Open Source Voting Systems

Presented to:
2015 State Certification Testing of Voting Systems National Conference

Paul W. Craft
Kathleen A. McGregor

May, 19, 2015
Introduction

One concern raised in the aftermath of Election 2000 was the inability of concerned citizens to access and review the source code used in voting systems. There were accusations that certification testing was only black box testing and that non-disclosure agreements signed by those who had access to source code kept the “real truth” from getting out. That raised a call for the full disclosure of the source code used in voting system software. This call continues to the present day. Voting system vendors are reluctant to disclose intellectual property that requires substantial investments in time and money to create. Long before Election 2000, state officials realized that access to source code could create the opportunity to modify these systems and the distribution of source code was banned or restricted as a security measure.

Voting system vendors were viewed with suspicion, and the prices they charged for systems sales, licensing, maintenance and support were widely viewed as excessive. About this time, open source software projects were becoming successful and viewed in the computing world as good alternatives to proprietary systems with licensing fees.

Today, proponents of open source voting systems are proposing approaches that range from simply disclosing all of the source code for voting systems to developing and implementing systems that use a collaborative open source development methodology as well as everything in between. There are currently three major election jurisdictions in the United States working to develop their own voting system. Each of them appears to be planning some form of open source disclosure or development, or free software.

The January 2014 report by the Presidential Commission on Election Administration states:

…to usher in the next generation of voting machines, the standards and certification process of new voting technology must be reformed so as to encourage innovation and facilitate the adoption of widely available, off-the-shelf technologies and “software-only” solutions.¹

Many people believe that the “software only solutions” called for in the report should be open source voting systems.

The goal of this paper is to provide policy makers with a resource to help them understand what is meant by the term “open source” and illustrate how using open source to develop voting systems might have an impact on regulating, certifying and using voting systems.

Definitions

**Source Code** is the form of computer instructions and comments written by a programmer in a programming language such as Visual Basic or C. It contains the algorithms, logical instructions, logical functions and mathematical formulas that operate the software. It should contain comments designed to assist with code review and future revisions of the source code by explaining what the various parts of the source code are doing. Most commercial software developers consider their source code to be a trade secret and an asset they have either created or acquired at significant expense.

**Object Code** is the form of computer software understandable to a computing device. Source code is compiled into object code or is interpreted by an interpreter at run-time. While the computing device understands the object code, it is difficult for humans to read or modify. Most commercial computer programs are only distributed in object code.

**Closed Source**, also known as proprietary source or secret source code, refers to source code that is considered a trade secret and is protected by prohibitions on disclosure, licensing restrictions, copyrights, and/or patents. In general only a small number of individuals in the company that owns the code will be given access to it.

**Escrowed Source Code** is closed source code which a customer or regulatory body requires to be placed into escrow to ensure their ability to maintain the system in the event that the developer of the code ceases to exist or ceases to maintain the system. When a triggering event, such as a bankruptcy of the developer occurs, the escrow agent releases the source code to the customer. Jurisdictions may require escrow as part of their system purchase requirements and states may require escrow as part of their certification requirements.

**Shared Source** is a term used by Microsoft under their Shared Source Initiative. Through the Shared Source Initiative Microsoft licenses product source code to qualified customers, enterprises, governments, and partners for debugging and reference purposes. Licensees are allowed access to code under restricted conditions subject to nondisclosure agreements. Although Microsoft is given credit for creating the term Shared Source, it is an approach that has always been used for systems that were required to be reviewed by third parties, for certifications, or systems sold to clients who required the source code be provided. It is the process that many election jurisdictions have used in certifying voting systems for the last twenty-five years.

---

^2 “Microsoft Shared Source Initiative”, Microsoft Corporation

**Disclosed Source** is code that is considered to be proprietary but is freely shared with interested parties. While it may be freely examined, it is still considered to be the intellectual property of its owner and may be protected by licensing restrictions, copyrights and/or patents. Production of derivative works without a license may be prohibited. Many of the citizens who called for open source after Election 2000 were, in fact, talking about requiring disclosed source for voting systems.

