The security paradox, disclosing source code to attain secure electronic elections

Dimitrios Zissis, Dimitrios Lekkas
Department of Product and Systems Design Engineering, Syros, Greece
dzissis@aegean.gr
lekkas@aegean.gr

Abstract: In recent years we have witnessed the amalgamation of government services and electronic systems. Citizens and state interactions have changed focus to human centered electronic approaches, introducing citizens with electronic services that have simplified bureaucratic mechanisms and reduced response time. All efforts of implementing electronic governance have led to the next step in this process, digitalization of the decision making process, electronic voting. Electronic voting is an evolutionary step in the integration of electronic governments which will inevitably be introduced by all democracies across borders. This paper will view the opportunity of introducing electronic voting and explore the characteristics of such a system to provide a secure and trustworthy platform. Setting the field and clarifying disambiguation will result in viewing the critical issues involved in such a system. Transparency and audit ability may be one of the most decisive rudiments of such a system as it is directed in increasing citizen's confidentiality ,focal point of this research is the selection involving the "disclosability" of e-voting system source code. Approaching the issue from information's systems security perspective and taking into account relations with the open source initiative, it is evaluated that for electronic voting to harness the power of scientific review and secure coding, security must not depend on obscurity.

Keywords: e-democracy, e-voting, open source electronic voting, disclosed electronic elections

1. Towards electronic elections

Contravening the common tradition of opening academic papers with references to current events, this paper returns to the 5th century BC. During this time, in classical Athens the first elections are believed to have been held; closely after these, claims of electoral fraud appeared in history books. During elections, pieces of broken pottery were used as voting tokens, named ostraka, on which candidate names were carved and these were deposited in vases which were used as election ballots. It is intriguing that in a well, bellow the acropolis, 190 ostaka (voting tokens) were discovered (Broneer, 1938). From the handwriting they appear to have been written by only 14 individuals and bear the name of Themistocles; these were evidently meant for distribution to voters. Although being written by a number of individuals does not indicate election fraud, being dumped inside a well may prove intention to hide them. Until present day, elections are always threatened by electoral fraud and methods of improving the electoral processes are continuously being explored. These first elections differ widely from the form of elections we are accustomed to; elections in classical Athens were an example of direct democracy, as opposed to today's representative democracy. In a direct democracy citizens did not elect representatives but voted on legislation and executive bills in their own right. Direct democracy stands in contrast to representative democracy, where sovereignty is exercised by a subset of the people, usually on the basis of election.
Discussions involving the form of democracy have re-emerged with the birth of a new interactive form of democracy, electronic democracy. E-democracy comprises the use of electronic communications technologies, such as the Internet in enhancing democratic processes within a democracy. Making use of electronic participation channels, citizens have wide access to rich information and are more engaged in the decision making process. It is a political development still in its infancy, as well as the subject of much debate and activity within government, civic-oriented groups and societies around the world.
E-Democracy is a form of direct democracy in which primarily the Internet and other complimentary communication technologies are used to revolutionize bureaucratic processes and information channels between government and citizens. Many advocates think that also important to this notion are technological enhancements to the deliberative process. Electronic direct democracy is sometimes referred to as EDD and is perceived as an open democracy. Openness here means public access to the information needed for the making of public decisions. Increased public access (i.e. less secrecy) also gives information to adversaries, thereby increasing their strength (Kantrowitz, The Weapon of Openness , 1989). The "weapon of openness" is the net contribution that increased openness (i.e. less secrecy)

makes to the survival of a society. Bohr believed that the gain in strength from openness in a democracy exceeded the gains of its adversaries, and thus openness was a weapon. (Kantrowitz, 1992). Athenian philosopher Plato (Greek: Πλάτων) believed that an uninformed and disengaged public was the greatest threat to democracy. Plato's critique on democracy still poses the question whether citizens of today's democracies are interested and informed enough to participate meaningfully in the democratic process. It is an undeniable necessity of a strong democracy to rest on the shoulders of informed and engaged citizens. It is a reality that many scholars today may not favor Plato's utopian form of government, but it is an undeniable fact that his ideas have deeply influenced the society we live in. James Madison was the forth president and one of the founding fathers of the United States, he is considered by many as the father of the constitution, he was the principal author, wrote:
"A popular government without popular information, or the means of acquiring it, is but a prologue to a farce or a tragedy, or perhaps both. Knowledge will forever govern ignorance, and a people who mean to be their own governors must arm themselves with the power which knowledge gives." (Madison, 1822)
Electronic democracy and electronic participation channels are envisioned as the tool necessary to provide the public with the spectrum of information necessary to evidently strengthen the system of governance.

