Invited Paper: Towards a Robust Distributed Framework for Election-Day Voter Check-in[⋆]

Alexander A. Schwarzmann[1]

Augusta University, Georgia, USA, email aschwarzmann@augusta.edu

Abstract. Electronic poll books are computerized distributed systems that replace paper-based voter lists used to enable eligible voters to cast their ballots on the Election Day. These systems have the potential for speeding up voter check-in at the polling place, and making voter records more accurate by reducing human errors in dealing with printed voter lists and post-election transcription. At the same time, electronic poll books are non-trivial distributed computing systems, and ensuring correctness, security, integrity, fault-tolerance, and performance of such systems is a challenging problem. In fact we are not aware of a single commercially available system that does not contain major deficiencies and risk factors. This paper focuses on the distributed system aspects of electronic poll book solutions and identifies the obstacles that are inherent in any distributed system that must deal with failure and asynchrony while providing a consistent and dependable service. We review several requirements that need to be satisfied by electronic poll book systems, we discuss selected important results from distributed computing research that the commercial developers of electronic poll book systems appear to not be aware of. We then present a wider landscape, including social and political science aspects, we survey broader research issues, and discuss system implementation considerations. This paper brings for the first time to the attention of the research community an in-depth presentation of an important new problem of immediate relevance. Moreover, the electronic poll book technology is an attractive application domain for the research results in dependable and secure distributed computing.

1 Introduction

Electronic voting systems are an integral component of the modern electoral procedures and an essential part of any democratic society. Such systems are composed of several entities working in concert: e.g., "Voter Registration Systems", "Election Management Systems", "Voting Terminals", and "Electronic Poll Books". The first of these has been computerized for some

[⋆] Supported in part by NSF Award 2131538, with the senior personnel A. Aleroud, C. Busch, R. DeFrancisco, D. Kowalski, G. Murray, N. Panwar, R. Rahaeimehr, A. Schwarzmann, E. Tremel, J. Heslen, C. Albert (Augusta Univ.), and Sh. Dolev (Ben Gurion Univ.). This paper also cites results obtained in collaboration with L. Michel, A. Russell, and M. Desmarais (Univ. of Connecticut).

time using well-understood database technologies. In the United States, the HAVA act of 2002 [39] mandated a nationwide modernization of the voting infrastructure and led to the broad adoption of computerized solutions for the next two components. Consequently, all 50 States initiated efforts to select digital voting solutions. The private sector rushed to market with hastily created and often inadequately engineered solutions. The result was an adoption of products that suffered from severe flaws: poor engineering, lack of resilience against the most elementary tampering or simple mis-configuration, the illusion of safety from misuse of cryptography, and, in general, an under-appreciation for the complexity of the electronic election systems that need to be built as sophisticated and secure distributed systems. The third component, Electronic Poll Books, provides the key function that ensures that all eligible voters can cast their vote, and cast it at most once, thus the imperative of "one voter, one vote." These systems are the focus of this presentation. They are relatively new, and they are being widely adopted to replace the old manual processes. The broad concerns surrounding the integrity and security of electronic election systems used in the 2016 and 2020 Presidential Elections underscore the need for a rigorous scientific approach to designing and implementing all such systems.

Contributions. This paper brings for the first time to the attention of the research community an in-depth view of an important new multidimensional problem of immediate relevance. The electronic poll book technology is an attractive and worthy application domain for synergistic research encompassing distributed computing, fault tolerance, cryptography and authentication, networking and network security, consisted distributed storage, and human factors. The research directions suggested here have the promise of high impact in an important area of political elections that are so essential for any democratic society. Unlike some research advances whose applicability to solving real life problems may need to wait for many years before having impact, in the space of electronic check-in systems we have a rewarding situation where research advances can directly benefit a crucial aspect of our increasingly digital civilization, viz. the integrity and security of the electronic electoral process.

On the Broad State of Technology. Dislocation of theory and practice in computing can lead to significant (and even sensational) problems with basic computing infrastructure, and indeed, a number of striking examples arises in the context of voting technology. Numerous technical reports have documented serious issues with electronic election systems, Electronic election systems [7,40,43,44] are perceived to be inherently complex from a security and verification standpoint. Our own work had exposed a number of vulnerabilities in the design and implementation of electronic voting systems, in their use of cryptography, their logging capabilities and communication software that an attacker could exploit to alter the result of an election, e.g., [13,14,27,29,30,37,45]. Technical issues have been reported by essentially all researchers and evaluators who have conducted independent assessment of security and integrity of electronic election systems, ranging from the well-known problems with the use of paperless DRE systems (direct recording, electronic) and the security vulnerabilities in all examined stand-alone systems to severe problems in Internet voting systems [48]. In addition to raising concerns about the dependability of electronic voting systems and suitability of adopted solutions, the ensuing reexamination led a number of states to abandon their chosen solutions and switch technology altogether at a remarkable cost to the U.S. taxpayer [42], e.g., in Maryland [18], where the decision was made to migrate to Optical Scan technology with voter-verified paper ballots, which emerged as the safest option due to its reliance on voter generated paper audit trail.

Whereas researchers exposed a number of vulnerabilities in the design, implementation, use of cryptography, logging capabilities, and communication software of voting systems that an attacker could exploit to alter the result of an election, it is not surprising that the bulk of these problems can be traced back to the divergence from, and the ignorance of, established results in algorithmics, verification, validation, cryptography, and sound engineering practices.

In this work we focus on the problem of voter check-in in elections, the area that received surprisingly little attention from the community of researchers working on electronic election systems. While substantial research has been dedicated to electronic tabulation of votes, ballot processing, vote aggregation, and audits, the very first step that enables voters to cast their votes on the election day and that ensures the "one voter, one vote" imperative has not been the target of sufficient research to our knowledge.

Document Structure. In Section 2 we describe the e-pollbook landscape in the US. In Section 3 we present the high-level requirements that address the distributed nature of e-pollbooks as replacement for the manual process, and provide specific examples of jurisdictional requirements. In Section 4 we presented selected distributed systems facts that make it challenging to implement e-pollbooks. In Section 5 we present a wide landscape that encompasses social and political science dimension, broader research needs, and implementation considerations. We conclude in Section 6.

