Voting Credential Management System for Electronic Voting Privacy

Arijet Sarker, SangHyun Byun, Wenjun Fan, Maria Psarakis, Sang-Yoon Chang
University of Colorado Colorado Springs
{asarker, sbyun, wfan, mpsaraki, schang2}@uccs.edu

***Abstract*—Electronic voting requires voting privacy to protect the voter anonymity. We present a novel design framework for credential management called Voting Credential Management System (VCMS) which preserves voting privacy against advanced attackers who do not only monitor the voting transactions and communications but are also capable of compromising an authority involved in the credential management and generation. Such requirement against the advanced threat model based on an authority compromise is inspired by the recent attacks in voting privacy and is adopted in the state of the art credential management systems. VCMS achieves such properties by building on the well-established cryptographic primitives and by separating the voting token (the VCMS output credential used for the voting) and the intermediate key token (which is used within VCMS and bridges the registration/certificate with the voting token). VCMS is specifically applicable to electronic voting and is simpler than other sophisticated credential management systems achieving comparable security properties.**

***Index Terms*—Electronic Voting, Credential Management, Public-Key Infrastructure, Voting Privacy**

I. Introduction

Electronic voting (e-voting) [10] is increasingly in use because of its potential to reach and involve greater number of voters (e.g., to facilitate voting from abroad) and its time and cost saving features. For example, countries such as Estonia, Brazil, Netherlands, and Norway adopted e-voting for national elections. The voting privacy to protect the voter information against unauthorized attackers is critical. The information needed protection for voting privacy is in two forms: first is to hide the personal information, e.g., used for registration, (*personal-information privacy*) and the second is to anonymize each vote so that the ballot/vote cannot be tracked to its voter (*voting anonymization*). In our paper, we achieve both of these objectives (we briefly discuss about personal-information privacy in Section VI-A), but our novel contribution focuses on the latter voting anonymization.

Unfortunately, privacy incidents occur, such as the attack on the US presidential elections in 2016 [1] and from the Defcon Voting Village [2], where the attacker compromised an authority providing voting services. In view of these attacks, we design a secure credential management system for e-voting, called Voting Credential Management System (VCMS). VCMS has a strong privacy property since it preserves voter anonymization even when the attacker compromises a credential management authority. This threat model is inspired by one of the most advanced PKI system which is being proposed and designed for vehicular networking and adopted by the state of the art e-voting systems (which we discuss in greater details in Section II-2).

Because we defend against such advanced threat, VCMS builds on the traditional PKI framework (e.g., based on a Certificate Authority) and is more complex than the credential management systems trusting the authority participants not to leak or infer the voting privacy information (passive attacks). However, our VCMS design minimizes the complexity and only adds the needed components for e-voting (e.g., VCMS is much simpler than that used in vehicular networking as discussed in Section VI-B).

VCMS provides the voters with voting token credentials, which can then be used to anonymously authenticate the voter ballots to ensure that the voter is a registered voter. Our contribution with VCMS focuses on the generation and delivery of such voting token so that the anonymity (incapable of linking between the ballot and the voter) holds even when an authority in the system is compromised by an adversary attacking the voting anonymization.

II. Related Work

1) Voting Anonymization: This subsection will review the related work in voting anonymization. An e-voting scheme often requires the existence of an anonymous channel to prevent the voter's privacy leakage [15]. There are several methods to be used to protect the voting privacy. A blind signature allows an authority to sign an encrypted message without knowing the message content, although it is vulnerable against time-based information leakage [9]. A mix-net provides an anonymous channel through large amounts of computation for multiple mixers to prove the correctness of their mixing data [7]. A homomorphic encryption based approach ensures the submitted ciphertext containing a encoded vote, whereby the final tallying can be in a publicly verifiable manner [8]. Also, blockchain [11] is incorporated to this research area. These schemes rely on the pre-established keys/credentials while our work in VCMS focuses on establishing the credentials.