2. Electronic Voting

It is apparent that the terms electronic democracy and electronic voting are interoperably linked. Electronic voting is a vital and indispensable aspect of electronic democracy. Combining technical complexity and the political processes leading to the development of an E-Democracy application framework. Electronic voting provides the need for citizens to express their timely opinion on civil affairs such as legislation, representatives (if any) and others. Currently a universally acceptable definition for e-voting is lacking. The term is being unambiguously used for a variety of applications ranging from vote casting over electronic networks to electronic voter registration.

In general, two main types of e-voting can be identified (Buchsbaum, 2004):

- e-voting supervised by the physical presence of representatives of governmental or independent electoral authorities, like electronic voting machines at polling stations or municipal offices, or at diplomatic or consular missions abroad;
- e-voting within the voter's sole influence, not physically supervised by representatives of governmental authorities, like voting from one's own or another person's computer via the internet (i-voting), by touch-tone telephones, by mobile phones (including SMS), or via Digital TV, or at public open-air kiosks - which themselves are more venues and frames for different machines, like, e.g., PCs or push-button voting machines, with or without smart card readers.

Electronic voting is envisioned as having the capacity of introducing a number of benefits and advantages to the electoral process; these include (Council of Europe-Commitee of Ministers, 2004):

- facilitating the participation in elections and referendums of all those who are entitled to vote, and particularly of citizens residing or staying abroad;
- widening access to the voting process for voters with disabilities or those having other difficulties in being physically present at a polling station and using the devices available there;
- increasing voter turnout by providing additional voting channels;
- bringing voting in line with new developments in society and the increasing use of new technologies as a medium for communication and civic engagement in pursuit of democracy;
- reducing, over time, the overall cost to the electoral authorities conducting an election or referendum; economies of scale
- delivering voting results efficiently, effectively and more reliably
- providing the electorate with a better service, by offering a variety of voting channels;

The issue of security in the context of the electoral process is referred to as one of the most important constraints in the implementation of electronic voting. The Caltech-MIT Voting Technology Project states: “Security is as important as reliability in guaranteeing the integrity of the voting process and public confidence in the system. People do not use things in which they have no confidence. Losing confidence in elections means losing confidence in our system of government.” (MIT, 2001)

Electronic voting security includes a wide spectrum of fields, procedures, issues and actors which are relative to the technological approach taken. It effectively relates to the procedures and standards that are put into place to overcome technological security shortcomings (Mohen, 2001) (Williams, 2004) (Xenakis, 2004). Crucial to the success of such a system is the decisions made on system characteristics and elements. Beforehand selections must guide design though the implementation of the identified technologies. Evolution in the fields of cryptography and information systems security have made technological solutions available for many recognized issues of electronic voting, making it possible to counteract eminent threats and attacks on such a system. It is central to stress the importance of the transparent implementation of selected technological solutions. Currently, voting system software is one of the most opaque aspects of electronic voting, as it is large, complex and generally unavailable for inspection (Hall, 2006). Government and states purchase voting software and equipment from commercial vendors, which retain source code and system design. Transparency and audit ability may be one of the most decisive rudiments of such a system, as it is directed at increasing citizen’s confidentiality, the focal point being the selection involving the “disclosability” of e-voting system source code. System disclosability refers to the system software, hardware, microcode, and any custom circuitry being open for random inspections (including examination of the documentation by appropriate evaluators), despite demands for secrecy from the system vendors (Neumann, 1993). Electoral transparency is defined as having four primary aspects: access, public oversight, comprehension and accountability (Hall, 2006).

In this paper we explore the disclosability issue of electronic voting and support the initiative that electronic voting source code should be open and available to all members of the public. Additionally we support the notion that a hybrid open source design approach should be followed during the design, implementation, testing and monitoring of electronic voting systems.

3. Security Principles and Cryptography

To anyone unfamiliar with information systems security, it appears paradoxical that open, fully disclosed source code, evidently increases overall system security, so the most obvious way in securing a system would be by keeping it secret.

There are at least three levels of openness, each with its own benefits, drawbacks, and peripheral issues:

1. Public disclosure of algorithms and protocols
2. Public disclosure of source code
3. Public or open contribution of source.

