InformationWeek

RogueWave SOFTWARE

Open Source Support

Open source software: The infrastructure impact

> Tackle open source (either commercially supported or self-supported) as inevitable investments that by being properly managed, will yield considerable total cost of ownership (TCO) and 'business value' benefits. When unmanaged (or undermanaged), these same OSS technologies will instead introduce considerable technical, security and legal risks to the enterprise.
>
> – GARTNER, *WHAT EVERY CIO MUST KNOW ABOUT OPEN-SOURCE SOFTWARE,* MARCH 2017

Open source software (OSS) runs most enterprises today. From Linux operating systems to web servers, application servers, databases, and custom business applications, OSS is everywhere. With OSS-dominant IT systems driving the operational and analytical hearts of businesses, it's essential to keep them running 24/7/365 at maximum performance levels, with up-to-date security and full compliance with all OSS licenses.

While the benefits of using OSS are well known, the implications of its extensive growth on managing and maintaining the infrastructure to achieve these objectives aren't. Keep reading to learn how to address the security, configuration, and support challenges that accompany the deployment of OSS.

PART I: THE IMPACT OF OPEN SOURCE SOFTWARE ON TODAY'S ENTERPRISES

PART II: HOW TO ACHIEVE OPEN SOURCE SUCCESS

PART III: WORKSHEET

Part I:

The impact of open source software on today's enterprises

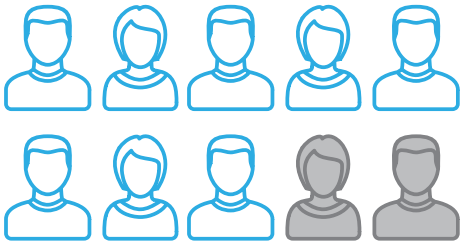
Open source is easy to
get but hard to manage

Enterprises run on open source

The days of single-vendor enterprise environments are over. It's true that there's plenty of Microsoft, Oracle, and other commercial software playing critical roles in large organizations worldwide, but over the past couple of decades, open source software (OSS) has been "eating the enterprise." While the open source landscape is still maturing, there's no question that OSS products have "won," as OSS has worked its way into the mission-critical IT workloads of the majority of IT organizations around the globe.

OSS has found its place as a full-fledged, enterprise infrastructure and production staple. For example, what began as a movement among a loosely-organized group of programmers and developers, has led to the emergence of Linux as a proven, sustainable alternative to established operating systems such as HP-UX, AIX, and Solaris. No longer the alternative, OSS is the default choice across enterprise stacks.

It's easy to get, harder to manage



80%

OF DEVELOPERS REPORT THEY HAVE DEPLOYED OSS IN THEIR APPS OVER THE PAST YEAR.[1]

30%

GROWTH OF OSS PRESENCE IN IT PORTFOLIOS THROUGH 2020.[2] (CAGR)

Explore any large IT enterprise and you'll find OSS operating systems, application servers, web servers, databases, proxy servers, NoSQL data stores, application frameworks, logging utilities, data analytics and big data applications, search engines, software development tools, code repositories, and applications, from security to office productivity tools, to graphics tools, email systems, containers, messaging platforms, and much more.

The burgeoning presence of OSS within your production applications is probably more extensive than you realize. You likely know about the most common packages in use such as Red Hat, CentOS, Apache, and Tomcat. But there are undoubtedly hundreds of packages being integrated and used, which you're likely unaware of, as developers adopt the same solutions that their peers are using or take advantage of "free code" to meet aggressive deadlines. It's also not unusual for developers to use dozens of OSS packages in a single application.

[1] *CIO Magazine*, "How long to build a custom app?," Feb 2016

[2] SiliconAngle, "Oracle CEO 2025 industry predictions," Oct 2015

Commonly known	Common, but lesser known
    	    

You may know about the big names but did you know that the lesser-known packages are equally as popular? **Do they exist in your environment?**

Here are just some of the ways that OSS finds its way into mission-critical applications:

