October 2015
Issue No: 1.1

Good Practice Guide

Open Source Software – Exploring the Risk

Customers may continue to use this guidance. The content remains current, although may contain references to legacy SPF policy and classifications.

CESG

NATIONAL TECHNICAL AUTHORITY
FOR INFORMATION ASSURANCE

Good Practice Guide No. 38

Open Source Software – Exploring the Risk

Issue No: 1.1
October 2015

Executive Summary

This guidance seeks to assist a range of IA professionals in exploring the risks associated with the use of Open Source Software (OSS) products. It does so by prompting a number of 'whole lifecycle' issues and questions which potential users should ask themselves when making software choices, not just of OSS, but also of proprietary products. This is because there are no 'right' or 'wrong' answers when it comes to the security of OSS versus that of proprietary (typically closed source code) products. There are good and bad examples of each in this respect and no one type is inherently more, or less, secure than the other.

Whether to adopt OSS or not is ultimately a business decision that should be taken in the light of understanding the risks involved.

A number of common misconceptions regarding OSS are also explored which, if not understood and handled correctly from the outset, can give rise to a variety of risks. Two issues in particular are dealt with: the belief that use of OSS comes 'cost-free' (invariably, for business, it does not) and that OSS and Open Standards are the same thing (they are not).

This guidance supports the Government ICT Strategy[I] objective of creating a level playing field for open source software solutions. It does not evaluate, recommend or otherwise offer judgement on the following:

- Specific OSS products;
- Savings in running costs that an organisation *may* realise by the adoption of OSS over proprietary products;
- The legal risks that may arise, for example from issues concerning copyright, intellectual property, or infringement of licences.

[I] "Government ICT Strategy", March 2011 (www.cabinetoffice.gov.uk).

Contents:

Chapter 1 - Introduction

Key Principles

- There are no 'right' or 'wrong' answers regarding the security of OSS as opposed to that of proprietary products. It is for the business to decide based on the balance of benefit versus risk
- No one type of software is inherently more secure than the other. Organisations should explore a number of common security requirements and issues against which any software product, OSS or proprietary, can be tested
- Use of CESG's Assurance Framework is recommended to establish confidence in any of the security aspects of OSS

Definition

1. Open Source Software (OSS), often referred to as just 'open source', or Free / Libre Open Source Software (FOSS or FLOSS), is computer software for which the human readable source code and various other rights are made available in the public domain under the terms of a licence that meets the Open Source Definition (OSD), the custodian of which is the Open Source Initiative (OSI) (reference [a]). OSS often comes in pre-compiled binaries or complete operating system distributions and is also often incorporated into commercial products.

2. The licence usually permits users to use, change and improve the software and redistribute it in modified or unmodified form for others to use or improve. Owing to its inherent flexibility and perceived reduced running costs, OSS is rapidly gaining in popularity over traditional, proprietary software products where the source code is closed. A proprietary product can be regarded as one whose licence falls outside of the OSD definition of OSS. Such licences are described as 'closed source'.

OSS versus Closed Source Security

3. The security argument surrounding OSS is, that with its source code available to all, it is self-evidently less secure than its closed source counterparts. However, the opposing view is that the accessibility of code promotes early detection of vulnerabilities and encourages fixes that therefore lead to a more secure product. This is the so-called "thousand eyes" argument.

4. This guidance takes the view that no one particular type of software is inherently more, or less, secure than the other and does not favour one type over the other. Each must be approached on a case-by-case basis. Instead, it aims to highlight common security requirements and issues against which any particular software product can be tested, be it open or closed source. Use of CESG's Assurance Framework is recommended to establish confidence in any of the security aspects of OSS. This framework is a way of thinking about how assurance can be achieved throughout the lifecycle of an ICT system or service. For further information, see CESG Good Practice Guide No. 30 (GPG 30), Assurance of ICT Systems and Services (reference [b]).

