Case No. 18-15189

IN THE UNITED STATES COURT OF APPEALS
FOR THE NINTH CIRCUIT

OPEN SOURCE SECURITY, INC. AND BRADLEY SPENGLER,
Plaintiffs-Appellants,

v.

BRUCE PERENS,
Defendant-Appellee.

ON APPEAL FROM THE UNITED STATES DISTRICT COURT FOR THE
NORTHERN DISTRICT OF CALIFORNIA, NO. 3:17-CV-04002-LB
THE HONORABLE LAUREL BEELER, UNITED STATES MAGISTRATE JUDGE, PRESIDING

APPELLEE'S SUPPLEMENTAL EXCERPTS OF RECORD

Jamie Williams
Aaron Mackey
ELECTRONIC FRONTIER FOUNDATION
815 Eddy Street
San Francisco, California 94109
(415) 436-9333
jamie@eff.org
amackey@eff.org

Melody Drummond Hansen
Heather J. Meeker
O'MELVENY & MEYERS LLP
2765 Sand Hill Road
Menlo Park, California 94025-7019
(650) 473-2600
mdrummondhansen@omm.com
hmeeker@omm.com

Cara L. Gagliano
O'MELVENY & MEYERS LLP
Two Embarcadero Center, 28th Floor
San Francisco, California 94111-3823
(415) 984-8899
cgagliano@omm.com

Counsel for Defendant-Appellee

APPELLEES'
SUPPLEMENTAL EXCERPTS OF RECORD

INDEX

CERTIFICATE OF FILING AND SERVICE

I hereby certify that I electronically filed the foregoing with the Clerk of the Court for the United States Court of Appeals for the Ninth Circuit by using the appellate CM/ECF system on August 15, 2018.

I certify that all participants in the case are registered CM/ECF users and that service will be accomplished by the appellate CM/ECF system.

Dated: August 15, 2018

/s/ *Jamie Williams*
Jamie Williams

Attorney for Appellee Bruce Perens

MELODY DRUMMOND HANSEN (S.B. #278786)
mdrummondhansen@omm.com
HEATHER J. MEEKER (S.B. #172148)
hmeeker@omm.com
O'MELVENY & MYERS LLP
2765 Sand Hill Road
Menlo Park, California 94025-7019
Telephone: +1 650 473 2600
Facsimile: +1 650 473 2601

CARA L. GAGLIANO (S.B. #308639)
cgagliano@omm.com
Two Embarcadero Center
28th Floor
San Francisco, California 94111-3823
Telephone: +1 415 984 8700
Facsimile: +1 415 984 8701

Attorneys for Defendant
Bruce Perens

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO

OPEN SOURCE SECURITY, INC., and BRADLEY SPENGLER,

,

Plaintiffs,

v.

BRUCE PERENS, and Does 1-50,

Defendants.

Case No. 3:17-cv-04002-LB

REQUEST FOR JUDICIAL NOTICE IN SUPPORT OF DEFENDANT BRUCE PERENS'S NOTICE OF MOTION AND SPECIAL MOTION TO STRIKE THE FIRST AMENDED COMPLAINT PURSUANT TO THE CALIFORNIA ANTI-SLAPP STATUTE, CODE OF CIV. PROC. SEC. 425.16, AND MOTION TO DISMISS THE FIRST AMENDED COMPLAINT WITH PREJUDICE PURSUANT TO FED. R. CIV. P. 12(b)(6)

Hearing Date: December 14, 2017
Time: 9:30 a.m.
Location: Courtroom C, 15th Floor
Judge: Hon. Laurel Beeler

REQUEST FOR JUDICIAL NOTICE

I. INTRODUCTION

Pursuant to Rule 201 of the Federal Rules of Evidence, Defendant Bruce Perens respectfully requests that the Court take judicial notice of the following documents submitted in support of Mr. Perens's Motion to Dismiss and Special Motion to Strike Plaintiffs First Amended Complaint:

1. An August 8, 2016 memorandum issued by the Executive Office of the President, Office of Management and Budget titled "Federal Source Code Policy: Achieving Efficiency, Transparency, and Innovation through Reusable and Open Source Software." A true and correct copy of the memorandum, downloaded from https://obamawhitehouse.archives.gov/sites/default/files/omb/memoranda/2016/m_16_21.pdf, is attached hereto as **Exhibit 1**.

2. An October 16, 2009 memorandum issued by the United States Department of Defense Chief Information Officer titled "Clarifying Guidance Regarding Open Source Software (OSS)." A true and correct copy of the memorandum, downloaded from http://dodcio.defense.gov/Portals/0/Documents/OSSFAQ/2009OSS.pdf, is attached hereto as **Exhibit 2**.

3. A webpage published and maintained by the United States Department of Defense Chief Information Officer titled "DoD Open Source Software (OSS) FAQ," available at http://dodcio.defense.gov/Open-Source-Software-FAQ/. A true and correct copy of a printout of the webpage is attached hereto as **Exhibit 3**.

II. ARGUMENT

In ruling on a motion to dismiss under Federal Rule of Civil Procedure 12(b)(6), a court may consider matters outside the pleadings if they are properly subject to judicial notice. *See MGIC Indem. Corp. v. Weisman*, 803 F.2d 500, 504 (9th Cir. 1986). Here, the Court should take judicial notice of the U.S. government publications attached as Exhibits 1 through 3, which provide useful background information about open source software and the Linux Operating

System. Federal courts are entitled to take judicial notice of facts outside the record—like those in Exhibits 1 through 3—that "can be accurately and readily determined from sources whose accuracy cannot reasonably be questioned." *See* Fed. R. Evid. 201(b). Official government documents, including materials on official government websites, fall within this category. *See Paralyzed Veterans of Am. v. McPherson*, No. C064670SBA, 2008 WL 4183981, at *5–*6 (N.D. Cal. Sept. 9, 2008); *Hansen Beverage Co. v. Innovation Ventures, LLC*, No. 08-CV-1166-IEG POR, 2009 WL 6597891, at *2 (S.D. Cal. Dec. 23, 2009); *Vaquero Energy, Inc. v. Herda*, No. 1:15-CV-0967-JLT, 2015 WL 5173535, at *4–*5 (E.D. Cal. Sept. 3, 2015).

III. CONCLUSION

Based on the foregoing, Defendant respectfully requests that the Court take judicial notice of Exhibits 1-3.

Dated: October 31, 2017

MELODY DRUMMOND HANSEN
HEATHER J. MEEKER
CARA L. GAGLIANO
O'MELVENY & MYERS LLP

By: */s/ Melody Drummond Hansen*
Melody Drummond Hansen
Attorneys for Defendant
Bruce Perens

Exhibit 1

EXECUTIVE OFFICE OF THE PRESIDENT
OFFICE OF MANAGEMENT AND BUDGET
WASHINGTON, D.C. 20503

August 8, 2016

M-16-21

MEMORANDUM FOR THE HEADS OF DEPARTMENTS AND AGENCIES

FROM: **Tony Scott**
United States Chief Information Officer

Anne E. Rung
United States Chief Acquisition Officer

SUBJECT: **Federal Source Code Policy: Achieving Efficiency, Transparency, and Innovation through Reusable and Open Source Software**

The U.S. Government is committed to improving the way Federal agencies buy, build, and deliver information technology (IT) and software solutions to better support cost efficiency, mission effectiveness, and the consumer experience with Government programs. Each year, the Federal Government spends more than $6 billion on software through more than 42,000 transactions.[1] A significant proportion of software used by the Government is comprised of either preexisting Federal solutions or commercial solutions. These solutions include proprietary, open source, and mixed source[2] code and often do not require additional custom code development.

When Federal agencies are unable to identify an existing Federal or commercial software solution that satisfies their specific needs, they may choose to develop a custom software solution on their own or pay for its development. When agencies procure custom-developed source code, however, they do not necessarily make their new code (source code or code) broadly available for Federal Government-wide reuse. Even when agencies are in a position to make their source code available on a Government-wide basis, they do not make such code available to other agencies in a consistent manner. In some cases, agencies may even have difficulty establishing that the software was produced in the performance of a Federal Government contract. These challenges may result in duplicative acquisitions for substantially similar code and an inefficient use of taxpayer dollars.

This policy seeks to address these challenges by ensuring that new custom-developed Federal source code be made broadly available for reuse across the Federal Government.[3] This is

[1] *M-16-12: Improving the Acquisition and Management of Common Information Technology: Software Licensing.* Office of Mgmt. & Budget, Exec. Office of the President, June 2, 2016. https://www.whitehouse.gov/sites/default/files/omb/memoranda/2016/m-16-12_1.pdf.

[2] *See* Appendix A for definitions of key technical terms used throughout this policy document.

[3] *See* Section 6 of this policy for additional information about limited exceptions.

consistent with the *Digital Government Strategy's* "Shared Platform" approach, which enables Federal employees to work together—both within and across agencies—to reduce costs, streamline development, apply uniform standards, and ensure consistency in creating and delivering information.[4] Enhanced reuse of custom-developed code across the Federal Government can have significant benefits for American taxpayers, including decreasing duplicative costs for the same code and reducing Federal vendor lock-in.[5]

This policy also establishes a pilot program that requires agencies, when commissioning new custom software, to release at least 20 percent of new custom-developed code as Open Source Software (OSS) for three years, and collect additional data concerning new custom software to inform metrics to gauge the performance of this pilot.[6]

While the benefits of enhanced Federal custom-developed code reuse are significant, additional benefits can accrue when source code is also made available to the public as OSS. Making source code available as OSS can enable continual improvement of Federal custom-developed code projects as a result of a broader user community implementing the code for its own purposes and publishing improvements. This collaborative atmosphere can make it easier to conduct software peer review and security testing, to reuse existing solutions, and to share technical knowledge.[7] Furthermore, vendors participating in or competing for future maintenance or enhancement can do so with full knowledge of the underlying source code. A number of private sector companies have already shifted some of their software development projects to an OSS model, in which the source code of the software is made broadly available to the public for inspection, improvement, and reuse.

Several Federal agencies and component organizations have also begun publishing custom-developed code as OSS or without any restriction on use. Some of these include:

- The White House: "We the People" is a White House service that allows the American people to easily and interactively petition their Government. The source code for this website is freely available as OSS;[8]

[4] *Digital Government: Building A 21st Century Platform To Better Serve The American People*, Office of Mgmt. & Budget, Exec. Office of the President, May 23, 2012. https://www.whitehouse.gov/sites/default/files/omb/egov/digital-government/digital-government.html.

[5] "Vendor lock-in" refers to a situation in which the customer depends on a single supplier for a product and cannot easily move to another vendor without sustaining substantial cost or inconvenience. Vendor lock-in can potentially raise costs and stifle innovation and it can result in reduced competition on future related software acquisitions.

[6] *Clinger Cohen Act of 1996*. 40 U.S.C. §§ 11301-11303.

[7] Department of Defense Chief Information Officer. *Clarifying Guidance Regarding Open Source Software (OSS)*. October 16, 2009. "The continuous and broad peer-review enabled by publicly available source code supports software reliability and security efforts through the identification and elimination of defects that might otherwise go unrecognized by a more limited core development team." http://dodcio.defense.gov/Portals/0/Documents/FOSS/2009OSS.pdf.

[8] "We the People" petitions are accessible at https://petitions.whitehouse.gov/. The source code for "We the People" is available at https://github.com/WhiteHouse/petitions.

- 18F[9] and the Consumer Financial Protection Bureau (CFPB):[10] Both of these organizations have policies that establish a default position to publish source code that is custom-developed by or for the organization. For example, both organizations contribute to the source code for the eRegulations platform,[11] a web-based interface for public viewing and commenting on proposed changes to Federal regulations. The eRegulations platform, which originated at CFPB, is being used by other Federal agencies[12] and continues to be improved based on public feedback;[13]

- The Department of Education: This agency's "College Scorecard" is a citizen-facing OSS website and accompanying application programming interface (API) that provides free tools to help potential students make informed decisions about which colleges or universities to attend;[14] and

- The Department of Defense (DOD): This agency issued a memorandum[15] in 2009 that, among other things, describes the many benefits of OSS that should be considered when conducting market research on software for DOD use.[16]

The Administration made a commitment, as part of its *Second Open Government National Action Plan,*[17] to "develop an open source software policy that, together with the Digital Services Playbook, will support improved access to custom software code developed for the Federal

[9] 18F (https://18f.gsa.gov/) is a digital services delivery team within the General Services Administration. The 18F Open Source Policy is described at https://18f.gsa.gov/2014/07/29/18f-an-open-source-team/ and can be accessed at https://github.com/18F/open-source-policy/blob/master/policy.md.

[10] CFPB's source code policy is described at http://www.consumerfinance.gov/blog/the-cfpbs-source-code-policy-open-and-shared/ and can be accessed at https://cfpb.github.io/source-code-policy/.

[11] "eRegulations," CFPB's platform to read regulations, is accessible at http://www.consumerfinance.gov/eregulations/.

[12] The Bureau of Alcohol, Tobacco, Firearms, and Explosives (ATF) has adopted a beta version of "eRegulations," accessible at https://atf-eregs.18f.gov/.

[13] The publically accessible open source repository for submitting comments and proposing improvements to the "eRegulations" platform is accessible at https://github.com/eregs/notice-and-comment. 18F also developed https://analytics.usa.gov—jointly with the U.S. Digital Service—to provide a window into how people are interacting with the Federal Government online and made the source code available online (https://github.com/18F/analytics-reporter). The cities of Philadelphia, PA (http://analytics.phila.gov/) and Boulder, CO (https://bouldercolorado.gov/stats) were able to reuse the code to provide their own citizens with real-time information on how city government websites serve citizens.

[14] The Department of Education's College Scorecard is accessible at https://collegescorecard.ed.gov/. The open source repository for the website and API that runs the College Scorecard is available via 18F's GitHub repository, accessible at https://github.com/18F/college-choice.

[15] Department of Defense Chief Information Officer. *Clarifying Guidance Regarding Open Source Software (OSS).* October 16, 2009. http://dodcio.defense.gov/Portals/0/Documents/OSSFAQ/2009OSS.pdf

[16] The Department of Defense's OSS FAQ states that "continuous and broad peer-review, enabled by publicly available source code, improves software reliability and security through the identification and elimination of defects that might otherwise go unrecognized." *Frequently Asked Questions regarding Open Source Software (OSS) and the Department of Defense (DoD)*, accessible at https://dodcio.defense.gov/OpenSourceSoftwareFAQ.aspx.

[17] *The Open Government Partnership: Announcing New Open Government Initiatives as part of the Second Open Government National Action Plan for The United States of America*. September 2014. Page 2. https://www.whitehouse.gov/sites/default/files/microsites/ostp/new_nap_commitments_report_092314.pdf.

government." [18] This policy fulfills that commitment in an effort to improve U.S. Government software development and make the Government more open, transparent, and accessible to the public.

1. Objectives

This policy will accomplish the following objectives:

- Provide a policy to agencies[19] on considerations that must be made prior to acquiring any custom-developed code;
- Require agencies to obtain appropriate Government data rights to custom-developed code, including at a minimum, rights to Government-wide reuse and rights to modify the code. Agencies shall make such custom-developed code broadly available across the Federal Government, subject to limited exceptions;[20]
- Require agencies to consider the value of publishing custom code as OSS;
- Establish requirements for releasing custom-developed source code, including securing the rights necessary to make some custom-developed code releasable to the public as OSS under this policy's new pilot program; and
- Provide instructions and resources to facilitate implementation of this policy.

2. Scope and Applicability

The requirements outlined in this policy apply to source code that is custom-developed for the Federal Government, subject to the limited exceptions outlined in Section 6 of this document. Source code developed for National Security Systems (NSS), as defined in 40 U.S.C. § 11103, is exempt from the requirements of this policy. For NSS, agencies shall follow applicable statutes, Executive Orders, directives, and internal agency policies.

The policies in this document do not apply retroactively (*i.e.*, they do not require that existing custom-developed code be retroactively made available for Government-wide reuse or as OSS). However, making such code available for Government-wide reuse or as OSS, to the extent practicable, is strongly encouraged.

---

[18] The Digital Services Playbook was developed by the U.S. Digital Service and consists of key "plays" that can help the Government build effective digital services. It encourages agencies to "default to open" and seek contracts that specify that "software and data generated by third parties remains under [the U.S. Government's] control, and can be reused and released to the public as appropriate and in accordance with the law." It also requires an explanation "[i]f the codebase has not been released under an open source license." https://playbook.cio.gov/.

[19] For the purposes of this policy, an agency is one that meets the definition of executive agency under the Clinger Cohen Act of 1996. *See* Appendix A.

[20] *See* Section 6 of this policy for additional information about limited exceptions.

The agencies' Chief Information Officers (CIO), Chief Acquisition Officers (CAO), and other key stakeholders should promptly begin working together to implement this policy. Agencies are expected to issue internal policies, as necessary, to support these efforts and should expect their progress to be evaluated in accordance with accountability mechanisms described in Section 7.

3. Three-Step Software Solutions Analysis

Agencies must obtain sufficient rights to custom-developed code to fulfill both the Government-wide reuse objectives and the open source release objectives outlined in this policy's pilot program.

In meeting their software needs, agencies must conduct the three-step analysis outlined below. This analysis is intended to leverage existing solutions—consistent with principles of category management[21] and shared services[22]—and suitable commercial solutions, while mitigating duplicative spending on custom-developed software solutions. These steps are consistent with the Office of Management and Budget's (OMB) long-standing policy on investments in major information systems.[23] Moreover, consistent with OMB's memorandum on Technology Neutrality,[24] agencies must consider open source, mixed source, and proprietary software solutions equally and on a level playing field, and free of preconceived preferences based on how the technology is developed, licensed, or distributed.

- ***Step 1 (Conduct Strategic Analysis and Analyze Alternatives)*:** Each agency must conduct research and analysis prior to initiating any technology acquisition or custom code development. The strategic analysis should consider not only agency mission and operational needs, but also external public initiatives and interagency initiatives such as Cross-Agency Priority Goals. Having conducted the strategic analysis, agencies shall then conduct an alternatives analysis, evaluating whether to use an existing Federal software solution or to acquire or develop a new software solution. The alternatives analysis shall give preference to the use of an existing Federal software solution.[25]

- ***Step 2 (Consider Existing Commercial Solutions)*:** If an agency's alternatives analysis concludes that existing Federal software solutions cannot efficiently and effectively meet

[21] *See Transforming the Marketplace: Simplifying Federal Procurement to Improve Performance, Drive Innovation, and Increase Savings*, Office of Mgmt. & Budget, Exec. Office of the President, December 4, 2014. https://www.whitehouse.gov/sites/default/files/omb/procurement/memo/simplifying-federal-procurement-to-improve-performance-drive-innovation-increase-savings.pdf.

[22] *M-16-11: Improving Administrative Functions Through Shared Services*, Office of Mgmt. & Budget, Exec. Office of the President, May 4, 2016. https://www.whitehouse.gov/sites/default/files/omb/memoranda/2016/m-16-11.pdf.

[23] *See OMB Circular No. A-11, Appendix J – Principles of Budgeting for Capital Asset Acquisitions*, Office of Mgmt. & Budget, Exec. Office of the President, July 1, 2016. https://www.whitehouse.gov/sites/default/files/omb/assets/a11_current_year/app_j.pdf.

[24] *Technology Neutrality*, Office of Mgmt. & Budget, Exec. Office of the President, January 7, 2011. https://www.whitehouse.gov/sites/default/files/omb/assets/egov_docs/memotociostechnologyneutrality.pdf.

[25] Existing Federal software solutions are those for which appropriate rights are already held by the Government, which may include commercial or custom-developed software solutions.

the needs of the agency, the agency must explore whether its requirements can be satisfied with an appropriate commercially-available solution.[26]

- ***Step 3 (Consider Custom Development):*** If an agency's alternatives analysis concludes that an existing Federal software solution or commercial solution cannot adequately satisfy its needs, the agency may consider procuring custom-developed code in whole or in conjunction with existing Federal or commercial code. When commissioning new custom-developed software, agencies must consider the value of publishing custom code as OSS and negotiate data rights reflective of its value-consideration. Agencies must also obtain sufficient rights to fulfill this policy's objectives related to Government-wide code reuse and the open source pilot program.

Agencies must also consider several factors throughout each stage of the three-step analysis:

A. Hybrid Solutions: Solutions containing a mixture of existing Federal, commercial, and/or custom-developed solutions should be considered throughout each step of the analysis.

B. Modular Architecture: Agencies should consider modular approaches to solution architecture. As discussed in the *Digital Government Strategy*, modularity can reduce overall risk and cost while increasing interoperability and technical flexibility.

C. Cloud Computing: Consistent with OMB strategy, agencies are encouraged to evaluate safe and secure cloud computing options throughout each step of the analysis.[27]

D. Open Standards: Regardless of the specific solution selected, all software procurements and Government software development projects should consider utilizing open standards whenever practicable in order to increase the interoperability of all Government software solutions. Open standards enable software to be used by anyone at any time, and can spur innovation and growth regardless of the technology used for implementation—be it proprietary, mixed source, or OSS in nature.

E. Targeted Considerations: Agencies must select a software solution that best meets the operational and mission needs of the agency, taking into consideration factors such as performance, total life-cycle cost of ownership, security and privacy protections, interoperability, ability to share or reuse, resources required to later switch vendors, and availability of quality support. These considerations should be taken into account during all three steps of the analysis.

[26] Preference must first be given to procurement of existing commercial solutions through best-in-class vehicles identified by category management policies.

[27] *Federal Cloud Computing Strategy*, Office of Mgmt. & Budget, Exec. Office of the President, February 8, 2011. https://www.whitehouse.gov/sites/default/files/omb/assets/egov_docs/federal-cloud-computing-strategy.pdf.

4. Government-Wide Code Reuse

Ensuring Government-wide reuse rights for custom code that is developed using Federal funds has numerous benefits for American taxpayers. To realize these benefits, agencies must comply with the following requirements:

A. Secure Rights for Government Reuse and Ensure Delivery of Source Code

Agencies that enter into contracts for the custom development of software shall—at a minimum—acquire and enforce rights sufficient to enable Government-wide reuse of custom-developed code. Agencies must ensure appropriate contract administration and use of best practices to secure the full scope of the Government's rights, including—but not limited to—sharing and using the code with other Federal agencies.

Additionally, in order to ensure the ability to exercise these rights, agencies must use best practices to ensure delivery of the custom-developed code, documentation, and other associated materials from the developer throughout the development process.

B. Inventory All Custom-Developed Code and Make It Available Government-Wide

Securing adequate rights to enable Government-wide reuse of custom-developed code is a critical first step in gaining efficiencies in Federal software purchasing; however, without broad and consistent dissemination of the code across the Federal Government, these efficiencies cannot be fully realized. Therefore, in addition to securing the rights discussed above, agencies shall do the following:

i. Maintain a Code Inventory: As part of their broader responsibility to maintain an up-to-date inventory of agency information resources, agencies shall make custom-developed code and related information available to all other Federal agencies[28] by creating and maintaining an enterprise code inventory that lists all new code that is custom-developed for the Federal Government; and

ii. Make Custom-Developed Code Available: Agencies shall make custom-developed code available for Government-wide reuse and make their code inventories discoverable at https://www.code.gov ("Code.gov"), pursuant to the limited exceptions outlined in Section 6 of this policy.

Agencies may refer to Section 7 of this document for additional information regarding their individual responsibilities related to implementing this policy.

[28] *See* Section 6 of this policy for additional information about limited exceptions.