2 Electronic Poll Books

It is interesting that while the premature deployment of immature technology resulted in numerous documented cases of serious problems with voting terminals and election management systems, another component of the electoral process—the use of registered voter lists to allow voters to cast votes—still broadly relies on perilously inadequate manual solutions using paper pollbooks, i.e., printed voter lists. The use of a questionable manual process can cause poll opening delays, long lines, and errors, possibly disenfranchising numerous voters [5]. According to the National Conference of State Legislatures, the majority of jurisdictions nationwide are using the manual process, however the usage of electronic solutions is growing rapidly. In the 2020 Presidential Elections, already millions of voters were checked-in electronically. The emerging computerized solutions, "electronic poll books", are a key component of an electronic election system whose basic purpose is to ensure "One voter, one vote." Electronic pollbooks, or "e-pollbooks", are available from a number of vendors, and while the integrity and security of these systems is paramount to the entire electoral process, vendors are again rushing to produce e-pollbooks based on naive premises and software fraught with deficiencies (cf. patent [26]). Naturally, the integrity and security of this component is paramount to the entire electoral process. Without the scientific approach, the desire to modernize and ease the administrative process can again lead to the premature adoption of severely deficient solutions. Indeed, there were a number of reports in the press on the failures of e-pollbook systems during the 2020 US Presidential Election.

It is noteworthy that while the electronic tabulators used on the election day are stand-alone systems, an e-pollbook system is conceptually more complex because it is inherently a dynamic distributed system where multiple check-in devices must operate in concert in providing "one voter, one vote" guarantee, with security and integrity, and despite possible failures and the resulting need to dynamically reconfigure this distributed system on the fly.

We have worked for over a decade with several jurisdictions in New England and New York on security and integrity of electronic election systems. In particular, we provided substantial contributions to the definition of e-pollbook requirements in the States of Connecticut [36] and New Hampshire [20]. We have also participated in evaluating several e-pollbook systems, and while non-disclosure and legal considerations prevent is from identifying specific systems, we must report that the state of the technology in e-pollbook implementations leaves much to be desired. To start with, some systems fail to synchronize properly, leaving a non-trivial window of vulnerability that allows a speedy and mischievous vote to cast more than one ballot. Also, despite the claims of reconfigurability, it is generally impossible to replace crashed devices, and it is not easy to introduce new devices to deal with long lines. Typically, reconfiguration is only possible offline, and even then the systems do not implemented controls to identify and authenticate configured devices adequately. Other systems do not secure wireless communications adequately, allowing nefarious actors to eavesdrop and even launch an impersonation attacks. In one system the network traffic is authenticated and encrypted, yet the keys are self-signed by a third party using a personal email address (@gmail.com). (Here we also note that for practical reasons wired connection are almost never used, thus keeping the digital doors open for attempted attacks, including denial of service, which can be trivially accomplished by inexpensive wireless jamming devices.) Some system implementations also insist on relying on remote servers, where communication over the Internet on the election days creates the potential for obvious problems. Existing implementations do not perform sufficient input checks, e.g., when scanning a barcode on an identity card, and some systems can be crashed with the help of a maliciously designed barcode, or even allow injection attacks that can disenfranchise voters.

In 2017, to address the challenges associated with the use of technology in the elections, the National Academies of Sciences, Engineering, and Medicine (NASEM) appointed an ad-hoc committee, the Committee on the Future of Voting: Accessible, Reliable, Verifiable Technology. The committee held several meetings, where we have also reported on the topic of security challenges, concerns, and increasing vulnerabilities associated with e-pollbooks [2]. NASEM published an extensive committee report "Securing the Vote: Protecting American Democracy" in 2018 [2]. The report summarizes its findings regarding e-pollbooks as follows.

> "Eligible voters may be denied the opportunity to vote a regular ballot if pollbooks are inaccurate. Internet access to e-pollbooks increases the risks associated with the use of e-pollbooks to manage elections. Cyberattacks can alter the voter registration databases used to generate and update pollbooks. If pollbooks are altered by external actors, eligible citizens might, on election days, be denied the right to vote or ineligible individuals might be permitted to vote. Cyberattacks could also compromise the record of who actually voted on Election Dayor disrupt an election in numerous other ways. If an e-pollbook is connected to a remote voter registration database and there is no offline backup, a denial-of-service cyberattack could force voting to be halted."

Among its recommendations, the report states that "Congress should authorize and fund" the development of "security standards and verification and validation protocols for electronic pollbooks." Our own work in this area is funded by a NSF award within the Secure and Trustworthy Cyberspace (SaTC) program.

The distributed computing community is well positioned to address the challenges inherent in e-pollbook solutions. Indeed, there are a number of approaches to the underlying problem that revolves around a collection of computing devices reliably and securely maintaining distributed and replicated databases (i.e., voter lists) in the face of equipment failures. Equally importantly, the research in distributed computing identified a number of problems that are notoriously difficult to solve or that even cannot be solved in the most general setting. The lack of adoption of existing techniques and certain obliviousness of the non-trivial challenges in providing solutions could be attributed to a lack of awareness from practitioners or, equally likely, to the difficulty of specializing and implementing the known approaches in this application domain.

The main part of this paper focuses specifically on the distributed system aspects of e-pollbook solutions. We also include a broader discussion of an approach to researching and developing such systems that in our opinion needs to include a sound computer science foundation, including the cybersecurity dimension, advanced system development, and also a social and political dimension that reflects the societal and legal aspects.

3 Electronic Pollbooks as a Distributed System

In this section we describe the basic setting for poll books, the technological challenges and specific questions regarding e-pollbook implementations.

3.1 The Manual Process. Consider the objectives of officials running an election at the scale of a precinct. Prior to the election, officials are accumulating in a database the collection of registered voters throughout the precinct. On Election Day, a paper listing is printed for each polling station indicating which voters are expected. If electoral procedures permit voters to register "on site", the official must also record individuals who desire to vote but are not on the voter list. As ballots are issued to eligible voters, their names get crossed off. Additionally, in some jurisdictions, absentee voters who were previously issued a ballot must be accommodated if they decide to vote in person. The process is meant to help achieve "one voter, one vote". Naturally, a rogue voter can go from polling station to polling station and attempt to vote several times in this way. In such a case, since the authorities collect voter credentials for on-site voting, they would have a legal recourse. To deal with high voter turn-out, the voter lists are partitioned either by street addresses or names, thus allowing for multiple check-in lines and increased "throughput" at a polling place. Multiple voting is impossible because a voter's name appears in only one partition of the voter list. Clearly absent are failures due to technology. From the procedural and safety standpoints, no further significant weaknesses exist with a paper solution. Yet, a paper process is slow, prone to human error, and work intensive before, during and after the election as the voter information must be collated and re-encoded in the voter database. High voter turn-out still results in long lines because the partition of the voter lists cannot be balanced dynamically. An electronic solution is appealing to streamline the process, but it also creates many difficulties.

