International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:1, No:8, 2007

Agent-Based Offline Electronic Voting

Mehmet Tahir Sandıkkaya, and Bülent Örencik

***Abstract*—**Many electronic voting systems, classified mainly as homomorphic cryptography based, mix-net based and blind signature based, appear after the eighties when zero knowledge proofs were introduced. The common ground for all these three systems is that none of them works without real time cryptologic calculations that should be held on a server. As far as known, the agent-based approach has not been used in a secure electronic voting system. In this study, an agent-based electronic voting schema, which does not contain real time calculations on the server side, is proposed. Conventional cryptologic methods are used in the proposed schema and some of the requirements of an electronic voting system are constructed within the schema. The schema seems quite secure if the used cryptologic methods and agents are secure. In this paper, proposed schema will be explained and compared with already known electronic voting systems.

***Keywords*—**Electronic voting, E-voting, Mobile software agents, Offline electronic voting.

I. Introduction

VOTING is one of the basic elements of the current republics. Due to the conveniences they will bring, it can be foreseen that electronic voting systems will be a requested element of democracy in the future. Electronic voting systems are designed to offer better performance and economy than the conventional ones. Speed and simplicity of tallying are other advantages [1]. Trustworthiness is another topic, and could be widely argued socially. However, it can be said that a system which doesn't get involved with the human factor is reasonably more secure. We can suppose electronic voting systems where security of the system is proved technically are more secure than conventional voting systems. In this paper, electronic voting refers to a voting system where authorized individuals cast their votes through a large-scale communication system and the agent refers to a software object executing a set of operations due to definitions set by a user or another software object.

Current electronic voting systems can be classified in three basic approaches: homomorphic cryptography based [4, 11-14], blind signature based [15-22] and mix-net based [5-10]. In homomorphic cryptography based systems, generally, encrypted ballots of each voter added into the same data package and decrypted as a whole using a master key that is different than any of the voters' key. Yet, this explanation can vary due to cryptographic algorithms used. In blind signature based systems, each voter blinded his/her ballot before the ballot had been signed by an authority, then sent the signed ballots to a collective pool that undertake tallying after unblinding them. Mix-net based systems composed of several hosts queued in an array. In these systems, each host in the array receive a group of encrypted ballots as input and while producing a permutation of the ballots as output, partially decrypting them; where at the end of the array, ballots are totally decrypted.

As far as known, use of agent-based approach in electronic voting systems is not examined yet. Agent-based approach has many advantages over its conventional counterparts. Agents can be customized just before the election and the authority can gain required control over voter's computer to execute a fair election. For example, all communication of the voter's computer can be interrupted during the voting phase to prevent any kind of trojan horses or remote control of the computers. In a large scale election, individual voters can not be trusted. Fortunately, agent-based approach can eliminate this problem by standardizing voters' behaviors. Any other unsigned voting interface might be altered to cast unfair votes. Also, agents reside on the voters' computers during the voting phase only which makes them more resistive to be altered. Consequently, agent-based approach may be the answer for an election system that voters can cast votes at home without any officer supervising the procedure with thoroughly analyzed security mechanisms.

Current approaches to electronic voting systems require one or more servers that have to calculate cryptologic algorithms for authorization, encryption, etc. Cryptographic calculations and the network load are generally the bottleneck of these systems. Using agent-based approach, a significant part of the cryptographic calculations can be run on the voters' computers instead of servers. Also, the number of network transactions is less than conventional algorithms as the agents conduct their data together with them. Unfortunately, this advantage of less transactions might be lost if the size of the agents increase.

II. Requirements of Electronic Voting Systems

Report of the National Workshop on Internet Voting [2] describes the requirements of electronic voting systems clearly. Seven of these requirements consider security issues, while the others consider economic or social issues. The seven requirements are as follows:

- Eligibility and Authentication—only authorized voters should be able to vote;

Manuscript received September 15, 2005. This work was supported by the Advanced Technologies Program of the Istanbul Technical University.

M. T. Sandıkkaya is with the Informatics Institute of Istanbul Technical University, Istanbul, 34469 Turkey (e-mail: tahir@be.itu.edu.tr).

B. Örencik is with the Computer Engineering Department, Electrical and Electronics Faculty of Istanbul Technical University Istanbul, 34469 Turkey (e-mail: orencik@cs.itu.edu.tr).

