Open Sourcing Federal Software

Amanda Li

January 19, 2018

Many federal U.S. agencies have already adapted to leverage open source software (OSS) as core components in their system capabilities. However, *custom* federal code has not made its way into the open source (OS) community until recently, and not without controversy. Should government agencies release their software to the public? Should the U.S. Department of Defense (DoD), the largest federal agency, open source their code?[1] The answers come through dissecting current U.S. legislation, analyzing the concerns and benefits of releasing DoD software, and ultimately concluding with how to best juxtapose OSS and government.

Current Status of Open Source in the U.S. Government

Ten years ago, releasing federal projects as OSS was out of the question. What was being considered, however, was the equitable treatment of open source solutions in developing federal software projects. In 2011, the U.S. Office of Management and Budgeting issued the Technology Neutrality Memorandum. The policy requires the consideration and treatment of proprietary, open source, and mixed software solutions, "free of preconceived preferences based on how the technology is developed, licensed or distributed."[2]. The unbiased, merit-based selection of technological investments became federal precedent. Technology Neutrality remains particularly relevant because agencies always seek to utilize existing software *before* creating custom software. Slowly but surely, OSS was making its way into U.S. legislation.

A mere five years later, the U.S. government started to consider open-sourcing federal code. Thus came the Federal Source Code Policy of 2016. It sought to make custom-developed source code broadly available across government agencies, improving the efficiency, consistency, and cost of software production. Perhaps more importantly, it also established a pilot program requiring all agencies to release at least 20% of new custom software as OSS for three years.[3] The experiment is, at press time, still ongoing.

The program's goal is to increase efficiency, transparency, and participation between the U.S. government and its citizens. In doing so, agencies ought to prioritize the release of code that may be potentially useful to a broader community. They must also use a consistent measure for the specified 20% minimum, be that number of lines of code, number of modules, or cost. Federal agencies ought to leverage the existing communities, engage in open development practices, regulate their release schedules, communicate with users and contributors, consider non-federal contributions for integration, and adequately document their source code. These guidelines likely come as no surprise to the OSS developer.

There are notable exceptions, though, to what types of software can be published. Firstly, agencies must not publish any source code restricted by existing regulation (i.e. patents, classified information). Secondly, no source code can be released that may create an identifiable risk to a) national security, confidentiality of government information, or individual privacy, or b) the stability, security, or integrity of agency systems and personnel, or c) the agency's missions, programs, or operations. Lastly, the CIO may veto the release of source code on the grounds of national interest. Vague language in the latter two exceptions, however, is arguably where the contention surrounding OSS and government stems from. What exactly does an "identifiable risk" and "national interest" consist of?

To begin, they tend to exclude legislative and regulatory software. Even prior to the 2016 pilot program, a handful of federal agencies began to publish their custom-developed source code

[1] The U.S. Department of Defense View
[2] Technology Neutrality View
[3] Federal Source Code Policy View

as OSS. "We the People" is a great example; it is a platform where anyone can petition online and actively engage with the U.S. Administration. All its source code is on a GitHub repository, starred by over a thousand users.[4] Another example is the Consumer Financial Protection Bureau (CFPB). Since 2012, the CFPB has defaulted to releasing custom code as OSS, bearing exceptions, and currently hosts over 200 repositories on GitHub with a steady number of public contributors.[5] Both of these examples are viewed as inherently harmless and are accepted by the public without backlash.

In 2008, a team from the National Security Agency (NSA) took interest in the research behind BigTable– Google's impressive (and proprietary) data storage system. The NSA team rebuilt a similar system from the ground up. Importantly, they added key security features that separated classifications of data, such that a user can only reach information they were authorized to access. The NSA released the project as an OSS called Accumulo, believing it could bolster the security measures of other federal projects. The Senate Armed Services Committee saw it differently. In a 600 page Senate bill in 2012, the committee banned the usage of Accumulo on the grounds of "replication", despite the significant new features made by NSA developers. After controversy broke on the ban's legitimacy, the NSA was allowed to use Accumulo, but not the rest of the DoD. Regardless, the press and public raised questions on the Senate's hidden intent in stopping open source development– specifically, on whether or not this action was issued out of fear.[6]

The DoD clearly attracts attention on the status of open source development in government, often due to the data, security, or military-driven nature of the projects themselves. In fact, the DoD can be thought of as an "edge case", particularly in terms of alleged risk and public opinion. Therefore, a discussion on OSS that addresses the DoD's concerns can be extended to address the concerns of less controversial departments.