Every one of these levels of openness has its own implications for security. Each also has associated costs that are often overlooked in discussions that focus only on benefits. The down side of any open process, for the development and review of security systems, is that open review might reveal security flaws and render users of flawed mechanisms, subject to attack. This argument is usually derided in the security community as “security through obscurity.”

It is a widely accepted principle in software security to never assume your secrets are safe. Kerckhoffs principle, which dates back to the 19th century, states that systems should be designed so that their security does not rely upon the secrecy of their design or implementation (Kerckhoffs, 1883) (Wagner, 2007). The reason is simple; if the leak of information about how the system works can compromise its security, then the system is fragile. (Hall, 2006). Security through obscurity means that some sort of secrecy or obfuscation is an important part of the security model. Keeping secrets is hard, and is almost always a source of security risk. Information security experts and cryptographers believe that software engineers should ``demand open source code for anything related to security'' (Schneier, 1999).Security through obscurity is the failure point of most types of computer systems.

- False sense of security
- Limited verification
- Vulnerabilities known by the wrong people

The claim that security through obscurity has negative implication on system security has its roots in cryptography, where algorithms have historically been designed to resist even an adversary who knew the algorithm. It seems to escape public knowledge that the baton used in relay races originates from the Spartan tool of military communication encryption .As early as the fifth century BC, they employed a device called the “skytale”. The earliest apparatus used in military cryptology and one of the few ever devised in the whole history of the science of transposition ciphers(Kahn, 1967,1973). The skytale consists of a staff of wood around which a strip of papyrus, or leather, or parchment is wrapped close-packed. The secret message was written on the parchment down the length of the staff; the parchment is then unwound and sent on its way. The disconnected letters make no sense unless the parchment is rewrapped around a baton of the same length, as the first then words leap from loop to loop, forming the message. Cryptography did not try to hide the existence of a message, but only the context, as it was assumed that adversaries would be able to gain possession of the message but not understand it.

Cryptography, secret writing, is the strongest tool for protection against many kinds of security threats. Well disguised data cannot be read, modified, or fabricated easily. Cryptography is rooted from higher mathematics: Group and field theory, computational complexity, and even real analysis, not to mention probability and statistics (Pfleeger & Pfleeger, 2006).

An encryption algorithm is only believed to be secure when:

1. It is based on sound mathematics
2. It has been analysed by competent experts and found to be sound. A review by critical, outside experts is crucial
3. It has stood the test of time. A new algorithm gains popularity people continue to review its foundations. Although a long period for successful use and analysis in not a guarantee of a good algorithm, the flaws in many algorithms are discovered relatively soon after their release.

All encryption algorithms are open to public scrutiny and their strength relies on higher mathematics. Even with the knowledge of the algorithms design it must be computationally infeasible to break, thus making it secure. Many developers find it exciting to write their own cryptographic algorithms, sometimes banking on the fact that if they are weak, security by obscurity will help them. Weak or flawed encryption provides only the illusion of protection and security. The RC2 and RC4 encryption algorithms were supposed to be RSA Security trade secrets and were not open to public scrutiny. They were both reverse engineered and posted anonymously to the Internet (Barnum & Gegick, 2005).

4. Open source

Open source is an approach to design, development, and distribution, offering practical accessibility to a product's source (goods and knowledge). Before open source became widely adopted, developers and producers used a variety of phrases to describe the concept; the term open source gained popularity with the rise of the Internet, which provided access to diverse production models, communication paths, and interactive communities. The open source design approach, involves concurrent input from diverse sources, with different approaches and priorities, and is directly opposed to centralized models of development. The principles and practices are commonly applied to the peer production development of source code for software that is made available for public collaboration. The result of this peer-based collaboration is usually released as open-source software; however open source methods are increasingly being applied in other fields of interest.