5. Open Source Software

5.1 Pilot Program: Publication of Custom-Developed Code as OSS

Each agency shall release as OSS at least 20 percent of its new custom-developed code[29] each year for the term of the pilot program. As discussed above, agencies must obtain sufficient rights to custom-developed code to fulfill the open source release objectives of this policy's pilot program.

When deciding which custom-developed code projects to release, each agency should prioritize the release of custom-developed code that it considers potentially useful to the broader community. Agencies should calculate the percentage of source code released using a consistent measure—such as real or estimated lines of code, number of self-contained modules, or cost—that meets the intended objectives of this requirement. Additional information regarding how best to measure source code will be provided on Code.gov.

Although the minimum requirement for OSS release is 20 percent of custom-developed code, agencies are strongly encouraged to release as much custom-developed code as possible to further the Federal Government's commitment to transparency, participation, and collaboration.

OMB expects all agencies to satisfy the requirements of this pilot program without exception. Agencies should—as part of their selection of custom-developed code to be released as OSS—refrain from selecting code that would fall under the exceptions outlined in Section 6 of this policy. In the event that an agency's CIO believes that the agency cannot satisfy the 20 percent requirement of the OSS pilot program (*e.g.*, because releasing code as OSS would create an identifiable risk to the detriment of national security), the CIO should consult with OMB.

Unless extended or supplanted by OMB through the issuance of further policy, the pilot program under this sub-section will expire three years (36 months) after the publication date of this policy; however, the rest of the Federal Source Code Policy will remain in effect. No later than two years after the publication date of this policy, OMB shall evaluate pilot results and consider whether to allow the pilot program to expire or to issue a subsequent policy to continue, modify, or increase the minimum requirements of the pilot program.

Within 120 days of the publication date of this policy, OMB shall develop metrics to assess the impact of the pilot program. Additional information on these topics will be available on Code.gov.

5.2 Participation in the Open Source Community

When agencies release custom-developed source code as OSS to the public, they should develop and release the code in a manner that (1) fosters communities around shared challenges, (2) improves the ability of the OSS community to provide feedback on, and make contributions to, the source code, and (3) encourages Federal employees and contractors to contribute back to the

[29] The definition of "custom-developed code" can be found in Appendix A.

broader OSS community by making contributions to existing OSS projects. In furtherance of this strategy, agencies should comply with the following principles:

A. Leverage Existing Communities: Whenever possible, teams releasing custom-developed code to the public as OSS should appropriately engage and coordinate with existing communities relevant to the project. Government agencies should only develop their own communities when existing communities do not satisfy their needs.

B. Engage in Open Development: Software that is custom-developed for or by agencies should, to the extent possible and appropriate, be developed using open development practices. These practices provide an environment in which OSS can flourish and be repurposed. This principle, as well as the one below for releasing source code, include distributing a minimum viable product as OSS; engaging the public before official release;[30] and drawing upon the public's knowledge to make improvements to the project.

C. Adopt a Regular Release Schedule: In instances where software cannot be developed using open development practices, but is otherwise appropriate for release to the public, agencies should establish an incremental release schedule to make the source code and associated documentation available for public use.

D. Engage with the Community: Similar to the requirement in the Administration's *Open Data Policy*, agencies should create a process to engage in two-way communication with users and contributors to solicit help in prioritizing the release of source code and feedback on the agencies' engagement with the community.

E. Consider Code Contributions: One of the potential benefits of OSS lies within the communities that grow around OSS projects, whereby any party can contribute new code, modify existing code, or make other suggestions to improve the software throughout the software development lifecycle. Communities help monitor changes to code, track potential errors and flaws in code, and other related activities. These kinds of contributions should be anticipated and, where appropriate, considered for integration into custom-developed Government software or associated materials.

F. Documentation: It is important to provide OSS users and contributors with adequate documentation of source code in an effort to facilitate use and adoption. Agencies must ensure that their repositories include enough information to allow reuse and participation by third parties. In participating in community-maintained repositories, agencies should follow community documentation standards. At a minimum, OSS repositories maintained by agencies must include the following information:
    i. Status of software (*e.g.*, prototype, alpha, beta, release, etc.);
    ii. Intended purpose of software;

[30] For the purposes of this policy, an "official release" is a release that is not in the alpha or beta test phases and, in the field of computer programming, would typically be designated with a version number 1.0.

iii. Expected engagement level (*i.e.*, how frequently the community can expect agency activity);
iv. License details; and
v. Any other relevant technical details on how to build, make, install, or use the software, including dependencies (if applicable).

6. Exceptions to Government Code Reuse

The exceptions provided below may be applied, in specific instances, to exempt an agency from sharing custom-developed code with other Government agencies. These exceptions do not apply to the OSS pilot program.[31] Any exceptions used must be approved and documented by the agency's CIO for the purposes of ensuring effective oversight and management of information technology resources.

Applicable exceptions are as follows:

1. The sharing of the source code is restricted by law or regulation, including—but not limited to—patent or intellectual property law, the Export Asset Regulations, the International Traffic in Arms Regulation, and the Federal laws and regulations governing classified information;

2. The sharing of the source code would create an identifiable risk to the detriment of national security, confidentiality of Government information, or individual privacy;

3. The sharing of the source code would create an identifiable risk to the stability, security, or integrity of the agency's systems or personnel;

4. The sharing of the source code would create an identifiable risk to agency mission, programs, or operations; or

5. The CIO believes it is in the national interest to exempt sharing the source code.

For excepted software, agencies must provide OMB a brief narrative justification for each exception, with redactions as appropriate.

7. Implementation

7.1 Roles and Responsibilities

The Federal Information Technology Acquisition Reform Act (FITARA)[32] creates clear responsibilities for agency CIOs related to IT investments and planning, as well as requiring that agency CIOs be involved in the IT acquisition process. OMB's FITARA implementation

[31] *See* Section 5 for additional information regarding the pilot program.
[32] FITARA was codified as part of the *National Defense Authorization Act for Fiscal Year 2015* (Title VIII, Subtitle D, H.R. 3979); accessible at https://www.congress.gov/bill/113th-congress/house-bill/3979.

guidance[33] established a "common baseline" for roles, responsibilities, and authorities of the agency CIO and the roles of other applicable Senior Agency Officials[34] in managing IT as a strategic resource. Accordingly, agency heads must ensure that CIOs and Senior Agency Officials, including CAOs, are positioned with the responsibility and authority necessary to implement the requirements of this policy. As appropriate, Senior Agency Officials should also work with the agency's public affairs staff, open government staff, web manager or digital strategist, program owners, and other leadership to properly identify, publish, and collaborate with communities on their OSS projects.

Moreover, in support of the objectives and requirements of this policy, agencies should strengthen internal capacity to efficiently and securely deliver OSS as part of regular operations. Additional information on this topic will be provided on Code.gov.

7.2 Code Inventories and Discovery

Inventories are a means of discovering information such as the functionality and location of potentially reusable or releasable custom-developed code. Within 120 days of the publication date of this policy, each agency must update—and thereafter keep up to date—its inventory of agency information resources to include an enterprise code inventory that lists custom-developed code for or by the agency after the publication of this policy. Each agency's inventory will be reflected on Code.gov. The inventory will indicate whether the code is available for Federal reuse, is available publicly as OSS, or cannot be made available due to a specific exception listed in this policy. Agencies shall fill out this information based on a metadata schema that OMB will provide on Code.gov.

7.3 Code.gov

Within 90 days of the publication date of this policy, the Administration will launch https://www.code.gov,[35] an online collection of tools, best practices, and schemas to help agencies implement this policy. The website will include additional materials such as definitions, evaluation metrics, checklists, case studies, and model contract language—with the goal of enabling collaboration across the Federal Government and advancing the Government's partnership with the public.

Additionally, Code.gov will serve as the primary discoverability portal for custom-developed code intended both for Government-wide reuse and for release as OSS. Note that Code.gov is not intended to house the custom-developed code itself; rather, it is intended to serve as a tool for discovering custom-developed code that may be available for Government-wide reuse or as OSS,

[33] *M-15-14: Management and Oversight of Federal Information Technology*, Office of Mgmt. & Budget, Exec. Office of the President, June 10, 2015. https://www.whitehouse.gov/sites/default/files/omb/memoranda/2015/m-15-14.pdf.

[34] Senior Agency Officials include positions that may include the Chief Acquisition Officer, Chief Operating Officer, Chief Financial Officer, Chief Technology Officer, Chief Data Officer, Senior Agency Official for Privacy, Chief Information Security Officer, and Program Manager.

[35] Code.gov will be modeled after Data.gov (https://www.data.gov) and Project Open Data (https://project-open-data.cio.gov/).

and to provide transparency into custom-developed code that is developed using Federal funds. This discoverability portal will be publically accessible and searchable via a variety of fields and constraints, such as the name of the project, its intended use, and the agency releasing the source code. Code.gov will evolve over time as a community resource to facilitate the adoption of good custom source code development, sharing, and reuse practices.

7.4 Code Repositories

Accessible, buildable, version-controlled repositories for the storage, discussion, and modification of custom-developed code are critical to both the Government-wide reuse and OSS pilot program sections of this policy. Agencies should utilize existing code repositories and common third-party repository platforms as necessary in order to satisfy the requirements of this policy.[36] Code.gov will contain additional information on this topic.

7.5 Licensing

Licensing is a critical component of OSS and can affect how the source code can be used and modified. Accordingly, when agencies release custom-developed code as OSS, they shall append appropriate OSS licenses to the source code. Additional information on licensing will be available on Code.gov.

7.6 Agency Policy

Within 90 days of the publication date of this policy, each agency's CIO—in consultation with the agency's CAO—shall develop an agency-wide policy that addresses the requirements of this document. For example, the policy should address how the agency will ensure that an appropriate alternatives analysis has been conducted before considering the acquisition of an existing commercial solution or a custom-developed solution. In accordance with OMB guidance,[37] these policies will be posted publicly. Moreover, within 90 days of the publication date of this policy, each agency's CIO office must correct or amend any policies that are inconsistent with the requirements of this document, including the correction of policies that automatically treat OSS as noncommercial software.

7.7 Accountability Mechanisms

Progress on agency implementation of this policy will be primarily assessed by OMB through an analysis of each agency's internal Government repositories, public OSS repositories, and code inventories on Code.gov, as well as data obtained through the quarterly Integrated Data

[36] Covered agencies should ensure access to these services. *See M-10-23: Guidance for Agency Use of Third-Party Websites and Applications,* Office of Mgmt. & Budget, Exec. Office of the President, June 25, 2010. https://www.whitehouse.gov/sites/default/files/omb/assets/memoranda_2010/m10-23.pdf.

[37] *See M-15-14: Management and Oversight of Federal Information Technology*, Office of Mgmt. & Budget, Exec. Office of the President, June 10, 2015. https://www.whitehouse.gov/sites/default/files/omb/memoranda/2015/m-15-14.pdf. This requires that IT policies be posted publicly at https://[agency].gov/digitalstrategy, and included as a downloadable dataset in the agency's Public Data Listing.

Collection (IDC), quarterly PortfolioStat sessions, the IT Dashboard, and additional mechanisms to be provided via Code.gov.[38]

[38] PortfolioStat is the core oversight tool used by OFCIO to improve both the efficiency and effectiveness of Federal IT. PortfolioStat's principle objectives are to serve as an overview of each agency's portfolio of IT investments and to oversee execution of OFCIO and OMB-wide policy. For information on the IT Dashboard, see https://itdashboard.gov/.

Appendix A: Definitions

Agency: For the purposes of this policy, an agency is one that meets the definition of executive agency under the Clinger Cohen Act of 1996. *See* 41 U.S.C. § 11101.

Code.gov: This platform is primarily intended to serve two distinct functions. First, it will act as an online collection of tools, guides, and best practices specifically designed to help agencies implement the framework presented in this policy. Second, it will serve as the primary discoverability portal for custom-developed code intended both for Government-wide reuse and for potential release as OSS. Code.gov is not intended to house the custom-developed code itself; rather, it is intended to serve as a tool for discovering custom-developed code that may be available for Government-wide reuse or as OSS, and to provide transparency into custom-developed code that is developed using Federal funds. This discoverability portal will be publically accessible and searchable via a variety of fields and constraints, such as the name of the project, its intended use, and the agency releasing the source code. Code.gov will be accessible at https://www.code.gov and will evolve over time as a community resource to facilitate the adoption of good custom source code development, sharing, and reuse practices.

Custom-Developed Code: For the purposes of this policy, custom-developed code is code that is first produced in the performance of a Federal contract or is otherwise fully funded by the Federal Government. It includes code, or segregable portions of code, for which the Government could obtain unlimited rights under Federal Acquisition Regulations (FAR) Pt. 27 and relevant agency FAR Supplements. Custom-developed code also includes code developed by agency employees as part of their official duties. For the purposes of this policy, custom-developed code may include, but is not limited to, code written for software projects, modules, plugins, scripts, middleware, and APIs; it does not, however, include code that is truly exploratory or disposable in nature, such as that written by a developer experimenting with a new language or library.

Mixed Source Software: A mixed source software solution incorporates both open source and proprietary code.

Open Source Software (OSS): Software that can be accessed, used, modified, and shared by anyone. OSS is often distributed under licenses that comply with the definition of "Open Source" provided by the Open Source Initiative (https://opensource.org/osd) and/or that meet the definition of "Free Software" provided by the Free Software Foundation (https://www.gnu.org/philosophy/free-sw.html).

Proprietary Software: Software with intellectual property rights that are retained exclusively by a rights holder (*e.g.*, an individual or a company).

Software: Refers to (i) computer programs that comprise a series of instructions, rules, routines, or statements, regardless of the media in which recorded, that allow or cause a computer to perform a specific operation or series of operations; and (ii) recorded information comprising source code listings, design details, algorithms, processes, flow charts, formulas, and related

material that would enable the computer program to be produced, created, or compiled. Software does not include computer databases or computer software documentation.[39]

Source Code: Computer commands written in a computer programming language that is meant to be read by people. Generally, source code is a higher level representation of computer commands as they are written by people and, therefore, must be assembled or compiled before a computer can execute the code as a program.

[39] As "computer software" is defined in 48 C.F.R. § 2.101. https://www.gpo.gov/fdsys/pkg/CFR-2002-title48-vol1/pdf/CFR-2002-title48-vol1-sec2-101.pdf.

Exhibit 2

DEPARTMENT OF DEFENSE
6000 DEFENSE PENTAGON
WASHINGTON, DC 20301-6000

CHIEF INFORMATION OFFICER

OCT 16 2009

MEMORANDUM FOR SECRETARIES OF THE MILITARY DEPARTMENTS
CHAIRMAN OF THE JOINT CHIEFS OF STAFF
UNDER SECRETARIES OF DEFENSE
DEPUTY CHIEF MANAGEMENT OFFICER
COMMANDERS OF THE COMBATANT COMMANDS
ASSISTANT SECRETARIES OF DEFENSE
GENERAL COUNSEL OF THE DEPARTMENT OF DEFENSE
DIRECTOR, OPERATIONAL TEST AND EVALUATION
INSPECTOR GENERAL OF THE DEPARTMENT OF DEFENSE
ASSISTANTS TO THE SECRETARY OF DEFENSE
DIRECTOR, ADMINISTRATION AND MANAGEMENT
DIRECTOR, COST ASSESSMENT AND PROGRAM EVALUATION
DIRECTOR, NET ASSESSMENT
DIRECTORS OF THE DEFENSE AGENCIES
DIRECTORS OF THE DOD FIELD ACTIVITIES
CHIEF INFORMATION OFFICERS OF THE MILITARY DEPARTMENTS

SUBJECT: Clarifying Guidance Regarding Open Source Software (OSS)

References: See Attachment 1

To effectively achieve its missions, the Department of Defense must develop and update its software-based capabilities faster than ever, to anticipate new threats and respond to continuously changing requirements. The use of Open Source Software (OSS) can provide advantages in this regard. This memorandum provides clarifying guidance on the use of OSS and supersedes the previous DoD CIO memorandum dated May 28, 2003 (reference (a)).

Open Source Software is software for which the human-readable source code is available for use, study, reuse, modification, enhancement, and redistribution by the users of that software. In other words, OSS is software for which the source code is "open."

There are many OSS programs in operational use by the Department today, in both classified and unclassified environments. Unfortunately, there have been misconceptions and misinterpretations of the existing laws, policies and regulations that deal with software and apply to OSS, that have hampered effective DoD use and development of OSS. Attachment 2 contains clarifying guidance to address some of those issues.

I have asked the Director, Enterprise Services & Integration, to work with your staffs and identify other barriers to the effective use of open source software within the Department, so we can continue to increase the benefits from the use of OSS. Additional information to clarify how existing DoD policies relate to open source software will be posted at http://www.defenselink.mil/cio-nii/cio/oss/. Questions concerning this memorandum should be directed to Daniel Risacher, Enterprise Services & Integration, at (703) 602-1098 or email, Daniel.Risacher@osd.mil.

David M. Wennergren
Performing the Duties of the
ASD(NII)/DoD CIO

Attachments:
As stated

ATTACHMENT 1

REFERENCES

(a) DoD Chief Information Officer (CIO) Memorandum, "Open Source Software (OSS) in the Department of Defense (DoD)," May 28, 2003 (superseded)
(b) Title 10, United States Code (USC), Section 2377
(c) Federal Acquisition Regulation (FAR), Sections 2.101, 12.000, 12.101
(d) Defense Federal Acquisition Regulation Supplement (DFARS), Section 227.7203-5
(e) Title 41, United States Code (USC), Section 253a
(f) Federal Acquisition Regulation (FAR), Section 10.001
(g) DoD Instruction 8500.2, "Information Assurance (IA) Implementation," February 6, 2003
(h) DoD Directive 8320.02, "Data Sharing in a Net-Centric Department of Defense," December 2, 2004
(i) DoD Directive 5230.24, "Distribution Statements on Technical Documents," March 18, 1987

ATTACHMENT 2

CLARIFYING GUIDANCE REGARDING OPEN SOURCE SOFTWARE (OSS)

1. GENERAL. This attachment provides clarification and additional guidance on the use and development of OSS. It does not change or create new policy, but is intended only to explain the implications and meaning of existing laws, policies and regulations.

2. GUIDANCE

a. In almost all cases, OSS meets the definition of "commercial computer software" and shall be given appropriate statutory preference in accordance with 10 USC 2377 (reference (b)) (see also FAR 2.101(b), 12.000, 12.101 (reference (c)); and DFARS 212.212, and 252.227-7014(a)(1) (reference (d))).

b. Executive agencies, including the Department of Defense, are required to conduct market research when preparing for the procurement of property or services by 41 USC Sec. 253a (reference (e)) (see also FAR 10.001 (reference (f)). Market research for software should include OSS when it may meet mission needs.

(1) There are positive aspects of OSS that should be considered when conducting market research on software for DoD use, such as:

(i) The continuous and broad peer-review enabled by publicly available source code supports software reliability and security efforts through the identification and elimination of defects that might otherwise go unrecognized by a more limited core development team.

(ii) The unrestricted ability to modify software source code enables the Department to respond more rapidly to changing situations, missions, and future threats.

(iii) Reliance on a particular software developer or vendor due to proprietary restrictions may be reduced by the use of OSS, which can be operated and maintained by multiple vendors, thus reducing barriers to entry and exit.

(iv) Open source licenses do not restrict who can use the software or the fields of endeavor in which the software can be used. Therefore, OSS provides a net-centric licensing model that enables rapid provisioning of both known and unanticipated users.

(v) Since OSS typically does not have a per-seat licensing cost, it can provide a cost advantage in situations where many copies of the software may be required, and can mitigate risk of cost growth due to licensing in situations where the total number of users may not be known in advance.

(vi) By sharing the responsibility for maintenance of OSS with other users, the Department can benefit by reducing the total cost of ownership for software,

particularly compared with software for which the Department has sole responsibility for maintenance (*e.g.*, GOTS).

(vii) OSS is particularly suitable for rapid prototyping and experimentation, where the ability to "test drive" the software with minimal costs and administrative delays can be important.

(2) While these considerations may be relevant, they may not be the overriding aspects to any decision about software. Ultimately, the software that best meets the needs and mission of the Department should be used, regardless of whether the software is open source.

c. DoD Instruction 8500.2, "Information Assurance (IA) Implementation," (reference (g)) includes an Information Assurance Control, "DCPD-1 Public Domain Software Controls," which limits the use of "binary or machine-executable public domain software or other software products with limited or no warranty," on the grounds that these items are difficult or impossible to review, repair, or extend, given that the Government does not have access to the original source code and there is no owner who could make such repairs on behalf of the government. This control should not be interpreted as forbidding the use of OSS, as the source code is available for review, repair and extension by the government and its contractors.

d. The use of *any* software without appropriate maintenance and support presents an information assurance risk. Before approving the use of software (including OSS), system/program managers, and ultimately Designated Approving Authorities (DAAs), must ensure that the plan for software support (*e.g.*, commercial or Government program office support) is adequate for mission need.

e. There is a misconception that the Government is always obligated to distribute the source code of any modified OSS to the public, and therefore that OSS should not be integrated or modified for use in classified or other sensitive DoD systems. In contrast, many open source licenses permit the user to modify OSS *for internal use* without being obligated to distribute source code to the public. However, if the user chooses to distribute the modified OSS outside the user's organization (e.g., a Government user distributes the code outside the Government), then some OSS licenses (such as the GNU General Public License) do require distribution of the corresponding source code to the recipient of the software. For this reason, it is important to understand both the specifics of the open source license in question and how the Department intends to use and redistribute any DoD-modified OSS.

f. Software source code and associated design documents are "data" as defined by DoD Directive 8320.02 (reference (h)), and therefore shall be shared across the DoD as widely as possible to support mission needs. Open source licenses authorize widespread dissemination of the licensed software, thus allowing OSS to be shared widely across the entire Department. One way to make software source code accessible across the

Department is to use the collaborative software development environment at https://software.forge.mil/, operated by the Defense Information Systems Agency.

g. Software items, including code fixes and enhancements, developed for the Government should be released to the public (such as under an open source license) when all of the following conditions are met:

(1) The project manager, program manager, or other comparable official determines that it is in the Government's interest to do so, such as through the expectation of future enhancements by others.

(2) The Government has the rights to reproduce and release the item, and to authorize others to do so. For example, the Government has public release rights when the software is developed by Government personnel, when the Government receives "unlimited rights" in software developed by a contractor at Government expense, or when pre-existing OSS is modified by or for the Government.

(3) The public release of the item is not restricted by other law or regulation, such as the Export Administration Regulations or the International Traffic in Arms Regulation, and the item qualifies for Distribution Statement A, per DoD Directive 5230.24 (reference (i)).

Exhibit 3

DoD Open Source Software (OSS) FAQ

Frequently Asked Questions regarding Open Source Software (OSS) and the Department of Defense (DoD)

This page is an educational resource for government employees and government contractors to understand the policies and legal issues relating to the use of open source software (OSS) in the Department of Defense (DoD). The information on this page does not constitute legal advice and any legal questions relating to specific situations should be referred to legal counsel. References to specific products or organizations are for information only, and do not constitute an endorsement of the product/company.

A collaborative version of this document is published in Intellipedia-U at **https://www.intelink.gov/wiki/Open_Source_Software_(OSS)_FAQ**

Contents

Defining Open Source Software (OSS)

Q: What is open source software (OSS)?

The 16 October 2009 memorandum from the DoD CIO, "**Clarifying Guidance Regarding Open Source Software (OSS)**" defines OSS as "software for which the human-readable source code is available for use, study, re-use, modification, enhancement, and re-distribution by the users of that software".