**Open Source** may have many meanings, depending upon who is using the term and the context in which it is used. It is sometimes used for those arrangements defined above as Shared Source or Disclosed Source.

The term "open source" usually refers to a license that allows the source code or design information to be used modified and/or shared under defined terms and conditions. One of the earliest definitions of an open source license was the Open Source Initiative (OSI), formed in 1998, as an educational, advocacy and stewardship organization. Their license requires that the code be disclosed and certain distribution and licensing conditions must be met. They define their open source license as follows:

Open source is not simply access to the source code. The distribution terms of open source software must comply with the following criteria:

1. **Free Redistribution**
   
The license shall not restrict any party from selling or giving away the software as a component of an aggregate software distribution containing programs from several different sources. The license shall not require a royalty or other fee for such sale.

2. **Source Code**
   
The program must include source code, and must allow distribution in source code as well as compiled form. Where some form of a product is not distributed with source code, there must be a well-publicized means of obtaining the source code for no more than a reasonable reproduction cost preferably, downloading via the Internet without charge. The source code must be the preferred form in which a programmer would modify the program. Deliberately obfuscated source code is not allowed. Intermediate forms such as the output of a preprocessor or translator are not allowed.
3. Derived Works
   The license must allow modifications and derived works, and must allow them to be distributed under the same terms as the license of the original software.

4. Integrity of The Author's Source Code
   The license may restrict source-code from being distributed in modified form only if the license allows the distribution of "patch files" with the source code for the purpose of modifying the program at build time. The license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number from the original software.

5. No Discrimination Against Persons or Groups
   The license must not discriminate against any person or group of persons.

6. No Discrimination Against Fields of Endeavor
   The license must not restrict anyone from making use of the program in a specific field of endeavor. For example, it may not restrict the program from being used in a business, or from being used for genetic research.

7. Distribution of License
   The rights attached to the program must apply to all to whom the program is redistributed without the need for execution of an additional license by those parties.

8. License Must Not Be Specific to a Product
   The rights attached to the program must not depend on the program's being part of a particular software distribution. If the program is extracted from that distribution and used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those that are granted in conjunction with the original software distribution.

9. License Must Not Restrict Other Software
   The license must not place restrictions on other software that is distributed along with the licensed software. For example, the license must not insist that all other programs distributed on the same medium must be open-source software.
10. License Must Be Technology-Neutral

No provision of the license may be predicated on any individual technology or style of interface.³

OSI issues approval of open source licenses for software and licensing agreements that comply with their definition of open source. However, there is no legal requirement that software be approved by OSI, nor does it have to meet the OSI definition of open source, in order to be called “open source.”⁴

**Free Software** is defined by the Free Software Foundation as:

... software that respects users' freedom and community. Roughly, it means that the users have the freedom to run, copy, distribute, study, change and improve the software. Thus, “free software” is a matter of liberty, not price. To understand the concept, you should think of “free” as in “free speech,” not as in “free beer”. We sometimes call it “libre software” to show we do not mean it is gratis.⁵

And,

“Free software” does not mean “noncommercial”. A free program must be available for commercial use, commercial development, and commercial distribution. Commercial development of free software is no longer unusual; such free commercial software is very important. You may have paid money to get copies of free software, or you may have obtained copies at no charge. But regardless of how you got your copies, you always have the freedom to copy and change the software, even to sell copies.⁶

---

³ “Open Source Definition”, Open Source Initiative
   [http://opensource.org/definition](http://opensource.org/definition)
   Licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).

⁴ “Frequently Asked Questions”, Open Source Initiative
   [http://opensource.org/faq#avoid-unapproved-licenses](http://opensource.org/faq#avoid-unapproved-licenses)
   Licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).

   Licensed under a [Creative Commons Attribution-NoDerivs 3.0 United States License](https://creativecommons.org/licenses/by-nd/3.0/us/).

   Licensed under a [Creative Commons Attribution-NoDerivs 3.0 United States License](https://creativecommons.org/licenses/by-nd/3.0/us/).
Although they are often used interchangeably with open source, the Free Software Foundation views its licenses as less restrictive than open source licenses. A significant difference between the two is that open source licenses do not allow the software to be sold or royalties be charged for its use and free software may be sold.

