1. Overview

Online Voter Registration ("OVR") is a way to improve the administration (and operational efficiency) of U.S. election voter rolls by automating the existing voter-registration process, thereby providing convenience for voters and cost savings for election officials ("EOs"). Designed and implemented right, it is a perfect “cloud-based” service for election administration.

One of the essential challenges to implementing OVR is integrating new OVR technology with pre-existing voter-records management systems ("VRMS") technology that itself was never designed to operate in an open Internet environment. That environment poses considerable security risk to the integrity of the voter registration ("VR") technology base, especially to the integrity of voter-records data.

This paper provides a detailed description of a “reference architecture” for OVR system deployment and integration with pre-existing VRMS technology. This architecture partitions the integrated system in a way that prevents the risks to Internet-connected OVR technology from being passed along to the legacy VRMS technology. This paper also provides some guidance on addressing residual risks to OVR integration—those that would remain after putting in place an OVR system based on this reference architecture.

Scope

One important foundational issue for OVR security must be stressed: OVR is constrained to operate with the existing regulations and practices of election administration. For example, a citizen can submit a voter registration request on a paper form, physically transported to an EO, or a voter can provide the same information in a digital form digitally transmitted to an IT system used by an EO. Digital accommodations exist in some implementations for capture of a registrant’s hand signature. These include data acquisition of a previously recorded hand signature from a motor vehicle department interaction, or capture of a hand signature with a
stylus on a tablet computer. But these accommodations do not make an essential change to the work process of submitting, receiving, and processing voter records requests. Regardless of variation in those processes, ultimately an EO is involved in deciding whether a request is complete and correct, and whether the applicant is eligible to be added to the voter rolls. The same is true for the processing a request to update an existing voter record, for example, for a change of address.

A similar point is required regarding the fraud model of OVR being the same as physical (offline) VR. Particularly in light of years of experience with identity theft and public disclosure of Personal Identifying Information (“PII”) of very large numbers of voters, it is important to recognize that VR has and continues to have a simple fraud model, particularly with reference to updates to existing voter records.

EOs are obligated to process any VR request that meets requirements for completeness and consistency, accompanied by an attestation that the submitter acknowledges and understands that any willful misrepresentation amounts to fraud and is most likely a felony in that State. For update requests, EOs are obligated to honor requests that meet all the rules for matching existing voters with PII provided on the request. For many voters, the requisite PII for that voter is available to adversaries who may wish to fraudulently pose as that voter in submitting a VR request.

The Reference Architecture described herein does not address the risks of fraudulent direct abuse of an OVR system in ways that parallel fraudulent abuse of paper-based VR. Instead, this paper describes technique to mitigate the technical risks from adversarial cyber-operations that target the voter list data of a legacy VRMS, through exploits of an Internet-connected OVR front-end system.

The objective is to gain the benefits of OVR while also maintaining integrity of the voter list data. The cyber-operations threat is much smaller in scope to VRMS without any OVR integration; OVR integration can significantly increase the attack surface of VR processing systems. This paper provides models and recommendations for how to limit this increase in risk, while implementing an OVR system that integrates with an existing VRMS.

2. Basic OVR Operational Requirements

Figure 1 on the next page illustrates the basics of an OVR system’s public facing technology, in the most common formulation of an OVR system.² A citizen uses a web browser to communicate over the Internet with a web-based application that presents dialogs for collecting the same voter registration request-form data that is required in a paper-based voter-registration process. The web application helps the user to create a complete and correct voter-registration application. When the user has completed the web form, the data collected is packaged into a digital voter-registration request, and digitally transmitted to a back-end system that warehouses the requests until processed by an election official. If the election official approves it, a new voter record is added to the official voter records database, or an existing record is updated.