- Uniqueness—no voter should be able to vote more than one time;
- Accuracy—election systems should record the votes correctly;
- Integrity—votes should not be able to be modified, forged, or deleted without detection;
- Verifiability and Auditability—it should be possible to verify that all votes have been correctly accounted for in the final election tally, and there should be reliable and demonstrably authentic election records;
- Reliability—election systems should work robustly, without loss of any votes, even in the face of numerous failures, including failures of voting machines and total loss of Internet communication;
- Secrecy and Non-Coercibility—no one should be able to determine how any individual voted, and voters should not be able to prove how they voted (which would facilitate vote selling or coercion).

III. Proposed Schema

A. Preliminaries

We propose an agent-based offline electronic voting system. The voting system consists of three phases and three parties. The phases are registration, vote casting and tallying. An optional vote changing phase can be added after the vote casting phase. The parties are the authority, the voter and the tallying pool. Each phase in the schema affects the following phases, so security issues of each phase have to be ensured. Fortunately, any security leakage that does not reveal the private key of the authority or the pool affects only the related voter, not the system.

In the registration phase, each individual voter has to be registered as an authorized voter. During registration, voters can be identified by officers or by biologic methods like fingerprints; voters' keys can be stored in smart cards or memory sticks, etc. Independent of the method used, we assume that each individual voter receive his/her unique key pair and any other data needed. It is clear that the security of the key pair is up to the voter if there is not any integrated hardware or password protection for the keys.

In the vote casting phase, each individual voter receive two agents, interact with the agents then send them back. One of the agents authorizes the voter on the authority side where the other carries the vote to the pool.

In the tallying phase, votes in the pool are decrypted and counted. At the same time, the private key of the pool and the ballots are announced so that everybody can audit whether the tally is correct.

The authority is the party that stores each individual voter's public keys. The authority also has its own key pairs and the public keys of the pool. One of the authority's key pair is used for creating votes and the other is used for changing votes due to voter's will. The authority stores two boolean arrays for voters. The voter is marked after receiving the agents in the first array. Second array is used to count the successful vote castings for each voter.

The pool is the party that receives ballots and the encrypted votes inside them. It is also the party does tallying. For the sake of trustworthiness, the pool can be expanded into several independent hosts, each have their own key pairs. Still, we assume a single fair host manages the pool party for simplicity.

The voter interacts with two agents: the register agent and the ballot agent. The register agent authorizes the user and ensures the authority whether the voter casts a valid vote. Register agent carries the one of the private keys of the authority and the public key(s) of the pool. The register agent can not predict the vote casted where the ballot agent stores and carries the vote to the pool and can not predict who the vote caster is.

Finally, used cryptologic methods, random number generators and the agent frameworks are assumed to be secure till the end of the paragraphs detailing the work flow.

B. Workflow

1. Registration Phase

It is assumed that each voter registers through authorized offices. A key pair belongs to the voter and the public key of the authority is delivered to the voter. Public key of the voter is stored in the authority's computer.

2. Offline Encrypting and Signing

The authority injects the public key(s) of the pool(s) and the private key of the authority which one is for vote creating into each register agent, encrypting the agents using a session key and finally encrypts the session key and signs the package with the related voter's public key. At the same time, the pool(s) signs the ballot agent using its private key.

3. Vote Casting

Vote casting consists of nine steps:

1. The voter sends a message combined of two parts to the authority to trigger the voting process. Voters identification number encrypted with authority's public key and concatenated to the identification number that is encrypted with voter's own private key to form the message.
2. The authority decrypts the first part of the incoming message using its private key, then decrypts the second part using voter's public key. If two parts of the message are same and meaningful, it searches for the voter's identity in the arrays of the registered voters. If the voter has not been registered, process canceled.
3. The authority searches for the voter's identity in the array of vote casted voters. If the voter has not already casted a vote, the register agent is sent to the voter as it is.
4. The voter decrypts the register agent appropriately and activates it. Register agent sends a message to the pool to receive the ballot agent and after a successful transaction activates the ballot agent.
5. The ballot agent asks the voter to cast a vote. The vote is concatenated with a random number, blinded [3] and sent to the register agent.
6. After signing the blinded vote with the authority's private key and sending it to the ballot agent, the register agent terminates itself.
7. The ballot agent unblinds the data to obtain the signed vote, digests the signed vote and encrypts

the signed vote with the pool's public key. The ballot agent stores the digest in the voter's computer and sends the signed vote to the pool. After acknowledging that the vote was casted, terminates itself.