Open source software (OSS) is defined as computer software for which the source code is freely available for review and scrutiny and is made available under a licence which conforms with the open source definition. OSS is viewed as having better quality higher reliability and more flexibility with additional benefits of being distributed for free. Proponents of open source software argue that due to the characteristic of open source, permitting peer review, massive parallel debugging leads to arguably securer code, than closed system software. This is often referred to as "Linus's Law":"Given enough eyeballs all bugs are shallow". The many eyeballs code review assertion has been criticized as for its effectiveness. Evidently success of an open source project is in accord with the characteristics of the project itself. Paraphrasing Raymond,(Raymond,2001) "The best OSS projects are those that scratch the itch of the best coders". Additionally high profile projects have the potential to motivate a greater developer community as increasing programmers status is a core incentive.Linus's Law has been proven effective in some high profile open source projects. (Robbins, 2005) As opposed to traditional software design methods, open source design begins only with a vision of what the final product should be; requirement analysis is an ad hoc procedure (Joseph Feller, 2005). A prototype is rapidly put into the community for circulation and the creative process begins. Release Early, Release Often. Open source projects are not subject to the economic concerns or contractual agreements that turn releases into major events in traditional development. (Robbins, 2005)Additional requirements or features can come from any member of the community reviewing the product. A core group of respected developers can guide this process and review the proposals by other members of the community. Implementation and testing is often going on in parallel with system specification. Often there will be competing designs and implementations, at most one of which will be selected for inclusion in the oss project. The process lacks most of the crucial elements of traditional project management such as project plans, system level designs, schedules and defined processes .The parallel debugging and testing feature in open source software design is captivating. Changes or fixes to a bug are sent to the core design team through a preselected reporting channel. After a rigorous design process and approval of the updates, they are assigned to developers with specific details to be updated in each module. After changes are performed additional code inspections, feature tests, integration, system tests and finally release to the customer follow.

Vincent Rijmen, a developer of the winning Advanced Encryption Standard (AES), encryption algorithm, believes that the open source nature of Linux provides a superior vehicle to making security vulnerabilities easier to spot and fix, ``Not only because more people can look at it, but, more importantly, because the model forces people to write more clear code, and to adhere to standards. This in turn facilitates security review'' (Rijmen, 2000). In reference to Linus Law and to openness, it is believed that programmers involved in open source projects code carefully knowing that their code will be heavily commented and reviewed. Additionally commercial deadlines can impose pressures as deadlines approach that cause programmers to work less carefully.

Proponents of open systems often counter argue that additionally to friendly eyes reviewing software many hostile can find weaknesses and use them to their own advantage. Whitfield Diffie, the co-inventor of public-key cryptography, is chief security officer and senior staff engineer at Sun Microsystems. "As for the notion that open source's usefulness to opponents outweighs the advantages to users, that argument flies in the face of one of the most important principles in security: A secret that cannot be readily changed should be regarded as a vulnerability. If you depend on a secret for your security, what do you do when the secret is discovered? If it is easy to change, like a cryptographic key, you do so. If it's hard to change, like a cryptographic system or an operating system, you're stuck. You will be vulnerable until you invest the time and money to design another system. This has long been understood in cryptography, where the principle of openness was articulated as far back as the 1870s (though it took over a century to come to fruition)." (Whitfield Diffie, 1998)
Additionally it is often stated that the second point is that a few expert eyes are better than several random ones; a dedicated organization with responsibility for the software is a better

custodian than the many eyes of the open-source community. An immense argument in favor of open source processes is their massive parallel debugging.

In particular, there seems to be an increasing tendency among the mass-market proprietary software developers to rush to market, whether the product is ready or not—in essence, letting the customers be the beta testers. Furthermore, efforts to reduce costs often seem to result in lowest-common-denominator products. Indeed, satisfying stringent requirements for security and reliability (for example) is generally not a goal that yields maximum profits. (Neumann, 2005)Then a straightforward economic analysis can in principle tell us the right time to roll out a product for beta testing. Alpha testers are more expensive, being paid a salary; as time goes on, they discover fewer bugs and so the cost per bug discovered climbs steadily. At some threshold, perhaps once bug removal starts to cost more than the damage that bugs could do in a beta release product, alpha testing stops. Beta testing is much cheaper; testers are not paid. (Anderson, Open and Closed System Are Equivalent, 2005) Typically, commercial software firms can ask users only to point at the problems: beta testers do not fix the bugs they just report them. It is also interesting to note that most commercial companies do not discourage their employees from working on open source projects. (Raymond, 2001)

Elias Levy, is the former moderator of one of the most popular security discussion groups – Bugtraq, Advocates derive their dogmatic faith in the implicit security of Open Source code from the concept of "peer review," a cornerstone of the scientific process in which published papers and theories are scrutinized by experts other than the authors. The more peers that review the work, the less likely it is that it will contains errors, and the more likely it is to become accepted. So does all this mean Open Source Software is no better than closed source software when it comes to security vulnerabilities? No. Open Source Software certainly does have the potential to be more secure than its closed source counterpart. But make no mistake; simply being open source is no guarantee of security. (Levy, 2000)

5. Transparent Elections-Threats/Attackers

Electronic voting system security is crucial in maintaining the public's faith in democracy. A breach in security could introduce risk and deficiencies to the whole electoral process and to democracy itself. As the needs for security are increasing, it is evident that electronic voting systems need to be evaluated and thoroughly tested. There is a growing debate on whether these systems should be tested only in government approved specialized laboratories or whether they should be open, so that anyone could examine and critique on them (Moynihan, 2004) Additionally private corporations are standing by their belief that limited disclosure of source code for security reasons can only be the answer.