Discussion

When policy makers discuss the use of, and make decisions regarding the requirements for or allowing or prohibiting open source voting systems, it is critical that all parties are clear on exactly what is meant by “open source.” Systems marketed and licensed as closed source systems frequently contain some open source elements. The OSI definition of “open source” allows for the distribution of “closed source” programs in an “open source” system. Some of the individuals and organizations that advocate open source for voting systems are, in reality, advocating disclosed source. They would like all source code used in voting systems to be available for inspection by the public, but do not necessarily advocate for open source during the period when the voting systems are under development. Many people use the terms open source and free software interchangeably.

In some states, legislators have discussed funding for developing an “open source” voting system, which, after being fully developed, could be licensed to other jurisdictions to help recover the cost of development. Such a system would not meet the requirements for an OSI open source license but could meet the definition of free software as defined by the Free Software Foundation.

There is no requirement that individuals using these terms give meaning to the terms consistent with the definitions. When using these terms in statutes or regulations it is important that the terms used be clearly defined within the body of the document.

The debate regarding the merits of closed source verses open source is ongoing. Arguments in favor of closed source include:

Other developers, particularly newcomer competitors, will copy and use the code and, from that code, will have a decided advantage in gaining the original developer’s knowledge and processes without making an investment in research and development.

Knowledge of the code makes it easier for attackers to find and exploit vulnerabilities in the system.
Users are able to modify the code and put their modifications into production, introducing potential for errors and professional embarrassment. This can also allow them to achieve independence from vendor support and maintenance.

Open source allows for unqualified people to examine the code and publish incorrect findings and raise unfounded concerns about the integrity of the system.

Royalties from proprietary systems fund professional research and ongoing improvements to the systems.

Analysis of the code by others can give rise to a variety of legal and public relations problems, including accusations of copyright infringement, criticism of code structure and disclosure of developer comments within the code that were never meant for publication.

Arguments in favor of open source include:

Other developers, particularly newcomers can copy and use the code and with the decided advantage of the original developer’s knowledge and processes can quickly get up to speed and start helping make improvements to the code.

Knowledge of the code makes it easier for white hat attackers to find and help mitigate vulnerabilities in the system.

Users are able to make modifications to the code to address bugs or develop enhancements, test their modifications and submit their modifications to the core group of developers for further testing and inclusion in future releases of the system. This can also allow them to independently resolve issues that are unique to their environment.

Open source allows for a large number of qualified people to examine the code. As a result, bugs are discovered quickly which allows for significantly more suggestions, improvements and extensions.

Open source projects benefit from donated programmer time and may be less expensive to develop and maintain.

The use of open source code has proven to be successful in numerous major projects. One only has to look to the development of Linux and the Apache Web Server to see the value of this approach.
The first five arguments in favor of closed source and open source are based upon different views of the same facts. Both approaches can produce high quality products, so there is little point in arguing their relative merits. The best approach for an organization developing a system will depend on the resources available and the comfort level that the project owners and key personnel have with one approach over the other.

Developing and maintaining open source systems is a collaborative process. Users and other interested individuals are encouraged to involve themselves as beta testers and co-developers. Most successful open source projects have an organized central control. This is usually a small group of core developers who provide the initial system design and coding, then act as a central point for version control, documentation and communications. The core developers evaluate, then accept or reject contributions and are the authority that defines the official releases of the system.

Compliance with the U.S. Election Assistance Commission’s Voluntary Voting System Guidelines (VVSG) with open source development presents a challenge. The task of bringing together the efforts of dozens, hundreds, or thousands of programmers to create a single, well-formed usable body of code is difficult. An individual who took part in one project to develop an open source voting system noted that convincing programmers to comply with the VVSG coding standards was difficult. Even within a proprietary organization, and hiring very experienced senior programmers, getting them to understand the importance of, and requirements for, code structure and documentation mandated in the VVSG standards is a daunting task. When utilizing volunteer programmers who possess varying degrees of talent and experience, meeting the standards was the biggest impediment to the project.