2) Certificate Management And Authority Compromise: VCMS constructs the credential management for voting systems and builds on the previous public-key infrastructure (PKI) designs such as that used for the public Internet, e.g., X.509. However, VCMS presents a more complex design than the trust management in Internet because of the application requirement for voting privacy.

VCMS is inspired by Security Credential Management System (SCMS) for vehicular networking [3], [14] in its threat model (authority compromise) and the corresponding design. Similarly to voting, vehicular networking requires *vehicular privacy* which prevents the certificate (being used in networking) to be linked to the vehicle in order to avoid an adversary from tracking the vehicle locations. SCMS presents such design defending not only against the adversary monitoring the vehicular networking but also against a single authority compromise [4], [5]; SCMS therefore involves many independently operating authorities and dynamic, pseudonym-based certificates. SCMS presents one of the most advanced credential management design and is driven by resourceful players in Crash Avoidance Metrics Partnership (CAMP), an industry consortium comprised of the major vehicle companies and government (USDoT and NHTSA).

The state of the art voting systems also adopt comparable threat model assuming the passive attacker compromising an authority compromise to breach the voting privacy, such as the state of the art voting system in Estonia [6] and in New South Wales [12]. While they provide relevant systems design to our work and can inform the use of VCMS, our work with VCMS focuses on the credential management and prioritize simplicity (introducing the components/authorities as needed) for easier analyses.

III. Problem Statement

A. Threat Model

Since our contribution focuses on voting privacy, we consider passive attackers significantly more advanced than just compromising a voter or its interactions with VCMS. In our threat model, the attacker can additionally monitor the transactions/communications between the authorities within VCMS (motivating the security protection of these communications using public-key cryptography) or can compromise an authority involved in the credential generation and management (motivating the separation of the authorities and the introduction of the intermediary Key Token within VCMS, as described in Section V). Our threat model based on authority compromise (which is stronger than just having a communication link or an endhost-voter compromised) is inspired by the attack incidents in voting privacy and by the credential management systems in the advanced e-voting systems [6], [12] and in other emerging networking technologies (e.g., SCMS for vehicular networking). The attacker succeeds in its goal of breaching voting anonymization if the attacker can track which voter generates which vote/ballot.

We also consider active threats to breach the VCMS integrity by the voters (e.g., to generate a new token or to modify a received token) or by the attackers compromising the communications medium within VCMS (e.g., spoofing). However, we assume that the authorities themselves do not launch active threats; such assumption can be realized by implementing accountability control on these authorities (which are pre-established and anchors VCMS), which control is outside of the scope of our contribution. In other words, the authority entities within the VMCS are trusted to correctly execute their roles so that the voting integrity and the system functionality are preserved while the voters and the communication connections between the VCMS nodes are less trusted which can provide sources of misbehavior. Our threat model is comparable with the threat model driving the SCMS design in vehicular networking (SCMS is described in Section II-2).

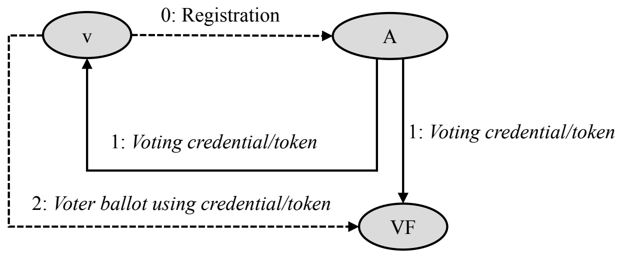

v
A
VF
0: Registration
1: Voting credential/token
1: Voting credential/token
2: Voter ballot using credential/token

Fig. 1: Credential management using one authority (vulnerable if the Authority is compromised).