3.2 Distribution and Consistency: Immediate Challenge. Shared storage services are at the core of most information-age systems and e-pollbooks are not an exception. Imagine an implementation that is based on a central server storage system. The server accepts requests from check-in devices to perform operations on its data objects (e.g., voter records) and returns responses. While this is conceptually simple, this approach already presents two major problems: (1) the central server is a performance bottleneck, and (2) the server is a single point of failure. The quality of service in such an implementation may degrade as the number of users grows, and the service becomes unavailable if the server crashes. Thus the system must, first of all, be available. This means that it must provide its services despite failures within the scope of its specification. In particular, the system must be able to mask device and communication failures up to a limit. The system must also support multiple concurrent accesses without imposing unreasonable degradation in performance. The only way to guarantee availability is through redundancy, that is, to use multiple check-in devices and servers and to replicate the data among these devices.

It is also critically important to ensure data longevity. A storage system may be able to tolerate failures of some servers, but over a long period it is conceivable that all servers may need to be replaced, because no servers are infallible. Additionally, it may be necessary to provide migration of data from one collection of servers to another as the needs dictate. The storage system must provide seamless runtime migration of data: one cannot stop the world and reconfigure the system in response to failures and changing environment. A major problem that comes with replication is consistency. How does the system find the latest voter record if the data is replicated? What the users (people and high level system components) should expect to see is the illusion of a single-copy object that serializes all accesses so that each operation that reads the object returns the value of the preceding write or update operation, and that this value is at least as recent as that returned by any preceding operation [23,38]. This notion of consistency is formalized as *atomicity* [32] or, equivalently, as *linearizability* [25], and we aim to provide this, most desirable, notion of consistency.

3.3 Specific Technical Questions. Here we present some immediate questions that any e-pollbook solution must address as a distributed system. These questions pose non-trivial challenges in implementing functional and dependable solutions. We primarily focus on the dynamic distributed system view of the required solutions, and in Section 5 we deal with a broader landscape.

1. Multiple front-end devices are needed to allow concurrent check-in. How does one ensure that device failures do not impact system functionality? How are the misbehaving devices removed from the system? How are new devices introduced to replace faulty devices and to deal with high voter turnout?
2. If electronic devices can fail, this raises questions about the status of voter information accumulated in each device. How is the information recovered? Passed on to other devices? Replicated in real-time? How does one transfer on the fly the relevant state to a spare device?
3. Where is the information (voter lists) held? On each device? On a server nearby on a LAN? In the (local) cloud for more reliability? Depending on the answer, one must question what to do in case of network failure. What to do if the network connectivity is lost? Experiences delays? How does one deal with transient (or prolonged or even permanent) network partitions?
4. Any dynamic replication involves some protocol that may itself be subjected to perturbations, e.g., denial of service attacks. How does one guarantee the legitimacy of messages exchanged among the participating devices?
5. When multiple devices add voters or check-off voters upon handing out ballots, how does one maintain a coherent view of the world guaranteeing consistency and enforcing the rule of voting at most once?
6. Last, but not least is the question of functionality of the system with respect to the voters and the election officials that the system is interacting with. Does the system comply with the law? Does it favor one type of voter and might it disenfranchise another? Is it is easy to use? Does it allow for all acceptable forms of identification to be used to check in a voter? Does it allow a manual override by a qualified election official in case of a problem?

Given the generally non-complimentary state of the extant electronic election systems, there is the real concern that existing and emerging commercial offerings for electronic poll-books might side-step the majority of these questions leaving jurisdictions with brittle systems that suffer from major shortcomings and that perform adequately only in benign and friendly environments. In the next section we present some existing requirements for e-pollbooks systems. We note that documenting such requirements is quite challenging. On one hand they must be readable by people without a technical background, e.g., citizens and election officials, on the other hand they must specific requirements that are most useful for technologists who will perform the needed research and development. Thus it is likely that the requirements officially adopted by jurisdictions are viewed as incomplete by the technologists.

3.4 Requirements for Electronic Poll Books as Distributed Systems. Some of the more comprehensive requirements for e-pollbook systems have been published by the States of Connecticut [36] and New Hampshire [20].. We extract and present several requirements from [36] that specifically address the necessarily distributed nature of e-pollbook solutions. We begin by stating several definitions for the terms used in the requirements.

Electronic poll book system (EPBS) – A collection of hardware and software including at least one *electronic poll book* and aiming to implement e-pollbook functionality that satisfies the requirements (in [36]).

Electronic poll book (EPB) – A component of the *electronic poll book system* that includes a user interface device and that is to be used by a poll worker to view and update voter registration records.

EPB system configuration (EPBSC) – A physical instance of an EPBS with all its components configured for use. An EPBS *configuration* consists of peripherals (e.g., printers, scanners, etc.) and a set of configured, networked EPBs. An EPBSC may contain auxiliary servers.

Voter record – The *voter registration record* and *voter activity record* of a voter.

Local voter database A collection of all *voter records* specific to a jurisdiction. The initial state of the *local voter database* is compiled and certified by the relevant authority. Poll workers make updates to the *local voter database* throughout the election by using the EPBS to reflect ongoing voter activity.

Voter list – A printable, exportable, and human-readable representation of the *local voter database.*

Completed update – An update to a *voter registration record* is *completed* if a query for said *voter registration record* on any active EPB within the EPBS returns the same data.

Quiescent – The EPBS is *quiescent* if all user-initiated updates have completed at all *electronic poll books.*

Reconfigure EPBSC – Configuring, adding, or removing any of the *electronic poll book configuration's* peripherals, *electronic poll books*, or auxiliary servers.

Requirements Relevant to the Distributed Nature of the System. We now present the most relevant requirements [36] that specifically deal with the distributed nature of any comprehensive e-pollbook solution. Broadly speaking, the requirements are formulated to ensure the following.

– Fault-tolerance: The system must not contain a single point of failure, and failures (up to a design limit) must not prevent the system from operating. Main types of failures are the failures of the physical system components or the software in these components, and communication failures.
– Service availability: The system must be able to provide the required service in the face of adversity and perturbations (again, up to its design limits).

– Data consistency: The data contained within the system (e.g., voter records) must be viewed consistently following any changes to the data.
– Data survivability: No data may be lost if certain components of the system fail (up to its design limit).
– System reconfigurability: The system must enable faulty components to be removed/replaced without requiring halting or restarting the overall system.

We now state the most relevant requirements [36] in an abridged form.

AR-2 : No single point of failure: The EPBS must be designed to tolerate any single point of failure scenarios.

AR-1 : At least three EPBs in an EPBS: An EPBS must support at least three (3) electronic poll books in a single polling location. Each of the electronic poll books must be usable concurrently. Should one of the electronic poll books become inoperable, the operation of the remaining electronic poll books must not be affected.