Careful legal review is required to determine if a given license is really an open source software license. The following organizations examine licenses; licenses should pass at least the first two industry review processes, and preferably all of them, else they have a greatly heightened risk of not being an open source software license:

- Open source software licenses are reviewed and approved as conforming to the **Open Source Definition** by the **Open Source Initiative (OSI)**. The OSI publishes **a list of licenses which have successfully gone through the approval process and comply with the Open Source Definition**.
- In practice, an open source software license must also meet the **GNU Free Software Definition**; the GNU project **publishes a list of licenses that meet the Free Software Definition**.
- Fedora reviews licenses and publishes a list of **"good" licenses that Fedora has determined are open source software licenses**.
- **Debian-legal** also examines licenses (for Debian) to determine if they meet the **Debian social contract**; the **Debian license information** lists licenses that are known to pass (or not pass) these criteria.

In practice, nearly all open source software is released under one of a very few licenses that are known to meet this definition. These licenses include the **MIT license**, **revised BSD license** (and its 2-clause variant), the **Apache 2.0 license**, the **GNU Lesser General Public License (LGPL)** versions 2.1 or 3, and the **GNU General Public License (GPL)** versions 2 or 3. Using a standard license simplifies collaboration and eliminates many legal analysis costs.

Q: What are synonyms for open source software?

"Open source software" is also called "Free software", "libre software", "Free/open source software (FOSS or F/OSS)", and "Free/Libre/Open Source Software (FLOSS)". The term "Free software" predates the term "open source software", but the term "Free software" has been sometimes misinterpreted as meaning "no cost", which is *not* the intended meaning in this context. ("Free" in "Free software" refers to freedom, not price.) The term "open source software" is sometimes hyphenated as "open-source software".

The DoD has chosen to use the term "open source software" (OSS) in its official policy documents.

Q: What are antonyms for open source software?

Commercially-available software that is *not* open source software is typically called *proprietary* or *closed source* software.

Q: Is this related to "open source intelligence"?

No. In the Intelligence Community(IC), the term "open source" typically refers to overt, publicly available sources (as opposed to covert or classified sources). Thus, Open Source Intelligence (OSINT) is form of intelligence collection management that involves finding, selecting, and acquiring information from publicly available sources and analyzing it to produce actionable intelligence.

In software, "Open Source" refers to software where the human-readable source code is available to the users of the software. (see above)

Q: Is there a name for software whose source code is publicly available, but does not meet the definition of open source software?

At this time there is no widely-accepted term for software whose source code is available for review but does not meet the definition of open source software (due to restrictions on use, modification, or redistribution). Such software could be described as "source available software" or "open-box software" (such terms might *include* open source software, but could also include other software). Obviously, software that does not meet the definition of open source software is not open source software.

OSS and DoD Policy

Q: What policies address the use of open-source software in the Department of Defense?

The following policies apply:

1. The DoD CIO issued a memorandum titled "**Clarifying Guidance Regarding Open Source Software (OSS)**" on 16 October 2009, which superseded a May 2003 memo from John Stenbit.
2. The Department of Navy CIO issued a memorandum with guidance on open source software on 5 Jun 2007. This memorandum only applies to Navy and Marine Corps commands, but may be a useful reference for others. This memo is available at **http://www.doncio.navy.mil/PolicyView.aspx?ID=312** .
3. The Open Technology Development Roadmap was released by the office of the Deputy Under Secretary of Defense for Advanced Systems and Concepts, on 7 Jun 2006. It is available at **http://www.acq.osd.mil/jctd/articles/OTDRoadmapFinal.pdf**

It is available at **http://www.acq.osd.mil/jctd/articles/OTRoadmapFinal.pdf** .

4. The Office of Management and Budget issued a memorandum providing guidance on software acquisition which specifically addressed open source software on 1 Jul 2004. It may be found at **http://www.whitehouse.gov/omb/memoranda/fy04/m04-16.html** .
5. US Army Regulation 25-2, paragraph 4-6.h, provides guidance on software security controls that specifically addresses open source software. This regulation only applies to the US Army, but may be a useful reference for others. The regulation is available at **http://www.army.mil/usapa/epubs/pdf/r25_2.pdf** .

In nearly all cases, OSS is commercial software, so the policies regarding commercial software continue to apply to OSS.

Q: Isn't using open source software forbidden by DoD Information Assurance Policy?

No. This misconception comes from a misinterpretation of DoD Instruction 8500.2, "Information Assurance (IA) Implementation", Enclosure 4, control DCPD-1.

The control in question reads:

DCPD-1 Public Domain Software Controls Binary or machine executable public domain software products and other software products with limited or no warranty such as those commonly known as freeware or shareware are not used in DoD information systems unless they are necessary for mission accomplishment and there are no alternative IT solutions available. Such products are assessed for information assurance impacts, and approved for use by the DAA. The assessment addresses the fact that such software products are difficult or impossible to review, repair, or extend, given that the Government does not have access to the original source code and there is no owner who could make such repairs on behalf of the Government.

This control is intended to limit the use of certain kinds of "binary or machine executable" software when "the Government does not have access to the original source code". As clarified in the 2009 DoD CIO Memorandum, this control does not prohibit the use of open source software, since with open source software the government *does* have access to the original source code.

In the **Desktop Application STIG version 3, release 1 (09 March 2007)**; in its section 2.4, it clearly states that DCPD-1 does not apply to open source software, for this very reason. The STIG first notes that "DoD has clarified policy on the use of open source software to take advantage of the capabilities available in the Open Source community as long as certain prerequisites are met. DoD no longer requires that operating system software be obtained through a valid vendor channel and have a formal support path, if the source code for the operating system is publicly available for review". It notes in particular that three cases for software are acceptable:

1. A utility that has publicly available source code is acceptable.
2. A commercial product that incorporates open source software is acceptable because the commercial vendor provides a warranty.
3. Vendor supported open source software is acceptable.

The DISA STIG also notes "4. A utility that comes compiled and has no warranty is not acceptable." Thus, a program must come with either source code or a warranty; if it has neither, then special dispensation is required, since it difficult to review, repair, or extend the program either directly or via someone else.

General information about OSS

Q: Is open source software commercial software? Is it COTS?

Open source software that has at least one non-governmental use, and has been or is available to the public, is commercial software. If it is already available to the public and is used unchanged, it is usually COTS.

U.S. law governing federal procurement (**U.S. Code Title 41, Chapter 7, Section 403**) defines "commercial item" as including "Any item, other than real property, that is of a type customarily used by the general public or by non-governmental entities for purposes other than governmental purposes (i.e., it has some non-government use), and (i) Has been sold, leased, or **licensed** to the general public; or (ii) Has been offered for sale, lease, or license to the general public ...". Thus, as long as the software has at least one non-governmental use, software released (or offered for release) to the public is a commercial item for procurement purposes.

Similarly, **U.S. Code Title 41, Chapter 7, Section 431** defines the term "Commercially available off-the-shelf (COTS) item"; software is COTS if it is (a) a "commercial item", (b) sold in substantial quantities in the commercial marketplace, and (c) is offered to the Government, without modification, in the same form in which it is sold in the commercial marketplace. Thus, OSS available to the public and used unchanged is normally COTS.

These definitions in U.S. law govern U.S. acquisition regulations, namely the **Federal Acquisition Regulation (FAR)** and the **Defense Federal Acquisition Regulation Supplement (DFARS)**. **DFARS 252.227-7014 Rights in Noncommercial Computer Software and Noncommercial Computer Software Documentation** defines "Commercial computer software" as "software developed or regularly used for non-governmental purposes which: (i) Has been sold, leased, or licensed to the public; (ii) Has been offered for sale, lease, or license to the public; (iii) Has not been offered, sold, leased, or licensed to the public but will be available for commercial sale, lease, or license in time to satisfy the delivery requirements of this contract; or (iv) Satisfies a criterion expressed in paragraph (a)(1)(i), (ii), or (iii) of this clause and would require only minor modification to meet the requirements of this contract."

There are many other reasons to believe OSS is commercial software:

- OSS is increasingly commercially developed and supported.
- OSS projects typically seek financial gain in the form of improvements. **U.S. Code Title 17, section 101** (part of copyright law) explicitly defines the term "financial gain" as including "receipt, or expectation of receipt, of anything of value, including the receipt of other copyrighted works."
- OSS licenses and projects clearly approve of commercial support

Q: Why is it important to understand that open source software is commercial software?

It is important to understand that open source software is commercial software, because there are many laws, regulations, policies, and so on regarding commercial software. Failing to understand that open source software is commercial software would result in failing to follow the laws, regulations, policies, and so on regarding commercial software.

In particular, U.S. law (**10 USC 2377**) requires a preference for commercial items for procurement of supplies or services. 10 USC 2377 requires that the head of an agency shall ensure that procurement officials in that agency, to the maximum extent practicable:

1. "acquire commercial items or nondevelopmental items other than commercial items to meet the needs of the agency;
2. require prime contractors and subcontractors at all levels under the agency contracts to incorporate commercial items or nondevelopmental items other than commercial items as components of items supplied to the agency;
3. modify requirements in appropriate cases to ensure that the requirements can be met by commercial items or, to the extent that commercial items suitable to meet the agency's needs are not available, nondevelopmental items other than commercial items;
4. state specifications in terms that enable and encourage bidders and offerors to supply commercial items or, to the extent that commercial items suitable to meet the

agency's needs are not available, nondevelopmental items other than commercial items in response to the agency solicitations;
5. revise the agency's procurement policies, practices, and procedures not required by law to reduce any impediments in those policies, practices, and procedures to the acquisition of commercial items; and
6. require training of appropriate personnel in the acquisition of commercial items."

Similarly, it requires preliminary market research to determine "whether there are commercial items or, to the extent that commercial items suitable to meet the agency's needs are not available, nondevelopmental items other than commercial items available" that "(A) meet the agency's requirements; (B) could be modified to meet the agency's requirements; or (C) could meet the agency's requirements if those requirements were modified to a reasonable extent." This market research should occur "before developing new specifications for a procurement by that agency; and before soliciting bids or proposals for a contract in excess of the simplified acquisition threshold."

An agency that failed to consider open source software, and instead only considered proprietary software, would fail to comply with these laws, because it would unjustifiably exclude a significant part of the commercial market. This is particularly the case where future modifications by the U.S. government may be necessary, since OSS by definition permits modification.

Q: Are "non-commercial software", "freeware", or "shareware" the same thing as open source software?

No.

Do not mistakenly use the term "non-commercial software" as a synonym for "open source software". As noted above, in nearly all cases, open source software is considered "commercial software" by U.S. law, the FAR, and the DFARS. **DFARS 252.227-7014** specifically defines "commercial computer software" in a way that includes nearly all OSS, and defines "noncommercial computer software" as software that does *not* qualify as "commercial computer software". In addition, important open source software is typically supported by one or more commercial firms.

Also, do not use the terms "freeware" or "shareware" as a synonym for "open source software". DoD Instruction 8500.2, "Information Assurance (IA) Implementation", Enclosure 4, control DCPD-1, states that these terms apply to software where "the Government does not have access to the original source code". The government *does* have access to the original source code of open source software, so these terms do not apply.

Q: How is OSS typically developed?

OSS is typically developed through a collaborative process.

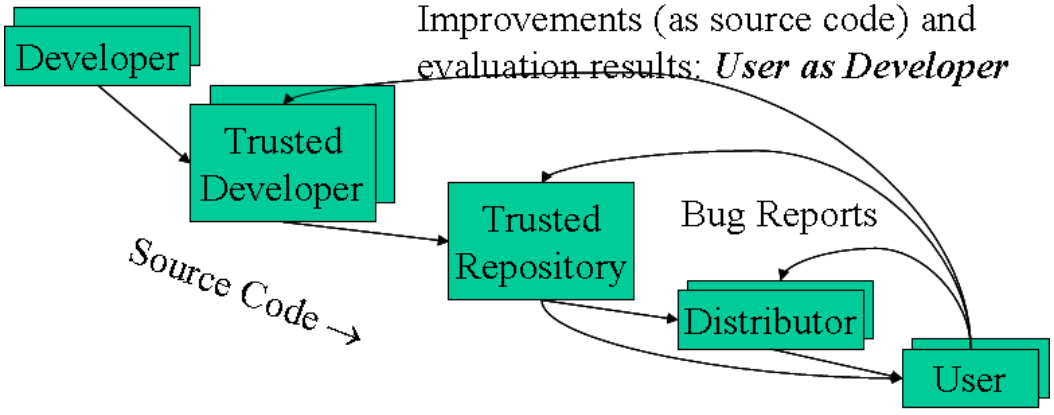
Most OSS projects have a "trusted repository", that is, some (web) location where people can get the "official" version of the program, as well as related information (documentation, bug report system, mailing lists, etc.). Users can get their software directly from the trusted repository, or get it through distributors who acquire it (and provide additional value such as integration with other components, testing, special configuration, support, and so on).

Only some developers are allowed to modify the trusted repository directly: the trusted developers. At project start, the project creators (who create the initial trusted repository) are the trusted developers, and they determine who else may become a trusted developer of this initial trusted repository. All other developers can make changes to their local copies, and even post their versions to the Internet (a process made especially easy by distributed software configuration management tools), but they must submit their changes to a trusted developer to get their changes into the trusted repository.

Users can send bug reports to the distributor or trusted repository, just as they could for a proprietary program. But what is radically different is that a user can actually make a change to the program itself (either directly, or by hiring someone to do it). Since users will want to use the improvements made by others, they have a strong financial incentive to submit their improvements to the trusted repository. That way, their improvements will be merged with the improvements of others, enabling them to use all improvements instead of only their own.

This can create an avalanche-like "virtuous cycle". As the program becomes more capable, more users are attracted to using it. A very small percentage of such users determine that they can make a change valuable to them, and contribute it back (to avoid maintenance costs). As more improvements are made, more people can use the product, creating more potential users as developers - like a snowball that gains mass as it rolls downhill.

This enables cost-sharing between users, as with proprietary development models. However, this cost-sharing is done in a rather different way than in proprietary development. In particular, note that the costs borne by a particular organization are typically only those for whatever improvements or services are used (e.g., installation, configuration, help desk, etc.). In contrast, typical proprietary software costs are per-seat, not per-improvement or service. However, it must be noted that the OSS model is much more reflective of the actual costs borne by development organizations. It costs essentially nothing to send a file or burn a CD-ROM of software; once it exists, all software costs are due to maintenance and support of software. In short, OSS more accurately reflects the economics of software development; some speculate that this is one reason why OSS has become so common so quickly.



Q: Isn't OSS developed primarily by inexperienced students?

No, OSS is developed by a wide variety of software developers, and the average developer is quite experienced. A **Boston Consulting Group** study found that the average age of OSS developers was 30 years old, the majority had training in information technology and/or computer science, and on average had 11.8 years of computer programming experience.

Q: Is open source software the same as "open systems/open standards"?

No, although they work well together, and both are strategies for reducing **"vendor lock-in"**. Vendor lock-in, aka lock-in, is the situation in which customers are dependent on a single supplier for some product (i.e., a good or service), or products, and cannot move to another vendor without substantial costs and/or inconvenience. Lock-in tends to raise costs substantially, reduces long-term value (including functionality, innovation, and reliability), and can become a serious security problem (since the supplier has little incentive to provide a secure product and to quickly fix problems found later).

An "Open System" is a "system that employs modular design, uses widely supported and consensus based standards for its key interfaces, and has been subjected to successful V&V tests to ensure the openness of its key interfaces" (per the DoD OSJTF). Thus, open systems require standards that are widely-supported and consensus-based; standards that meet these (and possibly some additional conditions) may be termed "open standards". Open systems and open standards counter dependency on a single supplier, though only if there is a competing marketplace of replaceable components. Indeed, according to **Walli**, "Standards exist to encourage & enable multiple implementations". Many governments, not just the U.S., view open systems as critically necessary. DoD Directive 5000.1 states that open systems “shall be employed, where feasible”, and the European Commission identifies open standards as a major policy thrust.

There are many definitions for the term "open standard". Fundamentally, a standard is a specification, so an "open standard" is a specification that is "open". Public definitions include those of the **European Interoperability Framework (EIF)**, the **Digistan definition of open standard** (based on the EIF), and **Bruce Perens' "Open Standards: Principles and Practice"**.

In the DoD, the **DISRonline** is a useful resource for identifying recommended standards (which tend to be open standards). DISRonline is a collection of web-based applications supporting the continuing evolution of the Department of Defense (DoD) Information Technology Standards Registry (DISR). **DAU has some information about DISRonline**. The **Open Systems Joint Task Force (OSJTF) web page** also provides some useful background.

Increasingly, many DoD capabilities are accessible via web browsers using open standards such as TCP/IP, HTTP, HTML, and CSS; in such cases, it is relatively easy to use or switch to open source software implementations (since the platforms used to implement the client or server become less relevant). As noted by the OSJTF definition for open systems, be sure to test such systems with more than one web browser (e.g., Internet Explorer and Firefox), to reduce the risk of vendor lock-in.

Q: How does open source software work with open systems/open standards?

Open standards can aid open source software projects:

- Open standards make it easier for users to (later) adopt an open source software program, because users of open standards aren't locked into a particular implementation. Instead, users who are careful to use open standards can easily switch to a different implementation, including an OSS implementation.
- Open standards also make it easier for OSS developers to create their projects, because the standard itself helps developers know what to do. Creating any interface is an effort, and having a pre-defined standard helps reduce that effort greatly.

Note that open standards aid proprietary software in exactly the same way.

OSS aids open standards, too:

- OSS implementations can help create and keep open standards open. A FLOSS implementation can be read and modified by anyone; such implementations can quickly become a working reference model (a "sample implementation" or an "executable specification") that demonstrates what the specification means (clarifying the specification) and demonstrating how to actually implement it. Perhaps more importantly, by forcing there to be an implementation that others can examine in detail, resulting in better specifications that are more likely to be used.
- OSS implementations can help rapidly increase adoption/use of the open standard. OSS programs can typically be simply downloaded and tried out, making it much easier for people to try it out and encouraging widespread use. This also pressures proprietary implementations to limit their prices, and such lower prices for proprietary software also encourages use of the standard.

With practically no exceptions, successful open standards have OSS implementations.

So, while open systems/open standards are different from open source software, they are complementary and can work well together.

OSS Licenses

Q: What is the legal basis of OSS licenses?

Software licenses, including those for open source software, are typically based on copyright law. Under U.S. copyright law, users must have permission (i.e. a license) from the copyright holder(s) before they can obtain a copy of software to run on their system(s). Authors of a creative work, or their employer, normally receive the copyright once the work is in a fixed form (e.g., written/typed). Others can obtain permission to use a copyrighted work by obtaining a license from the copyright holder. Typically, obtaining rights granted by the license can only be obtained when the requestor agrees to certain conditions. For example, users of proprietary software must typically pay for a license to use a copy or copies. Open source software licenses grant more rights than proprietary software licenses, but they are still conditional licenses that require the user to obey certain terms.

Software licenses (including OSS licenses) may also involve the laws for patent, trademark, and trade secrets, in addition to copyright.

Export control laws are often not specifically noted in OSS licenses, but nevertheless these laws also govern when and how software may be released.

Q: Are OSS licenses legally enforceable?

Yes, in general. For advice about a *specific* situation, however, consult with legal counsel.

The **U.S. Court of Appeals for the Federal Circuit's 2008 ruling on Jacobsen v. Katzer** made it clear that OSS licenses are enforceable, even if money is not exchanged. It noted that a copyright holder may dedicate a "certain work to free public use and yet enforce an 'open source' copyright license to control the future distribution and modification of that work... Open source licensing has become a widely used method of creative collaboration that serves to advance the arts and sciences in a manner and at a pace that few could have imagined just a few decades ago... Traditionally, copyright owners sold their copyrighted material in exchange for money. The lack of money changing hands in open source licensing should not be presumed to mean that there is no economic consideration, however. There are substantial benefits, including economic benefits, to the creation and distribution of copyrighted works under public licenses that range far beyond traditional license royalties... The choice to exact consideration in the form of compliance with the open source requirements of disclosure and explanation of changes, rather than as a dollar-denominated fee, is entitled to no less legal recognition. Indeed, because a calculation of damages is inherently speculative, these types of license restrictions might well be rendered meaningless absent the

less legal recognition. Indeed, because a calculation of damages is inherently speculative, these types of license restrictions might well be rendered meaningless absent the ability to enforce through injunctive relief." In short, it determined that the OSS license at issue in the case (the Artistic license) was indeed an enforceable license.

"Enforcing the GNU GPL" by Eben Moglen is a brief essay that argues why the GNU General Public License (GPL), specifically, is enforceable. U.S. courts have determined that the GPL does not violate anti-trust laws. In **Wallace vs. FSF**, Judge Daniel Tinder stated that "the GPL encourages, rather than discourages, free competition and the distribution of computer operating systems..." and found no anti-trust issues with the GPL. Similarly, in **Wallace v. IBM, Red Hat, and Novell**, the U.S. Court of Appeals for the Seventh Circuit found in November 2006 that the GNU General Public License (GPL) "and open-source software have nothing to fear from the antitrust laws". **German courts have enforced the GPL**.

Q: What are the major types of open source software licenses?

OSS licenses can be grouped into three main categories: Permissive, strongly protective, and weakly protective. Here is an explanation of these categories, along with common licenses used in each category (see **The Free-Libre / Open Source Software (FLOSS) License Slide**):

- Permissive: These licenses permit the software to become proprietary (i.e., not OSS). This includes the **MIT license** and the **revised BSD license**. The **Apache 2.0 license** is also a popular license in this category; note that the Apache 2.0 license is compatible with GPL version 3, but not with GPL version 2.
- Strongly Protective (aka strong copyleft): These licenses prevent the software from becoming proprietary, and instead enforce a "share and share alike" approach. In such licenses, if you give someone a binary of the program, you are obligated to give them the source code (perhaps upon request) under the same terms. This includes the most popular FLOSS license, the **GNU General Public License (GPL)**. There are two versions of the GPL in common use today: the older version 2, and the newer version 3.
- Weakly Protective (aka strong copyleft): These licenses are a compromise between permissive and strongly protective licenses. These prevent the software component (often a software library) from becoming proprietary, yet permit it to be part of a larger proprietary program. The **GNU Lesser General Public License (LGPL)** is the most popular such license, and there are two versions in common use: the older version 2.1 and newer version 3. An alternative approach is to use the GPL plus a **GPL linking exception term (such as the "Classpath exception")**.

Q: How can you determine if different open source software licenses are compatible?

In general, legal analysis is required to determine if multiple programs, covered by different OSS licenses, can be legally combined into a single larger work. This legal analysis must determine if it is possible to meet the conditions of all relevant licenses simultaneously. If it is possible to meet the conditions of all relevant licenses simultaneously, then those licenses are *compatible*.

Thankfully, such analyses has already been performed on the common OSS licenses, which tend to be mutually compatible. Many analyses focus on versions of the GNU General Public License (GPL), since this is the most common OSS license, but analyses for other licenses are also available. Resources for further information include:

- **GPL FAQ** (Focuses on compatibility between versions of the GPL and LGPL)
- **The Free-Libre / Open Source Software (FLOSS) License Slide**
- **Various Licenses and Comments about Them**
- **Maintaining Permissive-Licensed Files in a GPL-Licensed Project: Guidelines for Developers (Software Freedom Law Center)**
- **Fedora Licensing**

In brief, the MIT and 2-clause BSD license are dominated by the 3-clause BSD license, which are all dominated by the LGPL licenses, which are all dominated by the GPL licenses. By "dominate", that means that when software is merged which have those pairs of licenses, the dominating license essentially governs the resulting combination because the dominating license essentially includes all the key terms of the other license. This also means that these particular licenses are compatible. The Apache 2.0 license is compatible with the GPL version 3 license, but not the GPL version 2 license. The GPL version 2 and the GPL version 3 are in principle incompatible with each other, but in practice, most released OSS states that it is "GPL version 2 or later" or "GPL version 3 or later"; in these cases, version 3 is a common license and thus such software is compatible.