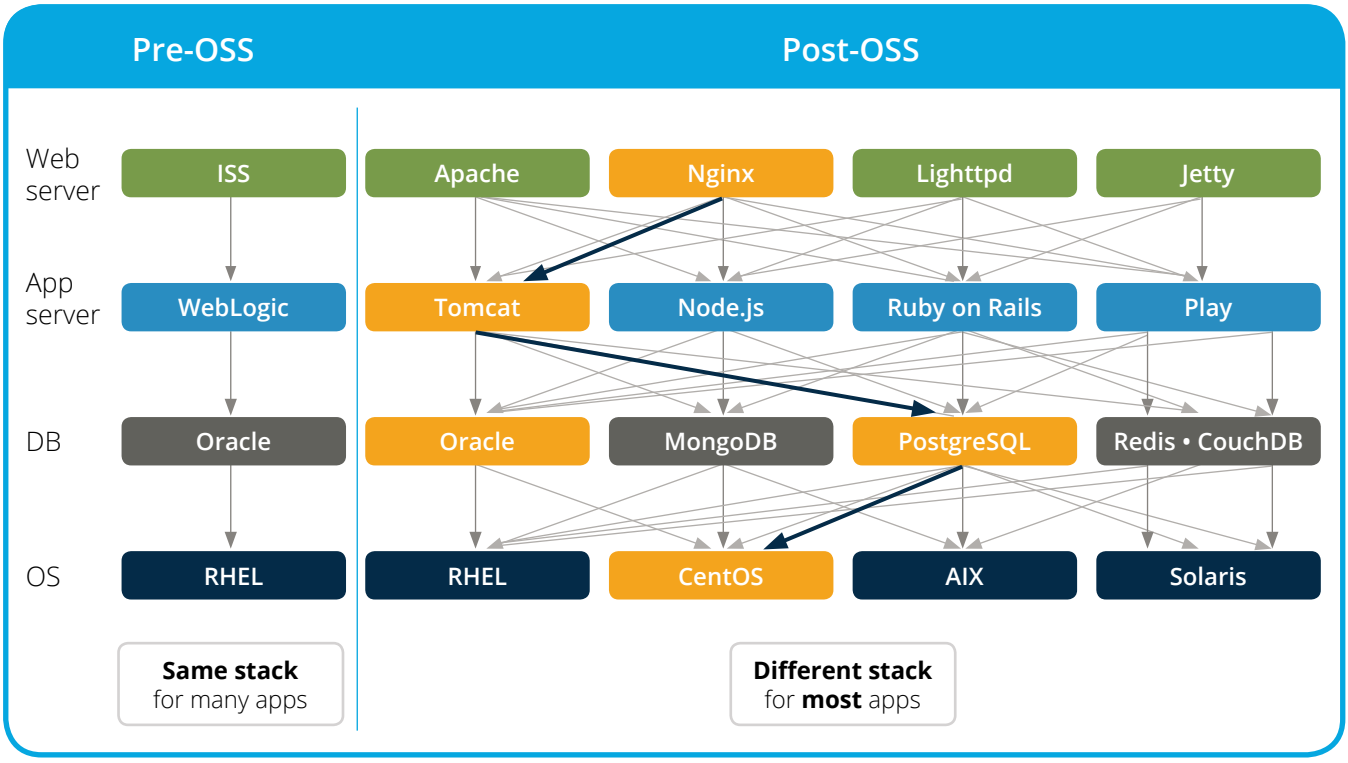
- It may be software you've written yourself using open source components
- It may be included within commercial software that you've licensed and purchased
- It may be software that was written by someone in your organization at some point, but was never properly documented or tracked
- It may be software that enters the enterprise as a hidden dependency of known OSS packages
- It may be software you inherited during a merger or acquisition of which you're not aware, or for which there is no documented history of acquisition, licenses, updates, dependency issues, etc.

According to Alan Zeichick, president and principal architect of Camden Associates, 47 percent of organizations report that they have no formal processes in place to track the use of OSS within their organizations. This means that nearly half of organizations are at least partially blind when it comes to understanding and being able to properly manage their IT enterprises.

These ongoing practices allow software packages that your organization did not build, does not own, and isn't necessarily aware of to "sneak" into production applications and systems. Maintaining them once deployed, though, is where the costs of OSS can become a pressing challenge.

The realities of OSS in production

The growing depth and breadth of OSS across the enterprise leads to complexity, which can quickly become a big problem for IT professionals.



OSS has increased the complexity of enterprise infrastructures.

As you can see in this diagram, in the pre-OSS days, enterprise infrastructures were straightforward. IT professionals were responsible for the management and maintenance of a fairly standard infrastructure stack. Now, with the presence of so much OSS in the typical enterprise, complexity takes on another dimension and presents significant management challenges.