A system grabbing the media's attention and of grave importance as an electronic voting system attracts a wide spectrum of threats of vast importance. Attacks on an electronic voting system can be categorized according to motive; such are publicity attacks, profit attacks, terrorist attacks and attacks which are motivated by creating instability in current government/democracy, figure 1.

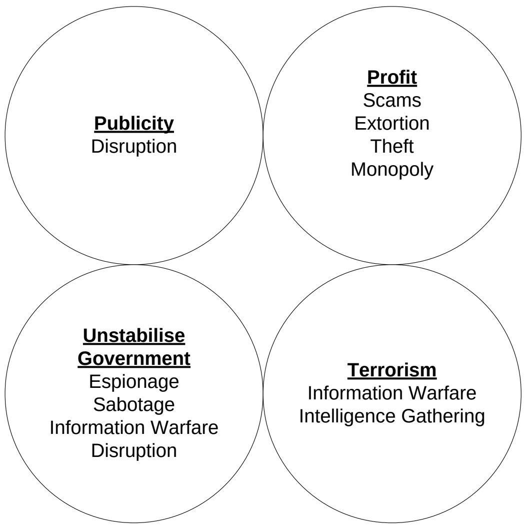

Figure 1 Categorized Threats to e-Voting

Pin pointing expected attacker's profiles confirms the importance of efficient security characteristics necessary for an information system of such scale.

E-Voting needs to be secured from the voters, election officials, programmers, technicians and system administrators (Jones, 2004). The threats posed could be internal e.g. the vendor, election officials. Or they could be external such as individuals, well funded agencies, states, parties, criminals, terrorists, many of whom cannot even be prosecuted (Jefferson, 2004) (Svensson, 2003) (A.Ballas, 2006)The motives of the attackers range from publicity (Mayniham, 2004), to foreign intelligence and terrorist acts (Philips, 2001),to governments manipulating the system for their benefit (Mercuri, 2004),figure 2.