The Department of Defense

The Department of Defense heads multiple other groups– the Army, Navy, Air Force, and NSA, to name a few. Software has become the grounds upon which modern planning, weaponry, and logistics systems stand. It is an infinitely renewable resource, given enough money and labor. Due to the importance and pervasiveness of national defense software, the move to open-source DoD projects sparks debate surrounding its cost, security, and utility.

Just like other governmental branches, the DoD outsources many of its software projects. While not inherently bad, federal reliance on discrete, disconnected software producers does come with drawbacks. To keep code hidden is to let duplication run amok, and that is a costly price to pay. In fact, an estimate of 75% of all code is written for a specific task and is never used for any other purpose.[7] Open sourcing DoD projects eliminates unknowingly replication of code. With rights and access to government-funded source code, various DoD agencies can draw on a shared pool of current systems, rather than single-handedly creating custom code. In addition to duplication concerns, open-sourcing federal projects breaks down the reliance on software contractors, who have built a wall of exclusivity around capabilities they've been paid to develop.[8] Open-sourcing code also prevents the extreme case: vendor lock-in. This occurs when an agency becomes so dependent on a single company or developer that the agency is unable to switch vendors without significant costs and labor. When developers around the world have access to the source code, they can supplement a lack in federal agency's programming power for a fraction of the price.

Opponents argue that the maintenance of the government's open-sourced code can be a hidden drain on federal resources.[9] Releasing custom federal code means that federal software developers now have to take care of those repositories, review pull requests, moderate the community, etc. in order to meet the requirements set forth by the 2016 policy. This process is laborious. The necessity, and therefore cost, of maintenance will rarely cease once proliferated. Even if a project is no longer useful to the federal government, either resources still would need to be continuously spent to moderate the community or the repository would cease to function properly (i.e. no review of contributions, drop in active participation).

[4] We the People GitHub View

[5] The CFPB's source code policy: open and shared View

[6] Senate Not Concerned About How Often NSA Spies On Americans, But Very Concerned That It Built Open Source Software To Do So View

[7] Open Technology Development: Lessons Learned and Best Practices for Military Software View

[8] Open Technology Development (OTD): Lessons Learned & Best Practices for Military Software View

[9] Open Technology Development: Lessons Learned and Best Practices for Military Software View

The open-sourcing of custom federal code has raised a fairly large amount of public concern surrounding its security implications as well. More so than companies or independent open source projects, the federal government, especially the DoD, maintain a responsibility to ensure the nation's safety in the eyes of the people. The existence of these fears themselves, regardless of validity, can negatively impact the U.S. political climate.[10] New, perhaps unforeseen contributor modifications to the code can negatively current missions utilizing the software. Those who work intermittently on an OS project are less likely to understand its intended use, and their contributions would lead to an inferior, unstable product. Black hat hackers can now presumably gain access to the source code without even needing to hack their way into the DoD. These are but a few of the concerns of those against OSS in government.[11]

Supporters for OSS in government, however, argue that the open sourcing federal projects will increase the security of the code. Chris Lynch of the Defense Digital Service says, "The whole idea of 'security through obscurity' is completely backwards. We need to understand where our weaknesses are in order to fix them, and there is no better way than to open it up to the global hacker community".[12] Lynch's opinion arguably stems from Linus' Law, coined by Eric Raymond, which claims that all bugs are shallow given enough eyes. With respect to government agencies and their code, this can be the case as well. There are, of course, a significant number of black hat hackers who attempt to insert malicious code into federal projects, but this has always been the case. Open sourcing the code allows for white hat hackers, who previously held no access, to analyze and improve upon the code. In addition, malicious code insertions are not as simple as sensational news sites claim them to be in OSS. Specifically, most OSS projects have a number of trusted developers who review, veto, and integrate contributor code. As per the 2016 policy's request to "follow open development practices", federal projects ought to do the same.[13]

Supporters also believe that releasing DoD projects as open source also improves the efficiency of producing and maintaining quality code. Open-sourcing the code allows a broader community of users and developers to continually improve federal projects. By essentially outsourcing the government's projects to the people themselves, development and deployment times can be significantly reduced. Furthermore, the agency is then able to receive direct feedback from users without the usual, lengthy bureaucratic process. This is not only fast and flexible, but smooth. Due to the respect for and prevalence of OS licenses, the public can utilize federal programs without untangling intellectual property rights to determine what is and is not allowed. Such an atmosphere of collaboration makes it easier to conduct software peer review, to reuse existing solutions, and to share technical knowledge.[14]

Open by Default

Perhaps the answer to this debate is an "open by default" policy. Rather than meeting the arbitrary 20% quota implemented in 2016, government agencies ought to release both custom-developed and contractor-developed code as the default practice. All rules exceptions detailed in the Federal Source Code Policy still apply. Open sourcing as much federal code as possible may be for the best; it is not a significant demand on resources, has proven to aid security in the past, and lends the public transparency on their government's projects.

