Global Information Assurance Certification Paper

GIAC Practical
Student: Angela Karnoupakis
Certification Attempting: GSEC
Version 1.4b
Option 1
Date: Sunday, February 22, 2004

Title: *Open Source: Secure use in government*

Abstract: Open Source Software is a resource that many commercial companies have begun to utilize with great success. Government agencies have proceeded into this realm with great reluctance. This paper will illustrate the benefits of Open Source Software and provide steps for the government user to research and implement Open Source Software in a strategic and secure manner.

Open Source Software Definition and Background

Open Source Software, a product behind a movement with a history that sounds straight out of a Robin Hood-like fairytale. Software engineers in the 1980's revolted against the industry market trend that began protecting source code against the publics that used them. These open source revolutionaries committed to sharing their source code within the community of hackers. In this community they would work towards the common goal of a secure, bug-free piece of software.

Open Source Software is defined as "A method and philosophy for software licensing and distribution designed to encourage use and improvement of software written by volunteers by ensuring that anyone can copy the source code and modify it freely.(Open Source Definition)"

In the late 1990's Open Source Software (including Linux, FreeBSD, Apache) started to receive recognition from outside their community of engineers. Management, network administrators, and investors started to take notice of this new technology and wonder if it would be a solid enough to work with in their own environment. Since then, the public sector has experimented with these products and is now utilizing them across the spectrum in order to have better control of their software while realizing the cost savings of Open Source Software.

The Value of Open Source Software in Government

Commercial Examples
For years, the commercial world has realized the myriad of benefits in the use of Open Source Software. Amazon was able to shave 25%, or $17 Million, off of their technology expenditures by migrating to a Linux based platform(Shankland

and Kane). Cost is not the only reason why the commercial sector is excited at the prospect of using Open Source Software. DaimlerChrystler's Chief Information Officer, who changed a supercomputer that ran crash tests over to Linux, stated "Everyone thinks that you do this just for the money, but we're seeing other factors." DaimlerChrystler was surprised to find that the system that ran the crash tests on Linux was easier to manage and had a higher level of performance than with her legacy applications(Koch).

Cost Savings

In many cases, the life cycle cost of Open Source Software is much lower than proprietary software. With little to no start-up cost on these products, the largest investment a user will find is in the set-up and maintenance of the product. Economies of scale also make Open Source Software a value to governmental agencies. Most Open Source Solutions do not charge a "per user" licensing fee, and as a result there are no additional cost passed along to larger organizations. For example, a city could provide an Open Source web server solution to its city council, school system, and police department. Wherever usage is common, there can be a single application solution at a minimal cost start-up cost with a shared life-cycle support cost.

Ease of Management

Open Source Software is often much easier to manage because the user is in control of the product support. Another benefit is that when doing initial research into a product, the user is able to do in depth evaluation of a product. An article in CIO Magazine said product support was more simple because, ". . .they don't have to depend on vendors with their own agendas, because when an open source application doesn't work, administrators can look at the source code, figure out why and write a fix themselves.(Koch)" This internal responsibility requires active ownership of applications, but it also allows for a much more flexible and adaptable product for the user.

Reduced Risk

The support of an Open Source Software application is also less susceptible risks caused by the whims of budget cuts, employee and priority changes that are a reality for many government agencies. Open Source Software offers an increased level of flexibility for organizations that are required to support applications despite the many changes in funding and people power they might be up against.

Another advantage of Open Source Software is found in minimized security risk. Through a reduced use of proprietary applications, government agencies can mitigate the risk of downtime due to large-scale cyber attacks. A greater diversity of products will lessen the opportunity for cyberattackers to take our large segments of our government. The January 2003 Mitre Report, "Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense" referred to this risk in their 2003 study: "Acquisition diversity reduces the cost and security

risks of being fully dependent on a single software product, while architectural diversity lowers the risk of catastrophic cyber attacks based on automated exploitation of specific features or flaws of very widely deployed products." Increased diversification and specialization of the applications used by governmental agencies will allow for an additional level of protection.

Open Source in Government

Until recently, the United States government has treaded lightly in open source technology. At first, the trepidation was one of security issues. How could the government possibly secure applications where all vulnerabilities were open and available to friends and foes alike? Internationally, countries like Brazil, China and Argentina mandated the use of Open Source Software in order to decrease dependencies on the use of foreign software sources and to encourage local software development economies. The European Union, United Kingdom and South Africa are leading the way in development of Open Source Software Policy. Many countries openly encourage the use Open Source Software products.