FR-1 : Adding a new EPB to the EPBS: The EPBS must provide means for the integration of an additional EPB into its configuration at any point throughout the election without requiring a shutdown or a restart.

FR-2 : Removing an EPB from the EPBS: The EPBS must provide means for the exclusion of an EPB from its configuration at any point throughout the election without requiring a shutdown, or restart of the EPBS. This action does not require physical access to the EPB that is to be excluded.

FR-21 : One voter/one vote within EPBS: The EPBS must guarantee that within an EPBS configuration a voter can be checked in at most once during normal connectivity.

RR-1.1 : Voter check-in during interruption of connectivity: In the event of a temporary interruption of connectivity within an EPBS, the EPBS must permit a voter to check-in.

RR-1.2 : Upon restoration of connectivity: In the event of a temporary interruption of connectivity within an EPBS, the EPBS must automatically restore voter list consistency across the EPBs after connectivity is restored.

RR-1.3 : Identify double voting: In the event of a temporary interruption of connectivity within an EPBS, the system must identify voters that have been checked in more than once during the interruption of connectivity.

RR-5 : Local voter database replicas: Within the EPBS there must exist at least two replicas (logical or physical) of the local voter database. These replicas must be stored in distinct physical storage components. (Note: together with AR-1 and RR-1.1 the number of replicas may need to be higher.)

RR-6 : Local voter database replica consistency: If the EPBS is in a quiescent state all replicas of the local voter database must be logically consistent.

RR-7 : Operational consistency: Any update to a voter record or to any other data pertaining to the election completed on one EPB must be seen as complete on all other EPBs.

These requirements are quite intuitive, and we consider them necessary for any implementation of an e-pollbook system. Next we identify several results that make it challenging to satisfy these requirements.

4 E-Pollbooks and the Distributed Systems Theory

We cite results from the distributed systems theory that stress the need for careful design in developing e-pollbook solutions. *A well-informed reader will be familiar with some if not most of these results. Some of these results are quite venerable, and they have been known for some time. This is why it is particularly surprising and troubling that all existing commercial e-pollbook systems appear to be oblivious of these results, and some claims regarding the capabilities of these systems are in conflict with the known facts.*

At first glance, the facts we cite here cast a pessimistic view on the ability to build dependable systems that coordinate their activities in non-trivial ways or that maintain replicated shared data with guaranteed consistency (e.g., if a data object is changed, then the following read of the object value must reflect the change). However, this does not mean that one cannot build reliable and usable e-pollbook systems. In order to succeed, one needs to understand the theoretical limitations and to make sensible assumptions about the nature of failures, communication, and asynchrony. The main point here is that any claims about a system that provides a solution that is able to deal with the requisite adversity and perturbations in the computation and networking medium, but that are not aware of these known results, is to be suspect.

In what follows we do not cross-reference the requirements from Section 3, but, as indicated earlier, the selected set of requirements deals collectively with the issues of fault-tolerance and availability (AR-1, AR-2), communication (RR-1.1, RR-1.2), agreement and consistency (FR-21, RR-1.3, RR-6, RR-7), and survivability (FR-1, FR-2, RR-5) of the shared data. (See [36] for details.)

Here we focus on the negative (impossibility) research results. Although specialized practical solutions exist for certain modified versions of the problems given here, we do not present them: not only the solution space is very large, but more importantly, it is the duty of responsible system designers to investigate relevant solutions when they will have started gaining the necessary insight.

4.1 Consistent Data Store with Device Crashes. Any e-pollbook system must be able to tolerate benign failures of individual devices, specifically, a failure where the faulty device stops at an arbitrary instant of time and does not perform any further actions. Such benign failures are known as crashes. E.g., a polling place may have several devices used to check in voters. A crash of such a device must not prevent other working devices from functioning, and the crash must not destroy the consistency of the shared data maintained by the system (of course the data must be replicated for survivability). E.g., if a voter was successfully checked in, then all operating devices must agree that this is the case, regardless of a crash. If this cannot be guaranteed, then an ill-motivated voter may attempt to vote more than once. Suppose the type of data we are interested in is consistent read/write data.[1] This is a basic data type, much simpler than data types that support more complicated read-modify-write

[1] Recall that here we are interested in an implementation of a data object that is consistent, i.e., atomic or linearizable, if the users that access the object are presented

operations. It turns out that any system of devices implementing such objects can tolerate the crashes of only a minority of the devices, e.g., [6,11]. *A system of N processors cannot implement a consistent read/write object where all object access operations terminate (complete) in the presence of F crashes if $N \leq 2F$.*

This means that to tolerate a single crash, three replicas are needed. To tolerate two crashes requires replication at five devices, etc. Any poll book solution that replicates its data in two locations and that claims to tolerate a single runtime crash cannot possibly be correct.

4.2 Coordinated Action with Link Failures. Given that multiple devices are necessary in any e-pollbook system, they must must provide their service consistently. Suppose several devices (say, individual poll book devices) need to agree on a common course of action, e.g., by deciding on a value that indicates what action to take. This is known as the agreement problem, and the correctness conditions for a solution are as follows: *(a) Agreement*: no two devices agree on different values, *(b) Validity*: if all devices propose the same value, then this is the only possible agreement value, *(c) Termination*: all non-faulty devices eventually decide. Now suppose that there are just two devices that never fail, but communication can be unreliable, e.g., messages can be lost because of failures or interference. If this is the case, one of the oldest results in distributed computing tells us that there is no protocol that always solves this agreement problem [24]. *There is no algorithm that solves the coordinated action problem for two processors that communicate using unreliable messaging.*

Needless to say, if the problem cannot be solved for two devices, it cannot be solved for any larger number of devices. Of course, this problem still needs to be solved in real systems. This is normally done by strengthening the assumptions about the model of computation or by relaxing the problem requirements [34]. For example, this can be done by limiting the types of failure that the system tolerates and by stating guarantees probabilistically, thus allowing a system to be incorrect with very small probability. (Similar approaches can be applied in solving other problems we describe in the sequel.) Incidentally, this problem is known as the "Two Generals Problem" in the literature. Here, two generals must launch a coordinated attack, lest they be defeated one at a time by the opposing force. The generals communicate by messengers that can be intercepted or destroyed. The commercially available e-pollbook solutions are routinely silent about the system behavior when communication may be unreliable or lossy.

