JOURNAL OF
EMERGING INVESTIGATORS

Article

Design and implementation of a cryptographically secure electronic voting infrastructure

Danielle Park[1] and Moshe Satt[2]

[1] Paramus High School, Paramus, New Jersey

[2] New York University, New York, New York

SUMMARY

Cyber security is the application of technologies, processes, and controls to protect against attacks on confidentiality, integrity, and availability. Cryptography maintains confidentiality by securing communications from being intercepted, provides integrity by preventing unauthorized modification of data, and provides availability by allowing data to be transmitted securely. There is currently limited to no application of cryptographic controls at election sites in today's voting environment due to the use of legacy systems and paper systems that do not support the technology required for encryption. This paper proposes an electronic voting solution to mitigate risk through the design and implementation of a secure, electronic voting app and infrastructure. Here, we present evidence, using a thorough National Institute of Standards and Technology (NIST) risk assessment, that removing human interaction remediates vulnerabilities within today's infrastructure and mitigates overall risk. We also extract multiple NIST Special Publication 800-53 family controls to analyze the vulnerabilities in today's voting infrastructure. Using our proposed secure electronic infrastructure, we mitigate the risk inherent in today's election environment, and we propose a model to secure our democracy and the future prospect of voting electronically.

INTRODUCTION

Free and fair elections are the foundation of any democracy. Thus, it is critical to maintain confidentiality, integrity, and availability of votes and voting systems in elections. The importance of recognizing greater confidence in election security and democracy begins with the implementation of secure voting. Today's election and electronic voting environment in the United States, associated with a lack of voting standards, fails to meet the cybersecurity standards of NIST to ensure public confidence in our democracy (1). We have found that vulnerabilities in today's voting environment include physical access with limited to no security on voting machines, unprotected backend servers, and no redundancy or backups. Additionally, threats of changing, deleting, and re-voting all come to a dangerous common denominator: a loss of public confidence and integrity in the election system (2). The transition to electronic voting eliminates the need to rely on proprietary voting machines and eliminates potential supply chain problems (manufacturers of software and hardware).

Preceding approaches to electronic voting have seen catastrophic consequences. Subject to attack, security issues arose as seen in the denial-of-service attack in 2009 with the Austrian Student Federation election (3). Additionally, in 2011 and 2013, e-voting in Norway was subject to software and physical implementation errors (3). At the same time, Estonian elections were subject to various vote manipulations in the form of malware and revocation of whole electronic votes (4). Consequences included failed verifications, coercion, false verifications, random factor exposure, and diverting verification. Evidently, the drawbacks are numerous, due to failing voting systems and thus there is a need for a reliable e-voting system.

The system we are presenting here in this paper is intended for voting sites. Many solutions designed previously have allowed for electronic voting remotely at home. This revealed several complexities and complications. These designed systems are subject to countless attacks from securing votes from home to the site of counting; we examined these e-voting systems such as Estonia's and concluded that while we can have a hybrid approach, we can securely vote with a methodology using standardized technology in-person. Regarding potential alternatives, we addressed taking our proposal to an at-home setting, however, this would raise issues regarding authorization, accounting, and identification. Identification with voting at home could warrant a national identification to be issued. The challenges of voting online would spike voter fraud as seen in previous elections. According to The Heritage Foundation, there have been 17 official findings of election fraud across the United States with 1,328 proven instances of voter fraud (5). The protocol would have to be national in our federal standard to complement our federally approved voting app with a digital certificate. By requiring either a physical token or biometrics in advance, the voting process would have to be in-person, and people may have privacy concerns. Our proposed infrastructure mitigates these fears that could potentially discourage voters from voting.

The purpose of this paper is to study whether electronic voting is the answer to protect election infrastructure. We hypothesize that we need electronic voting to secure elections because there are too many steps within the voting process where votes can be tampered with today. Implementing industry-standard algorithms and globally-accepted standards and protocols, electronic voting is the only option for ensuring the security of the vote for the high encryption standard used. Since the reliance on technology will create additional threats, we mitigate risks with cybersecurity.

We posit that only by voting electronically can we ensure the security of the vote from the moment it is cast until it is ultimately tallied. Our proposed election infrastructure incorporates a

voting app written with open-source software that would be federally approved. It utilizes a defense-in-depth approach to cybersecurity, including multiple layers of encryption through additive homomorphism, Paillier homomorphic cryptography, and Diffie-Hellman key exchange (6). It implements multiple layers of encryption sent through an encrypted Virtual Private Networking (VPN) tunnel to transmit and store all votes securely, both locally and in the cloud. Confidentiality of votes is performed by encrypting each vote with a unique, pseudo-randomly generated "vote" key. An attacker is not able (in polynomial time) to determine the candidate the vote was cast for. Any personally identifiable information that can connect a vote to a voter, such as Internet Protocol address or timestamp, is encrypted and stripped off when the vote is decrypted and counted (7).

To measure the risk qualitatively (using high, medium, and low risk levels), we utilize the NIST Special Publication (SP 800-53) for risk assessment (8). This paper highlights the most glaring cybersecurity risks, accidental and malicious, to today's electronic voting systems and mitigates those risks by applying the appropriate NIST SP 800-53 controls, which are the fundamental components of our proposed secure electronic election infrastructure (8).

After researching several cybersecurity controls and implementing the necessary voting capacity, we concluded that by removing human interaction, vulnerabilities within today's infrastructure are remediated and the overall risk is mitigated. Our designed infrastructure breaks the barriers to tactics used to discourage voting as seen in the past; our solution is accessible to everyone regardless of their race, gender, abilities, or income, providing an equalizer for all voters.