Note that this sometimes depends on how the program is used or modified. For example, the LGPL permits the covered software (usually a library) to be embedded in a larger work under many different licenses (including proprietary licenses), subject to certain conditions. However, if the covered software/library is *itself* modified, then additional conditions are imposed.

This need for legal analysis is one reason why creating new OSS licenses is strongly discouraged: It can be extremely difficult, costly, and time-consuming to analyze the interplay of many different licenses. It is usually far better to stick to licenses that have already gone through legal review and are widely used in the commercial world.

Q: Can OSS licenses and approaches be used for material other than software?

Yes. The **Creative Commons** is a non-profit organization that provides free tools, including a set of licenses, to "let authors, scientists, artists, and educators easily mark their creative work with the freedoms they want it to carry". A copyright holder who releases creative works under one of the Creative Common licenses that permit commercial use and modifications would be using an OSS-like approach for such works. **Wikipedia** maintains an encyclopedia using approaches similar to open source software approaches. Note that **Creative Commons does not recommend that you use one of their licenses for software**; they encourage using one of the existing OSS licenses which "were designed specifically for use with software".

Computer and electronic hardware that is designed in the same fashion as open source software (OSS) is sometimes termed **open source hardware**. The term has primarily been used to reflect the free release of information about the hardware design, such as schematics, bill of materials and PCB layout data, or its representation in a hardware description language (HDL), often with the use of open source software to drive the hardware.

Software/hardware for which the implementation, proofs of its properties, and all required tools are released under an OSS license are termed **open proofs**(see the **open proofs website for more information**).

Where it is unclear, make it clear what the "source" or "source code" means.

(See **GPL FAQ, "Can I use the GPL for something other than software?"**.)

Q: Is it more difficult to comply with OSS licenses than proprietary licenses?

No, complying with OSS licenses is much easier than proprietary licenses if you only use the software in the same way that proprietary software is normally used. By definition, OSS software permits arbitrary use of the software, and allows users to re-distribute the software to others. The terms that apply to usage and redistribution tend to be trivially easy to meet (e.g., you must not remove the license or author credits when re-distributing the software). Thus, complex license management processes to track every installation or use of the software, or who is permitted to use the software, is completely unnecessary. Support for OSS is often sold separately for OSS; in such cases,

you must comply with the support terms for those uses to receive support, but these are typically the same kinds of terms that apply to proprietary software (and they tend to be simpler in practice).

It is only when the OSS is modified that additional OSS terms come into play, depending on the OSS license. Since it is typically not legal to modify proprietary software at all, or it is legal only in very limited ways, it is trivial to determine when these additional terms may apply. The real challenge is one of education - some developers incorrectly believe that just because something is free to download, it can be merged or changed without restriction. This has never been true, and explaining this takes little time.

Q: Who can enforce OSS licenses?

Typically enforcement actions are based on copyright violations, and only copyright holders can raise a copyright claim in U.S. court. In the commercial world, the copyright holders are typically the individuals and organizations that originally developed the software. Under the current DoD contracting regime, the contractor usually retains the copyright for software developed with government funding, so in such cases the contractor (not the government) has the right to sue for copyright violation. In some cases, the government obtains the copyright; in those cases, the government can sue for copyright violation.

However, the government can release software as OSS when it has unlimited rights to that software. The government is not the copyright holder in such cases, but the government can still enforce its rights. Although the government cannot directly sue for copyright violation, in such cases it can still sue for breach of license and, presumably, get injunctive relief to stop the breach and money damages to recover royalties obtained by breaching the license (and perhaps other damages as well).

In addition, a third party who breaches a software license (including for OSS) granted by the government risks losing rights they would normally have due to the "doctrine of unclean hands". The **doctrine of unclean hands**, per law.com, is "a legal doctrine which is a defense to a complaint, which states that a party who is asking for a judgment cannot have the help of the court if he/she has done anything unethical in relation to the subject of the lawsuit. Thus, if a defendant can show the plaintiff had 'unclean hands,' the plaintiff's complaint will be dismissed or the plaintiff will be denied judgment." So if the government releases software as OSS, and a malicious developer performs actions in violation of that license, then the government's courts need not enforce any of that malicious developer's intellectual rights to that result. In effect, the malicious developer could lose many or all rights over their license-violating result, even rights they would normally have had! Since OSS licenses are quite generous, the only license-violating actions a developer is likely to try is to release software under a more stringent license... and those will have little effect once they cannot be enforced in court. In short, the government can enforce its licenses, even when it doesn't have the copyright.

See **GPL FAQ, "Who has the power to enforce the GPL?"**

OSS and Security/Software Assurance/System Assurance/Supply Chain Risk Management

Q: Does the DoD use OSS for security functions?

Yes. The **2003 MITRE study, "Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense"**, for analysis purposes, posed the hypothetical question of what would happen if OSS software were banned in the DoD, and found that OSS "plays a far more critical role in the DoD than has been generally recognized... (especially in) Infrastructure Support, Software Development, Security, and Research". In particular, it found that DoD security "depends on (OSS) applications and strategies", and that a hypothetic ban "would have immediate, broad, and in some cases strongly negative impacts on the ability of the DoD to analyze and protect its own networks against hostile intrusion. This is in part because such a ban would prevent DoD groups from using the same analysis and network intrusion applications that hostile groups could use to stage cyberattacks. It would also remove the uniquely (OSS) ability to change infrastructure source code rapidly in response to new modes of cyberattack".

Q: Doesn't hiding source code automatically make software more secure?

No. Indeed, vulnerability databases such as CVE make it clear that merely hiding source code does not counter attacks:

- Dynamic attacks (e.g., generating input patterns to probe for vulnerabilities and then sending that data to the program to execute) don't need source or binary. Observing the output from inputs is often sufficient for attack.
- Static attacks (e.g., analyzing the code instead of its execution) can use pattern-matches against binaries - source code is not needed for them either.
- Even if source code is necessary (e.g., for source code analyzers), adequate source code can often be regenerated by disassemblers and decompilers sufficiently to search for vulnerabilities. Such source code may not be adequate to cost-effectively *maintain* the software, but attackers need not maintain software.
- Even when the original source is necessary for in-depth analysis, making source code available to the public significantly aids defenders and not just attackers. Continuous and broad peer-review, enabled by publicly available source code, improves software reliability and security through the identification and elimination of defects that might otherwise go unrecognized by the core development team. Conversely, where source code is hidden from the public, attackers can attack the software anyway as described above. In addition, an attacker can often acquire the original source code from suppliers anyway (either because the supplier voluntarily provides it, or via attacks against the supplier); in such cases, if only the attacker has the source code, the attacker ends up with another advantage.

Hiding source code *does* inhibit the ability of third parties to respond to vulnerabilities (because changing software is more difficult without the source code), but this is obviously *not* a security advantage. In general, "Security by Obscurity" is widely denigrated.

This does *not* mean that the DoD will reject using proprietary COTS products. There are valid business reasons, unrelated to security, that may lead a commercial company selling proprietary software to choose to hide source code (e.g., to reduce the risk of copyright infringement or the revelation of trade secrets). What it does mean, however, is that the DoD will not reject consideration of a COTS product merely because it is OSS. Some OSS is very secure, while others are not; some proprietary software is very secure, while others are not. Each product must be examined on its own merits.

Q: What are indicators that a specific OSS program will have fewer unintentional vulnerabilities?

As noted in the **Secure Programming for Linux and Unix HOWTO**, three conditions reduce the risks from unintentional vulnerabilities in OSS:

1. Developers/reviewers need security knowledge. Knowledge is more important than the licensing scheme.
2. People have to actually review the code.
    1. This has a reduced likelihood if the program is niche/rarely-used, few developers, rare computer language, or not really OSS. Conversely, if it widely-used, has many developers, and so on, the likelihood of review increases. Examine if it is truly community-developed - or if there are only a very few developers.
    2. Review really does happen. Several static tool vendors support analysis of OSS (such as Coverity and Fortify) as a way to improve their tools and gain market use. There are many general OSS review projects, such as those by OpenBSD and the Debian Security Audit team. And of course, individual OSS projects often have security review processes or methods (such as Mozilla's bounty system). If there are reviewers from many different backgrounds (e.g., different countries), this can also reduce certain risks. When examining a specific OSS project, look for evidence that review (both by humans and tools) does take place.
3. Problems must be fixed. It is far better to fix vulnerabilities before deployment - are such efforts occuring? When the software is already deployed, does the project develop and deploy fixes?

Q: Is there a risk of malicious code becoming embedded into OSS?

The use of *any* commercially-available software, be it proprietary or OSS, creates the risk of executing malicious code embedded in the software. Even if a commercial program did not originally have vulnerabilities, both proprietary and OSS program binaries can be modified (e.g., with a "hex editor" or virus) so that it includes malicious code. It may be illegal to modify proprietary software, but that will normally not slow an attacker. Thankfully, there are ways to reduce the risk of executing malicious code when using commercial software (both proprietary and OSS). It is impossible to completely eliminate all risks; instead, focus on reducing risks to acceptable levels.

The use of software with a proprietary license provides absolutely no guarantee that the software is free of malicious code. Indeed, many people have released proprietary code that is malicious. What's more, proprietary software release practices make it more difficult to be confident that the software does not include malicious code. Such software does not normally undergo widespread public review, indeed, the source code is typically not provided to the public and there are often license clauses that attempt to inhibit review further (e.g., forbidding reverse engineering and/or forbidding the public disclosure of analysis results). Thus, to reduce the risk of executing malicious code, potential users should consider the reputation of the supplier and the experience of other users, prefer software with a large number of users, and ensure that they get the "real" software and not an imitator. Where it is important, examining the security posture of the supplier (e.g., their processes that reduce risk) and scanning/testing/evaluating the software may also be wise.

Similarly, OSS (as well as proprietary software) may indeed have malicious code embedded in it. However, such malicious code cannot be directly inserted by "just anyone" into a well-established OSS project. As noted above, OSS projects have a "trusted repository" that only certain developers (the "trusted developers") can directly modify. In addition, since the source code is publicly released, anyone can review it, including for the possibility of malicious code. The public release also makes it easy to have copies of versions in many places, and to compare those versions, making it easy for many people to review changes. Many perceive this openness as an advantage for OSS, since OSS better meets Saltzer & Schroeder's "Open design principle" ("the protection mechanism must not depend on attacker ignorance"). This is not merely theoretical; **in 2003 the Linux kernel development process resisted an attack**. Similarly, SourceForge/Apache (in 2001) and Debian (in 2003) countered external attacks.
As with proprietary software, to reduce the risk of executing malicious code, potential users should consider the reputation of the supplier (the OSS project) and the experience of other users, prefer software with a large number of users, and ensure that they get the "real" software and not an imitator (e.g., from the main project site or a trusted distributor). Where it is important, examining the security posture of the supplier (the OSS project) and scanning/testing/evaluating the software may also be wise.

The example of Borland's InterBase/Firebird is instructive. For at least 7 years, Borland's Interbase (a proprietary database program) had embedded in it a "back door"; the username "politically", password "correct", would immediately give the requestor complete control over the database, a fact unknown to its users. Whether or not this was intentional, it certainly had the same form as a malicious back door. When the program was released as OSS, within 5 months this vulnerability was found and fixed. This shows that proprietary software can include functionality that could be described as malicious, yet remain unfixed - and that at least in some cases OSS is reviewed and fixed.

Note that merely being developed for the government is no guarantee that there is no malicious embedded code. Such developers need not be cleared, for example. Requiring that all developers be cleared first can reduce certain risks (at substantial costs), where necessary, but even then there is no guarantee.

Note that most commercial software is not intended to be used where the impact of *any* error of any kind is *extremely* high (e.g., a large number of lives are likely to be immediately lost if even the slightest software error occurs). Software that meets very high reliability/security requirements, aka "high assurance" software, must be specially designed to meet such requirements. Most commercial software (including OSS) is not designed for such purposes.

Using OSS in DoD systems

Q: Does the DoD already use open source software?

Yes, extensively. The **2003 MITRE study, "Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense"**, identified some of many OSS programs that the DoD is *already* using, and concluded that OSS "plays a more critical role in the [Department of Defense (DoD)] than has generally been recognized".

Intellipedia is implemented using MediaWiki, the open source software developed to implement Wikipedia. This Open Source Software FAQ was originally developed on Intellipedia, using a variety of web browsers including Mozilla Firefox. Thus, this FAQ was developed using open source software.

Q: Is a lot of pre-existing open source software available?

Yes. Widely-used programs include the Apache web server, Firefox web browser, Linux kernel, and many other programs. **Estimating the Total Development Cost of a Linux Distribution** estimates that the Fedora 9 Linux distribution, which contains over 5,000 software packages, represents about $10.8 billion of development effort.

Q: Is there an "approved", "recommended" or "Generally Recognized as Safe/Mature" list of Open Source Software? What programs are already in widespread use?

No, the DoD does not have an official recommendation for any particular OSS product or set of products, nor a "Generally Recognized as Safe/Mature" list. The **2003 MITRE study, "Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense"** did suggest developing a "Generally Recognized As Safe" (GRAS) list, but such a list has not been developed.

Commercial software (including OSS) that has widespread use often has lower risk, since there are often good reasons for its widespread use. The MITRE study did identify some of many OSS programs that the DoD is *already* using, and may prove helpful. Examples of OSS that are in widespread use include:

- Apache - Web server
- Mozilla Firefox - Web browser
- Mozilla Thunderbird, Evolution - Email client
- OpenOffice.org - Office document suite
- OpenSSH - Secure Shell
- OpenSSL - SSL/cryptographic library implementation
- bind - DNS server
- Postfix, Sendmail - Mail servers
- gcc - Compiler suite
- GNAT - Ada compiler suite (technically this is part of gcc)
- perl, Python, PHP - Scripting languages
- Samba - Windows - Unix/Linux interoperability
- Mailman - mailing list manager
- MySQL and PostgreSQL - Relational Database System
- GIMP - Bitmap graphics editor
- MediaWiki - Wiki

There are many "Linux distributions" which provides suites of such software such as Red Hat Enterprise Linux, Fedora, Novell SuSE, Debian and Ubuntu. Other open source software implementations of Unix interfaces include Solaris, OpenBSD, NetBSD, and FreeBSD.

Again, these are examples, and not official endorsements of any particular product or supplier.

Q: What are some military-specific open source software programs?

Some more military-specific OSS programs used in the military include:

- **FalconView** - PC-based mapping application
- Open Source Software for Imagery & Mapping (**OSSIM**) - geospatial image viewing (with classified plugins)
- OSSIM Mapping ARchieve System (**OMAR**) - video indexing
- **BRL-CAD** - solid modeling (Army)
- **Optics** - MASINT toolset (with classfied plugins)
- **Delta3d** - Game/Simulation engine for modeling and simulation (e.g., for military training/exercises)

There are many others.

Q: Is there any quantitative evidence that open source software can be as good as (or better than) proprietary software?

Yes; **Why Open Source Software / Free Software (OSS/FS, FLOSS, or FOSS)? Look at the Numbers!** is a survey paper that "provides quantitative data that, in many cases, using open source software / free software (abbreviated as OSS/FS, FLOSS, or FOSS) is a reasonable or even superior approach to using their proprietary competition according to various measures.. (its) goal is to show that you should consider using OSS/FS when acquiring software". It points to various studies related to market share, reliability, performance, scalability, security, and total cost of ownership.

This is in addition to the advantages from OSS because it can be reviewed, modified, and redistributed with few restrictions (inherent in the definition of OSS).

That said, this does *not* mean that all OSS is superior to all proprietary software in all cases by all measures. Each government program must determine its needs, and then evaluate its options for meeting those needs.

Q: When a DoD contractor is developing a new system/software as a deliverable in a typical DoD contract, is it possible to include existing open source software?

Yes, it's possible. In nearly all cases pre-existing OSS are "commercial components", and thus their use is governed by the rules for including any commercial components in the deliverable. The use of commercial components is generally encouraged, and when there are commercial components, the government expects that it will normally use whatever license is offered to the public. Depending on the contract and its interpretation, contractors may be required to get governmental permission to include commercial components in their deliverable; where this applies, this would be true for OSS components as well as proprietary components. As with all commercial items, organizations must obey the terms of the commercial license, negotiate a different license if necessary, or not use the commercial item.

An alternative is to not *include* the OSS component in the deliverable, but simply *depend* on it, as long as that is acceptable to the government. This is often done when the deliverable is a software application; instead of including commercially-available components such as the operating system or database system as part of the deliverable, the deliverable could simply state that what it requires.

Q: When a DoD contractor is developing a new system/software as a deliverable in a typical DoD contract, is it possible to use existing software licensed using the GNU General Public License (GPL)? Can the DoD used GPL-licensed software?

Yes. There is no DoD policy forbidding or limiting the use of software licensed under the GNU General Public License (GPL).

The DoD already uses a wide variety of software licensed under the GPL. A **2003 MITRE study, "Use of Free and Open Source Software (FOSS) in the U.S. Department of Defense"**, identified many OSS programs that the DoD is *already* using that are licensed using the GPL. These included the Linux kernel, the gcc compilation suite (including the GNAT Ada compiler), the OpenOffice.org office suite, the emacs text editor, the Nmap network scanner, OpenSSH and OpenSSH for encryption, and Samba for Unix/Linux/Windows interoperability. This should not be surprising; the DoD uses OSS extensively, and the GPL is the most popular OSS license.

As with all commercial items, the DoD must comply with the item's license when using the item. There are two versions of the GPL: version 2 and version 3. The key issue with both versions of the GPL is that, unlike most other OSS licenses, the GPL licenses require that a recipient of a binary (executable) must be able to demand and receive the source code of that program, and the recipient must also be able to propogate the work under that license. The Free Software Foundation (FSF) interprets linking a GPL program with another program as creating a derivative work, and thus imposing this license term in such cases.

In most cases, this GPL license term is not a problem. After all, most proprietary software licenses explicitly forbid modifying (or even reverse-engineering) the program, so the GPL actually provides *additional* rights not present in most proprietary software. So if the program is being used and not modified (a very common case), this additional term has no impact. Even for many modifications (e.g., bug fixes) this causes no issues because in many cases the DoD has no interest in keeping those changes confidential.

However, if the GPL software must be mixed with other proprietary/classified software, the GPL terms must still be followed.

Q: Under what conditions can GPL-licensed software be mixed with proprietary/classified software?

Software licensed under the GPL *can* be mixed with software released under other license terms (e.g., proprietary or classified software), but only under conditions that do not violate any license. Such mixing can normally only occur when certain kinds of separation are maintained - and thus this becomes a design issue.

The **2003 MITRE study section 1.3.4** outlines several ways to legally mix GPL with proprietary/classified software:

- Distribution Mixing – GPL and other software can be stored and transmitted together. Example: GPL software can be stored on the same computer disk as (most kinds of) proprietary software.
- Execution Mixing – GPL and other software can run at the same time on the same computer or network. Example: GPL and (unrelated) proprietary applications can be running at the same time on a desktop PC.
- Application Mixing – GPL can rely on other software to provide it with services, provided either that those services are either generic (e.g., operating system services) or have been explicitly exempted by the GPL software designer as non-GPL components. Examples include GPL applications running on proprietary operating systems or wrappers, and GPL applications that use proprietary components explicitly marked as non-GPL. Windows Services for UNIX 3.0 is a good example of commercial use of GPL application mixing.

- GPL application mixing.
- Service Mixing – GPL can provide generic services to other software. These services must be genuinely generic in the sense that the applications that use them must not depend on the detailed design of the GPL software to work. An example is (connecting) a GPL utility to a proprietary software component by using the Unix "pipe" mechanism, which allows one-way flow of data to move between software components. This is the tightest form of mixing possible with GPL and other types of software, but it must be used with care to ensure that the GPL software remains generic and is not tightly bound to any one proprietary software component.

Often such separation can occur by separating information into data and a program that uses it, or by defining distinct layers. As long as a GPL program does not embed GPL software into its outputs, a GPL program can process classified/proprietary information. Thus, GPL'ed compilers can compile classified programs (since the compilers treat the classified program as data), and a GPL'ed implementation of a virtual machine (VM) can execute classified software (since the VM implementation runs the software as data). Similarly, a GPL'ed "engine" program can be controlled by classified data that it reads. In addition, a GPL'ed program can run on top of a classified/proprietary platform when the platform is a separate "System Library" (as defined in GPL version 3). Note that enforcing such separation has many other advantages as well.

The U.S. government can directly combine GPL and proprietary/classified software into a single program arbitrarily, as long as the result is never conveyed outside the U.S. government, but this approach should not be taken lightly. This approach may inhibit later release of the combined result to other parties (e.g., allies). When taking this approach, contractors hired to modify the software must not retain copyright or other rights to the result (else the software would be conveyed outside the U.S. government); see **GPL version 3 section 2, paragraph 2** which states this explicitly.

It can be argued that classified software can be arbitrarily combined with GPL code, beyond the approaches described above. The argument is that the classification rules are simply laws of the land (and not "additional" rules), the classification rules already forbid the release of the resulting binaries to those without proper clearances, and that the GPL only requires that source code be released to those who received a binary. While this argument may be valid, we know of no general counsel ruling confirming this. Anyone who is considering this approach should obtain a ruling from general counsel first (and please let the FAQ authors know!).

If a legal method for using the GPL software for a particular application cannot be devised, and a different license cannot be negotiated, then the GPL-licensed component cannot be used for that particular purpose. Note that this also applies to proprietary software, which often have even stricter limits on if/how the software may be changed.

Q: Is the GPL compatible with Government Unlimited Rights contracts, or does the requirement to display the license, etc, violate Government Unlimited Rights contracts?

The GPL and government "unlimited rights" terms have similar goals, but differ in details. This isn't usually an issue because of how typical DoD contract clauses work under the DFARS.

Any software that has a non-government use and is licensed to the public is *commercial software*, by definition, including OSS programs licensed to the government using the GPL. Normally the government only expects to get the usual commercial rights to commercial software, and not "unlimited rights". So if the software displays a license in a way that can't be legally disabled (as required by the GPL), there is no problem, because this is an ordinary commercial software license term. The same would be true if you used Microsoft Windows; you aren't normally permitted to disable the rights-display functions of Microsoft Windows either.

In contrast, the government normally gets "unlimited rights" only when it pays for development of that software, in full or in part. Software developed by government funding would typically be termed "noncommercial software", and thus falls under different rules. The government *does* have the right to take software it has unlimited rights to, and link it with GPL software. After all, the government can use unlimited rights software in any way it wishes.

Once the government has unlimited rights, it can release that software to the public in any it wishes - including by using the GPL. This is not a contradiction; it's quite common for different organizations to have different rights to the same software. The program available to the public may improve over time, through contributions not paid for by the U.S. government. In that case, the U.S. government can choose to use the version to which it has unlimited rights, or it can use the publicly-available commercial version available to the government through that version's commercial license (the GPL in this case).

Q: How can I evaluate OSS options?

OSS options should be evaluated in principle the same way you would evaluate any option, considering need, cost, and so on. In some cases, the sources of information for OSS differ.

Be sure to consider total cost of ownership (TCO), not just initial download costs. Even if OSS has no cost to download, there is still a cost for OSS due to installation, support, and so on (whether done in-house or through external organizations). Be sure to consider such costs over a period of time (typically the lifetime of the system including its upgrades), and use the same period when evaluating alternatives; otherwise, one-time costs (such as costs to transition from an existing proprietary system) can lead to erroneous conclusions. Include upgrade/maintenance costs, including indirect costs (such as hardware replacement if necessary to run updated software), in the TCO.