5. The OSS versus proprietary products security debate is subjective and emotive. Various surveys, statistics and opinions in recent years have purported to ‘prove’ the case one way or the other. The truth is that there are advantages and disadvantages regardless of what the type of software is selected. Both will have vulnerabilities and both may be subject to attack.

Chapter 2 - Making Software Choices

Key Principles

- A number of issues and questions applicable to OSS and proprietary products alike need to be considered when deciding on particular software
- Principal among these are licensing, provenance, quality of code, support, product development, vulnerability management and security-enforcing products
- Process and criteria for software selection should be the same for both open source and closed source products

Elements to Consider

6. Introduction of any software product, whether OSS or proprietary, carries an element of information risk. Be clear about the business requirement for the software and, in the case of OSS, whether it is already embedded in the intended product or will need to be obtained and installed separately. Ensure that the product's introduction is appropriately risk managed as part of the overall system architecture and that it is included in your software asset management (SAM) arrangements.

7. It is essential that the introduction of any software follows established organisational change control processes in order to minimise risk. Correct configuration, for example, will ensure that any functionality not required is disabled, while only specifically authorised (and trained) privileged users should be allowed to install new software.

8. Deployment within the organisation should be monitored to ensure that best use is made of the product, decisions about its development and lifecycle are only made by those authorised to do so, risks are managed and that licence terms are not infringed.

9. The use of an OSS product over a proprietary one does not affect the need to comply with Mandatory Requirement 32 of the HMG Security Policy Framework (SPF) (reference [c]) to perform an IS1 & 2 – Information Risk Management (reference [d]) of the ICT system or service in question.

Choosing between OSS and Proprietary Software

10. Due to the complexity of many organisations' ICT environments there is likely to be a requirement for a number of software products, both OSS and proprietary. It is important that any software product meets both business (functionality and cost) and security requirements appropriately and that the process and criteria for selection are the same for both types. In addition, the use of products (whether OSS or proprietary) that utilise common open standards is recommended to avoid issues of future vendor lock-in. The standard claimed should always be checked to ensure that it has been implemented accurately and in full. Any ill-defined elements could lead potentially to issues of interoperability.

11. Organisations should consider product support lifecycle, including the likelihood of a requirement for mid-life enhancement, and weigh this against how long any proprietary products are likely to remain supported by the vendor and what the consequences will be when that support comes to an end. Another equally important factor to consider is the length of time that any particular business function needs to be supported.

Licensing

12. The OSS licence determines how the user is permitted to use the software. There are a considerable number of licences approved by the OSI from the popular and most widely used to those covering specialist areas. Not all licence conditions are the same and it is essential that organisations understand the terms of the licence and what the licence permits them to do. There is no difference in this respect to the approach that should be taken to commercial licensing.

13. Some licences, for example, may only permit a 'not for profit' use and organisations should check carefully in this respect the category of activity that the licence deems them to come under. Other licences may insist that if code is developed for a specific purpose then it must fork[II] (see paragraph 33 for information about forking). Another aspect to be aware of is that some licences may insist that vulnerability assessments are made available to the entire community.

14. The most widely used types of licence (BSD, GNU GPL (see footnote [III]) and GNU LGPL) illustrate the sorts of differences involved:

    a. BSD - used in 1989 by the University of California to license the Berkeley Software Distribution (BSD). This licence is designed to maximise the use of software. Users are free to do whatever they want with it, to the extent that it can even be used in proprietary products;

    b. GNU GPL – the GNU Project's General Purpose Licence (often referred to simply as 'GPL') enforces "copyleft". This means that when redistributing the software, with or without changes, users must pass on the same freedom to others to make changes and redistribute in turn. The GPL licence is therefore designed to promote contribution to the OSS community;

    c. GNU LGPL – published by the Free Software Foundation, the GNU's Lesser General Public Licence, is less restrictive. Like the GPL it enforces "copyleft" on the software itself, but this does not extend to other software that merely links to it (although there are certain other restrictions applied). It is used primarily to cater for software libraries.