8. The voter created a message combined of two parts; voter's identification number encrypted with authority's public key and an acknowledgment message encrypted with voter's private key.
9. The authority marks the voter's identity in the array of vote casted voters.

4. Vote Changing

In the proposed schema each voter can change his/her voting strategy till the end of the election. The authority has another private key for this process which will be mentioned as "alternative private key" in the rest of the paper. A second public key has no use for the voter, so schema assumes that the voter has only the first public key of the authority.

1. The voter sends a message combined of two parts to the authority to trigger the voting process. Voters identification number encrypted with authority's public key and concatenated to the identification number that is encrypted with voter's own private key to form the message. The identification number is salted before the encryption.
2. The authority decrypts the first part of the incoming message using its private key, then decrypts the second part using voter's public key. If two parts of the message are same and meaningful, it searches for the voter's identity in the arrays of the registered voters. If the voter has not been registered, process canceled.
3. The authority searches for the voter's identity in the array of vote casted voters. If the voter has already casted a vote, a new register agent is created and sent to the voter. This version of register agent includes the *alternative* private key of the authority.
4. The voter decrypts the register agent appropriately and activates it. Register agent sends a message to the pool to receive the ballot agent and after a successful transaction activates the ballot agent.
5. The ballot agent asks the voter to cast a vote. The vote is concatenated with a random number, blinded [3] and sent to the register agent.
6. After signing the blinded vote with the authority's *alternative* private key and sending it to the ballot agent, the register agent terminates itself.
7. The ballot agent unblinds the data to obtain the signed vote, digests the signed vote, concatenates the previous vote's digest and encrypts the whole message with the pool's public key. The ballot agent stores the new digest in the voter's computer when erasing the old one. After sending the new vote and acknowledging that the vote was casted, terminates itself.
8. The voter created a message combined of two parts; voter's identification number encrypted with authority's public key and an acknowledgment message encrypted with voter's private key. Acknowledgment message is salted before encryption.
9. The authority adds another mark to the voter's identity in the array of vote casted voters.

5. Offline Tallying

After the vote casting phase finished, the pool filters the stored messages to capture the valid votes. The process is as follows: All packages including the original votes and the changed votes are decrypted using pool's private key. If there are more than one host in the pool side, all of the hosts have to join a coalition to decrypt the packages. The pool digests all votes and checks if the digests that was once concatenated to the changed votes are really mapping a vote. If this is the case, mapped votes are deleted; if not, the vote with the fake digest is deleted. At the end of this process, there must be only valid votes and valid digests which are mapping the votes in the pool.

All of the valid votes are decrypted appropriately using authority's public key (or *alternative* public key if the vote is changed) to read the plain text votes. At the same time, a table of valid digests published.

C. Security Issues

Seven requirements of a trustworthy electronic voting schema are ensured as follows:

- Eligibility and authentication is ensured during the registration phase by officers or biologic authentication methods. During the vote casting or vote changing phase, it is not possible to activate an instance of the register agent if the voter is not authorized.
- Uniqueness is ensured by the pool during the tallying phase. The pool accepts only the original votes or the votes which was fairly changed by the voter.
- Accuracy is ensured by the pool during the tallying phase. It can be ensured that a vote is stored correctly if the pool stores a digest mapping the vote.
- Deleted or added votes can be detected simply counting the vote casters in the authority and the votes in the pool. Any modification can be detected by the digests. Therefore, it has to be ensured if the digests are altered. If there is more than one host in the pool side, one fair party is enough to detect the modification. Finally, the voter can check if the digest of his/her vote is in the final tally.
- Verifiability and Auditability is ensured by the pool and the voter. Each voter can check if the digest of his/her vote is in the final tally. A fair independent court make the pool host(s) announce their private key(s) for re- tallying.
- Reliability is ensured by the agent-based approach. As the vote casting and tallying is executed independent of each other, any temporary connection problem during the vote casting or

tallying phase will not affect the whole system.

- Secrecy and Non-coercibility is ensured by the ballot agent as it creates random numbers and concatenates them to the votes. No one could claim that he/she has been created that precise number.

Some other security issues have to be discussed except these requirements; agent security problems, secrecy problems and DOS attacks make the whole schema suffer. Fortunately, there are some methods to resist.

