Podemos October Congress: Technical Overview

David Ruescas <david@agoravoting.com>
Eduardo Robles Elvira <edulix@agoravoting.com>
October 2014

1. Introduction

Podemos[1] is a political platform that will be carrying out an online election to choose a set of foundational resolutions as part of its foundational congress, taking place in October 2014 [2]. In order to meet this goal, Podemos have decided to use and contribute to Agora Voting. Agora Voting [28] is a free software project dating back to 2011. This document will provide an overview of the technical aspects of this project.

First we will describe the election and voting process. Then we will briefly list the most significant software/hardware components that make up the Agora Voting system (henceforth "the system"). We will then review the cryptographic primitives employed, and the resulting theoretical properties the system will exhibit under an explicit trust model. This will be done from two points of view; first in terms of typical properties considered for secure voting systems, and secondly in terms of threats. Finally, some possible alternative trade-offs will be considered.

2. The Election

2.1 Stakeholders

- Podemos - Political party conducting online primaries, administrators of the Registry (3.1)
- Eligible Voters - Supporters of Podemos who have completed pre-registration and are part of the census
- Agora Voting - Developers of the system, administrators of the Polling Station (3.2) and Election Authority

- OpenKratio [26] - Election Authority

Election Authorities serve as the electronic voting analogues of Scrutineers (poll-watchers) in traditional elections.

2.2 Questions

The election will decide a set of foundational resolutions according to one question using Plurality-at-large [3]:

¿?

Voters are allowed a maximum of 5 selections among the candidate resolutions.

2.3 Process

2.3.1 Key generation

The Election Authorities jointly create the election public key.

2.3.2 Voting phase

Once the voting period begins:

2.3.2.1 Voters access the registry webpage and provide personal information together with a password

2.3.2.2 The registry will validate the voter's data (DNI) according to a supporter census. If correct the voter is redirected to the Polling Station voting booth

2.3.2.3 The voter fills in the ballot, reviews it, and modifies if necessary

2.3.2.4 The ballot is encrypted and cast

2.3.3 Tabulation phase

Once the voting period is over:

2.3.3.1 The Election Authorities jointly execute the shuffling and decryption process, which produces ballot plaintexts

2.3.3.2 The plaintexts are tallied

2.3.3.3 The election results are published, including

- The final election results in the form of final tallies
- Ballot ciphertexts
- Ballot plaintexts
- Zero knowledge proofs of shuffle and decryption

2.3.3.4 Voters and third parties download and run the Election Verifier (3.4).

3. System Components

In this section we briefly describe the components that make up the voting system. At a high level, there are three items that take part in the voting process, plus the election verifier.

- The Registry, a web-based platform that serves to verify voters

- The Poll Station, a web-based platform through which voters fill and cast ballots

- The Election Authority Services, these are http web services that jointly create public keys, perform shuffling, decryption and tabulation

- The Election Verifier, a stand alone Python/Java application that verifies the published election data.

We list the most relevant software and infrastructure elements of each of these:

3.1 Registry

As the Registry is not developed nor administered by Agora Voting there are no technical details in this section. Please refer to Podemos for this information.

3.2 Polling Station

- Go rest api backend
- Ballotbox database
- Server SSL validation
- HMAC client authentication for registry validated voters
- Javascript encryption client libraries (SJCL, Helios)
- RNG: window.crypto.getRandomValues where available
- Fail2ban for DOS protection and against brute force registry attacks
- Cloudflare for DOS protection

3.3 Election Authority

- HTTP distributed queue for orchestration
- Client and server SSL validation
- Verificatum mixnet library
- OpenSTV tabulation library

3.4 Election Verifier

- Python/Java standalone application

4. Cryptographic primitives

The following cryptographic primities are used by the system, see also Wombat [12]

4.1 ElGamal homomorphic encryption scheme [5]

4.2 (t, n) Pedersen threshold encryption scheme, with n secret shares such that at least t parties must participate in message decryption [6, 7]

4.3 Universally verifiable mix-net producing zero-knowledge correctness proofs [8, 9]

4.4 Fiat-Shamir heuristic to transform honest-verifier zero knowledge proofs into non-interactive verifiable proofs [10]

4.5 Schnorr proof of plaintext knowledge [11] to make the scheme CCA2-secure

5. Trust model

The following assumptions will be made when evaluating the system's properties in later sections.

5.1 Assumptions pertaining to integrity

5.1.1 The voter trusts his/her voting device (computer/smartphone) to be operating correctly (in particular, the device, OS and browser are free of malicious software)

5.1.2 The voter trusts the administrators of the Polling Station to be operating honestly.

5.1.3 It is assumed that an attack that takes full control of the Registry/Polling Station is detected.

5.2 Assumptions pertaining to privacy

5.2.1 The voter trusts his/her voting device (computer/smartphone) to be operating correctly (in particular, the device, OS, and browser are free of malicious software).