15. Each licence should be assessed on a case-by-case basis. Dual licensing, for example, may be a particular issue. This situation arises when an OSS product

[II] Forking occurs when one or more strands OSS development diverge and follow separate paths.
[III] GNU is a recursive acronym that stands for 'GNU's not Unix'. GNU itself is a Unix-like operating system developed by the GNU Project beginning in 1984. An alternative expansion of GNU sometimes encountered is "Genuinely not Unix".

is available with a choice of licences depending on the use to which it is to be put. Whereas personal use by the individual may be free of charge, that by a corporate or commercial body could involve considerable cost.

16. In general, OSS licences are far less restrictive than proprietary ones. The flexibility that the user has to adapt the product also brings with it associated risks of mismanagement and the need to manage the business responsibly.

Provenance and Quality of Code

17. Any code can vary considerably in terms of trust, quality and provenance. For both proprietary and open source code a set of developers with a proven track record of quality coding practices and a reputation for delivering quality products will be a lower risk than a small-scale, less experienced or unknown set of developers. For example, there are instances of open source products that are well managed (sometimes by commercial organisations), widely deployed and used with sound configuration control processes. This contrasts with instances of open source products that have been developed by a very small number of amateur developers years ago and have not been maintained. However, it is clear that the same scenarios are equally possible with proprietary code development.

18. It is for individual organisations to assess the provenance and quality of software products they select, in just the same way that they should for any other product. They should assess whether the product comes from a mature community with an extended cadre of code developers and testers. Is there a clear policy about how contributions are evaluated and included and are any of the developers / testers (well) known individuals within the OSS community / ICT world in general? Is there evidence of contributions from respected technology organisations? How many downloads of the software have there been? While a high number may reflect a degree of trust in and satisfaction with a product, it may equally be spurious and warrants investigation. A more substantial and reliable measure of trust and quality in a product will be proof of its trouble-free deployment and use in the real world.

19. Confidence in the detail of the provenance of some OSS may be difficult to establish, as potentially anybody in the community can contribute to its development. However, its quality is likely to be much more reliable if many individuals have contributed to its development, it is a well-established product within the OSS community and there is a strict and reliable regime in place to check for errors or problems.

20. In common with proprietary software, there will probably be different versions of the same OSS, such as a more stable version with a set amount of functionality and a development version for those who want or need all the latest features immediately and are willing to risk using code that has not yet been tested completely. In this case, the users themselves will often become developers of the code. Care should be taken to select appropriate versions of products that meet the needs of the business.

21. Products should be purchased or obtained through a supply route with an appropriate amount of trust. For the supply of software products you should consider the provision of the product itself as well as updates and patches.

22. Apart from the more obvious means of building trust in the supply route, e.g. by only accepting digitally signed distributions, it can be something that is otherwise very difficult to quantify. Where software (both open source and proprietary) and updates can be downloaded from an internet source there should be an appropriate analysis of the reputation and standing of the source, such as the length of time the service has been available, how well known it is in the community and its reputation as a supplier in terms of availability and integrity of its products.

Support

23. Any software product selected for deployment should have a range of support options available. As with proprietary products, support for OSS is available on a payment basis, but many popular OSS products also make arrangements for free-of-charge support from the user-community.

24. You should make an assessment of what support requirements you have for any given solution (including product documentation) and this should form part of the assessment of whether any particular product set meets the overall business needs.

25. If you need or want a third party to provide for specific support requirements will you be able to switch provider seamlessly in the event of a problem? The option to do so with proprietary software seldom arises but, with OSS, the fact that issues of vendor lock-in often do not arise means that you can explore the market for the best option.

26. What is the quality and extent of any accompanying documentation available? The higher the quality of the OSS in question, the better the documentation is likely to be, but if it's lacking in any way you will probably need to meet your requirements by purchasing it elsewhere, if available, or pay to have it written for you.