The open source development model tends to rapidly produce new versions of a system, so it is reasonable to expect that there will be a higher risk associated with ensuring the correct certified version of the system is implemented. User co-developers will have to exercise great care to maintain separate development and production environments and ensure the correct version is installed prior to each election.

Licensing open source systems is free and may, or may not, be subject to the restrictions in the OSI license models. Under the OSI licenses, fees or royalties may not be charged for software distribution or use. For private sector voting system businesses dealing with OSI licensed open source systems, licensing fees will no longer be part of their business model. Their revenue must be derived from providing election jurisdictions with operational support, system maintenance and consulting. Systems licensed as free software may be developed in a collaborative manner similar to open source, yet the code may be sold or licensed for use.
In a collaboratively developed and maintained open source system, the involvement of an active group of user/co-developers should make system maintenance, including identifying bugs, developing bug fixes, improving functionality and implementing new features more efficient. However, without that kind of involvement, and contribution of resources to the system, it may also stagnate and users could be lost to other systems.

All voting systems require a high degree of technical support. Half of the nation’s counties have less than 16,500 registered voters. Elections in small jurisdictions are run with small or part time staff and rely on vendors for this support. Within the context of open source development, it is highly unlikely that a county with only 16,500 registered voters will have a computer programmer on staff to represent them as a co-developer on their system, or to analyze issues in their elections and communicate them to the rest of the users. If small counties adopt open source voting systems and escape the licensing fees associated with proprietary voting systems, they will need to transfer those resources to staffing for internal support or engaging contractors for support.

Whether a state has its own voting system certification program, relies on the US Election Assistance Commission certification program or simply allow purchasing processes to select appropriate systems, regulators and users of voting systems must exercise due diligence to determine whether a particular voting system meets their needs. They must determine that system functions as required by applicable state and federal law, is accurate and reliable. The basis for this determination must be well documented. There is no reason to expect the efforts required to evaluate a voting system and certify it for use within a jurisdiction will be substantively different for open source systems. The evaluation of documentation and artifacts of testing used to determine whether a system is certified must be retained to verify that every system in use within the jurisdiction is identical to the certified system. The process used for this verification will be substantially the same for open source systems and proprietary systems.

---

7 Mr. Kimball Brace “Basic Election Admin Facts, Need for Data” June 2014
Conclusions

The use of voting systems with some form of disclosed or open source is probably inevitable. Public concern regarding the integrity of voting systems and their overall transparency will force decision makers to require vendors to disclose at least some of the source code used in their voting systems. Companies that develop and market proprietary systems can use a disclosed source model and should have adequate protection of their intellectual property in existing copyright and patent law.

Jurisdictions that wish to develop their own systems will be well served by the free software approach. They can determine system requirements, select members for their core development group then enlist the contributions of user/co-developers. Eventually, they may be able to recover their development costs and, perhaps, the costs of certifying the system through licensing fees for its use.

The choice to use a proprietary or open development approach will depend on the resources available to the organization developing the system. Collaborative open source development requires an active, involved group of user/co-developers. Traditional proprietary commercial development takes substantial investment in development, marketing and user support.

There is nothing in the methodology of open source development that will diminish the certification requirements or the scrutiny given to systems built on such software. The effort required to evaluate and certify the systems remains the same regardless of how the software was developed, is maintained and supported.

Regulators should proceed with caution as they consider drafting legislation that either require or prohibit the use of disclosed source, open source or free software. It is important to ensure that the terms are clearly defined and consistently applied throughout the body of the text.
Sources:


“Microsoft Shared Source Initiative”, Microsoft Corporation

“The Open Source Definition” and “Frequently Asked Questions” The Open Source Initiative
http://opensource.org
Licensed under a Creative Commons Attribution 4.0 International License.

https://gnu.org/philosophy/free-sw.html
Licensed under a Creative Commons Attribution-NoDerivs 3.0 United States License.

Mr. Kimball Brace “Basic Election Admin Facts, Need for Data” June 2014.