By definition, open source software provides more rights to users than proprietary software (at least in terms of use, modification, and distribution). That said, other factors may be more important for a given circumstance.

The DoD does not have a single required process for evaluating OSS. The following externally-developed evaluation processes or tips may be of use:

- **How to Evaluate Open Source Software / Free Software (OSS/FS) Programs**
- **Navica's Open Source Maturity Model (OSMM)**
- **Capgemini's Open Source Maturity Model (OSMM)**
- **Top Tips For Selecting Open Source Software**
- **Business Readiness Rating™ (BRR)**
- **QSOS**

Q: How can I migrate to OSS?

Migrating from an existing system to an OSS approach requires addressing the same issues that any migration involves.

The IDA Open Source Migration Guidelines recommend:

- before starting have a clear understanding of the reasons to migrate;
- ensure that there is active support for the change from IT staff and users;
- make sure that there is a champion for change – the higher up in the organisation the better;
- build up expertise and relationships with the OSS movement;
- start with non critical systems;
- ensure that each step in the migration is manageable.

It also suggests that the following questions need to be addressed:

- how to ensure the interoperability of systems;

- how to ensure the interoperability of systems;
- how to support mobile users;
- how to securely identify remote users;
- how to build systems that are manageable.
- ensure that security is designed in from the start and not tacked on as an after thought.

It also recommends ensuring "that decisions made now, even if they do not relate directly to a migration, should not further tie an Administration to proprietary file formats and protocols". It also notes that OSS is a disruptive technology, in particular, that it is "a move away from a product to a service based industry".

Q: How can I get support for OSS that already exists?

You can support OSS either through a commercial organization, or you can self-support OSS; in either case, you can use community support as an aid.

Commercial support can either be through companies with specialize in OSS support (in general or for specific products), or through contractors who specialize in supporting customers and provide the OSS support as part of a larger service. Examples of the former include Red Hat, Novell, HP, Sun, IBM, DMSolutions, SourceLabs, OpenLogic, Carahsoft, and Mozilla.

Some have found that community support can be very helpful. The 1997 InfoWorld “Best Technical Support” award was won by the "Linux User Community". However, you should examine past experience and your intended uses before depending on this as a primary mechanism for support.

Q: How do GOTS, Proprietary COTS, and OSS COTS compare?

Government Off-the-Shelf (GOTS), proprietary commercial off-the-shelf (COTS), and OSS COTS are all methods to enable reuse of software across multiple projects. Thus, they are all strategies for sharing the development and maintenance costs of software, potentially reducing its cost.

GOTS is especially appropriate when the software *must not* be released to the public (e.g., it is classified) or when licenses forbid more extensive sharing (e.g., the government only has government-purpose rights to the software). If the software is not released to the public at all and it provides a direct military advantage, then the U.S. military (and its allies) may obtain a distinct military advantage (note that such software would normally be classified). Unlike proprietary COTS, GOTS has the advantage that the government has the right to change the software whenever the government chooses to do so. Unfortunately, the government must pay for *all* development and maintenance costs of GOTS; since these can be substantial, GOTS runs the risk of becoming obsolescent when the government cannot afford those costs. Also, since there are a limited number of users, there is limited opportunity to gain from user innovation - which again can lead to obsolescence. Even where there is GOTS/classified software, such software is typically only a *portion* of the entire system, with other components implemented through COTS components.

Proprietary COTS is especially appropriate when there is an existing proprietary COTS product that meets the need. Proprietary COTS tend to be lower cost than GOTS, since the cost of development and maintenance is typically shared among a larger number of users (who typically pay to receive licenses to use the product). Unfortunately, this typically trades off flexibility; the government typically does not have the right to modify the software, so it often cannot fix serious security problems, add arbitrary improvements, or make the software work on platforms of its choosing. If the supplier attains a monopoly or it is difficult to switch from the supplier, the costs may skyrocket. What is more, the supplier may choose to abandon the product; software escrow can reduce these risks somewhat, but in these cases it becomes GOTS with its attendant costs.

OSS COTS is especially appropriate when there is an existing OSS COTS product that meets the need, or one can be developed and supported by a wide range of users/co-developers. OSS COTS tends to be lower cost than GOTS, in part for the same reasons as proprietary COTS: its costs are shared among more users. It also often has lower total cost-of-ownership than proprietary COTS, since acquiring it initially is often free or low-cost, and all other support activities (training, installation, modification, etc.) can be competed. Its flexibility is as high as GOTS, since it can be arbitrarily modified. However, note that this cost discussion only applies if there are many users; if no user/co-developer community is built up, then it can be as costly as GOTS.

Q: What are the risks of failing to consider the use of OSS components or approaches?

For the DoD, the risks of failing to consider the use of OSS where appropriate are of increased cost, increased schedule, and/or reduced performance (including reduced innovation or security) to the DoD due to the failure to use the commercial software that best meets the needs (when that is the case). It also risks reduced flexibility (including against cyberattack), since OSS permits arbitrary later modification by users in ways that some other license approaches do not. In addition, ignoring OSS would not be lawful; U.S. law specifically requires consideration of commercial software (including extant OSS, regardless of exactly which license it uses), and specifically instructs departments to pass this requirements down to contractors and their suppliers.

DoD contractors who always ignore components because they are OSS, or because they have a particular OSS license they don't prefer, risk losing projects to more competitive bidders. If that competitor's use of OSS results in an advantage to the DoD (such as lower cost, faster schedule, increased performance, or other factors such as increased flexibility), contractors should expect that the DoD will choose the better bid. This does not mean that existing OSS elements should always be chosen, but they should be considered.

Q: Is there a large risk that widely-used OSS unlawfully includes proprietary software (in violation of copyright)?

No; this is a low-probability risk for widely-used OSS programs. A primary reason that this is low-probability is the publicity of the OSS source code itself (which almost invariably includes information about those who made specific changes). Any company can easily review OSS to look for proprietary code that should not be there; there are even OSS tools that can find common code. A company that found any of its proprietary software in an OSS project can in most cases quickly determine who unlawfully submitted that code and sue for infringement.

In addition, widely-used licenses and OSS projects often include additional mechanisms to counter this risk. The GPL and LGPL licenses specifically recommend that "You should also get your employer (if you work as a programmer) or school, if any, to sign a 'copyright disclaimer' for the program, if necessary.", and point to additional information. Many projects, particularly the large number of projects managed by the Free Software Foundation (FSF), ask for an employer's disclaimer from the contributor's employer in a number of circumstances. The Linux kernel project requires that a person proposing a change add a "Signed-off-by" tag, attesting that the "patch, to the best of his or her knowledge, can legally be merged into the mainline and distributed under the terms of (the license)."

In practice, OSS projects tend to be remarkably clean of such issues. For example, **Code Analysis of the Linux Wireless Team's ath5k Driver** found no license problems.

When considering any software (OSS or proprietary), look for evidence that the risk of unlawful release is low. Factors that greatly reduce this risk include:

- Widespread availability and use of the software (which increases the likelihood of detection)
- Configuration management systems that record the identity of individual contributors (which acts as a deterrent)
- Licenses or development policies that warn against the unlawful inclusion of material, or require people to specifically assert that they are acting lawfully (which reduce the risk of unintentional infringement)
- Lack of evidence of infrigement (e.g., an Internet search for project name + "copyright infringement" turns up nothing). Parties are innocent until proven guilty, so if there *is*

such a charge, investigate the charges' merits instead of presuming guilt.

Q: Is there a large risk to DoD contractors that widely-used OSS violates enforceable software patents?

Typically not, though the risk varies depending on their contract and specific circumstance. Note, however, that this risk has little to do with OSS, but is instead rooted in the risks of U.S. patent infringement for *all* software, and the patent indemnification clauses in their contract.

It is difficult for software developers (OSS or not) to be confident that they have avoided software patent infringement in the United States, for a variety of reasons. Software might not infringe on a patent when it was released, yet the same software may later infringe on a patent if the patent was granted after the software's release. Many software developers find software patents difficult to understand, making it difficult for them to determine if a given patent even applies to a given program. Patent examiners have relatively little time to review each patent, and do not have effective access to most prior art in software, which may lead them to grant patents for previously-published inventions or "obvious" inventions. The U.S. has granted a large number of software patents, making it difficult and costly to examine all of them. Recent rulings have strengthened the requirement for "non-obviousness", which probably renders unenforceable some already-granted software patents, but at this time it is difficult to determine which ones are affected. As a result, it is difficult to develop software and be confident that it does not violate enforceable patents. The DoD has not expressed a position on whether or not software should be patented, but it *is* interested in ensuring that software that effectively supports its missions can be developed in a cost-effective, timely, and legal manner.

U.S. government contractors (including those in the DoD) are often indemnified from patent infringement by the U.S. government as part of their contract. This greatly reduces contractors' risks, enabling them to get work done (given this complex environment). They can obtain this by receiving certain authorization clauses in their contracts. **FAR 52.227-1 (Authorization and Consent)**, as prescribed by **FAR 27.201-2(a)(1)**, inserts the clause that the "Government authorizes and consents to all use and manufacturer... of any invention (covered by) U.S. patent". The related FAR 52.227-2 (Notice and Assistance Regarding Patent and Copyright Infringement), as prescribed by FAR 27.201-2(b), requires the contractor to report to the Contracting Officer each notice or claim of patent/copyright infrigement in reasonable written detail. Specific patents can also be authorized using clause FAR 52.227-5 or via listed exceptions of FAR 52.227-3. See also **DFARS subpart 227.70--infringement claims, licenses, and assignments** and **28 USC 1498**.

As noted in **DFARS 27.201-1**, "Pursuant to 28 U.S.C. 1498, the exclusive remedy for patent or copyright infringement by or on behalf of the Government is a suit for monetary damages against the Government in the Court of Federal Claims. There is no injunctive relief available, and there is no direct cause of action against a contractor that is infringing a patent or copyright with the authorization or consent of the Government (e.g., while performing a contract)."

There are other ways to reduce the risk of software patent infringement (in the U.S.) as well:

- Some protocols and formats have been specifically devised and reviewed to avoid patents; using them is more likely to avoid problems.
- Prior art invalidates patents. Patents expire after 20 years, so any idea ("invention") implemented in software publicly available for more than 20 years should not, in theory, be patentable. Once an invention is released to the public, the inventor has only one year to file for a patent, so any new ideas in some software must have a patent filed within one year by that inventor, or (in theory) they cannot be patented. See **Prior Art and Its Uses: A Primer, by Theodore C. McCullough**
- OSS can often be purchased (directly, or as a support contract), and such purchases often include some sort of indemnification.
- Various organizations have been formed to reduce patent risks for OSS. The **Open Invention Network** (OIN[SM]) may in some cases provide some additional protection. OIN purchases patent rights; patents owned by OIN are available royalty-free to any company, institution or individual that agrees not to assert its patents against the "Linux System" (which includes a large set of OSS projects). **The Linux Foundations' Patent Commons** forum is a neutral forum where patent pledges and other commitments can be readily accessed and easily understood.

Q: How can I avoid failure to comply with an OSS license? What are good practices for use of OSS in a larger system?

The following are good practices:

- Educate all software developers that they must comply with all valid licenses - including both proprietary *and* open source software licenses. Explain the basic terms of the most common OSS licenses to them.
- Before including *any* software in a larger system (be it proprietary *or* OSS), review its license to ensure that the license will not impede anticipated uses.
- When including externally-developed software in a larger system (e.g., as a library), make it clearly separable from the other components and easy to update. Commercial software (both proprietary and OSS) is occasionally updated to fix errors (including security vulnerabilities), and your system should be designed so that it is relatively easy to accept these updates.
- Document from where and when any external software was acquired, as well as the license conditions, so that future users and maintainers can easily comply with the license terms.

Releasing software as OSS

Q: Has the U.S. government released OSS projects or improvements?

Yes, both entirely new programs and improvements of existing OSS. There are far too many examples to list; a few examples are:

- **Security-Enhanced Linux (SELinux)**
- **OpenVista**
- **Expect**
- **EZRO**
- Evergreen (by the State of Georgia),
- OpenSSL (this improvement was a Common Criteria evaluation)
- Bind implementation of DNSSEC
- GNAT Ada compiler
- BSD TCP/IP suite

Q: What are the risks of the government *not* releasing software as OSS?

If the government modifies existing OSS, but fails to release those improvements back to the main OSS project, it risks:

- Greatly increased costs, due to the effort of self-maintaining its own version

- Inability to use improvements (including security patches and innovations) by others, where it uses a "non-standard" version instead of the version being actively maintained

Similarly, the government develops runs the following risks when it develops new software but does not release it as OSS, it risks:

- Greatly increased cost, due to having to bear the *entire* burden of development costs
- Inability to use improvements (including security patches and innovations) by others, since they do not have the opportunity to aid in its development
- The development and release of a competing OSS project. In this case, the government has the unenviable choice of (1) spending possibly large sums to switch to the OSS project (which would typically have a radically different interface and goals), or (2) continuing to use the government-unique custom solution, leaving the U.S. systems far less capable that others' (including our adversaries)
- Questions about why the government - who represents "the people" - is not releasing software that they paid for back to "the people".

Clearly, classified software cannot be released back to the public as open source software. However, often software can be split into various components, some of which are classified and some of which are not, and it is to these *unclassified* portions that this text addresses.

Q: What are the risks of the government releasing software as OSS?

The key risk is the revelation of information that should not be released to the public. Classified software should *already* be marked as such, of course. This risk is mitigated by reviewing software (in particualr, for classification and export control issues) before public release.

Q: Can government employees develop software and release it under an open source license?

Not under typical open source software licenses based on copyright, but there is an alternative with the same practical effect.

Software developed by US federal government employees (including military personnel) as part of their official duties is not subject to copyright protection and is considered “public domain” (see 17 USC § 105). Public domain software can be used by anyone for any purpose, and cannot be released under a copyright license (including typical open source software licenses).

However, software written entirely by federal government employees as part of their official duties *can* be released as “public domain” software. This is not under a copyright license, it is *absence* of a license. By some definitions this is technically not an open source license (because no license is needed), but “public domain” software can be legally used, modified, and combined with other software without restriction. Thus, “public domain” software provides recipients all of the rights that open source software must provide. An example of such software is **Expect**, which was developed and released by NIST.

Government employees may also modify existing open source software. If some portion of the software was developed by persons who are not US government employees, then the software can be released under copyright license. (See next question.)

(See also **GPL FAQ, Question "Can the US Government release a program under the GNU GPL?"**)

Q: Can government employees contribute code to open source software projects?

Yes, but the following considerations apply:

As stated above, software developed by government employees as part of their official duties is not subject to copyright protection in the United States. If a government employee enhances or modifies a (copyrighted) open source software program, the resulting work is a “joint work” (see 17 USC § 101) which is partially copyrighted and partially public domain. The resulting joint work as a whole is protected by the copyrights of the non-government authors and may be released according to the terms of the original open-source license.

However, the public domain portions may be extracted from such a joint work and used by anyone for any purpose. For computer software, modern version control and source code comparison tools typically make it easy to isolate the contributions of individual authors (via "blame" or "annote" functions).

(See also **Free Software Foundation License List, Public Domain**)

(See also **GPL FAQ, Question "Can the US Government release improvements to a GPL-covered program?"**)

Q: Can contractors develop software for the government and then release it under an open source license?

In many cases, yes, but this depends on the specific contract and circumstances. Under the "default" DFARS and FAR rules and processes, the contractor often keeps and exercise the rights of a copyright holder, which enables them to release that software as open source software (as long as other laws and regulations are met).

For DoD contractors, if the standard DFARS contract clauses are used (in particular DFARS 252.227-7014) then the contractor who developed the software retains the copyright to the software and has the right to release it to others, even if the software was developed exclusively with government funds. In some cases a DoD contractor may be required to transfer copyright to the government for works produced under contract (see DFARS 252.227-7020). If this is the case, then the contractor cannot release the software as OSS without permission, because the contractor doesn't own the copyright.

Contractors for other federal agencies may have a different process to use, but after going through a process they can often release such software as open source software. If the contract includes the typical FAR 52.227-14 (Rights in data - general) clause, without any special alternatives or additions, then the contractor must make a written request for permission to assert copyright in works containing data first produced under the contract. As described in FAR 27.404-3, a contracting officer would generally grant such a request. Certain FAR clause alternatives (such as FAR 52.227-17) require the contractor to assign the copyright to the government. Again, if this is the case, then the contractor cannot release the software as OSS without permission, because the contractor doesn't own the copyright.

There are many alternative clauses in the FAR and DFARS, and specific contracts can (and often do) have different agreements on who has which rights to software developed under a government contract. The FAR and DFARS specifically permit different agreements to be struck (within certain boundaries). Thus, if there is an existing contract, you *must* check the contract to determine the specific situation; the text above merely describes common cases.

Contractors must still abide with all other laws before being allowed to release anything to the public. Obviously, contractors cannot release anything (including software) to the public if it is classified. The release of the software may be restricted by the International Traffic in Arms Regulation or Export Administration Regulation. The release may also be limited by patent and trademark law.

Q: Can the government release software under an open source license if it was developed by contractors under government

contract?

In many cases, yes, but this depends on the specific contract and circumstances. Under the usual "default" rules, the answer is "yes" if it was developed for the DoD under the DFARS. Under the "default" rules, the answer is typically "no" if it was developed for under the default FAR rules (used by many other federal agencies) unless the contract transferred the copyright to the government or was modified in some way to permit it.

If the contractor was required to transfer copyright to the government for works produced under contract (e.g., because the FAR 52.227-17 or DFARS 252.227-7020 clauses apply to it), then the government can release the software as open source software, because the government owns the copyright.

Under the DFARS, which is typically used for DoD contracts, the government can release software as open source software once it receives "unlimited rights" to that software. DFARS 252.227-7014(a)(15) defines "unlimited rights" as "rights to use, modify, reproduce, release, perform, display, or disclose computer software or computer software documentation in whole or in part, in any manner and for any purpose whatsoever, and to have or authorize others to do so". As noted in "Technical Data and Computer Software: A Guide to Rights and Responsibilities Under Federal Contracts, Grants and Cooperative Agreements" by the Council on Governmental Relations (CAGR), "This unlimited license enables the government to act on its own behalf and to authorize others to do the same things that it can do, thus giving the government essentially the same rights as the copyright owner." In short, once the government has unlimited rights, it has essentially the same rights as a copyright holder, and can then use those rights to release that software under a variety of conditions (including an open source software license), because it has the use and modify the software at will, *and* has the right to authorize others to do so.

If the standard DFARS contract clauses are used (see DFARS 252.227-7014), then unless other arrangements are made, the government has unlimited rights to a software component when (1) it pays entirely for the development of it (see DFARS 252.227-7014(b)(1)(i)), or (2) it is five years after contract signature if it partly paid for its development (see DFARS 252.227-7014(b)(2)). Before award, a contractor may identify the components that will have more restrictive rights (e.g., so the government can prefer proposals that give the government more rights), and under limited conditions the list can be modified later (e.g., for error correction). Where possible, software developed partly by government funds should broken into a set of smaller components at the "lowest practicable level" so the rules can be applied separately to each one. Note, however, that this may be negotiated; if the government agrees to only receive lesser rights (such as government-purpose rights or restricted rights) then the government does *not* have the rights necessary to release that software as open source software.

The rules for many other U.S. departments may be very different. Contracts under the federal government FAR, but not the DFARS, often use clause FAR 52.227-14 (Rights in Data - General). If all defaults are accepted, and no additional alternatives/amendments are added, by default the government does not receive the right to distribute to the public software it paid to develop; see FAR 52.227-14(c)(1)(iii). (This is actually a special case; the government normally *does* have the right to public release of copyrighted works it paid to develop.)

There are many alternative clauses in the FAR and DFARS, and specific contracts can (and often do) have different agreements on who has which rights to software developed under a government contract. The FAR and DFARS specifically permit different agreements to be struck (within certain boundaries). Thus, if there is an existing contract, you must check the contract to determine the specific situation; the text above merely describes common cases.

If the intent of a contract is to develop software to be released as open source software, it is best to expressly include release as OSS as part of the contract. This makes the expectations clear to all parties, which may be especially important as personnel change.

Other laws must still be obeyed. Classified information may not be released to the public without special authorization to do so. The release of the software may be restricted by the International Traffic in Arms Regulation, or Export Administration Regulation. The release may also be limited by patent and trademark law.

Q: Does releasing software under an OSS license count as commercialization?

In most cases, yes. U.S. law governing federal procurement (**U.S. Code Title 41, Chapter 7, Section 403**) defines "commercial item" as including "Any item, other than real property, that is of a type customarily used by the general public or by non-governmental entities for purposes other than governmental purposes (i.e., it has some non-government use), and (i) Has been sold, leased, or licensed to the general public; or (ii) Has been offered for sale, lease, or license to the general public ...". Thus, as long as the software has at least one non-governmental use, software released (or offered for release) to the public is a commercial item for procurement purposes, *even if* it was originally developed using public funds.

This does not mean that organizations will automatically arise to help develop/support it. Whether or not this will occur depends on factors such as the number of potential users (more potential users makes this more likely), the existence of competing OSS programs (which may out-compete the newly released component), and how difficult it is to install/use. Thus, components that have the potential to (eventually) support many users are more likely to succeed. Similarly, delaying a component's OSS release too long may doom it, if another OSS component is released first. If the OSS is intended for use on Linux/Unix systems, **follow standard source installation release practices** so that it is easier for users to install.

Q: What license should the government or contractor choose/select when releasing open source software?

It depends on the goals for the project, however, here are some guidelines:

- **Public domain where required by law.** You *must* release it as "public domain" (when releasing it at all) if it was developed by a US government employee as part of their official duties. Otherwise, choose some existing OSS license, since all existing licenses add some legal protections from lawsuits. (The "MIT license" is similar to public domain release, but with some legal protection from lawsuits.)
- **Release modifications under same license.** If it is a modification of an existing project, or a plug-in to it, release it under the project's original license (and possibly other licenses). This way, the software can be incorporated in the existing project, saving time and money in support.
- **Consider anticipated uses.** If it must work with other components, or is anticipated to work with other components, ensure that the license will permit those anticipated uses. In particular, will it be directly linked with proprietary or classified code?
- **Make sure it's really OSS.** Choose a license that has passed legal reviews and is clearly accepted as an OSS license. Choose a license that is recognized as an **Open Source Software license by the Open Source Initiative (OSI)**, a **Free Software license by the Free Software Foundation (FSF)**, *and* is acceptable to widely-used Linux distributions (such as being a **"good" license for Fedora**).
- **Use a widely-used existing license.** Choose a widely-used existing license; do *not* create a new license. This eliminates future incompatibility and encourages future contributions by others. **Bruce Perens noted back in 1999**, "Do not write a new license if it is possible to use (a common existing license)... The propagation of many different and incompatible licenses works to the detriment of Open Source software because fragments of one program cannot be used in another program with an incompatible license." **Many view OSS license proliferation as a problem**; **Serdar Yegulalp's 2008 "Open Source Licensing Implosion" (InformationWeek)** noted that not only are there too many OSS licenses, but that the "consequences for blithely creating new ones are finally becoming concrete... the vast majority of open source products out there use a small handful of licenses... Now that open source is becoming (gasp) a mainstream phenomenon, using one of the less-common licenses or coming up with one of your own works against you more often than not". As an aid, the **Open Source Initiative (OSI) maintains a list of "Licenses that are popular and widely used or with strong communities"**. Another useful source is the list of **licenses accepted by the Google code hosting service**. See the licenses listed in the FAQ question "What are the major types of open source software licenses?".
- **Choose a GPL-compatible license.** The GNU General Public License (GPL) is the most common OSS license; while you do not need to use the GPL, it is often unwise to choose a license incompatible with the majority of OSS. Thus, avoid releasing software under only the original ("4-clause") BSD license (which has been replaced by the

"new" or "revised" 3-clause licence), the "Academic Free License" (AFL), the now-abandoned "Common Public License" 1.0 (CPL), the "Open Software License" (OSL), or the "Mozilla Public License" (MPL).