Statistics in developed by Mitre Corporation in May of 2002 found that there were 249 identified uses of Open Source Software fielded in the United States Department of Defense. In early 2003, Mitre Corporation, on behalf of the Department of Defense, conducted an assessment of the use of Open Source Software. The report concluded that, "The main conclusion of the analysis was that Free and Open Source Software plays a more critical role in the DoD than has generally been recognized . . .that banning FOSS would have immediate, broad, and strongly negative impacts on the ability of many sensitive and security focused DoD groups to defend against cyberattacks.

The question was no longer IF the Department of Defense should use Open Source Software, but rather HOW will it use it. Many cities, county, state, and federal organizations are faced with the same predicament of how to find secure and functional open source solutions to bring to their users.

Steps for Secure Use of Open Source in Government

In the procurement of Open Source Software, as in any software procurement, the organization should have a methodical way of defining it's needs, reviewing products, and preparing for the financial and lifecycle support of the software being procured. It is wise to define the process and ensure that you have the approval of leadership prior to the start of the procurement process. Good communication and well established guidelines will ensure that all parties involved understand the process and are available for review at key points in the process. When an organization is first testing the waters for Open Source Software, it is advisable to begin with applications that are will not cause a major disruption in the usability of the system. A good place to start is with low-risk internet applications and those supported by the major providers of Open Source Software support including Dell, IBM, Sun, etc.

Preliminary Research
In preparation for reviewing Open Source Software, begin by defining the needs and expectations you have for the product. Ideally, you should gather a segment of your user population and identify what you require the application to accomplish. Specify what your current functionalities are and what increased level of functionality you would like to achieve with this new procurement. Be sure to review any procurement guidelines established by your organization. Frequently software procurement policy will suffice in the procurement of Open Source Software, but confirming this information early on will streamline your process.

It is advisable to use a wide range of research tactics when researching Open Source Software solutions. One resource that should not be overlooked is that of peer research. Speaking with your peers in the workplace will bring to light real-life experiences with that can help you build upon the "Lessons Learned" by others. This research should not be isolated to the government community. Discussing options and solutions with people in the commercial and non-profit sector will often provide a broader range of information. Consider organizations with similar needs. For example, an individual working within the Department of Defense would have similar data integrity and security needs as someone working in the banking or healthcare industry.

Formalized research is another way to learn more about the products that you are using. The internet provides a wealth of reviews that are available through simple web searches. Use a variety of search engines in order to obtain a wide and unbiased search. Open Source websites such as Free Software Directory http://www.gnu.org/directory/ are offers thorough reviews of products. Another resource would be reviewing lists of products that are considered to be mature. David Wheeler, Open Source Software expert, offers a list of mature produces at http://www.dwheeler.com/gram.html and also suggests his sites such as the European Union's Interchange of Data Between Administration site. All Open Source Software products in consideration should come from trusted sources that appropriately address your organization's acceptable level of risk.

Functionality
Returning to your list of required functionality needs that you developed earlier in your research, you should now compare your requirements against each of the products that you have identified. Assuming that you have a prioritized list of requirements, you can quantitatively review each product against the needs of your organization. Realize that many Open Source Software solutions will provide you a few of your needs while not addressing others. It is important to have a good understanding of the features that you are able to turn off or add on to a Open Source Software solution. Often times there are a myriad of features that are unnecessary for your organization. One of the great benefits of the Open Source Community is that options for personalization of products are often

just a message board away. Remember that each individualized functionality that you add will directly increase the time you spend on your product and as a result, increase the cost of your product life-cycle support.

Some solutions may have and unclear or unacceptable level of risk associated with "offshore developers". If you are concerned about the level of risk involved with your solution, it is advisable to utilize products that have been Common Criteria evaluated and that address the correct security target for your project. Be sure to document the pros and cons of each solution so that you can capture the information for your reference and for briefings to leadership.

Cost

For many organizations, cost is a major factor in pursuing an Open Source Software solution. This is why it is important that all costing research is thorough and provides a total cost for the lifecycle support of the product. When evaluating an Open Source Software solution along side a proprietary product, the initial costs will appear considerably lower. Only when taking long-term support costs into consideration can we get a true look at the costs of a product. Total ownership costs may include any of the following items: licensing fees, legal fees, training, hardware, support (provided by internal or external vendors), and installation. A good review of total life costs could be found in the costing for the current system that you would like to replace. A broad look at all possible costs associated with this procurement will give you help you to further reduce your list of possible Open Source Software solutions.