4.3 Availability, Consistency, and Network Partitions. All devices in the e-pollbook system must present a consistent view of the underlying data shared by the system. Because there must not be any single points of failure, the system must be distributed. If the system implementation is distributed, it must rely on some network for communication among its components. The implementation cannot assume that communication is always reliable; in particular, network failures may isolate some of the devices in the system. Clearly it is desirable for the service to be available and consistent,

with an illusion that there is a single copy of the object that is accessed sequentially regardless of how the object is implemented in the underlying distributed system.

however, the well-known "Brewer's conjecture" posits that it is not possible to simultaneously guarantee consistency, availability, and partition-tolerance [12]. *It is impossible for any distributed service implementation to provide the guarantees of (i) consistency, (ii) availability, and (iii) partition-tolerance.*

The above statement can be made more specialized for read/write objects as follows [22]. *It is impossible for any distributed service implementation of shared read/write data objects to guarantee (i) consistency, and (ii) availability, if the underlying asynchronous messaging system allows for message loss.*

This means, in particular, that if messages can be lost (e.g., due to jamming or denial-of-service attack), then either the data (e.g., voter records) may appear inconsistent or the service may be unavailable. The above result holds even if the system becomes synchronous, with known delays on the messages.

4.4 Reaching Agreement in the Presence of Crashes and Asynchrony. A polling place with several devices used to check in voters must be able to tolerate crashes. A crash must not prevent the overall system from taking coordinated actions. As before, the e-pollbook devices may not be in perfect synchrony with each other, e.g., processing delays and arbitrary timing of actions by the poll officials is likely to introduce some measure of asynchrony. Unfortunately, reaching agreement in the presence of even a single crash may be impossible in all cases for an asynchronous system, even if no message is ever lost. A seminal and venerable result from the distributed computing theory states the following [19]. *For an asynchronous system of processors that communicate using reliable channels there is no algorithm that solves the agreement problem and that guarantees termination in the presence of a single crash.*

This generally means that if a system relies on solving the agreement problem as part of its implementation, there may be some operations that never complete (or that are very slow). Thus, any system that claims that all of its devices are always in some type of agreement on certain values and that has good performance for all operations must make several non-trivial assumptions about the nature of failures and the constraints on asynchrony. If these assumptions are not explicitly stated, then the claims are to be taken with a grain of salt.

4.5 Agreement in the Presence of Malicious Failures. Given the plethora of malware and viruses that may affect a computer system, an e-pollbook system may also need to tolerate malicious failures of individual devices. Such malicious failures are called *byzantine* failures [33,41]. Here if a device fails, it does not stop as in a crash, but instead starts behaving arbitrarily, and in particular, it may perform malicious actions. This will be the case if a device is maliciously tampered with, or if it is infected with malware. For this setting, another seminal result states that a system of three devices (processors) cannot tolerate even a single byzantine fault [41]. *A system of three processors cannot solve the agreement problem in the presence of a single byzantine failure.*

Note that this result holds even if the processors are in a complete synchrony with each other and if there are no other perturbations, such as message delay or loss. For e-pollbook systems this means that a system with less than four devices cannot tolerate even a single malicious failure. The more general result dictates that any system of processors cannot tolerate malicious failures of even a third of the processors [41]. *A system of N processors cannot solve the agreement problem in the presence of F byzantine failures if $N \leq 3F$.*

Thus, in any system where the devices must reach agreement, the correct devices must outnumber the faulty devices by more than a factor of three-to-one. If this is not logistically feasible for a real installation, then the system cannot possibly claim to tolerate tampering with (or theft of) even one device.

4.6 The Problem of Reconfiguration in Dynamic Systems. Thus far we looked at static systems, i.e., a system where the universe of devices is fixed in the initial state. Providing e-pollbook solutions only for static systems is inadequate. Consider a polling place with three initial check-in devices. On the election day everything proceeds smoothly for a while. However the voter turnout is much higher than expected and the lines are getting long. Now suppose one of the three devices crashes. Even if the remaining two devices are operational and are able to provide the needed services, the lines of voters are getting really long now. A well-designed system must always allow for additional check-in devices to be introduced in order to cope with the faulty devices and the higher-than-expected voter turnout. Needless to say, this must be accomplished without halting the check-in process and without restarting any devices in the system.

The general problem of removing devices from a system and introducing new devices is known as the *reconfiguration* problem. In our context, we are not concerned with simply adding and removing devices: we need to also make sure that no data is lost and that the new devices are brought up to date with respect to the state of the data. Here all devices must have a consistent view of the state of the system. For a distributed system that is charged with maintaining consistent shared data the reconfiguration operation is described as follows [21].

Reconfiguration is the process of replacing one set of devices in a distributed system with an updated set of devices. In this process, the data is propagated from the old devices to the new set, and allowing devices that are not in the new configuration to safely leave the system. This changeover has no effect on data-access operations, which may continue to store and retrieve the shared data.

Development of algorithms implementing reconfiguration while providing uninterrupted and consistent access to data is an active area of research. A discussion of approaches to reconfiguration, including the use of Consensus and Group Communication Systems [9] can be found in [23,38]. The solutions to the reconfiguration problem are going to be difficult and fraught with impossibility results akin to those we discussed earlier. Regardless of how the reconfiguration of the set of devices is done, any practical implementation of e-pollbooks must address the challenge of deciding when to reconfigure. One approach is to leave this decision to the environment, e.g., the users of the system. Access to the reconfiguration service should be available to system administrators to enable reconfiguration based on policies, such as introducing new device in case of high voter turnout or removing misbehaving devices. For larger installations, reconfiguration could be enacted automatically when a failure of a certain number of devices is detected. This is a more complicated solution, but it has the potential of providing superior quality of service. Given that solutions to the reconfiguration problem are not routine, one must exercise caution. Any e-pollbook system that claims to provide reconfiguration features without supporting documentation and without rigorous arguments about the system's correctness must be carefully examined before any use on the election day.

5 A Broader Look at E-Pollbook Landscape

While this presentation focuses on the distributed systems aspects of e-pollbook systems, a comprehensive solution needs to consider a much broader landscape. There are additional dimensions of this problem that include not only the technological issues, but also societal and legal issues that impose separate requirements and that need to be incorporated. Furthermore, a substantial effort is needed to research, design, and implement a robust framework with rigorous semantics and security guarantees that provides a sound foundation for implementing e-pollbook solutions meeting the relevant requirements. To be successful, an approach to a solution needs to weave seamlessly key insights and contributions in distributed computing and cryptography into a general and implementable framework that is flexible, reusable and promotes the adoption of proven techniques with strong theoretical underpinnings.

Thus, to ensure that the technological approach encompassing theory and systems addresses meaningfully the societal, political, and legal needs it has envelop three interwoven areas: (1) Social and Political Science Dimension, (2) Computer Science as a Foundation for Software, and (3) Systems Implementation and Evaluation. This is illustrated in Figure 1.