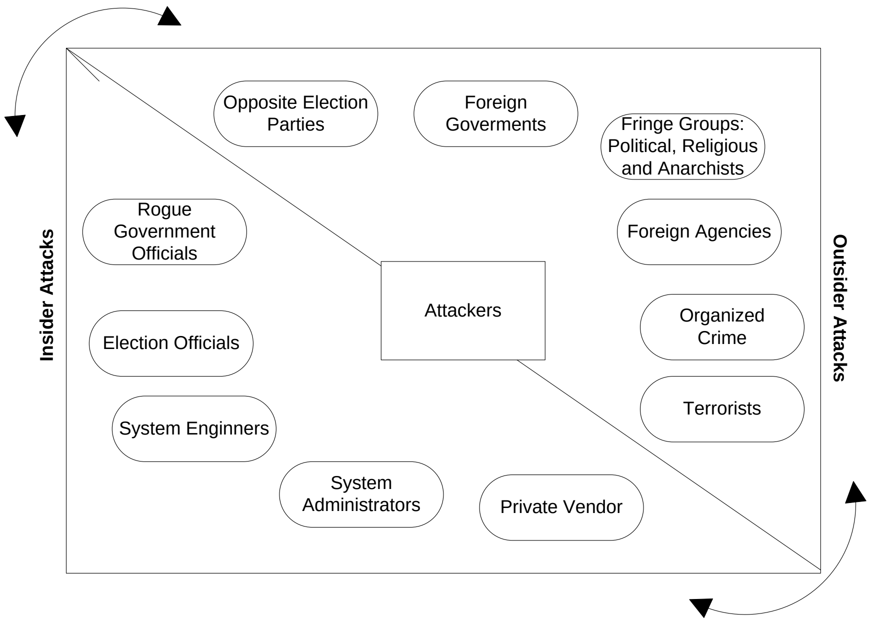

Figure 2 e-Voting attackers profiles

It is evidently obvious that relying on keeping system source code and design secret against such attacker profiles is inapplicable protection. Insider attacks make obscurity an irrelevant security measure.
It must be assumed that an attacker can obtain information about every system aspect -- assume the attacker has access to all source code and all designs. Even if this is not true, it is trivially easy for an attacker to determine obscured information. An example of such is that of the FBI spy Richard P. Hanssen who carried out the ultimate insider attack against U.S.classified networks for over 15 years. Hanssen was assigned to the FBI counterintelligence squad in 1985, around the same time he became a traitor to his country. Even the most secure networks are often amenable to insider attacks. Several studies show that the most common threat to companies is the insider attack, where a disgruntled employee abuses access. (Howard, 2002)

At any point an employee could consciously leave a backdoor to permit attacks on an electronic voting system. Backdoors, placed in software could be activated when a user tries to cast a vote, can invisibly monitor, or subvert the voting process. “The prevalence of so-called EasterEggs in many popular software packages demonstrates that this is a real possibility. (Easter Eggs are cute extras that a software developer adds to the application without authorization, for fun. One well-known example: Microsoft's Excel 97 spreadsheet application contains a full-fledged flight simulator, that can be launched using a secret sequence of keystrokes.)” (MIT, 2001).It is an undeniable fact the disclosed source code is the only protection to insider attacks of this nature.

Outsider attacks from well funded and organized threats, are equally as difficult to protect from, using obscurity, as insider attacks. Tools such as decompilers and disassemblers allow attackers to obtain sensitive information that may be stored in binary files. Also, inside attacks, which may be accidental or malicious, can lead to security exploits. Using real protection mechanisms to secure sensitive information should be the ultimate means of protecting your secrets. (Barnum & Gegick, 2005). Users don't want their personal data leaked. Keys must be kept secret to avoid eavesdropping and tampering. Many people make an implicit assumption that secrets in a binary are likely to stay secret, maybe because it seems very difficult to extract secrets from a binary. However keeping the "secrets" secret in a binary is incredibly difficult. One problem is that some attackers are surprisingly good at reverse engineering binaries.

Disclosing electronic voting software, firmware and hardware to public scrutiny is the single process of generating a trustworthy voting platform. Disclosed and open source software supports random access to the system by allowing a greater sphere of individuals the ability to scrutinize the detailed workings of a voting system. In the case of publicly available source, this access is available to all members of the public. Openness is necessary for the processes of trial and the elimination of error (Kantrowitz, The Weapon of Openness, 1992).It is important to clarify that open source software is not the same as disclosed source software. Vendors can continue to use traditional software development processes and subsequently disclose the resulting source code, without any need to adopt any of the other distinguishing features of open source software. Source code disclosure policies, licensing terms, and software development processes are three separate matters, and while open source software takes a particular stance on all three topics, it is source code disclosure that matters most to elections (Hall, 2006).

Trust in electronic voting system is a crucial factor; as a threat to the trust in electronic elections can pose a threat to the trust in our electoral processes and evidently our state of governance. Security through transparent operations is the only guarantee that can maintain trust in a system. Election security has to be viewed as a component of national security, since the very legitimacy of democratic government depends on elections that are fair, open, trustworthy, and seen to be so (MIT Serve, 2001). Electronic Voting requires a higher level of security than e-commerce, “e-commerce grade security is not good enough for public elections”. Trust in an e-commerce environment is based on the belief that in the event that the system should fail there are policies and guarantees in place to protect the customer. Ultimately the customer puts his trust in the commerce organization in belief that if something

goes wrong an honest corporation will have the moral will to do right, such is the example of trust in banking information systems
Evidently a customer puts his trust in the Bank and not the banks IS having the knowledge that in the worst case scenario he will be able to verify the mishappenings in an alternative method(visiting the local bank) and policies are at hand to protect him.
In a commercial setting, people can detect most errors and fraud by cross-checking bills, statements, and receipts; and when a problem is detected, it is possible to recover (at least partially) through refunds, insurance, tax deductions, or legal action. In contrast, voting systems must not provide receipts, because they would violate anonymity and would enable vote buying and vote coercion or intimidation. Yet, even though a voting system cannot issue receipts indicating how people voted, it is still vital for the system to be transparent enough that each voter has confidence that his or her individual vote is properly captured and counted, and more generally, that everyone else's is also.
Evidently for electronic voting to profit from all benefits of disclosing system software, it will have to incorporate features of the open source design process. A hybrid solution of open source design process involving a core guidance team leading the project and a supporting community are necessary. High reliability theory advocates that building a highly reliable system requires high levels of technical competence acquired through an environment that rewards error reporting and promotes continuous system improvement such as the reporting procedures implemented in open source design process (Moynihan, 2004).