RESULTS

Design

We proposed the feasibility of our voting system through the development of our system on the cloud using Azure. This allowed us to simulate a voting site with our code using every piece of our infrastructure, showing that they are all readily accessible in cloud technology. Thus, we tested and verified accessibility and feasibility by designing our infrastructure in the cloud and coding the necessary cryptographic components. Our hypothesis was examined through the creation of a secure and low-risk electronic voting election infrastructure. Our methods include a voting route, electronic voting system, homomorphic encryption, Diffie-Hellman Key Exchange, azure components, hash-based authentication code, and advanced encryption standards. To progress through each stage of this election scheme, each vote goes through various stages: the first stage involves storing the vote, the second phase is transmitting the vote and the third phase is to store the vote on the server.

Risks were determined with the NIST framework with the parameters as the control families; this was a qualitative assessment (high/medium/low risk) based on these parameters. We selected the most important controls while taking a thorough sampling in most areas relevant to voting infrastructure to show a diverse assessment. Our cyber solution at the federal level must be compatible with NIST standards. The overwhelming majority of controls are not adequate in today's methodology and environment whereas in ours, they are. Our thorough risk assessment demonstrated that by removing human interaction, vulnerabilities within today's infrastructure are remediated and the overall risk is mitigated as per the NIST SP 800-53 family controls including: Access Control (AC), Control Assessments (CA), Contingency Planning (CP), Policy and Procedures (PE), Personnel Security (PC), System and Communications Protection (SC), and System and Information Integrity (SI) (**Table 1**).

Assessment

The access control policy addresses procedures for facilitation and implementation of responsibilities, commitment, coordination among entities, and compliance of policies. This control requires that rules define conditions under which an access takes place in management. With the compliance of control number AC-4 in election infrastructure, this control would enforce approved authorization for the flow of information within the voting system, as the information flow enforcement is satisfied with the vote transmitting from the voting booth to the secure cloud server (both controlled and secured for the flow of vote end-to-end). Control number AC-20 establishes, identifies, and defines controls to be implemented on external systems with the established operation. External systems are crucial to process, store, or transmit votes using external systems. We implemented a single technology, vendor, and specific hardware and software which were used by applying these controls uniformly through enforcing election cybersecurity standards. We applied control number AC-25 by implementing a standardized, hardened tablet with minimal attack surface, the latest security patches, and the cloud server operating system that is configured similarly to create a tamperproof election infrastructure. The trusted platform module (TPM) is tied to the trusted computer base (TCB) boundary, holding tamperproof crypto keys and unique transport keys in the TPM hardware.

Securing, monitoring, testing, and assessing systems are pivotal to the control assessments family of NIST controls. The Security Assessment and Authorization control family includes controls that supplement the execution of security assessments, authorizations, continuous monitoring, plan of actions, and system interconnections. Control number CA-2 requires the scope of the development of a security assessment plan followed by both full assessment and application of cybersecurity controls. For control number CA-3, our proposed voting app resolves the issues of votes being exploited at transition points since our system applies controls that secure the vote as it traverses from the tablet to the server. The authorization control from CA-6 assigns an authorizing official for the voting system with our solution enforcing standards uniformly across all voting sites. Cybersecurity controls are primarily managed by the app developer, VPN provider, and cloud provider. The responsibility from the election site administrator to secure the data of the vote is completely removed. The development of a system-level continuous monitoring strategy must be monitored in accordance with control number CA-7 with built-in monitoring controls at every step in the process, including cybersecurity alerting and notifications. To apply control number CA-8, penetration testing goes beyond automated vulnerability scanning and is conducted by agents and teams with demonstrated skills with the secure version of GovCloud. Internal system connections from control number

CA-9 documents connections, terminates them after a defined condition, and reviews the connections needed for each internal connection, in this case, our voting tablet. We eliminate transitions by ensuring the vote stays in a digital form from start to finish. It is, therefore, able to be secured with encryption from end-to-end (ballot to count).

The Contingency Planning family of the NIST controls establishes necessary measures for the transfer and resumption of data in cases of backup on security controls in case of system compromise or breach. Control number CP-7 defines an outline for processing capabilities with recovering time at alternate processing sites; cloud environments have