B. Single Authority Problem

This subsection builds on our threat model in Section III-A and motivates our use of multiple authorities instead of one authority in VCMS system. More specifically, we present the simpler one-authority case, which aligns with the traditional schemes such as [13], and show that the authority breaches the voting anonymization if it is maliciously collecting data and transactions for the voting credential management. As depicted in Figure 1, the authority (A) takes registration information from voter denoted by v (Step: 0) and generates voting credential/token for the voter v. This voting credential/token is used to verify v's eligibility to vote and is sent to both v and Verifier (VF) (Step: 1) by A. The voter v uses (e.g., digitally signs) the Ballot using the voting credential/token and sends it to VF (Step: 2). VF verifies the voting credential/token from A and v. If they match, then VF processes the Ballot. In this case, A receives/processes the voter registration and generates/sends the corresponding voting credential/token directly to v. Because the receiving/processing and generation are not separate, A can breach the voting anonymization of the voter if it collects or stores such information at the time of the voting credential generation. To prevent such threat and defend voting anonymization even when an authority is compromised, we introduce greater complexity and multiple authorities to implement separation in VCMS in Section V; VCMS design also provides a more concrete use cases of cryptographic functions than in this section.

IV. VCMS Setup, Assumptions, and Notations

Before running VCMS to generate the token credential, we assume secure registration (which implementation varies across systems and voting applications) and that the keys for the voters and the authorities are established (which process is typically coupled with registration and involves another

Variable	Description	Variable	Description	Variable	Description
S	Set of Entities	V	Set of Voters	v	Voter
B_v	Ballot of Entity $v \in V$	K_i	Public Key of Entity $i \in S$	k_i	Private Key of Entity $i \in S$
C_v	Digital Certificate of $v \in V$	KT_v	Key Token of $v \in V$	$\widetilde{C}_v$	Info. to Generate Certificate of $v \in V$
VT_v	Voting Token of $v \in V$	h	Hash Function	f	Digital Signature
f^{-1}	Verifying Digital Signature	g	Asymmetric Encryption	g^{-1}	Asymmetric Decryption
s	Symmetric Encryption	z	Payload		

TABLE I: Variables and Notations in VCMS System

authority vouching for and certifying[1] the public keys, as opposed to self-certified keys). These (the registration also often involves physical/non-electronic means) are outside the scope of our contribution as we focus on generating the token credential after establishing registration and keys.

There are n number of voters, and each voter v is represented with indices, $v \in V$ where $V = \{ 1, 2, 3, ..., n \}$ is the set of voters. VCMS additionally involves the authority entities, so the set of the entities involved in VCMS is $S = \{$ RA, CA, KTA, VTA, VF, 1, 2, 3, ..., $n \}$ where $V \subset S$. The authority entities are described in Section V-A when we describe the VCMS architecture, and all of these entities within S are shown as nodes in Figure 2. For any entity i, where $i \in S$, K_i is the public key of the entity i and k_i is the private key of the entity i. For any voter v, where $v \in V$, C_v is the digital certificate for validating voter v for the VCMS token generation, and $\widetilde{C_v}$ is the information required to generate C_v (VCMS protects the confidentiality of $\widetilde{C_v}$ in case it includes personal/private information about the voter, but we further recommend that such information get stripped away by RA as they are not required for the functioning of VCMS.

We rely on public-key cryptography and define f and g as the cipher algorithms: f for digital signature for source-integrity protection and g for encryption for confidentiality protection. In other words, the sender i, for some $i \in S$ applying f_{k_i} corresponds to digitally signing using its private key, k_i and the receiver applying $f^{-1}_{K_i}$ corresponds to verifying the digital signature using the sender i's public key K_i. For encryption (so that only the receiver can correctly decrypt and retrieve the payload message), the sender applying g_{K_j} using the receiver j's public key, K_j, where $j \in S$, corresponds to encrypting and generating the ciphertext, and the receiver j applying $g^{-1}_{k_j}$ corresponds to decrypting the ciphertext and retrieving the payload message. We build on the well-established primitives for public-key cipher algorithms, e.g., RSA can be used for both f and g; digital signature scheme (DSS) can be used for f and ElGamal cipher can be used for g. Such public-key ciphers are known to be secure (anchored by the mathematical assumptions) and satisfy the followings (i.e., the functions are reversible with the correct keys): $f^{-1}_{K_i}(f_{k_i}(z)) = z$ for some payload z if (K_i, k_i) is the public-private key pair of some entity i, and $g^{-1}_{k_i}(g_{K_i}(z)) = z$.