The open source method could potentially lead to a more robust product. The term robust here is used in Neumann's sense—that is, an intentionally inclusive term embracing meaningful security, reliability, availability,and system survivability in the face of a wide and realistic range of potential adversities (Neumann 1999), (Krishnamurthy, 2005)
Using an open source design process to designing electronic voting systems will evidently lead to (Hall, 2006) (Joseph Feller, 2005) (Pfleeger & Pfleeger, 2006) (Howard, 2002):

- **Securer System**. Will evidently lead to a securer system due to mass parallel testing debugging and monitoring.
- **Less Complex & Higher Quality**. Complex design is never easy to understand, and is therefore more likely to include subtle problems that will be missed during analysis. Complex code tends to be harder to maintain as well. And most importantly, complex software tends to be far buggier.
- **Transparency**. Clear and complex free source code publicly available promotes transparency. Historically, one of the abiding principles of election administration has been that the best way to demonstrate that the election is honest is by inviting public scrutiny and being open and transparent about all aspects of the election.
- **Usability**. A not so obvious point of simplicity is usability.
- **Accountability**. Design and test teams can be held accountable for not performing duties.
- **Evaluation**. Source code disclosure would enable independent analysis of voting software.
- **Promote Interoperability**. Source code disclosure would eliminate one barrier to interoperability between equipment from different vendors, potentially enhancing competition between vendors
- **Increased Accuracy**. An efficiently and effectively operating system will operate as expected introducing increased accuracy into the electoral results.

6. Conclusion

In the authors opinion, evidently for electronic voting systems to be widely accepted, source code, firmware and hardware must be disclosed to public scrutiny. To gain the additional benefits of open source we support the notion that a hybrid open source design approach should be followed during the design, implementation, testing and monitoring of electronic voting systems. Additionally it is important to understand that an information system as vital as electronic voting needs to be thoroughly tested before widespread use. Such a system needs to be used in low risk elections (university student elections) before been adopted for high risk elections, as fixes to errors/bugs may not be patched in one cycle. Zero-day attacks occur when a vulnerability window exists between the time a threat is released and the time

security vendors release patches. A relevant time window must be given to design team to patch software taking into account that patches may be provided in the next election cycle. It is crucial that resolutions on these matters are rapidly decided before trust is lost in electronic voting system.
Selections about electronic voting software elements and characteristics will continue to remain a hot issue and essential decisions are necessary a priori to the deployment of any information system. The dissemination of source code is exceedingly beneficial, but not a panacea for all information technology security concerns. Multiple development models have to be evaluated and tested in numerous conditions to lead to a successfully deployed solution as currently there is no definitive evidence backing either claim. Advocates of closed system software often claim that the influx of changes, rapid release of software and introduction of new features and invariably flaws will continuously feed a vicious cycle of attack and countermeasure. Although the existence of successful OSS processes may be proof that high quality and widely deployed software can be achieved by implementing OSS development methods the exact means by which this has happened, and the prospects for repeating OSS successes, are frequently debated. It is necessary for us to take a holistic and system view of the software to make crucial choices on the architecture and design of electronic voting systems.