² Variants on this are possible; we’ll include them as this paper is updated
Figure 1 also illustrates a second requirement for most OVR systems. In many cases, voter-registration applicants are not sure whether they are registered; whether their record is up to date; whether they need to update their registration address (or other part of their voter record); or whether no action is required. In the paper-based process, a voter registration form can serve as a new registration request or as an update to an existing voter record if the applicant is already registered. However, a common capability of the OVR approach is to enable users to identify themselves to the OVR system, in order to find out whether they are currently registered, and whether they need to provide an update. If updates are necessary, the OVR system facilitates the submission of a voter registration update request.

In order to retrieve an existing voter record, or to update it, the user must provide a valid and matching set of personally identifiable information (“PII”) — typically elements of name, date of birth, driver’s license or other identity number, and sometimes elements of the registration address. When such PII does not match an existing voter-registration record, it is used as part of a new voter-registration request; when it does match, it is used as part of an update request.

Thus, an OVR system must be able to collect voters’ requests and hold them until they can be processed by an election official and integrated with the official voter records, and it must be able to retrieve existing records. The voter records database (“VRDB”) is a lynchpin of election integrity, because it is used to decide who is eligible to vote in each election, and because it provides the voter lists for every polling location and other voting method. An OVR system must preserve the integrity of its data when it is integrated with the VRMS.

OVR Security Goals and Basic Recommendations

Each of these two basic OVR functional requirements raises a basic question:

\[
\text{How should Internet-connected OVR system components be able to access data managed by a pre-existing VRMS, without creating new risk to the VRMS from the OVR system’s Internet connection?}
\]

One approach is direct integration, with the OVR system being able to connect directly to the VRDB to search for records and to store digital voter registration requests in the VRMS, for later processing by election officials. However, as experience in 2016 showed, this approach makes OVR systems tempting targets for cyber-adversaries who, when successful, can gain access to the core data, tamper with the data as stored, and exfiltrate the data.

Indirect integration is another approach, and it is the approach strongly recommended in this paper. With indirect integration:

- The Internet-connected OVR system is not connected to the official VRDB, but instead to a copy of the data stored in the VRDB;
• The Internet-connected OVR system does not send requests directly to the VRMS, but instead maintains its own database of pending requests.

The Internet-connected OVR system never communicates directly with the VRMS (which for maximum safety should not be connected to or accessible by the Internet at all), but rather receives data from existing voter records and transmits requests to create or modify records through an intermediary system. In this paper we refer to this intermediary system as an OVR Gateway, as illustrated in Figure 2.

The remainder of this paper describes how to meet the operational requirements above, while also implementing these two recommendations. The central issue is the means by which the OVR Gateway facilitates data exchange between the OVR system’s local database and data stores of the VRMS and VRDB so that the latter are insulated from the inevitable risks to the OVR system that arise from its Internet connection and accessibility.

3. OVR Gateway and Voter Data Exchange

The basic concept of the OVR Gateway shown in Figure 2 is a dedicated system that:

• Has access to the databases that are used by the Internet-connected OVR front end system;
• Is not directly connected to the Internet itself; and
• Is protected from access by the OVR front end system or any other Internet-connected system.

There are options for how an OVR Gateway can use its database access to “pull” and “push” data created by or used by the OVR systems. The distinctions between these alternatives all rest on the answer to one question:

**What is the protected and privileged method by which the OVR Gateway can access the data stores of the VRMS and VRDB?**

Figures 3 and 4 illustrate the simplest answer to this question: the use of “sneaker-net;” that is, an “air-gap” that is traversed by administrative staff physically transporting data on removable media that are used once. On the left of Figure 3, system administrative staff uses a data export utility to dump a copy of the VRDB, or more accurately, a list of records about each voter containing only data fields that are needed by the OVR system for voter record lookup. Many of the fields of the full VRDB can be omitted for simplicity and privacy, while any required Personal Identifying Information (PII) such as driver’s license number can be obscured for privacy, via encoding with cryptographic hash methods.
On the right of Figure 3, the removable media is used by an operator of the OVR Gateway, to copy the data from media, and uses a data import utility to re-fill the OVR system’s database with a current snapshot of the voter list.