It's created a situation in which a limitless permutation of stacks is possible. And, of course, IT is responsible for ensuring the high performance, security, and license compliance of these ever-changing stack combinations. Back in 2016, the popular stack of the day included jQuery, Bootstrap, and CoffeeScript. Now it's Angular.js, Node.js, and JavaScript ES7. What will it be a year from now? How do you find the skills necessary to maintain the "older" stacks?

Considering that 52 percent of custom applications are built in three months or less, and that the typical lifespan of an enterprise application is 20 years, the adoption of OSS has created an environment that fosters the constant integration of new technologies that must plan for long-term maintenance, security, and disaster recovery.

Think of your own enterprise:

- Are you aware of all the OSS that's in use?
- Do you know the version of each package and the compatibility issues among those versions?
- Are you up to date on all security patches?
- Are you fully aware of the known vulnerabilities for each package?
- How do you keep track of new vulnerabilities as they are reported?
- Do you understand how different configurations of OSS can affect availability, security, and performance of your numerous applications and systems?

Additionally, do you have a process in place to track and manage:

- OSS that has been "officially" acquired?
- OSS that has been downloaded on an ad-hoc basis without being recorded in a central repository?
- The applicable licenses for the OSS in use across the enterprise?
- Specific configuration guidelines and dependency issues for each OSS package?
- The way(s) in which OSS packages will be supported across the enterprise, from development to production?

Knowing the answers to key questions on open source usage and processes, helps avoid the following risks:

>> Production outages

>> Performance degradation

>> Failure to scale to meet increasing demands

>> Ever-evolving security threats

>> Delays in meeting business strategies and initiatives

> "47 percent of organizations report that they have no formal processes in place to track the use of OSS within their organizations."

The configuration quandary

The configuration of intertwined OSS packages and stacks has turned out to be a particularly critical and thorny factor when it comes to ensuring application performance, security, reliability, and scalability. A recent analysis of a large sampling of real-world support tickets handled by the Rogue Wave Open Source Support team revealed that:

> 80 percent of production issues are due to improper configuration or other problems in the environment.

Let's look at how configuration relates to information security. There is no lack of examples of data breaches and other exploits that can be traced to improper software configuration or unapplied security patches.

- Pinterest had problems with Apache ZooKeeper, which led it to take down its entire site because there was a Single Point of Failure (SPoF) in its architecture. The company tried adding capacity, but it didn't help. It also investigated alternative OSS solutions but none were more mature than ZooKeeper at the time. By re-architecting and re-configuring its system, the company was able to maintain full performance and keep the site up even when ZooKeeper crashed.
- The 2017 "MongoDB apocalypse" that made over 50,000 servers across the Internet vulnerable to data theft and corruption was caused by improper configuration.[3]
- OWASP itself says the following about Apache Tomcat: "Most weaknesses in Apache Tomcat come from incorrect or inappropriate configuration. It is nearly always possible to make Tomcat more secure than the default, out-of-the-box installation."[4]

The security challenges don't stop with getting the configuration of your OSS components right. For instance, the well-known Equifax breach was the result of not keeping up with a patch for a known vulnerability in Apache Struts, even when internal administrators mandated the update within a 48-hour deadline. The result was the exposure of extremely sensitive information on more than 143 million customers, causing financial and reputational damage.

Similarly, the Heartbleed episode was the result of a vulnerability in OpenSSL that was widely known for two years before becoming headline news. As a result, approximately 70 percent of all websites worldwide were affected, producing an estimated $500 million in recovery costs.

Examples like these underscore the need to understand what OSS is running across your enterprise and to have systems in place to assure you can keep up with, and act on, vulnerabilities, security hotfixes, patches, and new versions of all these packages.

[3] *WIRED*, "If You Want to Stop Big Data Breaches, Start with Databases," March 2017

[4] https://www.owasp.org/index.php/Securing_tomcat

Part II:

How to achieve open source success

Understand, assess, and decide on the best OSS support option

How to innovate with less OSS risk

Although there are challenges, well-managed OSS can provide overwhelming benefits. As Gartner says in its March 2017 report, What Every CIO Must Know About Open-Source Software.[5]

> Tackle open source (either commercially supported or self-supported) as inevitable investments that by being properly managed, will yield considerable total cost of ownership (TCO) and 'business value' benefits. When unmanaged (or undermanaged), these same OSS technologies will instead introduce considerable technical, security and legal risks to the enterprise.

Seeking new solutions and well-adopted software from open communities is often the easy part of using OSS to innovate and reduce time to market. The challenge begins once those applications become part of your critical infrastructure. Supporting all this software that you didn't build and do not own can become a full-time business.