27. The manner of introduction or conversion to OSS is crucial to minimising risk. A managed programme incorporating familiarisation, training, configuration and integration with existing proprietary products, transfer of business function from legacy software, etc., are all vital elements to consider. This is no different to any major upgrade involving proprietary products where similar issues concerning risks arise.

28. User training is equally important to minimise non-malicious risk that might result from a combination of user error and ignorance of the product and must be tailored to the OSS architecture and governance structure you decide to establish; for example, how many 'super-users' / privileged access / normal user accounts are required and in what ratio? Once again, this approach is no different to that which should also apply to closed source products.

Product Development

29. Will you need to develop the software? If so, do you have sufficient resource and technical expertise 'in house' to do so? If not, an external supplier will be required and all the same questions as above regarding provenance, reliability etc. will arise.

30. If you intend to develop your own organisational modifications to OSS you should consider to what extent, if at all, you wish or need to contribute modified code back to the OSS community. This is important as it may affect the OSS product selected and the type of licence required. While contributing code back is likely to result in greater peer review and community testing of the code, one aspect to consider is accreditation, where the intention to contribute modified code from an ICT system holding protectively marked information might be an issue.

31. Most licences permit modified code to be used within an organisation without further obligation. This may be beneficial for organisations using OSS to handle sensitive information, particularly so if the code modification itself is sensitive for any reason. However, this may involve increased costs in terms of code maintenance and support as opposed to contributing it back to the community to become part of a wider, updated version of which you may then take advantage at a later date.

32. Depending upon the licence in question the modular properties of OSS can often be developed to extend the functionality of a product such that different elements can be shared separately with different entities, e.g. other parts of your wider organisation, business partners, and so on.

33. Does the OSS product have a history of 'forking' and, if so, could this present future development or support problems for your organisation? 'Forking' occurs when one or more strands of OSS development diverge to such an extent that users have a choice to make in which to pursue, or even to seek a new OSS solution elsewhere. Indeed, some licences stipulate that forking must occur if code is developed for a specific function. The advantages and disadvantages of forking are subjective. Some argue that it leads to a dilution of developer resource across the strands, while others view it positively as another vehicle by which more options are provided and the quality of OSS is enhanced.

Vulnerability Management

34. Assurance can be gained by ensuring that the OSS is further evaluated and tested 'in house' before it is deployed operationally within the organisation. If your organisation does not have the capability to do this itself, a service would need to be bought in. The decision to do this will depend on how much trust you have in the OSS and the provider in question. The fact that a vulnerability assessor will have access to the source code may provide a greater degree of assurance than is the case with proprietary products where the source code is withheld.

35. All software products (not just those providing security functionality), whether OSS or proprietary, can introduce vulnerabilities into an ICT system. Some of these will be known and some unknown. It is critical that a robust process exists for managing vulnerability disclosure and development of patches for vulnerabilities. Equally important is the ability to distribute patches quickly, easily and with full confidence in their integrity, so as to minimise the period where exploits can be developed to target any vulnerabilities without an available patch.

36. Part of the product selection process should include an analysis of how quickly patches are issued once vulnerabilities become known and what vehicles are used to achieve this. Is there a reliable service direct from the supplier or community or will you be reliant upon a third party? How reliable is a particular third party likely to be in meeting your needs and what is their track record in this respect? The fact that source code is available does not automatically mean that it has been scanned thoroughly for problems.

37. A lot of the OSS and related patching in corporate use is made available through bundled ‘distributions’ and, as is the case with proprietary products, most organisations avail themselves of such a service. There is a good degree of pro-active co-operation between many distributions regarding vulnerability management, especially the major ones. Once again, organisations should research what is on offer from both the open source and closed source communities and take this into account when making software choices.

38. For patching, organisations may wish to consider client software products that have an element of ‘auto-update’ capability, rather than relying on an administrator to scan for available updates. However, it is for organisations themselves to decide, as part of their risk assessment, whether or not to adopt such a service. The benefits of ‘auto update’ notwithstanding, organisations should still consider their requirement for the assurance that can be gained from ‘in house’ evaluation and testing before deployment within the organisation (see paragraph 34).