Figure 4 illustrates the reverse data flow, for voter registration request (VRR) records collected by the Internet-connected OVR system. Administrative staff use a data export utility to copy to removable media a specific set of VRR records, and to mark them in the OVR database as having been copied. The media is then used with a data import utility that can copy from the media to the VRMS’s data store for VRRs that are yet to be processed by election officials.

A critical concept for this architecture is that the data import and export utilities are not modifications to the existing VRMS, nor do they require any changes to VRDB management. These utilities run on systems with access to data storage, in order to facilitate the information exchange. But it is only the human IT operators who use this access. The data exchange itself happens on the other side of the air gap. In this “air gap” version of the recommended OVR architecture, the OVR Gateway is a very simple dedicated system that sits idle except when used as that stage for IT operators to run tools to push or pull data from the databases of the Internet-connected OVR front end system.

**Automation Alternative to Air-Gap**

Figure 5 on the next page illustrates a more abstract version of the OVR Gateway architecture, in which the details of air-gap data exchange are omitted. Another possible approach, though technically more complex, is one in which the OVR Gateway is implemented as an autonomous system, with network access to both the data storage server of the existing VRMS and the data storage server of the Internet-connected OVR front end. In this approach, the OVR Gateway acts a bridge between the two systems, periodically performing the work of data push and pull that would be performed by operators in the air-gap approach.

This more complex approach to an OVR Gateway may be more operationally efficient, but it carries additional security management requirements. Although the OVR Gateway should not be directly connected to the OVR front end or the Internet, network access controls may not be
sufficient to prevent access to the OVR Gateway host from a host that is connected to the Internet. As a result, careful network segmentation and ongoing management of network-segmentation security are required, to ensure that any network configuration changes will preserve the segmentation required to protect the OVR Gateway.

Other technical measures would be beneficial as well, such as the use of application-level firewalls on either side of the OVR Gateway. These firewalls should ensure that only interaction on the “front” side of the OVR gateway is its own network access to data storage of the OVR front end; and that the only interaction on the “back” side of the OVR Gateway is its own network access to data storage of the VRMS. Any other network activity with the OVR Gateway host would be blocked, and all allowed OVR Gateway activity would be scanned to ensure that the activity is all and only the data exchange function of the OVR Gateway.

While this more complex architecture may be advantageous, we recommend controlling the VRMS and OVR systems first and using the air-gap approach initially. Once an OVR system is in production, and the operations team has gained experience with the controls required in addition to the air-gap, then IT management can consider the value of a follow-on effort to re-implement a more automated form of an OVR Gateway.

4. Architecture for 3rd-Party OVR Integration

In addition to the public OVR web interface for voters, another common function of an OVR system is to provide an interface for 3rd-party registrar (3PR) organizations that enable voter registration and their own OVR systems. These 3PR systems provide both web and mobile App interfaces for the 3PR organization’s constituency. By supporting integration with 3PR systems, a state-level OVR operation can expand the reach of its OVR service and leverage the resources and constituency outreach of 3PR organizations.

The typical method for 3PR integration is for an OVR system to have, in addition to a browser-based user interface, a web services interface that provides
programmatic access to the OVR system. Just as the web user interface is a portal to which users’ web browsers can get information and send voter registration records, the Web services interface is a portal to which 3PR systems can get similar information and make similar requests, but in a technically distinct format that is more suitable for integration.

One approach to 3PR integration is for the Internet-connected OVR system to offer both the Web user interface and the Web services interface in the same system (see Figure 6, previous page). While workable, this approach combines disparate functions that can be on different development, deployment, and testing schedules. A recommended alternative is to have a distinct system dedicated to delivering web services to 3PR systems.

One additional advantage of separation between these systems is that it facilitates adoption of the national data interoperability standard being developed by NIST specifically for OVR data interchange. When an OVR operator implements its web services interface according to this standard, 3PRs can integrate that OVR system in the same manner as others, limiting state-specific integration tasks. Likewise, OVR operators are relieved of the burden of developing a Web services interface from scratch and potentially introducing unnecessary, costly complexity that the standards efforts worked to iron out.