Moderation has been mentioned on both sides of the issue, with the anti-OS argument being that the cost of maintenance will quickly overtake any initial financial benefits. However, those opponents ignore the resources saved in financing software upkeep on the government's side. The opportunity to utilize the work of our nation's developers at large is worth the cost of management. After all, there exists lack of technical expertise in government already. The responsibility falls on government agencies to outsource their labor to third-party contractors. This is so prevalent that one military officer stated, "Most government organizations don't have software experts, they rely on a vendor... to supply expertise ... Real understanding is all on the vendor side. Vendors want to continue that relationship since that's how they get their money".[15] If money is always spent to maintain the software regardless, then it is in the agency's best interest to delegate part those

[10] Public concerned about security flaws in government open source code View
[11] The Department of Defense (DoD) and Open Source Software View
[12] Air Force Issues Challenge to "Hack the Air Force" View
[13] DoD Open Source Software (OSS) FAQ View
[14] Clarifying Guidance Regarding Open Source Software View
[15] Open Source Software in Government: Challenges and Opportunities View

resources to managing other developers– namely, those in the OS community.

The Department of Defense has previously crowd-sourced software development in order to improve security, and it proved extremely useful. Take, for example, Hack the Pentagon in April, 2016. This was a bug bounty hosted by HackerOne that encouraged contributors to find overlooked issues within Pentagon projects. Within a span of only 24 days, 138 vulnerabilities were discovered and patched by the public. In rebuttal to the anti-OS point on the so-called instability of public contributions, of course not all reports were valuable. In fact, 1051 reports were not accepted. That is why moderation is always key. The above 138 reports were validated by developers from the Defense Media Activity (DMA), and only then were they implemented.[16]

The cost of this initiative, including DMA moderation, HackerOne, and the participant's prizes, was $150,000.[17] Compared to the millions of dollars spent on federal project maintenance annually, this event was nowhere near expensive. Of course, that raises the question of whether or not public contributors should be paid for contribution to government projects, in keeping with how these bug bounties functions. In terms of usual open development practice, paying contributors is not standard unless they attain a certain level of activity within the project. It would also potentially set a national precedent, that may easily be understood as undermining OS community values of collaboration for the sake of improvement or understanding. So no, the government shouldn't automatically pay its public contributors. Rather, it may be in the government's interest to incentivize software agencies to support the government's OSS movement. This could manifest itself through a tax break, similar to that given to businesses "going green", or an agency-specific program.

The most important thing that the Hack the Pentagon proved was that security can definitely increase through crowdsourcing. In the words of Eric Fanning, secretary of the Army, "There are large numbers of technologists and innovators who want to make a contribution to our nation's security, but lack a legal avenue to do so".[18]

Stepping back, open sourcing federal government projects is just one step of the path to increased participation in the government's democratic process. Code.gov, the directory of released federal code, was another step. Data.gov, the home of the U.S.'s public data, was another. These are movements in the right direction. In the age of technology, OSS ought to serve as a bridge for open communication between the U.S. Government and its citizens.[19] By releasing the vast majority of federal source code, the public can take a deeper look into the way government programs run and impact their lives. The open source movement in government is about more than the developers inside and outside D.C. It is about accessibility, accountability, and transparency.

[16]Hack the Pentagon Fact Sheet View
[17]Hack the Pentagon Fact Sheet View
[18]Hack the Pentagon View
[19]White House opens the source code for Data.gov View

References

The U.S. Department of Defense View

Technology Neutrality View

Federal Source Code Policy View

We the People GitHub View

The CFPB's source code policy: open and shared View

Senate Not Concerned About How Often NSA Spies On Americans, But Very Concerned That It Built Open Source Software To Do So View

Open Technology Development (OTD): Lessons Learned & Best Practices for Military Software View

Open Technology Development: Lessons Learned and Best Practices for Military Software View

Clarifying Guidance Regarding Open Source Software View

DoD Open Source Software (OSS) FAQ View

The Department of Defense (DoD) and Open Source Software View

Public concerned about security flaws in government open source code View

Open Source Software in Government: Challenges and Opportunities View

Air Force Issues Challenge to "Hack the Air Force" View

DoD Open Source Software (OSS) FAQ View

Hack the Pentagon View

Hack the Pentagon Fact Sheet View

White House opens the source code for Data.gov View