Licensing

The key difference between Open Source Solutions and proprietary solutions is the licensing agreement. It is vital that you have a clear understanding of the licensing of the products you are reviewing as they will affect your organization and the functionality of your product. It is important that you review the entire licensing agreement and pay special attention to the following areas: code modification conditions (copylefting/non-copylefting), warranty, and computer library. A review of the licensing agreement will help to clarify the responsibility of both the user and the developer of the Open Source Software. Differing licensing practices may offer additional benefits to your organization. Note these differences when comparing products.

Be sure to review all licensing agreements with your legal representation. If your legal team is not familiar with the latest Open Source Software issues, you should consider requesting additional legal support. Preliminary legal reviews of licensing and legal issues will save your program a great deal of time and money if lawsuits are prevented.

Support

When analyzing Open Source Solutions against one another, it is prudent to consider the respective support options. Open Source Software products are

developed by a community, and that community will have an effect on the functionality of your product. It is important to review the activity of the community when considering a product. Signing up for bulletin boards and mailing lists will help you to assess the activity of this community. A productive group of developers will have an active dialogue and post changes and patches regularly. A product that may have fallen out of favor will lack activity and support. It is important to note this differences in community because it will directly effect your time investment and bottom line. Depending on the product that you are reviewing, you may chose to pay for support offerings from companies like Sun, Dell or IBM. Research the customer support offered by these companies and include this information in your product review.

Mid-Process Review
After identifying a handful of possible trusted solutions (less than 5), assessing the functionality, and identifying the total ownership costs of a product, this is a good time to review the options with a group of users and management. A "pulse-check" at this point will ensure that there is a consensus with the current research and that any issues are identified and expeditiously addressed. Use this Mid-Process Review to pick your top products for further review.

Testing and Evaluation
After the Mid-Process Review, the top products should be tested in an isolated lab environment that mimics your operational use. The product should be tested for and extended period of time reviewing the original requirements that were identified early on in the study. During the testing phase, review the product's actual performance against the stated requirements, costing estimates, technical requirements, etc. If the testing doesn't perform as planned, consider the next steps. Often, technical problems in the testing phase can be solved with Open Source Community interaction. This is a great way to gain a better understanding of how the community truly functions. If issues are irresolvable, consider dropping this product from your list.

Security Testing
Vital to any software testing phase is a thorough security testing phase. Depending on the budget associated with your project, you should attempt to attain the highest level of security testing and documentation afforded according to your budget.

A very modestly funded security testing project should take into consideration security documentation, product vulnerability notification, and product vulnerability research. At the very least, there should be a thorough review of all product security documentation. A complete review of product security documentation will help clarify how the product aligns itself with the security requirements that you identified early on in your process. Researching how (or if) the product reports vulnerabilities to it's users will help you to better understand how your security implementation will function after installation.

Also, utilize online reports of product vulnerabilities in order to mitigate the risk of the most common security issues facing your product.

In a more moderately funded testing project, a security specialist should review the program down to the code level to ensure the product will not put your system at risk. Reviewing source code, as opposed to binaries, allows you to take a more in depth look at security vulnerabilities and align the review to the most vital security needs of your organization. Other important code issues that should be considered include minimized privileges, code scanning, and simplicity of design.

If you have the resources and the security need, you may want to consider having the product tested at a commercial Common Criteria lab. This will provide a higher level of assurance and will give your organization and unbiased look at the specified security target identified in the product you are reviewing.

Select a Product

After the testing reports are completed, you should reconvene your panel of management and users in order to select your final product. Panelists should be given the product information in advance so that they might have ample time to review it. Depending on the depth of your program, several hours or days may be necessary for discussion of the Open Source Software solutions that are being reviewed. Proper note keeping at this event will help to keep a record of the ideas and decisions made at this meeting. The final responsibilities of this product selection team will be to identify the major milestones and responsible parties for the Implementation Team.