You need to develop a support policy for all your software, including OSS. There are several options at your disposal, boiling down to a few specific categories: self-support, community support, and commercial support.

Evaluate and understand the pros and cons of each to determine which approach will be most effective for your needs.

Self-support

Many organizations choose to rely on their internal IT teams and staff to support the open source packages they deploy. This is also the default support option (intended or not) when a developer adds OSS to a production system and doesn't communicate that fact to IT management.

The benefits of this model are relatively obvious. You can leverage your existing resources or hire specialists to maintain the products internally. Where necessary, you can find educational material to sharpen skills and designate code owners to assume responsibility for the various open source code that you've implemented. Ideally, these owners will be responsible for:

- Keeping ahead of new updates and functionality
- Being vigilant and proactive about security notifications
- Vetting software internally against business processes to ensure relevance
- Interacting with communities to request features and report bugs
- Dealing with the potential for poor documentation and slow community response
- Shepherding product knowledge to stakeholders

[5] Gartner, "What Every CIO Must Know About Open-Source Software," March 2017

Answer these questions to see if self-supporting OSS is right for you:

1. Are you and your IT colleagues sufficiently prepared for the management responsibilities of OSS and ready to remediate production outages, performance degradation, and any other issues that negatively impact your IT enterprise?
2. If you believe you have the expertise, is taking an in-house approach to supporting your OSS the most cost-effective use of your IT resources in terms of adding value to the organization?

Although many organizations feel that self-support is enough, the reality is that most open source problems are due to either environmental issues or misconfiguration of various packages. This means that the IT staff supporting these applications did not fully understand the way deployed solutions would interact in their environment, or that they didn't have a complete understanding of the best way to configure these applications.

On top of skills, consider how many people it would take to support your OSS deployments, given the tasks listed above. And that's just for keeping OSS running — how many more people would it take to add new functionality, adapt for increasing scale, or migrate to new systems?

So, despite the seeming benefits of self-support, this model is generally not sufficient to guarantee the performance and reliability of large-scale, business-critical systems.

Community support

The community can be your greatest resource when it comes to embracing open source but, like most things in the software world, there's a protocol. There are right and wrong ways of interfacing with these communities. The most successful organizations relying primarily on community support live by the following practices:

- Understanding that they're dealing with volunteer developers, in many cases, and that they're not paying for support. Patience and courteousness are key — there are no guaranteed service level agreements (SLA) here.
- Knowing that the most valuable currency is participation in the community. They are willing and prepared to donate their own resources.
- Supporting projects in many ways, including code development, documentation, donations, and evangelism. Communities tend to reciprocate in response to strong participation.
- Submitting clear, concise, and comprehensive bug reports and feature requests. One-liners tend to linger for months or even years in bug trackers.

At the end of the day, OSS communities are not your employees and they are not invested in the work you are doing. You can glean incredible knowledge from them and forge valuable relationships with committers, maintainers, and curators, but if you're looking for a guaranteed SLA to resolve issues within your deadlines and achieve true risk mitigation, your best option will be to engage a commercial support provider.

Commercial support

This model offers many options. There are companies that provide "enterprise" editions of the community software they maintain and those editions often come with options for SLA-based support and training. There are also organizations, such as the Linux Foundation and Rogue Wave Software, that provide the customized support team for you and help you build the skills you need internally. Though these options aren't free, when juxtaposed against the license costs associated with supporting a sophisticated commercial infrastructure, or the hiring costs associated with building and maintaining an internal support team, the value is compelling.

The benefits of utilizing a proprietary enterprise or third-party commercial support provider cover all aspects of OSS ownership including:

- Guaranteed SLAs that govern response and resolution times
- Proactive guidance, such as patching and vulnerability awareness
- Interaction with experts who have vetted the packages in dozens of environments for multiple business use cases
- Access to niche expertise and thought leadership
- Redundancy in skill sets and available resources

By selecting the right commercial provider for open source support, you can obtain both the freedom and flexibility gained from adopting free software, and the peace of mind that comes with commercial-grade enterprise support.

Making your OSS support decision

The best choice for your organization depends on your unique business requirements, and it's entirely possible to combine aspects of the three options above. Whatever you decide, the best way to extract the greatest benefit out of open source is to ensure that your business adopts a strong, strategic model for support, and utilizes it diligently to sustain your business operations, performance, and allows room for innovation.