- **Choose a license that best meets your goals.** Choosing between the various options - particularly between permissive, weakly protective, and strongly protective options - is perhaps the most difficult, because this selection depends on your goals, and there are many opinions on which licenses are most appropriate for different circumstances. A "permissive" license permits arbitrary use of the program, including making proprietary versions of it. A "protective" license "protects" the software from becoming proprietary, and instead enforces a "share and share alike" approach between parties. A "weakly-protective" license is a compromise between the two, preventing the covered library from becoming proprietary yet permitting it to be embedded in larger proprietary works. If the goal is maximize the use of a technology or standard in a variety of different applications/implementations, including proprietary ones, permissive licenses may be especially useful. However, if the goal is to encourage longevity and cost savings through a commonly-maintained library or application, protective licenses may have some advantages, because they encourage developers to contribute their improvements back into a single common project. In many cases, weakly protective licenses are used for common libraries, while strongly protective licenses are used for applications. Common licenses for each type are:
  - Permissive: MIT, BSD-new, Apache 2.0
  - Weakly protective: LGPL (version 2 or 3)
  - Strongly protective: GPL (version 2 or 3)

Licenses that meet all the criteria above include the **MIT license**, **revised BSD license**, the **Apache 2.0 license** (though Apache 2.0 is only compatible with GPL version 3 not GPL version 2), the **GNU Lesser General Public License (LGPL)** versions 2.1 or 3, and the **GNU General Public License (GPL)** versions 2 or 3.

In some cases, it may be wise to release software under multiple licenses (e.g., "LGPL version 2.1 and version 3", "GPL version 2 and 3"), so that users can then pick which license they will use. This can increase the number of potential users.

Q: How should I create an open source software project?

First, get approval to publicly release the software. One way to deal with potential export control issues is to make this request in the same way as approving public release of other data/documentation.

If it is an improvement to an existing project, release it to the main OSS project, in whatever format they prefer changes. Many prefer "unified diff patches", generated by "diff -u" or similar commands. Most projects prefer to receive a set of smaller changes, so that they can review each change for correctness.

If it is a new project, be sure to remove "barriers to entry" for others to contribute to the project:

- Use a common OSS license well-known to be OSS (GPL, LGPL, MIT/X, BSD-new, Apache 2.0) – don't write your own license
- Establish project website. Typically this will include source code version management system, a mailing list, and an issue tracker.
- Document the project's purpose, scope, and major decisions - users must be able to quickly determine if this project might meet their needs.
- Use typical OSS infrastructure, tools, etc. Requiring the use of very unusual development tools may impede development, unless those tools provide a noticeable advantage.
- Maximize portability, and avoid requiring proprietary languages/libraries unnecessarily. The more potential users, the more potential developers.
- The released version *Must run*. Small-but-running is better than big-and-not.
- Establish vetting process(es) before government will use updated versions (testing, etc.)
- Determine if there will be a government-paid lead.

Some documents that may help include:

- **"Producing Open Source Software: How to Run a Successful Free Software Project" by Karl Fogel**
- **Free Software Project Management HOWTO**
- **Software Release Practice HOWTO**
- **Recognizing and Avoiding Common Open Source Community Pitfalls**

Q: In what form should I release open source software?

OSS should be released using conventional formats that make it easy to install (for end-users) and easy to update (for potential co-developers). These formats may, but need not, be the same.

If you are releasing OSS source code for Unix-like systems (including Linux and MacOS), you should follow the usual conventions for doing so as described below:

- **Releasing Free/Libre/Open Source Software (FLOSS) for Source Installation**
- **GNU Coding Standards, especially on the release process**
- **Software Release Practice HOWTO**

Q: Where can I release open source software that are new projects to the public?

You may use existing industry OSS project hosting services such as **SourceForge**, **Savannah**, **Tigris**, **Google code**, **Apache Software Foundation** or **Microsoft CodePlex**. Each hosting service tends to be focused on particular kinds of projects, so prefer a hosting service that well-matches the project. Using industry OSS project hosting services makes it easier to collaborate with other parties outside the U.S. DoD or U.S. government.

DISA's **Forge.mil** is "a family of services provided to support the DoD's technology development community. The system currently enables the collaborative development and use of open source and DoD community source software. These initial software development capabilities are growing to support the full system life-cycle and enable continuous collaboration among all stakeholders including developers, testers, certifiers, operators, and users." It uses a variant of the software used by SourceForge.

If the project is likely to become large, or must perform filtering for public release, it may be better to establish its own website. Note that many of the largest commercially-supported OSS projects have their own sites.

Community Sites about OSS

Q: Where do OSS developers congregate and what conferences should I go to?

An outside DoD/IC discussion list can be found at: **Military - Open Source Software**.

The DoD CIO does not endorse any specific event or conference. That said, there have been a few conferences specifically focused on OSS in the government or military context, at which DoD CIO personnel have presented information on DoD policy and OSS. For example, in August 2009, there was a Military-OSS working group meeting in Atlanta, Georgia, info here **Mil-OSS**. In November 2009, The Government Open Source Conference (**GOSCON**) will be held in Washington, DC.

Atlanta, Georgia, this note **MIT OSS**. In November 2009, The Government Open Source Conference (**OSCON**) will be held in Washington, DC.

CHHABRA LAW FIRM, PC
ROHIT CHHABRA (SBN 278798)
Email: rohit@thelawfirm.io
257 Castro Street Suite 104
Mountain View, CA 94041
Telephone: (650) 564-7929

Attorney for Plaintiffs
Open Source Security Inc. &
Bradley Spengler

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN FRANCISCO DIVISION

OPEN SOURCE SECURITY INC. and BRADLEY SPENGLER

Plaintiff,

v.

BRUCE PERENS, and Does 1-50,

Defendants.

Case No.: 3:17-cv-04002-LB

PLAINTIFFS' OPPOSITION TO DEFENDENT'S MOTION FOR ATTORNEY'S FEES PURSUANT TO CALIFORNIA CODE OF CIVIL PROCEDURE § 425.16(C); DECLARATION OF ROHIT CHHABRA IN SUPPORT THEREOF; AND DECLARATION OF FEE EXPERT WITNESS WILLIAM NORMAN IN SUPPORT THEROF.

Hearing Date: March 29, 2018
Time: 9:30 a.m.
Location: Courtroom C, 15th Floor
Judge: Hon. Laurel Beeler

REDACTED VERSION OF DOCUMENT(S) SOUGHT TO BE SEALED – PURSUANT TO COURT ORDER

Table of Contents

Cases

I. INTRODUCTION AND BACKGROUND

Plaintiffs are a small business with one employee and three contractors providing a computer security product to a niche market of approximately 40 customers. Unlike Defendant, Plaintiffs did not have the means to afford a *dream team* of five attorneys and two support staff from a 5-star law firm, and had to resort to hiring a small law firm with one attorney. Ex. 1, Declaration of Rohit Chhabra (Chhabra Decl.) ¶3. Specifically, Plaintiffs were charged a reasonable hourly rate of $350/hour since this matter did not relate to any complex issues of Intellectual Property law. Chhabra Decl. ¶¶4, 7. If the Court were to grant Defendant's ridiculous and outrageous fee demands for a relatively simple matter, it would not only be unjustified but would perhaps also fulfill the ultimate objective of Defendant's blog post – to have "the desired effect"[1] of hurting Plaintiffs' business. No complex legal question was presented in this matter; specifically no issue related to intellectual property was presented or argued

Based on the Court's December 21, 2017 Order, Plaintiffs agreed that Defendant is the prevailing party for the anti-SLAPP motion and statutorily is entitled a reasonable attorneys' fee award. However, Defendant attempts to justify his outrageous fee demand claims and the hiring of a multi-million dollar law firm with specialization in intellectual property law, by stating that the underlying matter in this case was related to a complex legal issue involving intellectual property law. **Defendant's contention is patently incorrect.** The underlying premise of Plaintiffs arguments was based on a relatively simple legal argument whether, based on existing case law and American Jurisprudence, Plaintiffs could be held in violation of the GNU General Public License (GPL), and whether Defendant's statements based on his reputation could be considered as offering lay person opinion. *See* First Amended Complaint (FAC) ¶¶ 30 – 32, 49 (alleging that **all** the statements are false

[1] *See* FAC ¶ 45

because Plaintiffs' Access Agreement did not violate the GPL based on the principle that Plaintiffs had a right to choose their future business patrons); Plaintiffs never presented any argument related to any issue of intellectual property law. *See generally*, Motion for partial summ. judgment (ECF No. 24); Opposition to anti-SLAPP motion (ECF No. 38). Defendant incorrectly states that the FAC asserted complex legal issues. To the contrary, Plaintiffs claimed all nine statements presented by Defendant were false "**because** the Access Agreement does not violate the GPLv2". FAC ¶49 (emphasis added). Furthermore, neither did this Court find any complex legal issues to make its determination in this matter. *See* Order Dated Dec. 21. 2017 (ECF No. 53). Plaintiffs' counsel retained Attorneys' fee expert witness William Norman, to provide a fair and unbiased evaluation in this matter. Chhabra Decl.¶ 9. No sealed information was provided to Mr. Norman. *Id.* Neither Plaintiffs, nor Plaintiffs' counsel, or its agents or representatives asked Mr. Norman to modify or revise his assessment. Ex. 2, Declaration of Fee Expert Witness William Norman (Norman Decl.) ¶ 2; Chhabra Decl.¶ 9. Mr. Norman has 47 years of experience, has handled several complex business litigation matters, including approximately 15 anti-SLAPP matters; he has also appeared as an attorneys' fee expert on several occasions. Norman Decl. ¶1. A true and correct copy of Mr. Norman's publicly available experience and professional biography is attached hereto as Ex. 3.[2]

Mr. Norman also agrees with Plaintiffs' contention that nothing in this matter required a huge law firm with attorneys specializing in Intellectual Property matters; there was no need to hire an intellectual property based legal team with their exorbitant hourly rates. Norman Decl. ¶ 6. However, despite that, Defendant under seal submits Detailed Billing Entries (ECF No. 67 (Hansen Decl.), Ex. C) (Timekeeper Records) and demands $478,977.50 in attorneys' fees and an additional award of $188,687. 75 as a "success fee." Decl. Hansen ¶4. Not only does the demanded fee show inefficient

[2] available at: http://www.cwclaw.com/attorneys/attorneyBio.aspx?name=WilliamNorman

management, Defendant mistakes and forgets the legislative purpose of Code of Civil Procedure § 425.16(C) is to entitle recovery of reasonable attorneys' fees and costs, and not to unjustly enrich his counsels with outrageous and unfettered attorneys' fees or unconscionable alternative fee agreements like "success fees." Both are unreasonable and appalling by any standard for determining attorney's fees and costs in this matter. Norman Decl. ¶ 7, 10.

Exorbitant and unreasonable hourly attorney billing rates

Defendant also seeks recovery of fees based on hourly fee charges that exceed by hundreds of dollars per hour the average billing rates charged in the relevant legal community. Not only is Defendant's *dream team* overstaffed, their billing rates claimed can only be considered reasonable in ones' dream! Notably, all three associates who worked on this matter **were admitted in California in 2017**; two claimed "associates" **were not even attorneys (in any jurisdiction)** until December 2, 2017 – that is, 12 days before the Dec. 14 hearing in this matter. Chhabra Decl. ¶ 10. It is patently unreasonable to bill out a paralegal at [redacted], two non-attorney "interns" at [redacted] (for less than a month, ignoring December holidays, and then interestingly enough increasing their hourly rates to [redacted]) , a first year associate from [redacted] in 2017 to [redacted] in 2018, and two partners each billed at [redacted] and [redacted], and [redacted] and [redacted]/hour for 2017 and 2018 respectively. It is astonishing that even Defendant's paralegal has a claimed hourly billable higher that Plaintiffs' counsel; this is completely unheard of, even in matters involving complex intellectual property (patent) related issues. This further becomes extremely outrageous since Plaintiffs never argued any complex legal question in this matter related to a complex intellectual property issue. It is also problematic that Defendant argues that he needed to hire a multi-million dollar intellectual law firm because Plaintiffs sought to recover 3 Million dollars in damages from an individual. This is incorrect. Complaints are routinely filed based on information and belief, and Plaintiffs sought a recovery of 3 Million dollars in damages from Defendant and Does 1 -50 (a total of 51 defendants). Since discovery was never

initiated, Plaintiffs were never able to ascertain the correct number of defendants in this matter. Specifically, Plaintiffs sought damages of "an amount *to be determined at trial*, but in excess of $75,000 as to each defendant." FAC ¶ 86 (italics added). Thus, this also does not justify Defendant's imprudent actions of hiring a multi-million dollar law firm with specialization in intellectual property law. Further, Plaintiffs also cannot be held responsible for Defendant's counsel's actions for undertaking this non-intellectual property matter, when this matter could have easily been represented by any non-intellectual property lawyer. *See* Norman Decl. ¶ 6.

Serious Mismanagement Concerns

The unreasonableness and the inefficiency can also be recognized by Defendant's counsel's having assigned seven different time billers to the matter, including two partners, two non-admitted "associates" (legal interns) and a first year associate to the defense of this matter. As opined by attorneys' fee expert, Mr. Norman, "[m]ore timekeepers, especially those duplicating other timekeepers in the same levels, are extremely inefficient. Confusion, extra management time by the team leader, and excess intra-office conferencing result in greater cost and they often compromise the overall effort." Norman Decl. ¶ 7(b).

Furthermore, given the fact that 82%[3] of the billable hours involved work performed by non-admitted "interns" and a first year associate, the outrageous and absurd character of Defendant's demands can be recognized by determining a per page attorneys' fee charged:

- 137. 9 hours for a *23 page* First Anti-SLAPP Motion plus accompanying one page declaration totaling $83,606.50 (**$3,463 per page**);
- 77.6 hours for a mostly duplicative *24 page* Second anti-SLAPP motion and accompanying one page declaration totaling $43,669.50 (**$1,746.78 per page**);

[3] See Def. Motion' for Attys' fees at 13:9

- 109.5 hours for a *15 page* reply for the Second anti-SLAPP totaling $60,803.50 (**$4,053 per page**);
- 19.6 hours for a *3 page* Response to Plaintiff's supplemental brief totaling $10,798.50 (**$3,599 per page**);
- 87.5 hours for a *10 page* opposition to partial summary judgment with one page declaration totaling $49, 813 (**$4,528 per page**);
- 29.2 hours for a *6 page* motions for Surreply and Surreply to partial summary judgment totaling $17,477 **($2,912 per page)**;
- 131.8 hours for a 19 page motion for attorneys' fees, a 7 page declaration, and a 28 page expense report (totaling *59 pages*) for $76,602 **($1,298 per page).**

Other outrageous fees claimed by Defendant are:

- 141.6 hours for preparing for a Court hearing; needlessly involving excessive staff who played no active role in the hearing;
- 86.9 hours claimed by Defendant's counsel for case management; and
- 12.3 hours for three settlement communications via email (see Decl. Chhabra ¶ 11); and
- $188,167.75 unconscionable success fee that should be denied. Ninth Circuit law specifically prohibits success fee multipliers in statutory fee shifting awards, as discussed further herein.

Indeed, Defendant's lead counsel, Ms. Hansen, throughout her litigation career of more than a decade has successfully defended clients in defamation actions.[4] Decl. Hansen ¶ 9. Therefore, with

[4] An attorney who has defended defamation claims through the course of her decade long litigation career should reasonably know how to efficiently draft anti-SLAPP motions.

more than a decade of experience defendant clients in defamation actions, she should have reasonably exercised proper judgment and should have steered her inexperienced interns and first year associate in a manner that would have significantly reduced the needless hours of research and reviews performed by all three junior unexperienced researchers and motion drafters.

Even Plaintiffs' counsel, with no prior experience in defamation cases (but otherwise not new to addressing complex litigation matters), has been significantly more efficient than Defendant's *dream team* by singlehandedly addressing this matter, without any support staff, and by billing Plaintiffs a total of $80,175, for the entirety of this matter, representing 229 billable hours. Chhabra Decl. ¶ 5. In fact, Plaintiffs' counsel has not billed Plaintiffs more than 40 hours (generally less) for any motion or pleading, illustrating that there was no need for significant research of any complex legal issue. Chhabra Decl. ¶ 7.

Therefore, **the question now presented to the Court is** – If Plaintiffs' counsel, with no prior experience in addressing defamation cases, was able to provide efficient representation to his clients, why couldn't Defendant's counsel do the same for handling a substantially similar amount of work? Arguably, had Plaintiffs' counsel employed interns and junior associates, the fee charged to Plaintiffs would have been further lowered. Clearly, Defendant has failed to provide substantial evidence justifying such egregious mismanagement that warrants ***8.5 times the amount Plaintiffs' were charged for handling the same work.***

Respectfully, there can be no reasonable justification. While Ms. Hansen has not claimed she has "significant experience" in handling defamation cases, a decade long career of defending defamation cases is nothing less than substantial and impressive; it is enough to provide efficient management skills. The Court should therefore consider her experience, the number of hours expended by Plaintiffs' counsel, declaration of fee expert, Mr. Norman, and the Court's own expertise and experience in addressing similar actions to determine a reasonable fee award. See *Maughan v. Google*

Technology, Inc., 143 Cal.App.4th 1242, 1248-1251, 1253 (2006) ("the court determining based on its own experience and expertise in handling complex civil cases, reduced a $112,288.63 anti-SLAPP fee claim to $23,000 by reducing the claimed hours on the SLAPP motion from over 200 hours to 50 hours and further considering the attorney's experience handling such matters.); *Pecot v. Wong*, Case No. A139566, at *3, (Cal. Court of Appeal, 1st Appellate Dist., 4th Div., Jan. 18, 2018) ____ WL ____ (unpublished) (affirming a reduction of anti-SLAPP fee claim for approximately 159 claimed hours to $20,000 by reducing the number of hours and determining a reasonable fee, based on the court's own expertise and experience and considering fee expert witness testimony).

Furthermore, as explained below, 48% of the detailed Timekeeping Records have substantial miscalculations, showing inconsistent billing practices, and cannot be considered as a reliable source of evidence. Thus, Defendant cannot satisfy his burden and establish that the claimed charges are reasonable.

In order to assist the Court, Plaintiffs undertook the mammoth project of providing a detailed analysis of the Timekeeping Records, calculations (including corrections of the 48% errors), and determining a reasonable fee award along with the basis thereof, as discussed further below.

II. THE COSTS AND FEES DEFENDANTS REQUEST IN THEIR EXPENSE REPORT ARE UNREASONABLE, EXCESSIVE, AND UNNECESSARY

State law governs attorney's fees awards based on state fee-shifting laws, like California's anti-SLAPP statute. See *Northon v. Rule*, 637 F.3d 937, 938 (9th Cir.2011). The Northern District has recognized that a prevailing defendant, under section 425.16(c), shall only be entitled to recover attorney's fees and costs that a court deems are reasonable. *Loop AI Labs Inc. v. Gatti*, Dist. Court, Case No. 15-cv-00798-HSG at *2 (N.D. Cal. May 9, 2016) (citing *Minichino v. First California Realty*, No. C-11-5185 EMC, 2012 WL 6554401, at *3 (N.D. Cal. Dec. 14, 2012)); *Robertson v.*

Rodriguez, 36 Cal. App. 4th 347,362 (1995); *Dove Audio, Inc. v. Rosenfeld, Meyer & Susman*, 47 Cal. App. 4th 777, 785 (1996).

The proper method for calculating attorney's fees in California is the lodestar method. See *Ketchum v. Moses*, 24 Cal. 4th 1122, 1136 (2001). In assessing attorney's fees under this method, however, a Court must exclude those fees that are "excessive, redundant, [and] otherwise unnecessary." *Hensley v. Eckerhart*, 461 U.S. 424, 434 (1983); *see also Serrano v Priest*, 20 Cal 3d 25, 48 (1997) (explaining that a court assessing attorney fees begins with a lodestar figure that is based on the "careful compilation of the time spent and reasonable hourly compensation of each attorney ... involved in the presentation of the case.")

Since the Court's "role is not merely to rubber stamp the defendant's request, but to ascertain whether the amount sought is reasonable," *Robertson* at 361, any fee award must be established by "substantial evidence" supporting the award. *Macias v. Hartwell*, 55 Cal. App. 4th 669, 676 (1997). Therefore, the Court is "not bound by the amount sought by defendants and [has the] discretion to award them a lesser sum." *Robertson* at 362. Because Defendant requests an award that is unreasonable and excessive, Defendant's request for attorney's fees and costs must be substantially reduced.

A. A SUBSTANTIAL PORTION OF THE FEES AND COSTS SUBMITTED BY DEFENDANT IS NOT RECOVERABLE UNDER § 425.16(C)

Defendant presumes that he only had the right to file an anti-SLAPP motion and that no other motion could (or should) have been filed prior to the hearing of the anti-SLAPP motion. *See* Mot. Atty. Fees' at 3-4. However, Defendant fails to provide any case law that supports his contention.

The motion for partial summary judgment arose out of the facts based on statements made by Defendant – prior to the filing of the anti-SLAPP motion(s). Even if Defendant had not filed his anti-SLAPP motion, Plaintiffs would have filed the motion for partial summary judgment based on the Defendant's prior statement, since an issue of fact existed that reasonably questioned Defendant's

belief in the truth of his statements presented in the blog post. *See* Motion for Partial Summary Judgment ("MPSJ") (ECF No. 24). And since complaints are regularly filed on belief and information, these statements provided a showing that Defendant agreed with Plaintiffs, and thus there was no genuine issue of fact. Indeed, Plaintiffs had an arguable legal theory and wanted to debate that matter first before the filing of the Second anti-SLAPP motion; had the Court agreed with Plaintiffs, there would have been no need to file the Second anti-SLAPP motion. *See* Norman Decl. ¶8.

California's Anti-SLAPP statute allows a movant to recover "only those fees and costs incurred in connection with the motion to strike, not the entire action." *Paul for Council v. Hanyecz*, 85 Cal.App.4th 1356 (2001). Plaintiffs, therefore, are not responsible to pay any fees that are applicable to non-SLAPP motion matters or both the anti-SLAPP motion and other aspects of the litigation. The statute limits recovery to costs and fees that apply only to the motion to strike and this is clearly a rule of reason insofar as the purpose of an attorney's fees award under § 425. 16(c) is to compensate defendants for the additional cost of litigating the anti-SLAPP motion. Insofar, as research would have necessarily been performed were the anti-SLAPP motion never filed, Defendant should not be able to recover those fees as well. Nonetheless, Defendant attempts to subsume all research relevant to both the SLAPP motion and other aspects of the litigation even though that research would have needed to be performed regardless of whether the anti-SLAPP motion had been filed.

However, even if the Court were to disagree with Plaintiffs' contention, there was a substantial duplication in the arguments presented in the anti-SLAPP motion and MPSJ; attorneys' fees to file those additional motions should have been minimal. Norman Decl. ¶8.