39. While it is important to apply patches as soon as possible, just as with proprietary products organisations need to be confident that they are stable and have been tested properly (by your own or a partner/outsourced organisation) prior to application. The purpose is to ensure that, once applied, they do not impinge on the ICT system’s overall security, functionality and performance by introducing security vulnerabilities elsewhere.

40. It is sometimes asserted that more pre-release testing takes place with proprietary products but, as always, it is advisable to weigh up each product on a case by case basis, regardless of whether it is proprietary or not.

41. Well-established OSS products, in general, have a good record of developing and issuing patches once a vulnerability has been discovered. In some cases, where large developer communities are involved, theoretical exploits may be identified and fixed before any ‘real world’ exploitation has occurred. Over time this can generate a degree of hardening of the product in comparison to that of a smaller community.

42. Many companies that supply proprietary software products also have mechanisms to deal with responsible vulnerability disclosure. In these circumstances, vulnerabilities may be privately disclosed, allowing a patch to be developed and issued before development of vulnerability exploits is possible. However, one should consider that the time between issue of a patch and exploits becoming available can be extremely small (in some cases a matter of hours). Rapid patching is therefore extremely important.

43. Do you need, perhaps for reasons of security, to mask your connection to a particular OSS product or to the community in general? As long as the licence in question is not infringed, this may be achieved by various means, for example by conducting all dealings through designated individuals, anonymous engagement by use of generic e-mail, third party cut-out, and so on.

Use of OSS in Security Enforcing Products

44. It is important that products which provide critical or security-enforcing functions are appropriately robust and assured. There is no inherent IA benefit from either OSS or proprietary development processes or software. Any instance of a product should be assessed against functional and security requirements appropriate for the intended role of the product. Some OSS products will be more secure than their proprietary counterparts and vice versa.

45. The factors described above, such as provenance and quality of code, vulnerability management processes and support, all contribute to how robust any software product is.

46. There is no inherent reason why OSS cannot form part of a formally assured product or service. However, where OSS is used within a product you should ensure that the product developer truly understands the software and its functionality, rather than having just downloaded a module and integrated it into their product. This is especially important where the OSS performs a critical role, such as implementation of cryptography.

Chapter 3 - Common Misconceptions Dispelled

Key Principles

- There are associated costs with the use of OSS; it is rarely, if ever, cost-free
- OSS is not the same as Open Standards / Source Available / Open Content / Shareware / Free Software / Freeware

OSS is not Cost-Free

47. OSS is also referred to as Free Open Source Software (FOSS) or Free / Libre Open Source Software (FLOSS). The key element is that 'free' refers to the freedom that the user has to obtain the software, use it, change or develop it and then, usually, pass it back to the community for others to use and develop in turn. Use of the term 'free' does not mean that OSS by definition costs nothing to obtain, deploy and use.

48. As evidenced earlier in Chapter 2, the type of licence involved may incur costs and associated running costs should also be taken into account. The whole lifecycle running cost of any software product, be it OSS or proprietary, should always be considered. Such costs range from the initial feasibility study through to governance, procurement, import, staff training, configuration and deployment, software development, legacy software, patching, technical support, mid-life enhancement and finally, decommissioning / transfer to a new product or supplier.

49. It is important to be aware of such costs when contemplating the adoption of OSS and to make a fair comparison with any proprietary alternatives. There may otherwise be a temptation to decide in favour of a less suitable and potentially riskier software solution than might otherwise have been the case in the mistaken belief that OSS is the guaranteed cost-saving option.