5 questions

to guide your OSS support decision

1. What would the impact be if a key application or system went down for hours, days, or longer?

2. Does your staff have the necessary package-specific knowledge and skills to provide support across your entire enterprise? Do they know how all the packages are configured and work together?

3. Do you have a comprehensive inventory of all the versions of various OSS in production across your enterprise?

4. What system(s) do you have in place to keep up to date with the National Vulnerability Database and other sources of information on known OSS vulnerabilities?

5. How do you currently keep up with tracking and applying critical security and other patches?

Part III:

Worksheet

Assess your OSS risks

The first step towards better management of your OSS infrastructure is to determine the biggest areas of risk for the most critical applications in your environment. Typically, these applications are the ones that are end-user facing, core to the business, and the ones that would get C-level attention if something went wrong.

Using the table and descriptions of risk areas below, add your critical applications to the left column and write "Green," "Yellow," or "Red" for the five criteria listed across the table. If you don't know, assume the worst case, "Red."

Green. High-level of confidence that risk is mitigated and/or issues will not occur.	**Yellow.** Some risk is mitigated; confident that most issues can be dealt with by existing team. Do not know how to handle some issues.	**Red.** Unknown or most risk is not mitigated; existing team cannot handle most issues.

Description of risk areas

Technical risk

- **Maintaining versions:** Proactively knowing when a new package version is available and upgrading in a timely manner.
- **Scaled properly:** System and configuration of OSS packages support current, predicted, and unexpected load-volumes.
- **Performance:** OSS packages are properly configured to perform optimally, maximizing revenue.
- **Package selection:** There are many similar-looking packages for the same job — have you chosen the right ones?

License compliance

- **Rip and replace:** OSS packages come with legal obligations — violating those obligations can lead to removal of the package entirely and replacing with another component, causing release delays or costly updates to production systems.
- **Financial:** If OSS licenses are violated, it can lead to lawsuits and damages awarded.

Security

- **Vulnerabilities:** Failure to keep up with security patches can lead to data loss, theft, and corruption. Security is even harder to maintain if you don't know what OSS packages are in your environment.
- **Response team:** If your team doesn't have cybersecurity expertise across a broad range of packages, they won't be prepared to act immediately in an emergency.

Asset management

- **OSS inventory:** An OSS bill of materials (BOM) is critical to knowing what packages are in your environment and the versions that they're on.
- **OSS policy:** An OSS inventory is useful only if it reflects reality, which means all users of open source must follow a policy to track package additions, deletions, and locations within development and production systems.

Skillset/expertise

- **Sufficient resources:** Without 24/7 coverage in all supported geographies for all packages and combinations, you're at risk for excessive downtime.
- **Implementation/ configuration:** If you don't have expertise deploying your specific stacks of OSS packages in an enterprise production environment, you won't be able to properly configure them for scalability, security, and performance
- **Operations:** If you don't properly monitor and manage the OSS packages in your deployed stacks, they could exhibit symptoms that will lead to outages or performance problems and you won't know until the app actually goes down.

OSS risk assessment

Your critical application	Technical risk	License compliance	Security	Asset management	Skillset and expertise
Example application	Green	Red	Yellow	Red	Red
Application 1					
Application 2					

You have multiple support options but OSS enterprise support itself is essential

As OSS continues to proliferate across the enterprise, and even dominate many IT organizations, it is imperative that you have a well-designed and clearly-articulated OSS support model in place to ensure optimal performance, scalability, and security. The good news is that you have a number of options for this support.

When your organization is running at the speed of ultra-competitive business, however, it can be easy to feel that having to make even one more decision is just too overwhelming. In the case of deciding how to support your IT enterprise, avoiding, or putting off that decision could prove extremely costly… even catastrophic.

Rogue Wave Open Source Support can help you create an OSS support model that is tailored to your organization's unique requirements. Our open source architects resolve open source issues ranging from package selection and setup to integration and production problems — all within guaranteed SLAs.

To talk to a Rogue Wave Open Source Support architect now, visit roguewave.com/OSSexpert.

InformationWeek

RogueWave SOFTWARE

Rogue Wave helps thousands of global enterprise customers tackle the hardest and most complex issues in building, connecting, and securing applications. Since 1989, our platforms, tools, components, and support have been used across financial services, technology, healthcare, government, entertainment, and manufacturing, to deliver value and reduce risk. From API management, web and mobile, embeddable analytics, static and dynamic analysis to open source support, we have the software essentials to innovate with confidence. roguewave.com