By salting the messages in the first and eighth step of the vote changing phase and sending the identification number after encrypting with voter's private key in the first step of the vote casting phase, the schema resists against replay attacks which can pave the way for denial of service. (Salting is a technique that adds some random bits after the original message. It is useful when there is a possibility of sending the same message more than once.)

To keep the voting strategy of a voter secret, blinding method is used in the vote casting and vote changing phases. Blinding can easily achieved for an RSA encryption schema by executing the following algorithm: A random blinding factor b created and the number $x = v.b^e \pmod n$ is calculated using authority's public key and the vote, x is sent to the register agent. Register agent calculates the number $y = x^d \pmod n$ and sent it back to the ballot agent. Ballot agent unblinds the number by calculating $z = y / b \pmod n = v$.

Using more than one host in the pool side makes the schema resist against secrecy problems caused by traffic analysis. As the votes are encrypted in a known order using hosts' public keys, the hosts can behave just like a mix-net during the tallying phase. Performance drawbacks of mix-net approach are relatively unimportant as the tallying phase is executed offline.

Probably, the most problematic and important issue is the agent security. As we mentioned before, leakage of authority's private key is lethal for the security of the schema; even the software agents will not let people execute themselves unauthorized. It might be possible to create valid votes using authority's private key and sending them to the hosts at the pool side. However, detecting the surplus votes is easy by comparing the number of votes in the pool and the number of vote casted voters in the authority; it is not possible to find which votes are fake, causes a confused crowd. Fortunately, Hohl proposes a solution [23] against data leakage from an agent which can be used together with other methods [24] which protect agents against misbehaving agent frameworks. Hohl's method is messing up the code and data of the agent while keeping its functions unchanged till its expiration time. This method can be safely used in our schema as an election continues at most a few hours. Furthermore, it is possible to deliver fair frameworks to voters during the registration phase. Agents can check whether the frameworks are modified or not by digesting the framework codes before activating. It is also possible to inject a random number into voter frameworks during registration phase which makes each framework's digest unique, so register agents check each voter's framework independently.

IV. Conclusions and Future Work

Many electronic voting systems ensure the required security issues using many network transactions and cryptologic calculations. On the contrary, using agents, we observe that some security mechanisms and cryptologic calculations can be transferred into the voter's computer reducing the load on the servers during a large-scale election. Moreover, offline calculations make the schema attractive and eliminate the necessity of expensive servers. If the servers are distributed homogeneously, each server can act mostly as a file server instead of executing cryptologic calculations. Another advantage of using agent-based approach is the flexibility. Each computer can be customized and supervised remotely and any modification in the voting system can be transferred to all of the computers easily with the help of agents.

Such a schema with 1000 voters and a single host at the pool side can easily realized using Java language. In the simulation, 1024 bit *RSA* keys used for asymmetric cryptography, 168 bit *TripleDES* keys used for symmetric cryptography, *SHA1* is used for digesting and Java provided *SecureRandom* class used for random number generation. In the simulation, generation of a register agent takes between 2000 and 2500 milliseconds with a mean value of 2322 milliseconds and tallying of a vote takes between 170 and 470 milliseconds with a mean of 236 milliseconds on an ordinary computer which has 1000 Mhz of CPU speed and 512 MBytes of RAM. Extrapolating the results to a ten million voter scenario, calculations during the registration takes about 5 hours and tallying takes about 40 minutes with 1000 ordinary computers. It have to be noted that adding new hosts to the pool side affects the tallying time linearly. Therefore, 10 independent hosts at the pool side makes the tallying time about 400 minutes, means nearly 7 hours.

In the future, agent-based approach can be adapted on other electronic voting schemas. Using homomorphic cryptography in an agent-based electronic voting approach might be a better way of voter secrecy.