Family	№	NIST Control	Today's Voting Infrastructure	Proposed Election Infrastructure
AC	1	Access Control Policy and Procedures	No access controls for vote accessibility	Rigorous access controls at every transition point of vote
AC	4	Information Flow Enforcement	No chain of custody of votes	End-to-end control of votes
AC	20	Use of External Information Systems	Voting systems open to unsecure external vendors	Control uniformity and cybersecurity standards
AC	25	Reference Monitor	Inadequate patch management	Hardened tablet with automatic security patches
CA	2	Security Assessments	Does not apply necessary controls nor security control assessment	Assesses and applies necessary cybersecurity controls
CA	3	System Interconnections	No delineation of how vote is secured end-to-end	End-to-end security from tablet to server with controls
CA	6	Security Authorization	Administration likely not qualified to secure site, IT infrastructure, or vote	Cybersecurity controls managed by app developer and VPN and cloud provider
CA	7	Continuous Monitoring	Lacks monitoring controls from when vote is cast to when it is counted	Built-in monitoring controls at every step-in process with cybersecurity alerts
CA	8	Penetration Testing	No penetration testing or funding for testing at all	Internal teams perform pen-testing from cloud and internet providers funded by government
CA	9	Internal System Connections	No controls to secure paper to electronic vote form	Vote remains in digital form with encryption
CP	7	Alternate Processing Site	No alternate site to process or hold votes	Cloud environments have multiple sites with replication of encrypted data
CP	9	Information System Backup	No backup or security control applied with potential to lose all votes on server	Cloud facility has control across data centers spread nationally
PE	2	Physical Access Authorizations	Trivial access to data, votes, and server, leaving hacking at high risk	Multiple layers of physical access controls with authorized personnel
PE	3	Physical Access Control	No physical access control for backend IT systems, servers, and databases	Votes stored off-site in the cloud with high levels of physical access controls
PE	6	Monitoring Physical Access	Some external monitoring, but unmonitored and unsecured areas for IT equipment	Off-loading vote immediately to secured facility
PE	9	Power Equipment and Cabling	Servers placed in locations subject to power failures, leading to loss of votes	Multiple data centers with redundant power and emergency battery backup
PE	23	Facility Location Control	Voting sites are subjective to environmental or man-made threats	Cloud data centers are strategically located
PS	6	Access Agreements	No access agreement at state and local level for all election sites	Control applied at ISP and cloud service provider level
SC	13	Cryptographic Protection	Limited to no application of cryptographic protection with legacy systems unable to support encryption measures	Vote undergoes multiple layers of encryption with several cryptographic protections
SI	19	DE-Identification Control	Voter can personally identify through multiple channels at the voting site	Encryption of each vote separately to dissociated information from vote itself

Table 1: Risk assessment comparing today's voting infrastructure to our proposed election infrastructure determined in arrogance with NIST SP 800-53 family controls (8). High risk (red), medium-high risk (dark orange), medium risk (light orange), and low risk (yellow) were determined based on the standards outlined.

multiple sites with real-time replication of the encrypted data. Control number CP-7 on system backup effectively lowers security concerns by applying this control at the cloud facility, this control implemented across all the data centers, including real-time replication of the votes across multiple secure data centers spread around federally.

Access control and monitorization of voting sites are among the many challenges faced by the current voting environment; the Policy and Procedures family of NIST controls outlines these risks. Control number PE-2 of Physical Access Authorizations underlines the development, approval, and maintenance of administrative access to system facilities. We implemented multiple layers of physical access controls including this control whereby only authorized personnel are allowed access to this highly sensitive information. Control number PE-3 risks are mitigated by storing data (votes) off-site in the cloud and by not requiring an on-site data store. In control number PE-6, we satisfy the outlined procedure by off-loading the vote immediately to a secured facility to mitigate the risk of this control not being adequately applied by the implementation of a compensating control. Power equipment and cabling controls aim to protect power equipment and power cabling for the system from damage and destruction as specified in control number PE-9. Under control number PS-6, the control is applied at the internet service provider (ISP) and cloud service provider level, verifying individuals who require access to information and systems.

Cryptographic protection controls are outlined in the systems and communications protection family of NIST Controls under number SC-13; our solution hinges on the vote being cryptographically protected at every step along the way, with multiple layers of encryption as outlined in our crypto protections employment for effective implementation.

Control number SI-19 from the System and Information Integrity family of NIST controls strives to primarily protect votes from modification/deletion (integrity), but additionally strives to protect the confidentiality of which candidate a voter voted for and the availability of the election systems and voting sites. By encrypting each vote separately, we effectively separate the identifying information such as IP address and timestamp from the vote itself, thus providing the de-identification control required to address purpose, scope, roles, responsibilities, management, and coordination.

Storing the vote is encryption at rest, transmitting the vote is encryption in motion, and storing votes on the server is encryption at rest. We achieved a technical control of the proposed methodology that is more resilient to cyber-attack (**Figure 1**). Our algorithm is written in Python using the publicly available Crypto.Cipher package (**Input**). The secure cloud on the backend (**Output A**), the encrypted tablet on the front end (**Output B**), and the encrypted tunnel in between transport the secure data at rest and in motion. The secure cloud infrastructure is provisioned for open use by the public, but is owned, managed, and operated by the government and exists on the premises of the cloud provider in accordance with NIST SP-145 (12). This model is superior in security level.

DISCUSSION

We hypothesized that we need electronic voting to secure elections because there are too many steps within the traditional voting process where votes can be tampered

Input

```
encryptedVote = encrypt_AES_GCM(vote, secretKey)
print("This is the encrypted vote:")
print()
print({
   'ciphertext': binascii.hexlify(encryptedVote[0]),
   'aesIV': binascii.hexlify(encryptedVote[1]),
   'authTag': binascii.hexlify(encryptedVote[2])
})
decryptedVote = decrypt_AES_GCM(encryptedVote, secretKey)
print()
print("This is the decrypted vote:", decryptedVote)
f = open("election.txt", "a")
f.write(dt_string)
f.write(str(count1))
f.write(str(count2))
f.write(str(binascii.hexlify(encryptedMsg[0])))
f.write(str(binascii.hexlify(secretKey)))
```

Output

A

```
n.txt
[16/10/2021 22:04:31]
1/0
b'7f'
b'089e5005fd0b3fea8706f1fac3fb454738189211094cfbb25bd701ddacddd5c7'

[16/10/2021 22:04:33]
2/0
b'f4'
b'd27068905720cabcba9db9ff038e037f34c402bf6e8485d13abe8bfa846ab6c5'

[16/10/2021 22:07:12]
2/1
b'1b'
b'13e62096f9c4fa8aafe6a42a7db473f6407ed3c3e2590e4b7eb6dca760024f15'
```