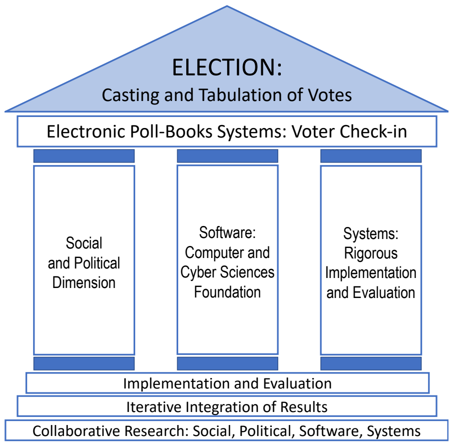
ELECTION:
Casting and Tabulation of Votes
Electronic Poll-Books Systems: Voter Check-in
Social and Political Dimension
Software: Computer and Cyber Sciences Foundation
Systems: Rigorous Implementation and Evaluation
Implementation and Evaluation
Iterative Integration of Results
Collaborative Research: Social, Political, Software, Systems

Fig. 1. Voters participate in the election proper, i.e., they cast their votes, only after they are admitted by means of the electronic poll-book system. The development of the systems rests on the three integrated pillars as shown.

5.1 Social and Political Science.

A 2020 poll conducted by well-regarded polling firm Ipsos found among US voters that 17% had waited in line more than one hour to vote, 4% were told their name was not on the registered voter list, 4% could not physically access the polling location, and 3% were told they did not have the correct identification [46]. Voter disenfranchisement can lead to a loss of system and government legitimacy [1]. Democracies are sustained and driven by the general principle of one person one vote. When that principle is challenged, whether due to social or administrative barriers, such as with a flawed voter check-in system, negative systemic and governing outcomes can result. Pew survey following the 2020 US Election found that more than 40% of respondents believed the election was not run well, and 15% were not confident that their vote was accurately counted.

Any e-pollbook system must offer features that enhance system legitimacy and efficiency. To accomplish this, it is important to conduct survey research, including elections administrators and citizens, that assesses the features of check-in systems. The results need to be incorporated into any e-pollbook system, and then evaluated, while keeping in mind the efficiency, usability, and availability of the implementation. Software artifacts need to be open-source and publicly available to assuage voter concerns of security and propriety.

5.2 Computing Theory Foundation. Any usable e-pollbook system is an inherently distributed system, consisting of multiple check-in devices and any supporting servers at the polling place (we take the position that within the current state-of-the-art it is unwise to rely on the Internet, or extend the system beyond a single polling place). In this section we survey some of the research topics that can be considered "theory." The problems that are more engineering in nature are deferred to the following section.

A framework for building secure and dependable e-pollbook systems must support dynamically changing collections of physical devices and provide resilient stores of objects supporting operations with provable semantics and consistency [23]. The underlying platform can not be assumed to be "nice." On the contrary, it is subject to perturbations, such as failures, delays, arrivals and departures of participating devices, and–ultimately–even malicious participants. The algorithms included and implemented in the framework must come with rigorous guarantees of fault-tolerance, security, and performance. These guarantees must include system reconfigurability to deal with the changing collection of participants, and consistency of the provided object storage.

Research in this area must include a definition of threat and failure model for which the algorithms and protocols are to be designed. An e-pollbook system must be resilient to ordinary failures of devices and networks, and to attacks by a motivated adversary. The election officials and voters are aware of the possibility for cyberattacks on election infrastructure and the States have mandated that polling places must not allow election-related devices to connect to the Internet [36,20], and the NASEM Committee on the Future of Voting recommends avoiding Internet-dependent election systems [2]. An e-pollbook system must operate in isolation within a polling place. Yet the system must tolerate attacks that can occur in the proximity to the polling place. One needs to assume that an adversary can monitor all network traffic at the polling place, and inject arbitrary packets into this network. The adversary can also cause traffic in the network to be dropped or delayed for an arbitrary, but finite, amount of time. Moreover, the adversary can compromise some of the devices, turning them into malicious participants that can also leak information to the adversary. All of these attacks can be launched by an attacker in close geographic vicinity of the public section of a polling place, especially if the devices comprising the system communicate with the over a wireless network (which is usually the case).

While the distributed computing research has a wealth of relevant results, much work remains in several ares. The data storage must be resilient and provide consistency guarantees supporting the at-most-once update semantics [28] (at most one vote). The imperfect biological units (i.e., voters and officials) must be able to interact with the system without destroying its semantics, here we note that election officials must be legally authorized to override that automated operation of the e-pollbook system. It is also fruitful to explore the provision of lightweight blockchain-style services [17], e.g., to implement audit logs and the monotone collection of data (yes, the reader expected us to say "blockchain" at least once). It is also relevant to extend the realm of byzantine fault tolerance to the current context (cf. [8,16]). While the byzantine model is heavy-handed, it does incorporate all malicious behaviors. Auditability is a must, and so the provision of immutable audit logs and efficient tools for examining such logs [4,37] is required. It is also necessary to adapt cryptographic techniques to deal efficiently with the security requirements, although much of this can be addressed through sound software engineering principles with respect to security, including authentication and encryption.

Catastrophic failures present another challenge. What should happen if the system completely fails? In the traditional distributed systems work one specifies an upper bound on the number of failures, then all bets are off when the limit is exceeded. This is unacceptable in the elections context, and solutions must deal with such scenarios. Here an approach based on self-stabilization needs to be explored (cf. [10,15]). But even this may not be enough, and for this reason there may be no way to avoid having an up to date printed record as the election progresses. In fact several states adopt the policy of a parallel use of the manual systems alongside the e-pollbook system to avoid disasters.

5.3 Systems: Development, Implementation, and Evaluation. In tandem with the theoretical efforts in the previous section, it is also necessary to develop and implement a middleware substrate, providing consistency guarantees, and to showcase this to solve a pressing implementation challenge in e-pollbook systems. The solutions to the research problems must also be shown be practical. The middleware must support dynamic storage [23] with at-most-once updates [28], providing a tamper-proof append-only audit log [4,37], and limiting the effect of compromised devices without imposing undue cryptographic overhead [35,47]. The middleware needs to also manage a reconfigurable collection of devices while tolerating benign and malicious failures. It must maintain a consistent view of the membership of devices, ensuring that only the authorized devices can participate in the e-pollbook operation and submit check-in requests. The reconfiguration of the system must be in response to failures and on demand (by election officials) without disrupting the check-in process. Note that a theft or an unauthorized access must cause or enable detection so that an immediate action can be taken. The system must have acceptable performance and must reasonably scale (in some jurisdictions there will be several tens of devices). This performance must not unduly degrade or degrade gracefully in the face of failures of devices or the network. This includes providing a reasonably short response time for check-in requests regardless of how many client devices are participating.