We also denote a symmetric cipher with s, and a cryptographic hash function with h (VCMS requires the hash function's one-way/preimage-resistant property). All of these cipher functions generate (pseudo-)random outputs. Table I lists the variables and notations used in our paper.

Our contribution focuses on the VCMS design. Beyond the design, we assume that there is no information leakage from implementation which can be used for privacy breach, e.g., timing-based or networking-based unintentional leakage.

V. VCMS Design

VCMS presents a design framework for the credential management system for electronic voting.

A. VCMS Architecture and Interactions

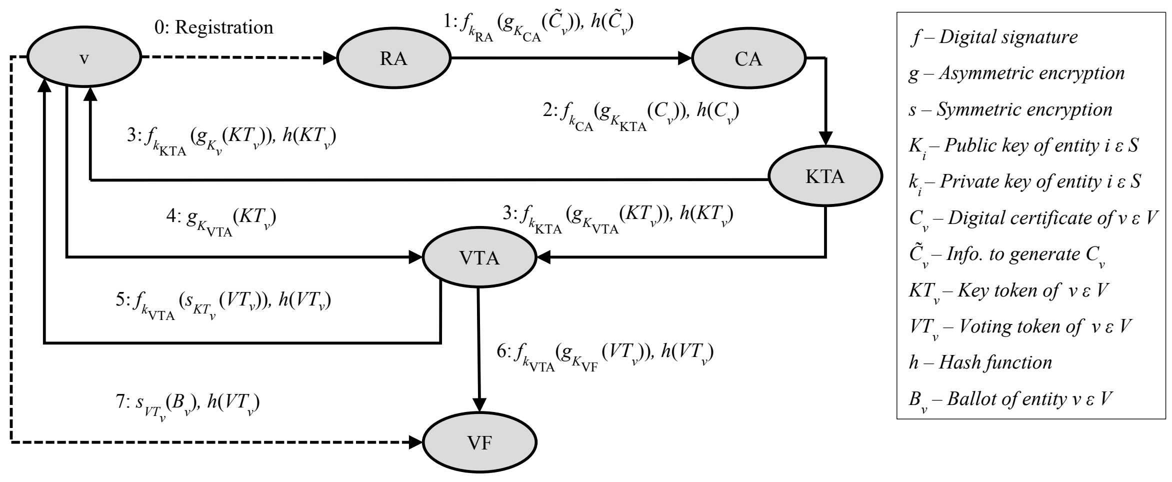
In this section, we describe the VCMS design architecture, which takes the voter registration and produces the voting token (VT) to the voter. To solve the SA problem of linking v to Ballot described in III-B, VCMS has four authorities: Registration Authority (RA), Certificate Authority (CA), Key Token Authority (KTA), Voting Token Authority (VTA). The voter v and the Verifier (VF) interact with VCMS. These authorities are independently operating, so that attacker compromising one authority does not lead to compromising another authority. This section corroborates with Figure 2. Afterward, we explain the VCMS design rationale in Section V-B.

RA: Registration Authority (RA) is responsible for conducting the Registration, including the verification of the eligibility of the voters to vote. RA also generates $\widetilde{C_v}$ for voter v which is unique to v but does not include privacy-sensitive information (e.g., as is in the case of the e-voting system in Estonia).

CA: Certificate Authority (CA) gets $\widetilde{C_v}$ of the v to generate C_v from RA. $\widetilde{C_v}$ includes K_v and v but the other information can be system-specific. RA sends $\widetilde{C_v}$, encrypted by K_{CA} and signed by k_{RA} with $h(\widetilde{C_v})$ (Figure 2, Step: 1). CA then verifies the message is coming from RA using K_{RA} and decrypt, $\widetilde{C_v}$ using k_{CA}. From $\widetilde{C_v}$, CA generates C_v (the certificate of v) including the public key of v, K_v (similarly to the public-key certificate used by X.509).