B

```
Vote 1 or 2:  1
now = 2021-10-16 22:00:17.760175
This is the 256-bit random key:
b"r\x81\xbd\x9e.cv\x1a][^@J`+\xb8\xb9<^q\x9f(\xe7n9\x1fd'\xaa+mE"

This the Encryption key: b'7281bd9e2e63761a5d5b5e404a602bb8b93c5e719f28e76e391f6427aa2b6d45'

This is the plaintext vote:
b'1'

This is the encrypted vote:

{'ciphertext': b'11', 'aesIV': b'3fcec11f303c1ff647cd5792894725db',
 'authTag': b'f80b51b44a6359add697ea615013d6ad'}

This is the decrypted vote: b'1'
date and time = [16/10/2021 22:00:17]
Total votes for candidate 1:   1
Total votes for candidate 2:  0
```

Figure 1: Encryption algorithm to encrypt the vote and secret key. The output represents the election text file (A) and simulates the interface behind the voting process (B). The text file shows encrypted storage of how votes are stored with a timestamp to replay loss of connectivity and retransmit votes that were after that time, but prior to restoring connection with the ephemeral vote key.

with. Thus, we implemented industry-standard algorithms and globally accepted standards and protocols, to reveal evidence that electronic voting is the only option for ensuring the security of the vote for the high encryption standard used. However, creating an electronic voting system still has many challenges to overcome to avoid compromising voting availability, especially with the cyber vulnerabilities that come with the implementation of an electronic voting scheme on a national scale (9). Although not addressed in this revision, there is a limitation to NIST SP 800-53 control PE-2. While we have addressed moving data off-site, we intentionally implemented cyber controls to reduce the need for physical security measures. In future studies, we will research more about remote voting and e-voting capabilities to enhance the physical security or physical voting environment, to further mitigate these risks. For example, coming to vote in-person cannot be anonymous. Our proposal, instead, is an adaptation to already existing voting sites, but does not allow for voters to vote in their homes; we can increase accessibility to voting for people who are unable to commute to the voting site. This is why there is still a medium risk, and cannot be low, due to intimidation factors (penalization, threat, hesitation) present in in-person voting at a voting site. The cost of physical measures, in turn, will be reduced on a national scale given that cloud infrastructure is more cost effective than in today's voting environment. Our cloud technology is the building block for creating opportunities to allow for remote voting.

Our NIST risk assessment revealed numerous vulnerabilities and found countless violations to SP 800-53. While the technical controls should be critical to ensuring election cybersecurity, legal mandates in combination with technical controls create an election infrastructure more resilient to cyber-attack. The in-person voting process should be uniform so that voters have the same seamless experience both at the election site or remotely; technology is equally accessible with the same controls. Although no specific technology is required, a cybersecurity mandate must be put into place with a national standard for election infrastructure from the federally approved app. All voting sites need to comply with cybersecurity measures as well as physical security measures. Additionally, compliance with NIST controls for election systems is necessary, leaving NIST control implementation up to each state.

In today's voting environment, election sites fail to meet the standards of implementing these controls for vote access. Leaving votes subject to tampering makes them vulnerable to malicious activities. Multiple variations of the chain of custody for a vote are seen as there is no federal mandate for a secure voting path; this leaves each state and voting site undergoing different approaches. When putting voting into perspective, violations include unauthorized monitoring of voting and manipulation of votes, both of which violate standards and pose a threat to the democratic process to society.

The election systems in use today lack an adequate reference monitor due to the lack of standardized operating systems and inadequate patch management. Services are exchanging information, however, paper votes are stored as a backup in case the system fails. However, the votes are now subject to tampering with insecure information exchange, which is often seen in voter fraud. This leap from analog to digital data puts the integrity and confidentiality of the original vote at risk of not being read, counted, or received properly. Currently, the administrator or authorizing official is not required to have any cyber training due to the lack of cybersecurity standards implemented nationally. If not qualified to implement and secure the IT infrastructure, then the site can no longer be secure or trusted to protect votes. The server hardware itself often does not utilize Redundant Array of Independent Disks (RAID) and a battery backup.

The government provides funding only if states agree to comply with the national standard; this provides enticements for states to follow the mandate while being provided funds for technology. Having compliance on a national scale is crucial to remove barriers to threats regarding information being shared between the government and the private sector. Private sector barriers include compensating controls and unknown vulnerabilities in software. Sharing information about vulnerabilities in the baseline of what is on the state level and what is on the federal level, modernization and implementation of stronger cybersecurity standards in the federal government embody zero trust architecture to improve software supply chain security, improve detection of cybersecurity incidents on federal government networks, and improve investigative and remediation capabilities within the election system.

Although creating electronic voting system protocols has been researched and proposed in the past, many methods are costly and have had controversy over the uncertainty of large-scale implementation. Creating an electronic voting system still has many challenges to overcome to avoid compromising voting availability, especially with the cyber vulnerabilities that come with the implementation of an electronic voting scheme on a national scale. Despite these concerns, electronic voting continues to advance with numerous countermeasures to vulnerabilities.

MATERIALS AND METHODS

Voting Route

We propose that a tablet can take a vote and encrypt it with a pseudo-random encryption key, securing it with a Diffie-Hellman generated symmetric key, encrypted in the VPN tunnel. This proposed method sends encrypted data and the key using the Diffie-Hellman generated transport key via the VPN tunnel (**Figure 2**). The vote and vote key would have to be transmitted to the server using the shared transport key inside of the tunnel, and then use homomorphic encryption to encrypt data so the total number of votes is decrypted in a single, secure process (10). The hash-based message authentication code (HMAC) function ensures that the vote count is accurate and is sent from the authorized party for public knowledge by digitally signing with a hash of a hash of a vote and secret key.