These goals must be achieved while also remaining resilient to adversarial behavior. The system must minimize the effect of failed or compromised devices, ensuring that malicious records they may introduce are not accepted by the system components. If a system deploys local servers in addition to the check-in device (usually tablets) the system must also tolerate malfunctions among the servers, including byzantine failures, and a few compromised servers should not spoil the integrity of the election. Finally, the system must implement the generation of trustworthy, auditable logs of all actions taken by client and server devices, and these logs must be recoverable despite any number of device failures.

The middleware must faithfully implement the guarantees of the underlying algorithms, including the guaranteed performance that depends on the perturbations in the underlying platform—the system must provide safety in all executions and conditional performance guarantees. The focus here is not on high volume data movers, but more on the integrity of the system state. Indeed, in the case of e-pollbooks, the information being exchanged by the devices boils down to the occasional on-site voter registration record creation and the "crossing-off" of identified voters whose arrival rate at the polling station is slow relative to the processing speed. However, resilience of the system is paramount as no information can be lost, corrupted, unavailable or inconsistent because of the failures (or disappearance, e.g., theft) of individual devices. It is equally important to retain the ability to inject new devices into the active pool to offset the loss of existing devices and without incurring any down time for *any* devices.

Because it is unavoidable that e-pollbook devices communicate wirelessly, it is important to incorporate traffic analysis to detect possible network attacks. Here a machine learning-based intrusion detection approach can be explored [3,31]. An e-pollbook system must also have means to communicate with the state-wide voter registration systems prior to the start of an election. This communication can be electronic and/or by means of removable media, in all cases the security, authenticity, and integrity of the voter data must be guaranteed.

The system must implement a rigorous framework for authenticating devices, software components, and communication, and for encrypting the relevant information (but note that audit logs must be authenticated but not be encrypted because the public must be able to inspect logs without relying on any system for decryption). Much of this can be accomplished by a careful application of known techniques, and intelligent the use of cryptographic tools (lest we create only a false sense of security [14]).

Lastly, research on and development of e-pollbook systems must be evaluated. To evaluate an implementation framework and the underlying algorithms, the following criteria must be kept in mind: (1) Simplicity: The system should be clear, with succinct, mathematically clear properties, and easy to understand and to use. (2) Applicability: It should be directly useful in developing a real e-pollbook application, such that a jurisdiction might be tempted to use. (3) Feasibility: It should be implementable with acceptably good performance.

The underlying algorithms and protocols must be evaluated according to their correctness in with respect to their specifications, their simplicity, and their degree of security, performance and fault-tolerance.

6 Discussion

We presented a view of e-pollbook solutions as distributed systems. We cited requirements that need to be satisfied by any robust e-pollbook system in order to guarantee fault-tolerance, availability, and consistency. We also present a broader view that societal issues, additional research directions, and the need for reference implementations. While it is also necessary to address certain security issues, including the security of underlying physical platforms, and catastrophic failure issues, these are outside of the intended scope. We provide key results from the distributed system research showing that it is challenging to build adequate poll book solutions, and that without being grounded in relevant research any solution is likely to be incorrect and provide only an illusion of fault-tolerance, consistency, and safety. Indeed, we are not aware of a single commercially available implementation that satisfies the overall requirements and that are consistent with the relevant research. An important conclusion is that e-pollbook development is an attractive application domain for the research in dependable distributed computing.

References

1. Deepening democracy: A strategy for improving the integrity of elections worldwide. In *Global Commission on Elections, Democracy and Security*. Kofi Annan Foundation, 2012. `https://aceproject.org/ace-en/topics/ei/onePage`.
2. Securing the Vote: Protecting American Democracy, September 2018. National Academies of Sciences, Engineering, and Medicine.
3. A. AlEroud and G. Karabatis. Bypassing detection of url-based phishing attacks using generative adversarial deep neural networks. In *Proceedings of the Sixth International Workshop on Security and Privacy Analytics*, pages 53–60, 2020.
4. T. Antonyan, S. Davtyan, S. Kentros, A. Kiayias, L. Michel, N. C. Nicolaou, A. Russell, and A. A. Shvartsman. Automating voting terminal event log analysis. In *EVT/WOTE*, 2009.
5. A. Aponte, J. Cruz, C. Jennings, D. MacDonald, and S. Wooden. Committee of Inquiry Report of Factual Findings. Tech. Rep. *City of Hartford Court of Common Council*, 2015.
6. H. Attiya, A. Bar-Noy, D. Dolev, D. Peleg, and R. Reischuk. Renaming in an asynchronous environment. *J. ACM*, 37(3):524–548, 1990.
7. M. Bernhard, J. Benaloh, J. Alex Halderman, R. L. Rivest, P. Y. A. Ryan, P. B. Stark, V. Teague, P. L. Vora, and D. S. Wallach. Public evidence from secret ballots. In *Electronic Voting*, pages 84–109, 2017.
8. A. Binun, T. Coupaye, S. Dolev, M. Kassi-Lahlou, M. Lacoste, A. Palesandro, R. Yagel, and L. Yankulin. Self-stabilizing Byzantine-tolerant distributed replicated state machine. In *Stabilization, Safety, and Security of Distributed Systems*, LNCS 10083, pages 36–53, 2016.