KTA: Key Token Authority (KTA) gets C_v from CA, encrypted by K_{KTA} and signed by k_{CA} with $h(C_v)$ (Figure 2, Step: 2). KTA verifies the message is coming from CA using K_{CA} and decrypt C_v using k_{KTA}. KTA generates the key token for the v, KT_v from C_v using any hash function.

[1]This certificate is for public key of the entities involved in VCMS (for this process, the entities' private key does not need to be shared with the public-key-certifying authority) while VCMS certificates, which also includes the public key information similarly to the certificates used in other applications, are eventually used for tokens.



v
0: Registration
RA
$1: f_{k_{RA}}(g_{K_{CA}}(\tilde{C}_v)), h(\tilde{C}_v)$
CA
$2: f_{k_{CA}}(g_{K_{KTA}}(C_v)), h(C_v)$
KTA
$3: f_{k_{KTA}}(g_{K_v}(KT_v)), h(KT_v)$
$4: g_{K_{VTA}}(KT_v)$
$3: f_{k_{KTA}}(g_{K_{VTA}}(KT_v)), h(KT_v)$
VTA
$5: f_{k_{VTA}}(s_{KT_v}(VT_v)), h(VT_v)$
$6: f_{k_{VTA}}(g_{K_{VF}}(VT_v)), h(VT_v)$
$7: s_{VT_v}(B_v), h(VT_v)$
VF
f – Digital signature
g – Asymmetric encryption
s – Symmetric encryption
K_i – Public key of entity i ε S
k_i – Private key of entity i ε S
C_v – Digital certificate of v ε V
$\tilde{C}_v$ – Info. to generate C_v
KT_v – Key token of v ε V
VT_v – Voting token of v ε V
h – Hash function
B_v – Ballot of entity v ε V

Fig. 2: VCMS System Architecture for generating token for voter v. Our contribution lies in between registration and casting the ballot in the overall voting process. The solid arrow lines represent our contribution while the dotted arrow lines represent secure registration (which we assume) and the actual use of the voting token for casting the ballot; the dotted arrow lines/transactions can occur offline/separately in time from VCMS VT_v generation and distribution.

The input of the hash function is C_v. KTA sends the key token KT_v to the voter v using v's public key for encryption g. Only v is able to decrypt KT_v. KTA sends KT_v, encrypted by K_v signed by k_{KTA} with $h(KT_v)$ (Figure 2, Step: 3). v verifies the message is coming from KTA using K_{KTA} and decrypt KT_v using k_v.

VTA: Voting Token Authority (VTA) gets KT_v from KTA, encrypted by K_{VTA} and signed by k_{KTA} with $h(KT_v)$ (Figure 2, Step: 3). VTA verifies that the message is coming from KTA using K_{KTA} and decrypt KT_v using k_{VTA}. v will request for voting token, VT_v of v by sending KT_v to the VTA, encrypted by K_{VTA} (Figure 2, Step: 4). VTA checks KT_v from v with the KT_v from KTA. If it matches, then VTA sends VT_v to v. The checking process ensures that if any malicious attacker sends any unauthorized Key Token, then VTA does not send any voting token to that attacker. VTA sends the voting token without linking it to a particular v. Only v will be able to decrypt the voting token, VT_v. VTA will send VT_v, encrypted by respective KT_v and signed by k_{VTA} with $h(VT_v)$ (Figure 2, Step: 5). v will be able to verify the message is coming from VTA using K_{VTA} and decrypt VT_v using KT_v.

VF: Verifier (VF) will receive two inputs- i) one from v and ii) another from VTA. VTA will send $h(VT_v)$ and VT_v, encrypted by K_{VF} and signed by k_{VTA} to VF (Figure 2, Step: 6). VF will verify $h(VT_v)$ from the two inputs. v will send B_v signed by VT_v and $h(VT_v)$ to VF (Figure 2, Step: 7). Only if they match, VF further processes the ballot, B_v, e.g., for counting votes.