To maintain confidentiality and integrity, encryption (using a pseudo-randomly generated key by our voting app) would operate on each vote for an individual candidate. By setting up a VPN tunnel between the voting site and the cloud, the site's IP address would no longer be visible. As a result, everything that goes into the tunnel is encrypted; both the internet provider and an attacker cannot decrypt any of the traffic inside the tunnel. The symmetric transport key would then be negotiated through a Diffie-Hellman key exchange protocol to securely send the encrypted vote and its associated key, which is needed to decrypt the encrypted vote. Diffie-Hellman is utilized because it enables two parties communicating over

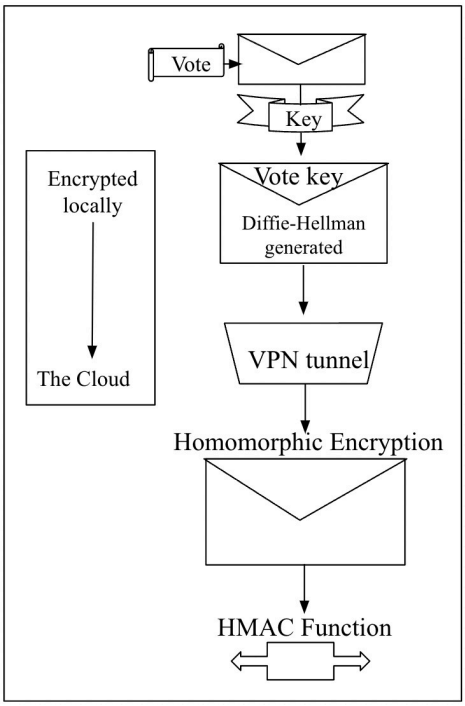

Figure 2: Proposed vote transmission. Each vote is encrypted locally and is transmitted to the cloud after going through a series of encryption and algorithmic functions. The vote is stored locally with the device key. Then, the encrypted vote is sent with the decryption key and is encrypted through the private transport key. Finally, a hash of a hash of vote and secret key is taken.

a public channel to establish confidentiality without it being transmitted over the internet. Using symmetric cryptography, this protocol is superior to others available for it is a public key that can both encrypt and decrypt the vote and data.

Structure of Electronic Voting System

To progress through each stage of this election scheme, each vote will go through various stages (**Figure 3**). The first stage involves storing the vote. The voter comes to vote at a typical voting booth, but instead of a piece of paper, they are presented with a tablet. With less surface area for attack, the voting software on the tablet reduces the attack surface. With greater surface area, there is more surface vulnerable to attack. Thus, surface area being anything that could potentially be compromised, we reduce the exposure to risk (threat or vulnerability).

Additionally, the tablet would not accept connections through the port; only charging is permitted through the Universal Serial Bus port. Mobile Device Management (MDM) policy is put into place for restrictions to restrict internet access and to only allow traffic via the VPN tunnel. It also hides all browser features. The Wi-Fi Protected Access 3 (WPA3) protocol is utilized with key agreement from the Simultaneous Authentication of Equals protocol, which supports elliptic curve cryptography. This is utilized since it is the latest and most widely accepted wireless Wi-Fi that is NIST recommended. Perfect forward secrecy and protection from offline brute force attacks ensure confidentiality as votes would not be able to be decrypted from any recorded data even if a password is obtained. Each vote has its own encryption/decryption key, and the vote key is only utilized temporarily until the vote is delivered.

The second phase is transmitting the vote. To tackle the vulnerability of physical tampering of votes, cloud servers and storage are utilized. Votes are stored locally on the device and then are sent to the cloud. Since both physical security and cybersecurity have a high risk based on our assessment of current voting circumstances, the cloud ensures the prevention of vote loss. Then, the vote is encrypted with a pseudo-random key and is saved locally before transmission. Thus, the encrypted vote and key are transmitted securely. The transport key ensures nobody can see or change a vote by interception.

The third phase is to store the vote on the server. When the server has the unencrypted vote, it needs to encrypt the vote to prevent modification; homomorphism is used. While the server is assigning the vote to a particular candidate, the server reverts to who the vote was cast for originally but cannot store the vote as who casted the vote before, and votes are encrypted with a public key. Each vote follows the same encryption, so each vote is re-encrypted. To avoid vulnerability through decrypting, homomorphic encryption is