9. K. Birman. A history of the virtual synchrony replication model. In *Replication: Theory and Practice*, LNCS, pages 91–120. Springer, 2010.
10. P. Blanchard, S. Dolev, J. Beauquier, and S. Delaët. Practically self-stabilizing paxos replicated state-machine. In *Networked Systems - Second Int-l Conf., NETYS 2014. Revised Selected Papers*, LNCS 8593, pages 99–121. Springer, 2014.
11. G. Bracha and S. Toueg. Asynchronous consensus and broadcast protocols. *J. ACM*, 32(4):824–840, Oct. 1985.
12. E. A. Brewer. Towards robust distributed systems (abstract). In *19th ACM Symp. on Principles of Distr. Comp., July 16-19, 2000*, page 7. ACM, 2000.
13. S. Davtyan, S. Kentros, A. Kiayias, L. Michel, N. Nicolaou, A. Russell, A. See, N. Shashidhar, and A. Shvartsman. Taking total control of voting systems: firmware manipulations on an optical scan voting terminal. In *2009 ACM Symp. on Applied Computing*, SAC'09, pages 2049–2053. ACM, 2009.
14. S. Davtyan, A. Kiayias, L. Michel, A. Russell, and A. A. Shvartsman. Integrity of electronic voting systems: Fallacious use of cryptography. In *Proc. of 27th Annual ACM Symp. on Applied Computing*, SAC '12, pages 1486–1493. ACM, Mar. 2012.
15. S. Dolev, C. Georgiou, I. Marcoullis, and E. M. Schiller. Self-stabilizing Byzantine tolerant replicated state machine based on failure detectors. In *Cyber Security Cryptography and Machine Learning, CSCML 2018, Proc.*, LNCS 10879, pages 84–100. Springer, 2018.
16. S. Dolev and M. Liber. Toward self-stabilizing blockchain, reconstructing totally erased blockchain. In *Cyber Security Cryptography and Machine Learning - Fourth Int-l Symp., CSCML 2020, Proc.*, LNCS 12161, pages 175–192. Springer, 2020.
17. S. Dolev and Z. Wang. Sodsbc: Stream of distributed secrets for quantum-safe blockchain. In *2020 IEEE Int-l Conf. on Blockchain*, pages 247–256, 2020.
18. E.Cox. New Voting Machines Finally on Horizon: Seven years later, maryland finally buying voting machines with a paper trail. *The Baltimore Sun*, December 16, 2014.
19. M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, Apr. 1985.
20. W. M. Gardner. *New Hampshire Electronic Poll Book System Request for Information V0.1.* New Hampshire Department of State, 2017.
21. S. Gilbert, N. Lynch, and A. Shvartsman. RAMBO: A robust, reconfigurable atomic memory service for dynamic networks. *Distributed Computing*, 23(4):225–272, December 2010.
22. S. Gilbert and N. A. Lynch. Brewer's conjecture and the feasibility of consistent, available, partition-tolerant web services. *SIGACT News*, 33(2):51–59, 2002.
23. V. Gramoli, N. Nicolaou, and A. A. Schwarzmann. *Consistent Distributed Storage.* Morgan & Claypool Publishers, 2021.
24. J. Gray. Notes on data base operating systems. In *Operating Systems, An Advanced Course*, volume 60 of *LNCS*, pages 393–481. Springer, 1978.
25. M. P. Herlihy and J. M. Wing. Linearizability: A correctness condition for concurrent objects. *ACM Trans. on Prog. Lang. and Systems*, 12(3):463–492, 1990.
26. T. Iredale and K. Clark. System and method for synchronizing electronic poll book voter databases. U.s. patent us8812594 b2, August 19 2014.
27. R. Jancewicz, A. Kiayias, L. Michel, A. Russell, and A. Shvartsman. Malicious takeover of voting systems: Arbitrary code execution on optical scan voting terminals. In *28th Annual ACM Symp. on Applied Comp.*, pages 1816–1823, 2013.
28. S. Kentros, A. Kiayias, N. C. Nicolaou, and A. A. Shvartsman. At-most-once semantics in asynchronous shared memory. In *Distributed Computing, 23rd Int-l Symp., DISC 2009.*, volume 5805 of *LNCS*, pages 258–273. Springer, 2009.

29. A. Kiayias, L. Michel, A. Russell, N. Sashidar, A. See, and A. Shvartsman. An authentication and ballot layout attack against an optical scan voting terminal. In *Electronic Voting Technology Workshop, EVT'07*. USENIX Association, 2007.
30. A. Kiayias, L. Michel, A. Russell, N. Shashidhar, A. See, A. Shvartsman, and S. Davtyan. Tampering with special purpose trusted computing devices: A case study in optical scan e-voting. In *Computer Security App-s Conf., ACSAC 2007. 23rd Annual*, pages 30–39, 2007.
31. C. Kolias, G. Kambourakis, A. Stavrou, and S. Gritzalis. Intrusion detection in 802.11 networks: empirical evaluation of threats and a public dataset. *IEEE Communications Surveys & Tutorials*, 18(1):184–208, 2015.
32. L. Lamport. On interprocess communication. part i: Basic formalism. *Distributed Computing*, 2(1):77–85, 1986.
33. L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982.
34. N. A. Lynch. *Distributed Algorithms.* Morgan Kaufmann Publishers Inc., 1996.
35. H. Maleki, R. Rahaeimehr, C. Jin, and M. van Dijk. New clone-detection approach for rfid-based supply chains. In *2017 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, pages 122–127, 2017.
36. D. W. Merrill. *Connecticut Electronic Poll Book System: Requirement Specification V1.0.* Office of the Connecticut Secretary of the State, 2015.
37. L. D. Michel, A. A. Shvartsman, and N. Volgushev. A systematic approach to analyzing voting terminal event logs. *USENIX Journal of Election Technology and Systems (JETS)*, 2(2):34–53, 2014.
38. P. M. Musial, N. C. Nicolaou, and A. A. Shvartsman. Implementing distributed shared memory for dynamic networks. *Commun. ACM*, 57(6):88–98, 2014.
39. One Hundred Seventh Congress of the United States of America. Help America Vote Act of 2002, 2002.
40. M. Pawlak, J. Guziur, and A. Poniszewska-Marańda. Towards the blockchain technology for system voting process. In *Cyberspace Safety and Security*, pages 209–223, 2018.
41. M. C. Pease, R. E. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.
42. P. Peisch. Procurement and the Polls: How Sharing Responsibility for Acquiring Voting Machines Can Improve and Restore Confidence in American Voting Systems. *The Georgetown Law Journal*, 97.877:877–915, 2009.
43. R. Rivest. Clipaudit: A simple risk-limiting post-election audit. *ArXiv*, abs/1701.08312, 2017.
44. R. L. Rivest and P. B. Stark. When is an election verifiable? *IEEE Security Privacy*, 15(3):48–50, 2017.
45. A. A. Shvartsman, A. Kiayias, L. Michel, and A. Russell. On the security and integrity issues of optical scan voting systems. In *County of Nassau Board of Elections against State of New York, New York State Board of Elections*, pages 1–23. Supreme Court of the State of New York, County of Nassau, March 19 2010.
46. A. Thomson-DeVeaux, J. Mithani, and L. Bronner. Why many americans don't vote. In *FiveThirtyEight*, 2020 (October 26).
47. M. van Dijk, C. Jin, H. Maleki, P. Ha Nguyen, and R. Rahaeimehr. Weak-unforgeable tags for secure supply chain management. In *Financial Cryptography and Data Security*, pages 80–98. Springer, 2018.
48. S. Wolchok, E. Wustrow, D. Isabel, and J. A. Halderman. Attacking the Washington, D.C. Internet Voting System. In *Financial Cryptography and Data Security - 16th Int-l Conf., FC 2012*, LNCS 7397, pages 114–128. Springer, 2012.