mac OS, and so on. These traditional builds are essential when ZCS is to share a hardware footprint with other applications or systems without virtualization technology. Moreover, many IT departments elect to do their own quality assurance and acceptance of particular OS releases that they then mandate within their data centers. By offering a software appliance, Zimbra is simply seeking to deliver greater enduser choice. We remain thrilled by any selection of Zimbra, whether that selection is a distribution for a particular OS, our software appliance, [SaaS via a Zimbra service provider](#), or the ZCS source code for a do-it-yourself alternative. Our excitement about the software appliance model stems from its ability to increase the reach of Zimbra technology to users that might not otherwise have deemed ZCS the right choice.

Will Zimbra provide ZCS hardware appliances?

Zimbra is not itself planning to get into the business of selling and supporting hardware for running the Zimbra Collaboration Suite. Our intent instead is to make it as easy as possible for customers to procure a server or servers with the appropriate specifications to meet their ZCS deployment needs. We are also gratified that many of Zimbra's [Value-Added Resellers \(VARs\) partners](#) are providing pre-configured ZCS hardware appliances as an option for their end-user customers (who in turn, then, derive all the benefits of a software appliance without the need to provision servers and storage).

Any concerns with security and the software appliance model?

The manufacturer of the software appliance takes on the added burden of ensuring that the distribution has been hardened for the appropriate level of network deployment. In the case of ZCS, this most often means having ports that are open for secure access over the Internet (generally only over Transport Layer Security/TLS for full privacy) or over virtual private networks (VPNs). Moreover, the reduction in footprint of the underlying OS afforded by the software appliance model can itself reduce vulnerability in that there is reduced “surface area” that can be attacked.

However, for deployments utilizing a particular certified release of an operating system such as RHEL or SuSE Linux (some data centers seek to run only a handful of approved OS versions at any given time), we recommend using the traditional ZCS packaging. That same general-purpose OS configuration can also be deployed with virtualization—RHEL on VMware Server, for example.

What were the technical issues in delivering ZCS as a software appliance using rPath?

The initial port of ZCS to the rPath platform was very much like any other Linux port: compile, package, fix the installer, test and ship. Features like rPath's mirroring servers for automatic update, entitlement servers, and the rBuilder platform with its automatic checking and inclusion of dependencies were key for delivering the full benefits of a software appliance. But for us, implementing these additional features drove the actual port from the normal two weeks to closer to six months. (Most of the unanticipated work was actually in cleaning up internal ZCS build dependencies—see below. Given the maturation of the rPath technologies and that clean up work, we could probably accomplish the port in a month or two were we starting again today.)

Our basic recipe was to create a local mirror of the rPath Linux distribution within a local rBuilder repository. Then a custom top-level rPath group was created to define the components needed by the ZCS software appliance, and a

separate server was setup to perform the actual builds. While porting ZCS to the Conary-based change sets, the automatic dependency checking was not only invaluable in reducing the size/complexity of the eventual distribution, but it also helped find missing dependencies in our other open source components!

After the initial port was completed, we set up an external server with the rPath rMirror software to provide automated updates. Other enhancements to complete the software appliance included custom entitlement scripts and services for the ZCS Network Edition, rPath Appliance Agent (rAA) customizations for the web-based install, and rebranding of the ISO installation and rAA interface. Custom web services were written to provide integration and license verification between the rPath entitlement service on the update server and the ZCS internal licensing server (for the ZCS Network Edition, but not the Open Source Edition).

Acknowledgements. The Zimbra team is deeply indebted to Eric Hahn, one of our board members and company mentors, who motivated our work in software appliances (and did so more than three years ago). We would also like to thank Marty Wesley and Rebecca Hirschfield of rPath who provided comments on a draft of this manuscript.