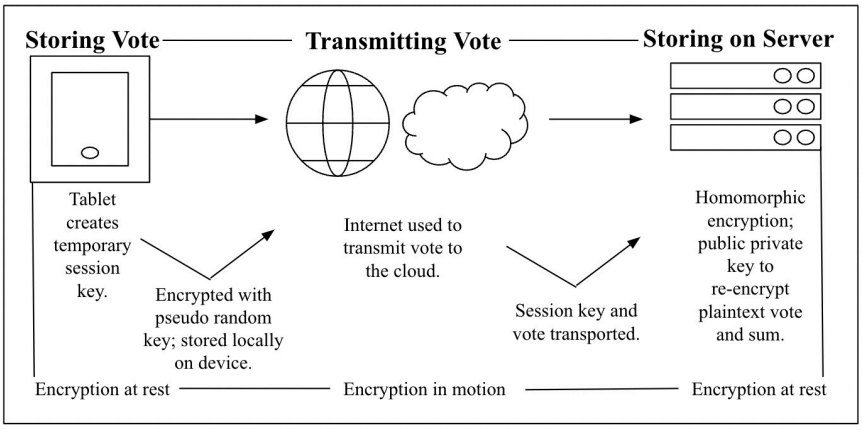

Figure 3: Structure of e-Voting system. The vote is stored on the tablet, transmitted from the internet to the cloud, and stored on the server. When storing the vote, the tablet creates a temporary session key which is encrypted with the pseudo-random key stored locally on the device. When transmitting the vote, the internet is used to send it to the cloud while the session key and vote are transported. When storing the vote on the server, homomorphic encryption is used for the public-private key to re-encrypt the plaintext vote and sum.

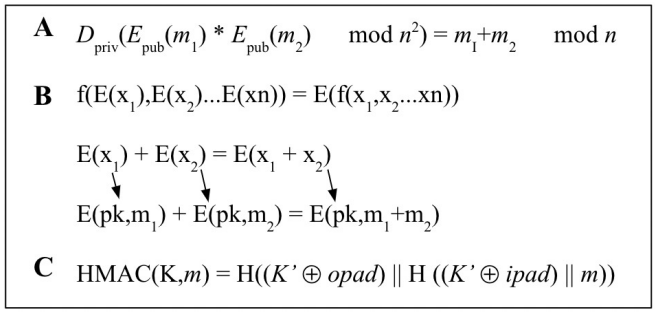

Figure 4: Mathematical encryption sequence for the public-private keys and HMAC function. (A) Additive homomorphism is applied to each vote received as the function is applied to the sum. (B) When two ciphertexts are multiplied, the result decrypts to the sum of their plaintexts. (C) A hash of a secret key and vote is taken using the HMAC function. Here, *H* is the hash function, *m* is the vote to be authenticated, *x* denotes the authenticated vote, *K* is the secret key, *K'* is a block-sized key derived from the secret key, *E* is encryption, ⊕ denotes XOR, || denotes concatenation, *opad* denotes outer padding, and *ipad* denotes inner padding.

used to immediately encrypt votes and perform the additive operation. At the server, votes are decrypted to count, so public keys can re-encrypt each plain text vote using homomorphic encryption (11).

Since the data is stored only on the tablet, the key is only known to the application and not to the user or administrator. The tablet will be protected with multi-factor authentication so even if the device is stolen, one cannot gain access to the voting tablet. Even the administrator who is able to unlock the tablet is still unable to unlock the vote. In the event the tablet is unlocked, one would still be unable to obtain data because the application encrypted the vote and discarded the vote key after the vote was confirmed to be received at the server. By encrypting everything with the public key on the backend server and keeping the private key stored in a hardware security module, the internet and network keep this voting scheme tamperproof.

Homomorphic Encryption

To leverage homomorphic encryption, a public/private key pair in the form of asymmetric cryptography is utilized; each vote is individually encrypted using the public key and then the encrypted votes are summed without decrypting them using homomorphism. We then use our private key to decrypt the sum of all the encrypted votes and count them in a single atomic operation without running a counter. This process is done all in one step; each vote is not decrypted individually because of additive homomorphism when decrypting the sum. This means that property cannot change during this read operation as each part runs independently of each process.

Additive homomorphism is applied to each vote received as the function is applied to the sum. Paillier cryptography supports the additive homomorphic property (**Figure 4A**). Applying this function to all votes, where *E* is the encryption function, the additive property and encryption loses the ability to decrypt each vote even if one were to obtain the private key; *E* needs a key and plaintext. Thus, when additive homomorphism is applied after adding the encrypted votes together, the ability to break back down into each vote is lost, making it completely anonymous. Additionally, *H* is the hash function which is used on *m*, the vote to be authenticated, with *x* authenticating the vote vote. *K* is the secret key, *K'* is a block-sized key derived from the secret key, ⊕ denotes XOR, || denotes concatenation, *opad* denotes outer padding, and *ipad* denotes inner padding.

The encryption function takes in two arguments as plaintext message and private key, where *E* is the encryption function and *pk* is the public key (**Figure 4B**). With homomorphic properties, when two ciphertexts are multiplied, the result decrypts to the sum of their plaintexts. The first vote gets encrypted with the encryption function using the public key, and then the second vote is encrypted with the public key. Then, the ciphertext is added and encrypted with the public key to count without decrypting. This way, nobody can see the vote coming in or its content, and it is already encrypted, so only the server is doing the decryption to the plaintext vote. Where *f* is a function that takes in already encrypted votes when applied, the encryption function runs, encrypting the whole set of votes. For *n-1* votes, this method encrypts *n* votes using one key, applying the public key to the *n*th vote. When the two ciphertexts apply the additive operation to them and encrypt the result with the public key, a third ciphertext is generated. The plaintext vote is represented as m_1 for the first vote and m_2 as the second vote; each is encrypted with the public key.

The votes are not individually decrypted to count them. As each vote comes into the server, after it is individually encrypted with the public key, it is cryptographically summed via homomorphic cryptography in its encrypted state without needing to decrypt them first (**Figure 4C**). Thus, at any time, the *n*th vote arrives at the server, there is an encrypted sum of the *n-1* votes stored on the server. The *n*th vote is immediately encrypted and summed with the *n-1* votes to create a sum of *n* votes on the server.