B. 48% OF THE TIMEKEEPING RECORDS HAVE "DOCTORED" FEES AND/OR HOURS CLAIMS AND CANNOT BE CONSIDERED AS A RELIABLE SOURCE; NEITHER CAN ANY DECLARATION BE CONSIDERED RELIABLE THAT IS BASED ON THE ERRONEOUS TIMEKEEPING RECORDS

Attorneys are required to "maintain accurate records of work done and time spent in preparing each client's case" as "a detailed billing record gains the advantage of being able to evaluate the worth

of the services provided." *Martino v. Denevi*, 182 Cal. App. 3d 553, 558 (1986). Even though testimony by any attorney regarding the number of hours worked is sufficient to justify that it is appropriate to grant attorney's fees, the reasonable value of the services rendered is still at the discretion of the Court. *Id.* at 558-59; see also *Wilkerson v. Sullivan*, 99 Cal. App. 4th 443, 448 (2002) (explaining that "[t]he reasonableness of attorney fees is within the discretion of the trial court.")

Although Defendant's counsel submits under declaration that the timekeeping records were contemporaneously maintained (Def. Fee Motion, 9: 21 -23; Hansen Decl. ¶20 - 27), and personally reviewed the records of fees and costs (Hansen Decl. ¶¶ 2, 26), approximately 48%[5] of the records have incorrect mathematical calculations by either presenting exaggerated hours claimed with substantially less fee listed, or by presenting exaggerated fees claimed for substantially less hours listed. Decl. Chhabra ¶ 13. With so many disparities one can reasonably infer that the Timekeeping Records are "doctored" for the sole purpose of meeting the purported amount and hours being claimed. Not only the number of hours and fees claimed for which Defendant seeks reimbursement is absurd, with 48% records reflecting incorrect calculations, the truthfulness and veracity of the Timekeeping Records, in its entirety, and any supporting Declaration therewith are justifiably questioned;[6] it is respectfully submitted the Timekeeper Records cannot be considered as trustworthy evidence, and thus any accompanying declaration relying on the Timekeeper Records should be stricken.

[5] 240 out of 502 records, excluding records related to sanctions.

[6] But of course Defendant's counsel is going to claim the 48% inaccuracies were an administrative "mistake," even after submitting a declaration, under penalty of perjury, that she reviewed "each of the billing records." However, it is improbable that a **700+ million** dollar law firm like O'Melveny would not even have the most primitive timekeeping software that can perform simple mathematical calculations. Furthermore, with seven resources working on this matter (out of which five are (now) attorneys), it is hard to believe this could be an "honest mistake." It can reasonably be inferred that Defendant's counsel **intentionally presented inaccurate data** (both in hours and time) to greatly exaggerate either the fee or hours claimed per task, while burying this data in a *7-point font* in an attempt to justify their unreasonable and outrageous fee claims.

Egregious examples of the claimed fee are:[7]

Excessive Fees Claimed in Timekeeping Records:

Date	Timekeeper	Timekeeper hourly claimed rate	Hours claimed	Fees claimed
[redacted]				
The list goes on. See Chhabra Decl. Exs. 1-A to 1-L for more details.				

Excessive Hours Claimed in Timekeeping Records:

Date	Timekeeper	Timekeeper hourly claimed rate	Hours claimed	Fees claimed
[redacted]				
The list goes on. See Chhabra Decl. Exs. 1-A to 1-L for more details.				

[7] Even after correcting the miscalculations, the mismanagement by Defendant's lead counsel can clearly be noted by the duplicative work and inefficiency performed by Defendant's *dream team*.

However, in good faith, Plaintiffs have corrected the claimed fees/ hours and summarized/ sorted them by "category," for the Court's convenience. *Id.*

Based on the corrected calculations Defendant's counsels' own timekeeping records indicate that the total inefficient and mismanaged hours they are in fact claiming is 642.3 hours, that is a reduction of 191.6 hours from the 833.9 hours claimed in Defendant's motion. Decl. Chhabra, Ex. 1-A. While Plaintiffs recognize that California does not require contemporaneously maintained records, and usually attorneys' declaration suffices for a fee motion, under the best evidence rule, any disparity and deviation in Defendant's counsel's declaration, from the contemporaneously maintained records, should be stricken out from such declaration.

Furthermore, with 48% errors, a question now exists as to the truthfulness and veracity of all the Timekeeping Records submitted by Defendant's counsels. Also, any attempt to provide "corrected" Timekeeping Records questions the premise of maintaining "contemporaneous" records and submitting them as proof. Therefore, Defendant has failed to provide sufficient information to determine whether the time spent and billed for various activities was, or was not, reasonable. With 48% of the timekeeping record not matching their claimed fees, it is fair to conclude Defendant's counsels have not maintained proper records and thus Defendant's counsels have failed to establish by "substantial evidence" supporting the award claimed. *Macias v. Hartwell*, *supra*, at 676. On these grounds, this motion should be dismissed with prejudice; however, at the very least the contemporaneous Timekeeping Records, and Ms. Hansen's declaration cannot and should not be considered trustworthy and should be stricken from record.

C. THE TIMEKEEPING RECORDS HAVE AMBIGUOUS OR INCOMPLETE INFORMATION

Furthermore, Defendant's Timekeeping Records are filled with incomplete and ambiguous information such that it is impossible to determine whether or not a particular expense is for purposes of the anti-SLAPP motion or training exercises for its junior associate and interns. The descriptions of

the Timekeeping Records do not provide any guidance whatsoever in determining whether a reasonable amount of time was spent on that activity. For most records, other than claiming that an activity for a motion was performed, there is no detail as to what specific portion of that activity was conducted. For example, there are numerous ambiguous expenses and duplicative entries related to:

a. 23 entries, excluding "sanctions" entries, related to "Conducting Legal Research" for a motion, "Conduct Additional Research," "Conduct Supplemental Research," and "Conduct related research" (and other variants) for a motion without providing anything more;

b. 107 entries, excluding "sanctions" entries, including conference, confer, or discussions, or additional conferences regarding a motion without providing more; and

c. 188 entries, excluding "sanctions" entries, related to revising or drafting a motion.

Decl. Chhabra ¶ 14.

Because the non-descript or ambiguous and duplicative billing expenses make it impossible to determine whether the time spent on those activities is reasonable, Plaintiffs cannot be obligated to pay for those expenses.

D. DUPLICATIVE, EXCESSIVE, IRRELEVANT AND INEFFICIENT PRACTICES

Counsels for Defendant are obligated to "make a good faith effort" to deduct from its Timekeeping Record and Expense Report all "hours that are excessive, redundant, or otherwise unnecessary, just as a lawyer in private practice ethically is obligated to exclude such hours from his fee submission." *Hensley*, 461 U.S. at 434; *see also Ketchum*, 24 Cal. 4th at 1132 (holding that "'padding' in the form of inefficient or duplicative efforts is not subject to compensation.") However, the record indicates Defendant's counsel has not done so. As illustrated numerous above, numerous duplicative and ambiguous records prevent reasonable time determinations.

However, 49 entries that do provide detailed research analysis performed. Decl. Chhabr Ex. 1-M. Those records highlight the duplication, inefficiency, and irrelevancy, which further reflects Ms.

Hansen's mismanagement in this matter (providing detailed objections on all the records). A few examples are:

1. C. Gagliano conducted "[redacted] on 10/3/17 – 10/5/17 and claimed [redacted] hours. Furthermore, if Defendant claims the first record of 10/3, highlighted in red, was a genuine mathematical error (and should have instead been [redacted] hours to justify the claimed $[redacted], then the total changes to [redacted] hours of "research" to determine [redacted] on a pending anti-SLAPP motion.

2. C. Gagliano spent [redacted] hours to "[redacted]" on 1/27/18 – 1/29/18 claiming [redacted]. Plaintiffs wonder what sort of "[redacted]" warranted [redacted] hours.

3. M. Rhoades spent [redacted] only to discover L.R. 79-5, on 1/23/18. Also, the Court's well written webpage on sealing documents shows up as the first link on Google when searching for "e-file under seal northern district."

These are just a few of the outrageous examples; a complete list of research activity (where details were provided in Timekeeping Records) and objections thereto are provided at Chhabra Decl. Ex. 1-A to 1-M. In sum, just for "research" Defendant's *dream team* claimed (without corrections) hours: [redacted]; Fee: [redacted]; and with corrected calculations, claimed: hours [redacted] fee: [redacted], for irrelevant, duplicative and/or inefficient research. Decl. Chhabra Ex. 1-M.

Plaintiffs cannot be held responsible for such inefficiencies.

E. MOTION DRAFTING: TIME SPENT DRAFTING, REVISING, CONFERRING AND RESEARCHING THE MOTIONS WAS EXCESSIVE

For the Court's convenience, Plaintiffs have compiled and sorted the Timekeeping records by motion, and also provided corrected calculations, where necessary, to illustrate the unreasonableness and exorbitant fees/hours claimed by Defendant in this matter along with a basis of objection. *See* Decl Chhabra Ex. 1-A through 1-M.

As can be observed, predominantly all entries state 'performing research', 'motion drafting', or 'conferences', and are repeated numerous times. Further, these entries have numerous errors, inconsistent billing calculations, and are irrelevant, duplicative, or simply show inefficiency. All objections are stated in the extreme right column of each entry.

Since no complex intellectual property claim was asserted, see FAC ¶ 49 (stating that all statements in Defendant's blog were false because Plaintiffs did not violate the GPL). Also see FAC ¶¶ 21, 22, 30 and 31 (explaining the basis of why Plaintiffs claimed that they did not violate the GPL). However, 33.3 hours were spent for researching this issue, without proper guidance to a first year associate. Defendant provides no justification why there has been extreme inefficiency, especially when no complex legal issue was presented.

In sum, since Ms. Hansen has over a decade of litigation experience and has handled several defamation matters (Decl. Hansen ¶ 9), she could have easily prevented such frivolous and needless research and actions. Plaintiffs cannot be held liable for a training school created by O'Melveny's attorneys.

F. HOURLY FEES CLAIMED ARE UNREASONABLE

"In determining a reasonable hourly rate, the district court should be guided by the rate prevailing in the community for similar work performed by attorneys of comparable skill, experience, and reputation." *Chalmers v. City of Los Angeles*, 796 F.2d 1205, 1210-11 (9th Cir. 1986) (citing *Blum v. Stenson*, 465 U.S. 886, 895 n.11). The relevant community for purposes of determining the prevailing market rate is generally the "forum in which the district court sits." *Camacho v. Bridgeport Fin., Inc.*, 523 F.3d 973, 979 (9th Cir. 2008).

In determining the reasonableness of Defendant's counsel's fees, this Court must weigh several factors including the attorney's skill required and employed in handling the matter, the attorney's learning, and the attorney's experience in the particular type of work. *Clejan v. Reisman*, 5 Cal. App.

3d 224, 241 (1970). The lodestar approach begins by multiplying "the numbers of hours reasonably expended [with the] reasonable hourly rate." *PLCM Group, Inc v. Drexler*, 22 Cal. 4th 1084, 1095 (2000) (emphasis added). In so doing, a court should use the prevailing rates of comparable private attorneys as the "touchstone" for determining a reasonable rate for an attorney. *International Longshoremen's Warehousemen's Union v. Los Angeles Export Terminal, Inc.*, 69 Cal. App. 4th 287, 303 (1999).

Notwithstanding Defendant's claims to the contrary, O'Melveny's hourly billing rate for its attorneys and support staff is outrageous. O'Melveny staffed this case with seven individuals representing six different billing rates ranging from [redacted]. All of these billing rates are in excess of the normal prevailing rate for attorneys practicing in San Francisco Bay Area, California, including Menlo Park and San Francisco and also exceeds the experience and similar expertise in this type of litigation. Norman Decl. ¶ ¶ 1, 2, and 7(a).

It is respectfully submitted, Defendant cannot provide any reasonable justification why intellectual property attorneys from a huge law firm were selected to represent him in this matter, and thus his counsel's fee should be adjusted accordingly.[8]

Mr. Norman has provided estimated maximum hourly rates based on the complexity in this matter ranging from $180 to $550 per hour. Norman Decl. ¶7(a). In fact, Mr. Norman's estimated hourly rates exceed those that have been approved and recognized by various courts discussing similar, if not more, complex legal issues, in the Northern District. *See LOOP AI LABS INC. v. Gatti, Dist. Court, 5-cv-00798-HSG (*finding that the requested hourly rates of $230 for associates having four years experience, $365 per hour for attorney with 16 years of experience in complex intellectual

[8] Plaintiffs' counsel, while primarily an intellectual property attorney, offered a discounted hourly rate to Plaintiffs in this matter (and has since maintained the same rate) as it was reasonably determined that this was not going to be an intellectual property related matter. See Chhabra Decl. __.

property litigation matters, and $440 per hour for a partner with are reasonable are within the range of reasonable rates in the Northern District [of California]); citing *Henry v. Bank of Am. Corp.*, No. C 09-0628 RS, 2010 WL 3324890, at *3 (N.D. Cal. Aug. 23, 2010) (approving rates of $225 per hour for an associate and $515 per hour for the partner); *Minichino*, 2012 WL 6554401, at *5 (finding attorneys with nine and fourteen years of experience reasonably had billing rates ranging from $450-555); *Braden v. BH Fin. Servs., Inc.*, No. C 13-02287 CRB, 2014 WL 892897, at *6-7 (N.D. Cal. Mar. 4, 2014) (approving rates of $610 per hour for partners, and $310 per hour for managing attorney with over eight years of experience)).

Furthermore, even if this is considered as a complex matter, Mr. Norman's expert testimony as to the prevailing rate for a Bay area attorney are comparable to the Laffey Matrix, when adjusted to the Bay area, which are a widely recognized compilation of attorney and paralegal rate data which is regularly prepared and updated by the Civil Division of the United States Attorney's Office for the District of Columbia and used in fee shifting cases in complex litigation matters and frequently accepted by the Northern District. *See* https://www.justice.gov/usao-dc/file/796471/download (last visited March 7, 2018).[9] As noted by former Chief Judge Walker of this Court, "adjusting the Laffey matrix figures upward by approximately 9% will yield rates appropriate for the Bay area" by using the locality pay differentials within the federal courts as a reference. *In re HPL Technologies, Inc. Securities Litigation*, 366 F. Supp. 2d 912, 922 (N.D. Cal. 2005) (determining the Laffey Matrix as a "well-established objective source for rates" and finding it adequate for a complex securities fraud class action).

In fact, as late as last month Chief Magistrate Judge Hon. Joseph C. Spero recognized the Laffey Matrix, adjusted to the Bay area as an accurate prevailing rate. *Lane Zhao v. SuminTsai*, 17-cv-

[9] A true and correct copy is attached hereto. Chhabra Decl. ¶ 15, Ex. 4

07378-JCS (N.D. Cal, Feb. 2018); Also see *Brinker v. Normandin's* 14-cv-03007-EJD (HRL) (N.D. Cal., Feb 2017); *Garcia v. Stanley*, 14-cv-01806-BLF, (N.D. Cal. March 2017) (finding an hourly rate of $500/hour in the San Francisco Bay area reasonable when the Laffey Matrix provides a reference range of from $608 to $747 per hour) (citing *In re HPL Technologies, Inc. Securities Litigation, supra*). Plaintiffs confirm, according to the locality pay differentials within the federal courts, Judge Walker's assessment of approximately 9% upwards differential for the Bay area remains correct as of today. Decl. Chhabra ¶ 16, Ex. 5, 6.

The following table shows the comparable rates between Mr. Norman's unbiased assessment and the adjusted Laffey Matrix. Further, since a substantial amount of work in this matter was performed in 2017 (with the exception of the fees motion itself), using the Laffey Matrix of 2016-2017, provides an adequate reference point to determine the prevalent rate, even if this matter is considered as a complex legal matter:

Experience	**Per hour rates 2016-2017 (Laffey)**	**Per hour rates 2016-2017 (Laffey adjusted 9% for San Francisco Bay Area)**	**Mr. Norman's estimated hourly fee for this matter based on its complexity (See Norman Decl. 7(a))**
21-30 years experience	$543	$591	$475 - $550
11-15 years experience	$465	$507	$425 - $450
Less than 2 years experience	$291	$317	$230 - $240
Less than 1 year experience	$240[10]	$261	$210 - $215
Law Clerk/ non-admitted	$157[11]	$171	$180 - $195
Paralegal	$157	$171	$190 - $220

[10] No data is provided for an associate with less than 1 year experience in the Laffey Matrix (for a complex litigation matter), but a good faith estimate is provided based on Mr. Norman's estimated maximum for a relatively simple matter.

[11] See Laffey Matrix fn.6, attorney not admitted to bar compensated at "Paralegals & Law Clerks" rate.

It should be noted that while the Laffey Matrix is generally considered as an average fee for complex litigation, Mr. Norman has not determined this matter to be a complex issue and thus his fee estimates are understandably lower than the rates displayed in the Laffey Matrix. Thus, Plaintiffs request this Court to consider Mr. Norman's hourly rate assessment as more accurate than the Laffey Matrix. Nonetheless, the above, provides substantial evidence that any attorney hourly rate determination higher than the adjusted Laffey Matrix in this matter should be considered as unwarranted.

G. NO SUBSTANTIAL FEE CLAIMS SINCE 2018 ARE WARRANTED

Defendant's counsels claim to have substantially worked on this matter in 2018. Specifically, Defendant's counsels claim to have expended H. Meeker ([redacted] hours); M. Hansen ([redacted] hours); C. Gagliano ([redacted] hours); E. Ormsby ([redacted] hours); M. Rhoades ([redacted] hours). However, except for three terse email communications, Plaintiffs have not communicated with Defendant (except when directed by the Court on January 18, 2018). Chhabra Decl. ¶ 11. Thus, except for the fees motion the Court should strike any hours claimed by Defendant. Moreover, the demonstrated inefficiency and duplicative work performed by Defendant's counsels existed throughout this matter, and therefore the hours claimed are unjustified

H. ACCORDING TO NINTH CIRCUIT LAW SUCCESS FEE AGREEMENTS PROVIDING MULTIPLIERS ON FEE SHIFTING CASES ARE NOT ALLOWED

In Federal Court, contingency multipliers are not allowed in fee shifting cases. *See Gates v. Deukmejian*, 987 F.2d 1392 (9th Cir. 1992). In *Gates*, the court, in declining to apply a multiplier on a contingency case in the fee shifting context stated:

> In *Dague* the Supreme Court addressed whether, in determining an award of attorney's fees under section 7002(e) of the Solid Waste Disposal Act, 90 Stat. 2826, as amended 42 U.S.C. § 6972(e), or section 505(d) of the Federal Water Pollution Control Act, 86 Stat. 889, as amended, 33 U.S.C. § 1365(d), a court "may enhance the fee above the 'lodestar' amount in

> order to reflect the fact that the party's attorneys were retained on a contingent-fee [**31] basis and thus assumed the risk of receiving no payment at all for their services. *City of Burlington v. Dague*, 120 L. Ed. 2d 449, 112 S. Ct. 2638, 2639 (1992).
>
> In its June 24, 1992 opinion in *Dague* the Court answered this query with a resounding "no," when it held "that enhancement for contingency is not permitted under the fee shifting statutes." *Id.* at 2643-44. Although the Solid Waste Disposal and Federal Water Pollution Control Acts and not § 1988 were at issue in *Dague*, the *Dague* Court expressly noted that the language of both of these sections "is similar to that of many other federal fee-shifting statutes, see, e.g., 42 U.S.C. §§ 1988, 2000e-5(k), 7604(d); our case law construing what is a 'reasonable' fee applies uniformly to all of them." *Id.* at 2641 (citing *Flight Attendants v. Zipes*, 491 U.S. 754, 758 n. 2, 105 L. Ed. 2d 639, 109 S. Ct. 2732 (1989)).
>
> Given the Court's holding in *Dague*, it is clear that contingency multipliers are no longer permitted under § 1988. Thus, we reverse the portion of the district court's amended order awarding a 2.0 [**32] contingency multiplier in this case.

Gates, 987 F.2d at 1403.

Defendant, on the other hand cites no Ninth Circuit case law to justify his position. However, if this Court declines to apply Ninth Circuit law, which Plaintiffs believe would be an error, Plaintiffs acknowledge that in California superior courts "[a]n enhancement of the lodestar amount to reflect the contingency risk is "[o]ne of the most common fee enhancers … ." *Graham v. DaimlerChrysler Corp.*, 34 Cal. 4th 553, 579 (2004).

"The purpose of a fee enhancement, or so-called multiplier, for contingent risk is to bring the financial incentives for attorneys enforcing important constitutional rights … into line with incentives they have to undertake claims for which they are paid on a fee-for-services basis." *Ketchum v. Moses*, 24 Cal. 4th 1122, 1132 (2001). Thus, as explained in *Ketchum*, the lodestar enhancement "is intended to approximate market-level compensation for such services, which typically includes a premium for the risk of nonpayment or delay in payment of attorney fees." *Id.* at p. 1138. However, here, the attorneys are not sole practitioners, in fact they had five attorneys with hourly rates ranging from [redacted], working on a simple Anti-SLAPP motion -- compared to Plaintiffs' lone lawyer with an hourly rate of $350. Since the claimed hourly rates by Defendant's counsels are already approximately

8.5 times above the market-level compensation, a contingency multiplier-based lodestar enhancement for a huge multi-million dollar law firm cannot be warranted.

Furthermore, Defendant states that he was given an alternative fee agreement with a low fixed cost for litigation of the anti-SLAPP motion. Fee motion, at 16:14. The agreement specifically stated that in case the Court rendered a favorable decision, he would be awarded attorneys fee and a success fee of 1.5 times the standard rates. *Id.* 16:19- 21. If Defendant did not prevail he would have only been responsible for the substantially discounted fee for the representation. *Id.* Defendant then claims that his counsel bore the risk if the Court's ruling would have been unfavorable to him.

This circular argument is flawed. First, *Gates*, *supra*, does not allow a multiplier in matters involving fee shifting statutes in Federal Court. Given that the Ninth Circuit ruled that Anti-SLAPP motions apply in federal court, the federal standard for denying a multiplier under *Gates*, should apply.

Moreover, the contingent nature of the work was mitigated by the fact that there is a statutory right to recover attorneys' fees for this work. Thus, there was no risk. Even the agreement clarified that if the Court did not rule in Defendant's favor, he would have not paid anything over the fixed substantially low fixed cost. The purpose of the anti-SLAPP statute is to protect the client, not the attorney. Norman Decl. ¶10.

Defendant cannot have it both ways, he cannot argue on the one hand that it was outrageous for Plaintiffs to refuse to dismiss this case early on, or should have let the Court ruled on the initially filed motion, and this failure caused increased fees, while on the other hand, claim that this was a complex case requiring extensive attorney time utilizing five attorneys at exorbitant billing rates.

In any case, the Ninth Circuit law should apply, and the success fee multiplier should be denied.

I. NO COURT HAS EVER GRANTED AN AWARD THAT IS REMOTELY SIMILAR TO THE AMOUNT REQUESTED BY DEFENDANTS

A review of reported decisions in California suggests that Defendants' request for attorney's fees and costs is facially unreasonable. These decisions indicate that movants are rarely granted more than $60,000 pursuant to the anti-SLAPP statute. Furthermore, these cases were just as complex, if not more so, than the current litigation. Below is a list of awards of attorney's fees and costs that have been deemed reasonable by the California Court of Appeal or the California Supreme Court since 2000:

• $77,835.25: *Bernardo v. Planned Parenthood Federation of America*, 115 Cal. App. 4th 322 (2004) (affirming award of reasonable attorney's fees to a national charitable organization annually serving over four million people in suit regarding controversial scientific and medical issues that were of public importance and required expert input, scientific data, and worldwide studies)

• $55,900: *Tuchscher Development Enterprises, Inc. v. San Diego Unified Port Dist.*, 106 Cal. App. 4th 1219 (2003) (lawsuit against a port district for breach of contract and numerous business tort claims based on alleged conspiracy to disrupt agreement to develop commercial property).