Figure 7 shows an advantageous elaboration, in which much of the complexity of the data management and security isolation is moved to the OVR Gateway itself. In this architecture, the OVR Gateway has sole control of the data storage used for OVR (whether or not the data integration is air-gapped), isolated from direct access from the Internet. The OVR Gateway uses a type of integration approach called an Application Programming Interface (API) to encapsulate and protect the data. The OVR web UI and web service components use this API to both: make voter lookup requests and receive lookup results; and to submit VRR requests and receive confirmation that the VRR requests were stored for later review by election officials.

5. Recommended Practices for Operational Security

The security controls on OVR technology presented in previous sections specifically address safety concerns for OVR data interchange. These controls are only part of a complete approach to implementing an OVR system or revisiting existing OVR implementations; other controls address additional VR operational security issues. To conclude this report, we summarize some of these additional VR operational security issues and recommend some priorities for addressing the mix of operational and architectural measures described in this report.
Operational Security Issues Separate From OVR Data Interchange

In addition to the OVR-specific measures described above, there are several other operational security issues that should be addressed in any OVR system. These apply both to pre-existing VRMS systems and to new systems added for OVR.

For pre-existing VRMSs, this report’s reference architecture has the benefit of avoiding new Internet-related operational security requirements that might arise if VRMSs were not isolated from new OVR systems. However, several kinds of security controls are relevant for VRMSs and for management of the data in the VRDB: access control on access to databases; logging of all transactions on the VRDB and on other stored VRMS data; regular backup of VRDB contents; and periodic test of restore from backup, to name a few. These and several other typical database operations management practices remain relevant to VRMS operations, whether or not OVR is supported by a state’s VRMS.

For OVR system components, this report’s reference architecture has the benefit of limiting the operational security controls largely to those typical of Internet-facing web application, rather than imposing additional VR-specific requirements that might arise if pre-existing VRMS components were not isolated from OVR components. Among these typical controls are:

- Limiting public access to a single HTTPS port;
- Use of TLS proxy;
- Web application-level firewall filtering of requests;
- Network segmentation to limit public facing hosts’ access to other hosts;
- Web application security scanning for vulnerabilities including SQL injection;
- Host and network intrusion detection;
- Use of API keys for authentication and control of access by 3PR systems to OVR web service interface; and
- Carefully controlled and audited IT administrative access, particularly for management of API keys.

In addition to such controls, technical measures may be needed to support fraud monitoring, such as logging each OVR request along with IP source address. Privacy measures are also relevant, for example, eliding PII in log records.

Beyond these fairly typical security measures for public-facing web applications, there are data security concerns more specific to OVR. The Internet-facing components of an OVR system should be able to read, but not write the data derived from a copy of voter list data, and they should be able to append to, but not modify or delete VRR “inbox” data. Although network-level filtering and DBMS-level access controls can be used for this purpose, the recommended approach is shown in Figure 7 (previous page), where both these types of OVR front-end data are managed by an OVR Gateway and are accessible only via an API. The API implementation should ensure that only read operations are available on voter list data, and that only an append operation is available for the “inbox.”
6. Summary

This paper has presented a reference architecture for the system integration process of updating an existing Voter Registration (VR) system by re-using an existing Voter Records Management System (VRMS), and adding new software components to serve as the Online Voter Registration (OVR) front end.

The key principle is that Internet-connected systems should _never_ have direct access to systems that access the Voter Records Database (VRDB). The “air-gap” approach is simplest to set up, and simplest for IT management, but has ordinary operations that can be labor intensive. Other variants of the reference architecture automate some of the data transfer, but have additional complexity for system integration, and require additional IT operations support.

Finally, each of the VRMS back-end, and the OVR front-end systems have significant IT operations requirements for operational security. This reference architecture is consistent with typical operations practices, including but not limited to those summarized herein. Such typical security practices remain relevant for operational security. The objective of this reference architecture is to maintain the sufficiency of those typical practices, while avoiding any new and VR-specific security measures that would be required if Internet-connected systems were able to access the voter list data of record.