Open Standards

50. OSS and open standards are not the same thing. While there is no one single agreed definition of an open standard, HMG defines these as standards which:

    - Result from and are maintained through an open, independent process;
    - Are approved by a recognised specification or standardisation organisation. (The specification/standardisation must be compliant with Regulation 9 of the Public Contract Regulations 2006);
    - Are thoroughly documented and publicly available at zero cost or low cost;
    - Have intellectual property made irrevocably available on a royalty free basis; and
    - As a whole can be implemented and shared under different development approaches and on a number of platforms (reference [e])

51. Both OSS and proprietary software can, and frequently do, support open standards, such as standards for office automation file formats. Nevertheless,

while the OSS ethos promotes the concept of open standards, you cannot automatically assume that an OSS product will adopt them.

52. The use of open standards has a number of benefits, including avoidance of vendor lock-in to any particular product or service provision and facilitation of system interoperability and information sharing. Data can continue to be accessed and used as new software comes along and there is no dependency on proprietary software, the development and support of which can be discontinued at any time.

Source Available / Shared Source

53. Despite the OSI's definition of OSS, the latter is sometimes used to describe products where the source code is available to view, but is not accessible for modification or redistribution. Such products are more accurately referred to as 'source-available' or 'shared source'.

Open Content

54. Open content describes any kind of creative work, including articles, pictures, audio and video that is published in a format that explicitly allows the copying of the information. The issue of open content can sometimes emerge in conjunction with OSS. As is the case with open standards, OSS and open content are not the same thing. The principles behind the latter are that the opportunities presented by developments in the ICT world should be capitalised upon to maximise the spread of information and knowledge. While this represents a similar kind of ethos to that of OSS, there is no relation to source code, open or otherwise. Probably the best-known example of an open content development is Wikipedia.

Shareware

55. 'Shareware' encompasses a wide range of software solutions, usually distributed under licence. The software can often be used free of charge (with the option of a paid-for version) and can usually be redistributed by the individual. However, the source code cannot be modified to suit ones own purposes. Only OSS provides the user with the freedom to do this.

Free Software

56. OSS and 'free software' are similar, although there are subtle differences. Free software is defined by the Free Software Foundation's (FSF) 'four freedoms' (reference [f]) and seeks to emphasise the users' freedom to copy, modify and redistribute code. The OSI's definition, on the other hand, highlights that the source code is viewable by all and that high-quality software can be developed as a result. OSS and free software share an almost identical set of licences.

Freeware

57. Freeware is cost-free (depending upon licensing conditions) proprietary software provided by a company or organisation. The source code is usually closed.

References

[a] www.opensource.org/about

[b] CESG Good Practice Guide No. 30, Assurance of ICT Systems and Services (UNCLASSIFIED) – latest issue available from the CESG website.

[c] HMG Security Policy Framework, 2011 Tiers 1–3 (NOT PROTECTIVELY MARKED). Available at www.cabinetoffice.gov.uk

[d] HMG Information Assurance Standard No 1 & 2, Information Risk Management (UNCLASSIFIED) – latest issue available from CESG website.

[e] Procurement Policy Note 3/11 31 January 2011 – Use of Open Standards when specifying ICT Requirements. (www.cabinetoffice.gov.uk/sites/default/files/resources/PPN%20311%20Open%20Standards.pdf

[f] www.fsfe.org/about/basics/freesoftware.en.html

Glossary

FLOSS	Free (Libre) Open Source Software
FOSS	Free Open Source Software
FSF	Free Software Foundation
HMG	Her Majesty's Government
IA	Information Assurance
ICT	Information and Communications Technology
IS1	Information Assurance Standard No. 1
ITU	International Telecommunications Union
OSS	Open Source Software
OSI	Open Source Initiative
OSD	Open Source Definition
SAM	Software Asset Management

CESG provides advice and assistance on information security in support of UK Government. Unless otherwise stated, all material published on this website has been produced by CESG and is considered general guidance only. It is not intended to cover all scenarios or to be tailored to particular organisations or individuals. It is not a substitute for seeking appropriate tailored advice.

CESG Enquiries
Hubble Road
Cheltenham
Gloucestershire
GL51 0EX

Tel: +44 (0)1242 709141
Email: enquiries@cesg.gsi.gov.uk