• $7,296.15: *Kashian v. Harriman*, 98 Cal. App. 4th 892 (2002) (lawsuit against an environmental organization and its attorney alleging causes of action for unfair competition and for defamation following newspaper's report on defendant lawyer's request that Attorney General conduct an investigation into the plaintiff's business dealings).

• $45,000: *Schroeder v. Irvine City Council*, 97 Cal. App. 4th 174 (2002) (lawsuit against the City of Irvine, its city council, and individual council members seeking injunctive and declaratory relief on the grounds that defendants' "Vote 2000" program was an illegal expenditure of public funds).

• $65,386.61: *Rosenaur v. Scherer*, 88 Cal. App. 4th 260 (2001) (affirming trial court's award of attorneys' fees for $65,386 in action for defamation and slander stemming from comments made during a bitterly fought local initiative campaign concerning the commercial development of real property).

• $9,300: *Dowling v. Zimmerman*, 85 Cal. App. 4th 1400 (2001) (reduction of attorney's fees from an original request of $61,862.50 in case stemming from numerous unlawful detainer actions, petitions for restraining orders, and a suit alleging almost a dozen causes of action).

•The only reported decision in which the court reported on the reasonableness of a fee award obtained by Defendants' counsel is *Dove Audio, Inc. v. Rosenfeld, Meyer & Susman,* 47 Cal. App. 4th 777 (1996). In Dove Audio, the son of famed actress Audrey Hepburn hired the law firm of Rosenfeld, Meyer & Susman ("Rosenfeld") to contact other parties that had been bilked out of royalty payments in anticipation of filing a complaint with the Attorney General. *Id.* at 780. The plaintiff, Dove Audio, then sued the law firm for libel and interference with economic relationship. *Id.* Rosenfeld, represented by Defendants' counsel, then successfully demurred and was granted their motion to strike pursuant to § 425.16. *Id.* at 780-81. On appeal, Dove Audio challenged the award of attorney's fees in the amount of $28,296. *Id.* at 785. The court of appeal upheld the award on the grounds that although the award was "generous," the court's determination did not "exceed[] the bounds of reason." *Id.* (emphasis added). Here, Defendants' counsel's Expense Report absolutely exceeds the bounds of reason and exceeds its own request for fees and costs in Dove Audio. The decision in Dove Audio was more complex than the present litigation. Dove Audio involved multiple celebrities; understanding of the sophisticated way in which music royalties are calculated; due diligence in identifying, and communicating with, potential celebrity plaintiffs; correspondence with a governmental agency to initiate an investigation; and complex legal issues. *Id.* at 779-784. In the present case, however, Defendant predominantly argued the *Coastal Abstract* case (including his opposition to summary judgment), stating that this was a disputed legal issue. If $28,296 were considered generous in *Dove Audio*, certainly a similar amount would be considered generous in this case as well.

However, Plaintiffs are aware that in this case more than one motion was filed. Therefore, Plaintiffs sought the independent and unbiased evaluation of Mr. Norman who after reviewing all the

pleadings in this matter opined that the total number of hours claimed by Defendant should be between 231 and 305 hours without the summary judgment motions and between 271 and 360 hours if the Court considers the summary judgment motions intertwined with the anti-SLAPP motions. Norman Decl. ¶ 7(c). Although Plaintiffs believe Mr. Norman has been generous to Defendants (since Mr. Norman has not examined the under seal, detailed Timekeeper Records and the numerous irrelevant, duplicative, and inefficient practices employed by junior associates without proper guidance), however, Plaintiffs submit to his independent and unbiased assessment and request this Court to accept Mr. Norman's evaluation in its entirety.

J. CALCULATION OF REASONABLE FEE

When using the lodestar method, "court[s] [are] not required to set forth an hour-by-hour analysis of the fee request." *Gates v. Deukmejian*, 987 F.2d at 1399. Courts can "make across-the-board percentage cuts either in the number of hours claims or in the final lodestar figure as a practical means of [excluding unreasonable hours] from a fee application." *Id.* When performing such reductions, the court should explain its reasoning. *Gonzalez v. City of Maywood*, 729 F.3d 1196, 1203 (9th Cir. 2013).

III. CONCLUSION

Plaintiffs respectfully submit the Court, considering Mr. Norman's unbiased assessment, substantially reduce the number of hours and fees claimed by Defendant's counsels. In order to assist the Court with the pertinent calculations, Plaintiffs have provided a fee calculation worksheet submitted herewith as Chhabra Decl. 17, Ex. 7. Based on the evaluation, any award, including the fees for the summary judgment motions, the Court is requested to grant Defendant a reasonable fee award between **$65,248** and **$100,448,** as deemed appropriate.

(a) Statement of Decision with Specific Findings

Given the fact that Plaintiffs are a small business operation with limited resources, any monetary reward against Plaintiffs is bound to hurt their business operations. However, based on this Court's Dec. 21 Order, Plaintiffs understand they are responsible for Defendant's statutorily granted attorneys' fees and hope the Court finds the detailed analysis with calculations, submitted herein, reasonable. If, however, this Court disagrees with Plaintiffs' attempt to evaluate a fair and reasonable fee, Plaintiffs request this Court to provide a statement of decision with specific findings.

(b) Stay on Fees, Pending Appeal

Since this matter is currently on appeal, Plaintiffs request any monetary judgment be stayed until Appellate determination.

Date: March 8, 2018

Respectfully Submitted,

CHHABRA LAW FIRM, PC

s/*Rohit Chhabra*
Rohit Chhabra
Attorney for Plaintiffs
Open Source Security Inc. & Bradley Spengler

CHHABRA LAW FIRM, PC
ROHIT CHHABRA (SBN 278798)
Email: rohit@thelawfirm.io
257 Castro Street Suite 104
Mountain View, CA 94041
Telephone: (650) 564-7929

Attorney for Plaintiff
Open Source Security Inc.

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN FRANCISCO DIVISION

OPEN SOURCE SECURITY INC. Plaintiff, v. BRUCE PERENS, and Does 1-50, Defendants.	Case No.: Complaint For: 1. DEFAMATION PER SE 2. DEFAMATION PER QUOD 3. FALSE LIGHT 4. TORTIOUS INTERFERENCE WITH PROSPECTIVE ADVANTAGE **DEMAND FOR JURY TRIAL**

COMPLAINT

Plaintiff Open Source Security, Inc. ("OSS" or "Plaintiff") alleges against Defendant Bruce Perens ("Defendant") and Does 1-50 (Collectively, including Defendant Perens, "Defendants"), the following:

///

///

///

INTRODUCTION

1. Defendant is a computer programmer, known for his creation of the Open Source Definition and co-founder of the Open Source Initiative. This action arises from Defendants' abusive and false claims made on a blog post[1] ("Posting"), on Defendant's website, http://www.perens.com (the "Website"), regarding Plaintiff's business, which has resulted in substantial harm to Plaintiff's reputation, goodwill, and future business prospects. A true and correct copy of the Posting is attached hereto as **Exhibit A**.

PLAINTIFF

2. Plaintiff is a company based in Pennsylvania, and a resident of Pennsylvania.

DEFENDANTS

3. Defendant is an individual who wrote the defamatory Posting at issue, and based on information and belief, owns and operates the Website, and further based on information and belief, is a citizen and resident of Berkeley, California.

4. Defendant Doe 1 is a company or individual that provides the server(s) to host the Website, doing business in California.

5. Defendant Doe 2 is a company or individual that helped write the defamatory Posting at issue, doing business in California.

6. Plaintiff is not aware of the true names, identities, and/or capacities of defendants sued herein under the fictitious names of "Does." Based on information and belief, Plaintiff alleges that each Doe defendant is responsible in some manner forming the basis of this complaint. It is further alleged that Plaintiff's injuries were directly or proximately caused by such defendants. Plaintiff will amend this complaint to allege their true names when ascertained.

7. It is alleged each defendant aided and abetted the actions of the defendants set forth below, in that each defendant had knowledge of those actions, provided assistance and benefited from those actions, in whole or in part. Each of the defendants was the agent of each of the remaining

[1] Bruce Perens, *Warning: Grsecurity: Potential contributory infringement and breach of contract risk for customers,* BRUCE PERENS (Jun 28, 2017, updated Jul 10, 2017), http://perens.com/blog/2017/06/28/warning-grsecurity-potential-contributory-infringement-risk-for-customers/ (last visited Jul 14, 2017).

defendants, and in doing the actions hereinafter alleged, was acting within the course and scope of such agency and with the permission and consent of other defendants.

JURISDICTION

8. This Court has subject matter jurisdiction over this action under 28 U.S.C. § 1332. Diversity of citizenship exists since the parties are citizens of different states. Further, the amount in controversy exceeds $75,000 with respect to Plaintiff's claims against each Defendant.

VENUE

9. Venue is proper in the Northern District of California under 28 U.S.C. §1391(b)(2), as a substantial part of the events giving rise to the claims at issue in this lawsuit occurred in this District.

INTRADISTRICT ASSIGNMENT

10. Assignment to the San Francisco Division of this Court is appropriate under Civil L.R. 3-2(d), in that, based on information and belief, Defendant resides in the County of Alameda. In addition, this action involves dissemination of the defamatory Posting using the Google search engine and Google, Inc. has a substantial presence in San Francisco. Further, Cloudflare, Inc., through its services, shields the true location of the server hosting the Website, including the identity of defendant Doe 1, provides managed Domain Name Service (DNS) to the Website, and is headquartered in San Francisco.

FACTS COMMON TO ALL COUNTS

11. Plaintiff provides kernel hardening security software code ("Patches") under the trade name of Grsecurity® for the Linux® Operating System to clients throughout the United States and all over the world via their website[2].

12. The Patches are released under the GNU General Public License, version 2 ("GPLv2").[3]

///

///

///

[2] Open Source Security, Inc., *Grsecurity*, http://www.grsecurity.net (last visited Jul 16, 2016).

[3] See Open Source Security, Inc., *Download*, GRSECURITY, https://grsecurity.net/download.php (last visited Jul 16, 2016).

13. Section 6 of the GPLv2[4] provides, in part:

> Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein.

14. As defined by the GPLv2 the Patches that have already been distributed, or provided to a client, by Plaintiff are the Program over which the license applies. [5]

15. Patches are distributed contingent upon a subscription agreement[6] ("Subscription Agreement"). A true and correct copy of the Subscription Agreement is attached hereto as **Exhibit B**.

16. Under the Subscription Agreement, clients are informed that they have all rights and obligations granted by the GPLv2 for the Patches in their possession.[7]

17. The Subscription Agreement provides OSS the right to terminate a client's subscription, thereby *only* limiting a client's access to *future* updates or versions (that is, Patches that have not yet been developed, created, or released by Plaintiff), if the Patches are redistributed outside of the explicit obligations under the GPLv2 to the client's customers.[8]

18. There is no explicit or implicit term, section, or clause in the GPLv2 that is applicable over *future* versions or updates of the Patches that have not yet been developed, created, or released by Plaintiff.

19. The Subscription Agreement does not apply further restrictions on a client's *ability* to redistribute the Patches in their possession, or restrict their *ability* to exercise their rights for Patches in their possession, in accordance with the terms and conditions of the GPLv2.

///

///

[4] Free Software Foundation, *The GNU General Public License, version 2,* THE GNU OPERATING SYSTEM AND THE FREE SOFTWARE MOVEMENT (June 1991), https://www.gnu.org/licenses/old-licenses/gpl-2.0.html (last visited July 16, 2017)

[5] See *Id.*, Section 0.

[6] Open Source Security, Inc., *Stable Patch Access Agreement*, GRSECURITY (Oct. 2, 2016).

[7] *Id*. at section "*Redistribution"*

[8] *Id.*

20. Plaintiff has been targeted by outside businesses and individuals, including Defendants, who have wrongfully and maliciously accused Plaintiff, by virtue of the Subscription Agreement, of violating the terms of the GPLv2.

21. Defendants published statements in the Posting on June 28, 2017.

22. Defendants, in the Posting, stated that customers "should avoid the Grsecurity product sold at grsecurity.net because it presents a contributory infringement and breach of contract risk."[9]

23. Defendants further stated that Plaintiff was in violation of the GPLv2, and thus "[a]s a customer, … [Plaintiff's clients] would be subject to both contributory infringement and breach of contract by employing this product in conjunction with the Linux kernel under the no-redistribution policy currently employed by Grsecurity."[10]

24. The statements in the Posting are false because Plaintiff has not violated the GPLv2.

25. The statements in the Posting are false because the Grsecurity product does not present a contributory infringement or breach of contract risk to Plaintiff's clients.

26. Defendants are not aware of any legal authority holding that Plaintiff has violated the terms of the GPLv2.

27. Defendants are not aware of the existence of any legal authority that can even remotely suggest that the Subscription Agreement may have violated the terms of the GPLv2.

28. Defendants are not aware of any legal authority holding that the Grsecurity product presented a contributory infringement and breach of contract risk to Plaintiff's customers.

29. Defendants are not aware of the existence of any legal authority that can even remotely suggest that the Grsecurity product presents a contributory infringement and breach of contract risk to Plaintiff's customers.

30. The Posting is available on the front (home) page of the Website.

[9] Ex. A: *Warning: Grsecurity: Potential contributory infringement and breach of contract risk for customers, Supra*, at ¶ 1.

[10] Ex. A: *Id.* at ¶¶ 4–5.

31. With an estimated Internet traffic of 16,560 unique visitors each month[11] to the Website, the Posting is widely disseminated and read by thousands of people.

32. Defendant is recognized and well known in the Open Source community. [12]

33. Defendant is aware that "publicity [is] a tool" available to him.[13] A true and correct copy of the cited webpage is attached hereto as **Exhibit C**.

34. The Posting was also partly reproduced, linked, and discussed on www.slashdot.org ("Slashdot").[14]

35. Slashdot is a website well known by programmers and software developers in the Open Source community and has an Internet traffic of approximately 3.2 million unique visitors each month.[15]

36. The Posting was seen and read by hundreds, if not thousands, of consumers and prospective clients of Plaintiff, as well as by professional colleagues and business partners.

37. "If a speaker says, 'In my opinion John Jones is a liar,' he implies a knowledge of facts which lead to the conclusion that Jones told an untruth. Even if the speaker states the facts upon which he bases his opinion, if those facts are *either incorrect or incomplete,* or if his assessment of them is *erroneous*, the statement may still imply a false assertion of fact." *Milkovich v. Lorain Journal Co.* 497 U.S. 1, 18 (1990) [emphasis added]).

38. The Posting is not constitutionally protected speech because it includes a false assertion of fact. See *Gertz v. Robert Welch, Inc.*, 418 U.S. 323 (1974).

---

[11] *perens.com Traffic Worth,* SITEWORTHTRAFFIC.COM, http://www.siteworthtraffic.com/report/perens.com (Jul 16, 2017) (last visited Jul 16, 2017).

[12] *Bruce Perens*, WIKIPEDIA, https://en.wikipedia.org/wiki/Bruce_Perens (last visited Jul 16, 2017)

[13] Bruce Perens, Commenting to *Re: Why does no one care that Brad Spengler of GRSecurity is blatantly violating the intention of the rights holders to the Linux Kernel?* DEBIAN.ORG, (Jun 14, 2017), https://lists.debian.org/debian-user/2017/07/msg00814.html (last visited Jul 16, 2017)

[14] *Bruce Perens Warns Grsecurity Breaches the Linux Kernel's GPL License*, SLASHDOT (Jul 9, 2017, 2:10 pm), https://linux.slashdot.org/story/17/07/09/188246/bruce-perens-warns-grsecurity-breaches-the-linux-kernels-gpl-license (last visited Jul 16, 2017).

[15] *Slashdot.org Traffic Worth*, SITEWORTHTRAFFIC.COM, http://www.siteworthtraffic.com/report/slashdot.org (Jul 16, 2017) (last visited Jul 16, 2017). Also see, *Slashdot*, WIKIPEDIA, https://en.wikipedia.org/wiki/Slashdot (last visited Jul 16, 2017).

39. Defendants performed the alleged acts intentionally, and acted with malice, oppression, and fraud with the sole purpose to generate negative publicity against Plaintiff's business as it was "more effective than writing to" Plaintiff about their disagreement with the Subscription Agreement.[16] A true and correct copy of the cited webpage is attached hereto as **Exhibit D**.

40. Defendants acted with malice, oppression, and fraud, despite being informed by Dr. Richard Stallman, the President of the Free Software Foundation, that forming an opinion on the Subscription Agreement was a complicated task that required "a lot of time to think about []… ."[17]

41. The statements in the Posting have caused Plaintiff extraordinary damages, including loss of potential customers and loss of good will.

CLAIMS FOR RELIEF

FIRST CLAIM

Defamation *Per Se*– Libel

(Against all Defendants)

42. Plaintiff repeats and re-allege each and every allegation of the foregoing paragraphs as if fully set forth herein.

43. Readers of the Posting reasonably understood that the statement(s) in the Posting were about Plaintiff.

44. Readers reasonably understood the statement(s) in the Posting to mean that Plaintiff's conduct, characteristics, or a condition were incompatible with the proper exercise of their lawful business, trade, profession or office.

45. The statements in the Posting are false.

46. The Defendants together and each of them acting in concert, jointly and severally, and individually, have defamed Plaintiff by knowingly, intentionally, willfully, or negligently publishing statements about OSS which they knew or should have known to be false.

---

[16] Bruce Perens, Commenting to *Re: Why does no one care that Brad Spengler of GRSecurity is blatantly violating the intention of the rights holders to the Linux Kernel?* DEBIAN.ORG, (Jun 19, 2017), https://lists.debian.org/debian-user/2017/06/msg00759.html (last visited Jul 16, 2017)

[17] *Id.* Also see Richard Stallman Commenting to *Re: Why does no one care that Brad Spengler of GRSecurity is blatantly violating the intention of the rights holders to the Linux Kernel?* DEBIAN.ORG, (Jun 19, 2017) https://lists.debian.org/debian-user/2017/06/msg00758.html (last visited Jul 16, 2017)

47. Defendants failed to use reasonable care to determine the truth or falsity of the statements in the Posting.

48. Defendant further stated:

> I am bothered by the sort of action that Open Source Security Inc. is doing, and felt that informing the customers (albeit indirectly, in places like Slashdot) was the best way to effect a change. This was a case where publicity was the most effective means of effecting change … .[18]

49. Defendants intended to injure Plaintiff in its trade or profession by developing a wrongful fear that Plaintiff's clients may be subject to legal liability if they continued to use the Grsecurity® product.

50. As a proximate result of the Posting, Plaintiff has suffered loss of business and professional reputation.

51. Plaintiff has suffered general and special damages, including, without limitation, lost revenue and profits as a function of damage to Plaintiff's business reputation; diminution in the pecuniary value of Plaintiff's goodwill, administrative costs in connection with Plaintiff's efforts to monitor and counteract the negative publicity, and other pecuniary harm.

52. Defendants' false statements in the Posting, or relating to the Posting, have caused Plaintiff damages in an amount to be determined at trial, but in excess of $75,000 as to each defendant.

53. The negative and false posts were created and published by Defendants with malice and/or oppression as the content of the Posting contains false, defamatory statements that were known by Defendants to be false and the Posting was deliberately published with the intention of destroying Plaintiff's reputation and the reputation of Plaintiff's services, and to cause Plaintiff to lose its ability to continue its business. Plaintiff is entitled to punitive damages.

SECOND CLAIM

Defamation *Per Quod*– Libel

(Against all Defendants)

54. Plaintiff repeats and re-allege each and every allegation of the foregoing paragraphs as if fully set forth herein.

[18] Bruce Perens, commenting on *Bruce Perens Warns Grsecurity Breaches the Linux Kernel's GPL License*, SLASHDOT, (Jul 9, 2017, 4:27 pm), https://slashdot.org/comments.pl?sid=10840323&cid=54774713 (last visited Jul 16, 2017).

55. The Posting tended to discourage others from associating or dealing with Plaintiff, since doing so presented "a contributory infringement and breach of contract risk."

56. The statements in the Posting were a substantial factor in causing Plaintiff harm and damages as alleged in paragraphs 50–53.

THIRD CLAIM
False Light
(Against all Defendants)

57. Plaintiff repeats and re-allege each and every allegation of the foregoing paragraphs as if fully set forth herein.

58. Defendants published the Posting on the Website.

59. Defendants further discussed the contents of the Posting with readers of Slashdot, attempting to convince them that the statements in the Posting were an accurate analysis of the law. A true and correct copy of various comments by Defendant on Slashdot are attached hereto as **Exhibit E**.

60. Defendant publicized the Posting and continued to show Plaintiff in a false light by making the Posting available on the Website, abusing a position of power based on his recognition in the Open Source community, and further by engaging in a discussion about the content of the Posting with readers of Slashdot.

61. The false light created by the Posting is highly offensive to a reasonable person in Plaintiff's position since the Posting attempts to destroy Plaintiff's reputation and the reputation of Plaintiff's services, and attempts to cause Plaintiff to lose its ability to continue its business.

62. Defendants knew the Posting would create a false impression about Plaintiff and/or acted with reckless disregard for the truth.

63. Defendants were negligent in determining the truth of the information in the Posting or whether a false impression would be created by its publication.

64. Plaintiff was harmed and damages occurred, as alleged in paragraphs 50–53.

65. Defendants' conduct was a substantial factor in causing the harm to Plaintiff.

66. In publicizing the Posting on the Website and further discussing the matter on Slashdot, Defendant publicized the Posting to the public at large or to so many people that the Posting was substantially certain to become public knowledge.

FOURTH CLAIM

Intentional Interference with Prospective Relations

(Against all Defendants)

67. Plaintiff repeats and re-allege each and every allegation of the foregoing paragraphs as if fully set forth herein.

68. Plaintiff and many other potential clients were in an economic relationship that probably would have resulted in an economic benefit to Plaintiff.

69. Defendant knew of the economic relationship.

70. By publishing the Posting, and urging that Plaintiff's current and potential clients "should avoid the Grsecurity product sold at grsecurity.net because it presents a contributory infringement risk," Defendants intended to disrupt the economic relationship.

71. Defendants engaged in wrongful conduct through misrepresentation, fraud, deceit, malice, or oppression.

72. The relationship has been disrupted.

73. Plaintiff has been harmed as alleged in paragraphs 50–53.

74. Defendants' wrongful conduct was a substantial factor in causing Plaintiff the harm.

75. Defendants intentionally interfered with an economic relationship between Plaintiff and numerous potential clients that probably would have resulted in an economic benefit to Plaintiff.

PRAYER FOR RELIEF

With regard to all counts, Plaintiff prays that judgment be entered against Defendant Bruce Perens and Does 1-50, each and every one of them, acting in concert, jointly and severally, for compensatory actual damages in excess of $2 million resulting from their financial, reputational and professional injury to Plaintiff, as well as equitable relief as may be appropriate, and such other relief the Court may deem just and proper. Plaintiff further prays for an award of punitive damages in an amount in excess of $1 million, to punish Defendants for their outrageous, deceitful, unprecedented, vicious and malicious conduct toward Plaintiff designed so to discourage the public from conducting business with Plaintiff.

Plaintiff further seeks an Injunctive relief, including a preliminary and permanent injunction enjoining restraining Defendants from engaging in the conduct described above.

JURY DEMAND

Plaintiff requests this case be tried to a jury on all issues triable by a jury.

Dated this 17th July 2017.

Respectfully Submitted,

CHHABRA LAW FIRM, PC

/s/Rohit Chhabra
Rohit Chhabra
Attorney for Plaintiff
Open Source Security Inc.