The Paillier scheme re-encrypts each vote with a public key. Meanwhile, the private key is used to decrypt and get the sum without decrypting the individual votes, ensuring integrity. Since votes are encrypted, the ISP is not able to see who is casting individual votes. Additionally, the VPN hides the endpoints, storing no logs, thereby hiding the identity of voting sites and the resultant voter identities.

Diffie-Hellman

Diffie-Hellman Key Exchange allows for secure, symmetric key exchange using a public channel (6). The key functions as a transport key, encrypting both the vote and the vote key. The key exchange in the public channel allows users to share a key without needing to have already shared a key previously. Creating the random transport key, pseudo-random numbers are negotiated between a server and client to generate a transport key. In case of a lost internet connection, votes must be encrypted and queued on the device. Our election scheme has the client randomly produce an encryption key and, once the circuit comes back, the random temporary transport key is negotiated. Encrypting the data locally on the device, the vote is sent, decrypted, then re-encrypted with the negotiated key when the vote is detected, keeping it encrypted with the vote and key sent along the secure VPN tunnel. Our voting site will hide all traffic inside of a VPN tunnel, so the ISP is only aware of connectivity to the VPN, not where the votes are.

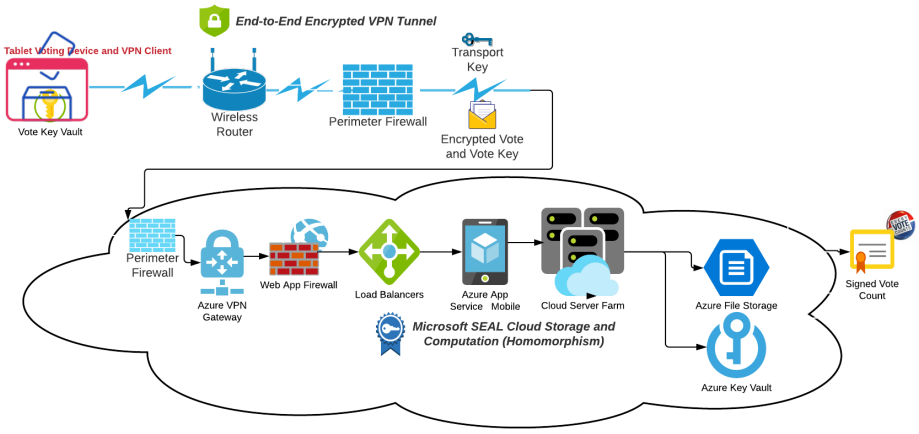

Figure 5: System diagram of end-to-end encryption from the moment the vote is cast to when it is tallied. From the vote key vault, the tablet voting device and VPN client have end-to-end encryption through the VPN tunnel. Within the cloud, the VPN gateway is met with the web app firewall that precedes the load balancers to the app service; the cloud server farm sends the vote to the Azure key vault and file storage and the signed vote count is transmitted.

Azure Components

The Azure cloud computing services are built, tested, and deployed to manage applications and services of election data centers. With end-to-end encryption, the system diagram shows the election infrastructure composed of Azure components (**Figure 5**). This system is feasible by using current technology by leveraging public cloud infrastructure. The vote key vault supports storing the voting software and Hardware Security Module (HSM) backed keys, secrets, and certificates. Perimeter networks enable secure connectivity between cloud networks and on-premises or physical datacenter networks with connectivity to and from the internet. Azure VPN Gateway connects on-premises election networks to Azure through Site-to-Site VPNs using IP Security (IPsec) and Internet Key Exchange (IKE). The Azure Web Application Firewall protects voting applications from web vulnerabilities. The load balancer stops routing the traffic to a failed virtual machine in the pool to make our application resilient to any software or hardware failures in that pool of virtual machines. Azure App Service provides authentication, data query, offline synchronization, and push registration capabilities to the voting app while File Storage enables the migration of on-premises file share-based applications.

Hash-Based Message Authentication Code

The voting protocol of the HMAC function confirms the final stage of certifying votes by ensuring authenticity that the vote count was generated by the authorized party or government body and integrity that the vote count remains unchanged (**Figure 6**). No counter is utilized; the sum of the *n* votes is re-encrypted with the public key. Before election day, the secret key would have already been exchanged in a meeting of the board of elections; the board of elections uses this cryptosystem to securely receive and count votes to later release with trust that the votes were not tampered with and remained anonymous. HMAC has great resistance towards cryptanalysis attacks, making it more secure than any other authentication code. HMAC has been made compulsory to implement in IP security, confirming the security and the accuracy of the vote count, verifying both the data integrity and the authentication of the vote.

According to the HMAC keyed-hash message authentication code, where *K* is the secret key, *K'* is the secret key (in the case where the key is larger than the block size, *K'* is a hash of the key, in the case where the key is smaller than the block size it is padded with zeros to equal the block size), *m* is the plaintext vote count, and the *opad/ipad* are two distinct values (0x5c and 0x36) that perform the "exclusive or" (XOR) logical operation with the inner and outer keys respectively, to mitigate against certain attacks on HMAC (**Figure 4C**). This method uses symmetric cryptography and receives, concatenates, and then hashes the vote. In adaptation to the HMAC formula, our proposed election infrastructure hashes a concatenation of three things: the key, a hash of the key, and the plaintext vote count. This allows us to maintain a secure record of the vote count at any point in time, for an election with *n* votes, from when the first vote is received until vote *n* is received.