7. References

A.Ballas. (2006). E-Voting: The Security Perspective. London School of Economics.
Anderson, R. (2005). Open and Closed System Are Equivalent. B. F. Joseph Feller, *Perspectives on Free and Open Source Software* (σ. 133). London: MIT Press.
Barnum, S., & Gegick, M. (2005). Never Assuming that Your Secrets Are Safe. Cigital, Inc.
Broneer, O. (1938). Hesperia.
Buchsbaum, T. M. (2004). E-Voting: International Developments and Lessons Learnt. Electronic Voting in Europe Technology, Law, Politics and Society.
Council of Europe-Commitee of Ministers. (2004). Recommendation Rec(2004)11 of the Commitee of Ministers to the member states on legal, operational, and technical standards for e-voting. 898th meeting of the minsters Deputies.
Hall, J. L. (2006). Transparency and Access to Source Code in Electronic Voting. USENIX/ACCURATE Electronic Voting Technology (EVT'06) Workshop.
Hissam, C. B. (2005). Making Lightning Strike Twice. In B. F. Joseph Feller, *Perspectives on Free and Open Source Software* (p. 143). London: The MIT Press.
Howard, M. &. (2002). Writing Secure Code. Microsoft Press.
Jefferson, D. A. (2004). Analyzing internet voting security. Communications of the ACM 47, (pp. 57-64).
Jones, D. W. (2004). Auditing elections. Communications of the ACM 47.
Joseph Feller, B. F. (2005). Perspectives on Free and Open Source Software. MIT Press.
Kantrowitz, A. (1989). The Weapon of Openness . Foresight Background No. 4, Rev. 0 .
Kantrowitz, A. (1992). The Weapon of Openness. Στο C. a. Lewis, Nanotechnology Research and Perspectives (pp. 303-311.). Cambridge: MIT Press.
Kerckhoffs, A. (1883, Jan.-Feb). La cryptographie militaire. Journal des sciences militaires .
Krishnamurthy, S. (2005). An Analysis of Open Source Business Models. P. o. Software, *Joseph Feller, Brian Fitzgerald, Scott A. Hissam, Karim R. Lakhani* (p. 279). London: MIT Press.
Levy, E. (2000, 4 17). Wide Open Source;Is Open Source really more secure than closed? Elias Levy says there's a little security in obscurity. . SecurityFocus .
Macintosh, A. (2003). Working Group 4 to the European Commission. Ανάκτηση 03 05, 2004, από http://www.eu-forum.org/summit/docs/WG4e-democracy-FINAL%20RESULTS.doc
Madison, J. l. (1822, August 4).
Mercuri, R. T. (2004). The code of elections. Communications of the ACM 47(10), (pp. 53-57).
MIT, C. (2001). Voting: What Is, What Could Be, Report of the CalTech MIT Voting Technology Project.
Mohen, J. a. (2001). The case for Internet voting. Communications of the ACM 44, (pp. 72-85).

Moynihan, D. P. (2004). Building Secure Elections- EVoting Security, and Systems Theory. Administration Review , pp. 515 - 528.
Neumann, P. (1993). Security Criteria for Electronic Voting. National Computer Security Conference Baltimore. Maryland.
Neumann, P. G. (1999). Robust open-source software. Communications of the ACM 42
(2): 128–129.
Neumann, P. G. 2004. Principled assuredly trustworthy composable architectures.
Technical report, Computer Science Laboratory, SRI International, Menlo Park, CA.
Final report, SRI Project 11459. Available from: http://www.csl.sri.com/neumann/
chats4.html.
Pfleeger, C. P., & Pfleeger, S. L. (2006). Elementary Cryptography. Στο C. P.-P. Group, & S. L.-R. Corporation, Security in Computing, Fourth Edition. Prentice Hall.
Philips, D. M. (2001). Gauging the risks of internet elections. Communications of the ACM 44, (pp. 73-85).
R.Michael Alvarez, T. E. (2004). Point, Click and Vote, the future of internet voting. Washington,D.C.: Brookings Institution Press.
Raymond, E. S. (2001). The Cathedral and the Bazaar. Cambridge: O'Reilly.
Rijmen, V. (2000). LinuxSecurity.com Speaks With AES Winner.
Robbins, J. (2005). Adopting OSSE practices by adopting OSSE Tools. B. F. Joseph Feller, *Perspectives on Free and open Source Software* (p. 251). London: MIT Press.Schneier, B. (1999, September 15). Open Source and Security. Crypto-Gram. Counterpane Internet Security .
Sean Barnum, M. G. (2005). Never Assuming that Your Secrets Are Safe. Cigital, Inc.
Svensson, J. a. (2003). "E-voting in Europe: Divergent democratic practice. Information Policy 8(1) , pp. 3-15.
Wagner, D. (2007). Written testimony before the committee on house administration,elections subcommittee .
Whitfield Diffie. (1998). Privacy on the Line: The Politics of Wiretapping and Encryption. MIT Press.
Whitfield Diffie, S. L. (1998). Privacy on the Line,The Politics of Wiretapping and Encryption. MIT Press.
Williams, B. J. (2004). Implementing voting systems- the Georgia method. Communications of the ACM 47, (pp. 39-42).
Xenakis, A. a. (2004). Procedural Security in Electronic Voting. 37th Hawaii International Conference on System Sciences.