B. Design Rationale

Our design is anchored by our setup in Section IV, e.g., applying f using the sender's private key is for digital signature to protect source and message integrity (since only the sender holds the private key, no other entities can generate such message) and applying g using the receiver's public key is for encryption for confidentiality (since only the receiver holds the corresponding private key, no other entities can decrypt the ciphertext). The hash function h provides protection on message integrity even when there is no structure in the payload and enables efficient verification/matching. All the transactions/communications in Figure 2 also have causal relations in steps, e.g., Step: 0 (Registration) needs to occur before Step: 1 (RA's transmissions).

The VCMS transactions are protected in source integrity, message integrity, and in confidentiality against our threat model in Section III-A. We achieve these objectives via the use of f for digital signature, g for encryption, and h for message integrity protection (so that the receiver can compute its own hash and see if they match, which assures that the message has not been modified by an adversary enroute). There are two exceptions in VCMS transactions to these use of techniques (involving f, g, and h). First exception is Step: 4 when the voter v send the KT_v to VTA, in which case the source integrity is protected implicitly because of the causal relations from Step: 3 (i.e., only the voter v could have correctly retrieved the KT_v in Step: 3); the hash output is also omitted since VTA already received that from Step: 3 from KTA, which increases security by enforcing the VTA to collaborate with KTA to receive the required $h(KT_v)$. The second exception is the use of Step: 7 from the voter v to VF which delivers the actual ballot B; in this step, the voter v uses symmetric encryption, using VT_v as the key, instead of using f for a digital signature so that an attacker monitoring the transaction cannot identify which voter it is.

We mainly use public-key cipher because of the strong security properties of the current algorithms supporting public-key cryptography. VCMS can afford the relatively high computational overhead (compared to the symmetric ciphers) because VCMS is for credential generation and requires only one VCMS execution per voter (in contrast to other more frequent networking/cryptography applications, such as using the credentials for data communications in networking).

VI. VCMS Analysis

A. Privacy Analysis

Voting privacy can be in two forms, as defined in Section I: personal-information privacy and anonymity. As for personal-information privacy, we rely on secure registration and secure RA, as RA is directly involved in taking the voter's personal information and verifying them for voting eligibility. However, beyond RA, VCMS does not deal with personal information since RA omits privacy-sensitive information for its output, $\widetilde{C}$. (Voting anonymity, in contrast, is preserved even when RA is compromised and attempts to link the ballot to the voter.)

For anonymity, VCMS provides a sophisticated design with great degrees of separation using both independently- and autonomously-operating authorities and the intermediate credentials ($\widetilde{C}$, C, and KT) in order to break the linkability between the ballot and the voter identity. VCMS addresses the threat model in Section III-A and preserves anonymity even in the case of a single authority compromise. Breaking VCMS requires the compromise of multiple authorities, which is difficult to realize for the attacker (SCMS described in Section II-2 also deems the feasibility/risk of multiple-authority compromise to be significantly lower and therefore focuses more on the single-authority compromise). For example, an attacker compromising KTA (linking C_v - including K_v - and KT_v), VTA (link KT_v and VT_v), and either the VF or the link for the ballot (linking B_v and KT_v) breaches the voting anonymization (linking B_v and K_v).

B. VCMS Simpler than SCMS

The state of the art SCMS (described in Section II-2) provides an inspiration and a baseline for our VCMS design. Therefore, we include analyses in this section comparing VCMS and SCMS. VCMS can become significantly less complex than SCMS because of the credential requirements for the voting applications. This is because the VCMS output, the voting token, is used only once (which is why VCMS produces tokens as opposed to other longer-lasting certificates) and the relative scarcity of the voting event enables the overhead of running VCMS for every voting event. In contrast, the certificates generatad by SCMS are for multiple usage (to be used regularly during the car vehicle operations) and thus the certificates are pseudonym-based and dynamically used, triggering greater complexity in the SCMS design to support such properties. While SCMS can be adapted for voting purpose, we design VCMS for simplicity and only focus on the security objectives needed for voting; voting does not require pseudonyms and dynamic and multiple-use certificates, which enables a simpler design, e.g., VCMS has four authorities compared to SCMS having eight authorities [3].