For a 512-bit vote count, we concatenate with 256 bits of the secret key (assuming a 256-bit secret key length) and with 512 bits of the hashed key, totaling 1280 bits. While the entire input is 1280 bits, the output of the outer hash remains 512 bits because it is a Secure Hash Algorithm (SHA3-512) hashing function. The SHA3-512 algorithm takes in arbitrary

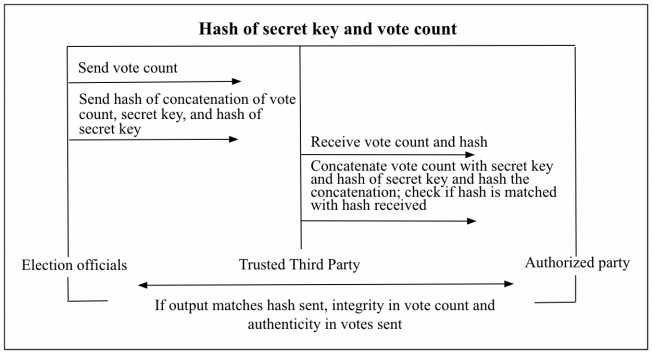

Figure 6: Voting protocol of HMAC Function. This function begins with the HMAC method; data (vote count) and the hash are transmitted securely. Then, the vote count is taken and hashed; the hash is concatenated with the secret key and hashed again. If the output matches the hash that was transmitted, these factors are true.

bit lengths in blocks of 576 bits at a time, resulting in a single 512-bit hash.

Advanced Encryption Standard 512 (AES-512)

For the vote key and transport key, our election infrastructure utilizes the AES-512 algorithm for symmetric cryptography, transforming one block of vote data at a time using the cryptographic keys. Virtually impenetrable using brute-force methods, protection is applied and processed for vote information that was already protected, making it suitable for high security. The plaintext votes and key size organized in bytes remain with the 512-bit input block. Each is processed in multiple rounds throughout the encryption process to result in the same 512-bit length ciphertext. The public and private key cryptography for asymmetric encryption utilize the Elliptic Curve Digital Signature Algorithm (ECDSA). Specified in NIST Federal Information Processing Standards (FIPS) 186, the 512-bit ECDSA is Paillier compatible and maintains verification within the federal government. In compliance with NIST SP 800-186 (13), elliptic curves generate verified key lengths for secure interoperability within a random bit generator for the key pairs.

APPENDIX

Election Infrastructure Source Code: https://github.com/daniellekpark/SecureVoting

ACKNOWLEDGMENTS

The authors would like to thank the New York University Courant Institute of Mathematical Sciences for facilitating contact between them and the New York University Tandon School of Engineering for supporting the continued development of their research.

Received: July 30, 2021
Accepted: January 19, 2022
Published: June 12, 2022

REFERENCES

1. Asplund, Mikael, and Simin Nadjm-Tehrani, eds. *Secure IT Systems: 25th Nordic Conference, NordSec 2020, Virtual Event, November 23-24, 2020, Proceedings*. Vol. 12556. Springer Nature, 2021.
2. Ryan, Peter Y., and Thea Peacock. "A Threat Analysis of Prêt à Voter." *Towards Trustworthy Elections*, 2010, pp. 200–215., doi:10.1007/978-3-642-12980-3_12.
3. Gjøsteen, Kristian. "The Norwegian Internet Voting Protocol." *Lecture Notes in Computer Science*, 2012, pp. 1–18., doi:10.1007/978-3-642-32747-6_1.
4. Heiberg, Sven, and Jan Willemson. "Verifiable Internet Voting in Estonia." *2014 6th International Conference on Electronic Voting: Verifying the Vote (EVOTE)*, 2014, doi:10.1109/evote.2014.7001135.
5. The Heritage Foundation. "Voter Fraud Map: Election Fraud Database." *The Heritage Foundation*, 2022, www.heritage.org/voterfraud.
6. Nan Li, "Research on Diffie-Hellman key exchange protocol," *2010 2nd International Conference on Computer Engineering and Technology*, 2010, pp. V4-634-V4-637, doi: 10.1109/ICCET.2010.5485276.
7. Paillier, Pascal. "Public-Key Cryptosystems Based on Composite Degree Residuosity Classes." *Advances in Cryptology — EUROCRYPT '99*, 1999, pp. 223–238., doi:10.1007/3-540-48910-x_16.
8. "Security and Privacy Controls for Information Systems and Organizations." *NIST Special Publication 800-53* , no. 5, 2020, doi:10.6028/nist.sp.800-53r5.
9. Suwandi, Rifki, *et al*. "Secure e-Voting System by Utilizing Homomorphic Properties of the Encryption Algorithm." *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 16, no. 2, 2018, p. 862., doi:10.12928/telkomnika.v16i2.8420.
10. Saproo, Saksham, *et al*. "Online Voting System Using Homomorphic Encryption." *ITM Web of Conferences*, vol. 32, 2020, p. 03023., doi:10.1051/itmconf/20203203023.
11. Jabbar, Ihsan, and Saad Najim Alsaad. "Design and Implementation of Secure Remote e-Voting System Using Homomorphic Encryption." *Int. J. Netw. Secur.* 19.5 (2017): 694-703.
12. Mell, P M, and T Grance. "The NIST Definition of Cloud Computing." *NIST Special Publications*, Sept. 2011, doi:10.6028/nist.sp.800-145.
13. Regenscheid, Andrew. "Recommendations for Discrete-Logarithm Based Cryptography:" *NIST Special Publications*, 30 Oct. 2019, doi:10.6028/nist.sp.800-186-draft.