Project Implementation

Now that you have selected a product, you must finalize the plans that will bring your Open Source Software solution to your organization. Your initial research and testing will be folded into several project-planning tools including an extensive budget and a project plan. The budget should incorporate all the changes made during the testing phase and should prepare for the entire life of the product. The project plan should include major milestones. An example of major milestones would be installations, product start dates, product refreshes and updates. It is important not to neglect to inclusion of plans to migrate from the legacy application. Often times schedules can suffer due to the absence of transition planning from one system to the next. Another pitfall that teams often run in to is support planning. The resources of an information technology team are often stretched thin, and the ramping up of a new program can often lead to delays in the project plan. Any critical path milestones should be identified and closely tracked by a member of the project implementation team. It is vital that budget, resource and project plans are all inter-related and reflect the support required to implement your new Open Source Software.

Success Criteria

It is important that success criteria are developed as a metrics to the Open Source Software program. This will help clarify how the product met the needs of your organization. Quantifiable metrics will show the strengths and weaknesses of your new Open Source Software program and will help defend or rework any future projects. Return to the requirements that you developed in the requirements development phase. Find ways to measure if your Open Source Software solution is meeting the needs of your organization. For example, if your company experienced server down time, due to problems with your legacy application, tracking down time would be a valuable metric. Showing how this metric improves with time and against the legacy application's metric will illustrate the value added in the Open Source Software solution. Another important metric is that of cost. Initially your new Open Source Software Solution will have a distinct advantage over your legacy application, so all metrics to this effect should be considered for their long-term value. Be sure to define the period of collection for each metric and keep up to date on gathering this data, as it will help you make the case for future Open Source Software projects.

Conclusion

Open Source Software is a force in the software development world of the new millennium. For governmental organizations to ignore the opportunities offered by this line of products would be to the detriment of organizations as well as they communities they serving. Governmental organizations have the opportunity to take advantage of yet another asset of the free economy. Open Source Software products offer benefits that include cost savings, reduced risk, and ease of management. The commercial sector has been taking advantage of these technologies and there is a great deal to be gained by governmental adoptions of Open Source Software Solutions. Andy Stein, Chief Information Officer for the city of Newport News, Virginia expanded the idea to illustrate the greater good of Open Source Software, "There is only so much one can develop and accomplish working alone. . . In local government, I will accomplish in 2 years more than I could in 25 years in the private sector.(Adelstein)" By utilizing this vehicle for technological progress, government agencies can be on the forefront of software development instead of the tail end, where they have been for so many years.

In many ways the procurement process of an Open Source Software solution is much the same as with a proprietary solution. By identifying the major differences in lifecycle support and clarifying the differences in the long term, we can safely identify products that will support the needs of our organizations without sacrificing security. With the proper preparation and support, professionals working in the governmental sector can offer a wider selection of products to their organizations and further strengthen the security and functionality of their information systems.

References

Adelstein, Tom. “Open Source in Government: Newport News, VA”. 15 January 2004. O’Reilly Linux Devcenter.com. 18 January 2004 http://linux.oreillynet.com/pub/a/linux/2004/01/15/andy_stein_interview.html.

Briggs, Julie, & Peck, Dr. Matthew. “QuinetiQ Analysis of Open Source Solution Implementation Methodologies”. February 2003. The Center of Open Source & Government. 21 December 2003 http://www.ogc.gov.uk/embedded_object.asp?docid=1000435.

Koch, Christopher. “Your Open Source Plan”. 15 March 2003. http://www.cio.com/archive/031503/opensource.html. CIO Magazine. 12 December 2004.

“Open Source Definition”. www.opensource.org/docs/definition/php, 18 November 2003.

Shankland, Stephen, Kane, Margaret, and Lemos, Robert. “How Linux saved Amazon Millions”. 30 October 2001. Cnet News.com. 10 February 2004 http://news.com.com/2100-1001-275155.html?legacy=cnet .

United States, Department of Defense written by Mitre. “Use of Free and Open Source Software in the U.S. Department of Defense”. 2 January 2003. 2 December 2003 http://www.mitre.org/work/sepo/library/docs/dodfoss.html.

Wheeler, David. “How to Evaluate Open Source Software/Free Software Programs”, 21 November 2003. 21 January 2004 www.dwheeler.com/oss_fs_eval.html.

Williams, Sam. “A Timeline of Open Source in Government”. 15 July 2002. O’Reilly Linux Devcenter.com. 18 January 2004 http://linux.oreillynet.com/pub/a/linux/2002/07/15/osgov_timeline.html.