VII. Conclusion

We present Voting Credential Management System (VCMS) which preserves voting privacy even against an adversary capable of compromising an authority (which attack vector has been demonstrated to be effective against voting privacy in real world). VCMS presents a sophisticated design framework relying on separations in both the authorities and the credentials (introducing an intermediary Key Token to produce the final output of voting token). VCMS is tailored to the e-voting applications and minimizes complexity while achieving voting anonymization.

Acknowledgment

This material is based upon work supported by the National Science Foundation under Grant No. 1922410. This research is also supported in part by Colorado State Bill 18-086.

References

[1] A. L. Abba, M. Awad, Z. Al-Qudah, and A. H. Jallad, "Security analysis of current voting systems," in *2017 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*. IEEE, 2017, pp. 1–6.

[2] M. Blaze, J. Braun, H. Hursti, D. Jefferson, M. MacAlpine, and J. Moss, "Defcon 25 voting machine hacking village: Report on cyber vulnerabilities in us election equipment," *Databases, and Infrastructure*, 2017.

[3] B. Brecht, D. Therriault, A. Weimerskirch, W. Whyte, V. Kumar, T. Hehn, and R. Goudy, "A security credential management system for v2x communications," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 12, pp. 3850–3871, 2018.

[4] J. Brorsson, P. S. Wagner, and M. Hell, "Guarding the guards: Accountable authorities in vanets," in *2018 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2018, pp. 1–4.

[5] C.-W. Chen, S.-Y. Chang, Y.-C. Hu, and Y.-W. Chen, "Protecting vehicular networks privacy in the presence of a single adversarial authority," in *2017 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2017, pp. 1–9.

[6] D. Clarke and T. Martens, "E-voting in estonia," *Real-World Electronic Voting: Design, Analysis and Deployment*, pp. 129–141, 2016.

[7] S. Guasch and P. Morillo, "How to challenge and cast your e-vote," in *International Conference on Financial Cryptography and Data Security*. Springer, 2016, pp. 130–145.

[8] A. Kiayias and M. Yung, "Tree-homomorphic encryption and scalable hierarchical secret-ballot elections," in *Financial Cryptography and Data Security*, R. Sion, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 257–271.

[9] K. Kim, J. Kim, B. Lee, and G. Ahn, "Experimental design of worldwide internet voting system using pki," 2001.

[10] T. Kohno, A. Stubblefield, A. D. Rubin, and D. S. Wallach, "Analysis of an electronic voting system," in *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*. IEEE, 2004, pp. 27–40.

[11] P. McCorry, S. F. Shahandashti, and F. Hao, "A smart contract for boardroom voting with maximum voter privacy," in *International Conference on Financial Cryptography and Data Security*. Springer, 2017, pp. 357–375.

[12] NSW Electoral Commission, "iVote® refresh project for the 2019 NSW State election," https://www.elections.nsw.gov.au/NSWEC/media/NSWEC/Reports/iVote%20reports/iVote-Refresh.pdf, 2019, [Online; accessed 16-April-2020].

[13] D. Springall, T. Finkenauer, Z. Durumeric, J. Kitcat, H. Hursti, M. MacAlpine, and J. A. Halderman, "Security analysis of the estonian internet voting system," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014, pp. 703–715.

[14] W. Whyte, A. Weimerskirch, V. Kumar, and T. Hehn, "A security credential management system for v2v communications," in *2013 IEEE Vehicular Networking Conference*. IEEE, 2013, pp. 1–8.

[15] B. Zhang and H.-S. Zhou, "Statement voting," *Financial Cryptography and Data Security 2019*, 2019.