References

[1] Ku, Ho, "An e-Voting schema against Bribe and Coercion," in Prooceedings of the 2004 IEEE International Conference on e-Technology, e-Commerce and e-Service, Taipei, 2004.
[2] "Report of the National Workshop on Internet Voting: Issues and Research Agenda," Internet Policy Institute, Maryland, 2001.
[3] M. Stadler, J.M. Piveteau, and J. Carmenisch, "Fair blind signatures," Advances in Cryptology - Eurocrypt '95, St.Malo, 1995, pp. 209-219.
[4] Benaloh. "Verifiable Secret-Ballot Elections," Ph.D. Thesis, Yale University, New Haven, 1996.
[5] D. Chaum. "Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms," Communications of the ACM, vol. 24, no. 2, 1981, pp. 84-88.
[6] M. Abe, F. Hoshino. "Remarks on Mix-Networks Based on Permutation Networks," presented at the PKC 2001, Cheju Island, Korea.
[7] D. Boneh, P. Golle. "Almost Entirely Correct Mixing With Applications to Voting," presented at the 9th ACM-CCS Conference, Washington, USA, 2002.
[8] J. Furukawa, K. Sako. "An Efficient schema for Proving a Shuffle," presented at the CRYPTO 2001, Santa Barbara, USA.
[9] P. Golle et al., "Optimistic mixing for Exit-Polls," presented at the Asiacrypt 2002, Queenstown, New Zealand.
[10] M. Jakobsson, A. Juels, R.L. Rivest. "Making Mix Nets Robust for Electronic Voting by Randomized Partial Checking," presented at the USENIX Security Symposium, San Francisco, USA, 2002.

[11] O. Baudron et al., "Practical Multi-Candidate Election system," presented at the PODC 2001, Rhode Island, USA.
[12] R. Cramer et al., "A Secure and Optimally Efficient Multi-AuthorityElection schema," presented at the Eurocrypt 1997, Konstanz, Germany.
[13] P. Fouque et al., "Sharing Decryption in the Context of Voting or Lotteries," presented at the Financial Cryptography 2000, Anguilla, British West Indies.
[14] B. Schoenmakers. "A Simple Publicly Verifiable Secret Sharing schema and its Applications to Electronic Voting," presented at the Crypto 1999, Santa Barbara, USA.
[15] G. Dini, "A Secure and available electronic voting service for a large-scale distributed system," Future Generation Computer Systems, vol. 19, 2002, pp. 69-85.
[16] C.-C. Chang, W.-B. Wu. "A secure voting system on a public network," Networks, vol. 29, no. 2, 1997, pp. 81-87.
[17] D. Chaum. "Elections with unconditionally secrets ballots and disruption equivalent to breaking RSA," Proceedings of Eurocrypt'88, Davos, Switzerland, 1988, pp. 177-182.
[18] J.D. Cohen, M.J. Fischer. "A robust and verifiable cryptographically secure election schema," Proceedings of the 26th IEEE Annual Symposium on Foundations of Computer Science, 1985, pp. 372-382.
[19] K.R. Iverson, "A cryptographic schema for computerized general elections," Proceeding of Advances in Cryptology— CRYPTO '91, Santa Barbara, USA, 1991, pp. 405-119.
[20] H. Nurmi, A. Salomaa, L. Santean, "Secret ballot elections in computer networks," Computer Security, vol. 10, no.6, 1991, pp. 553-560.
[21] R.S.-N.A. Baraani-Dastjerdi, J. Pieprzyk, "Secure voting protocol using threshold schemas," Proceedings of the 11th IEEE Annual Computer Security Applications Conference, New Orleans, USA, 1995, pp. 143-148.
[22] C. Boyd, "A new multiple keys cipher and an improved voting schema," Proceedings of Advances in Cryptology— EUROCRYPT'89, Hounthalen, Belgium, 1989, pp. 617-625.
[23] F. Hohl, "Time Limited Blackbox Security: Protecting Mobile Agents From Malicious Hosts," Lecture Notes in Computer Science, vol. 1419, 1998, pp. 92-113.
[24] A. Wagner, "Implementing Mobile Agent Security In An Untrusted Computing Environment," Proceedings of the 8th International Conference on Telecommunications, Zagreb, 2005, pp. 591-594.

Mehmet T. Sandıkkaya born in Ankara, 1979. Sandıkkaya obtained his B.Sc. degree from the Electrical Engineering Department of Istanbul Technical University, Istanbul, Turkey in 2002.

He was with Aselsan in 2000 and with IBM in 2001 as a Trainee. He is working as a Research Assistant since 2002 with Informatics Institute of Istanbul Technical University, Istanbul, Turkey.

Eng. Sandıkkaya is a student member of IEEE since 2000 and the chair of IEEE Power Engineering Society Student Branch Chapter at Istanbul Technical University since 2004.