5.2.2 The voter trusts that at least 1 of the election authorities or the administrators of the Registry are honest.

5.3 Assumptions pertaining to incoercibility

For completeness, we will state one assumption with consequences for receipt-freneess, although this assumption can be dropped without significant changes as incoercibility is not a property of the system (see 6.6)

5.3.1 The election authorities trust that the voter is not injecting the voting client device with malicious software to access the encryption randomness

6. System properties

In this section we will evaluate system according to (most of) properties defined in [19]; see also [20]. For each property we state whether the system satisfies the property, adding comments if necessary. Note that these assessments are tentative conclusions to the best of our knowledge. Please refer to [19] for the full description of each of these properties.

6.1 Eligibility

Assessment: satisfied

Comments: The Registry validates voters against a pre-existing census.

6.2 Privacy

Satisfied

Provided by the cryptographic scheme. Depends on 5.2.1 and 5.2.2

6.3.1 Individual verifiability

Satisfied

Provided by the cryptographic scheme. Depends on 5.2.1

6.3.2 Universal verifiability

Satisfied

Provided by the cryptographic scheme. Depends on 5.2.1

6.3.3 Eligibility verifiability [21]

Not satisfied

There exists no data in the system linking encrypted ballots to corresponding registry information, and therefore this information is not available as part of the publicly verifiable election data. Hence eligibility verifiability is not satisfied. See 8.1

6.4 Dispute-freeness

Partially satisfied by universal verifiability

6.5 Accuracy

Provable per election as 6.3.2 is satisfied

Here we restrict the notion of accuracy to refer to the correct recording and tallying of votes cast by eligible voters, leaving out issues related to 6.1. Because universal verifiability is satisfied, this property is provable for any given election. Depends on 5.2.1.

6.6 Incoercibility

Not satisfied

The problem of coercion is in general not solved for electronic or traditional paper based elections. Coercion can happen if the coercer is present during vote casting, or it can happen remotely as the system is not entirely receipt-free (6.10).

6.7 Scalability

Probably satisfied

It is impossible to know whether the system will adequately handle the volume of voting activity until the election is carried out. However, the system has been designed for scalability (see section 3) and has been tested to handle loads greater than expected voter turnout.

6.7.1 Practicality

A practical voting scheme should not have assumptions and requirements that may be difficult to implement on a large scale.

Strongly satisfied

This is the key advantage of internet voting, motivating higher participation as voters do not have to physically visit the polling station.

6.8 Robustness

Satisfied for authorities.

The system is by definition robust against corrupt authorities given 4.2. Other aspects of robustness are interpreted as problems of availability, see 7.14.

6.9 Fairness

Satisfied

Depends on 5.2.2

6.10 Receipt-freeness

Partially satisfied

The property is partially satisfied if assumption 5.3.1 holds. However, there are other ways to provide a receipt, see for example [22].

7. Threats

In this section we will analyze the system from an adversarial point of view, in terms of specific threats. Although there is considerable overlap with the previous section it is worthwhile to consider this perspective as it is closer to practical, concrete attacks.

Following [13] and [14], we can classify threats into three major groups

- Threats to Integrity
- Threats to Privacy
- Threats to Availability

Integrity refers to the correctness of the election results, including voter eligibility, ballots and tally. Privacy refers to the confidentiality of the voter's ballots. Availability refers to the guarantee that any and all eligible voters are able to participate without inconveniences or barriers.

Although [14] refers to a system (Wombat) which includes paper ballots we can still reuse and adapt many of the threats listed there. Some of these threats have also been listed in [15]. Each section shows a brief description of the threat (copied from its original source if applicable), a classification of the applicability of the threat (eg possible, not possible), and short comments. Note that these assessments are tentative conclusions to the best of our knowledge. Please refer to [14] for the full description of some of these threats.

Threats to Integrity

7.1 Registration Frauds

A voter is identified but casts his vote in another person's name. In this case, one person can cast more than one vote using the names of other voters.

Assessment: Not possible

The Registry uses a census.

7.2 Repeating

A voter casts his vote more than once.

Not possible

Ruled out by conjunction 5.1.2 & 5.1.3

7.3 Ballot Stuffing

"Stuffing" votes without voters, or entering more votes into the system, without the casting process.

Not possible

Ruled out by conjunction 5.1.2 & 5.1.3. Stuffing is possible if the administrators (Registry or Polling Station) are malicious.

7.4 Altering Ballots

The voter knows the candidate he voted for, but the ballot has been altered and does not reflect his vote

Not possible

Ruled out by 5.1.1. Possible if the voting device is compromised.

7.5 Altering Returns

A ballot selection is entered into the voting system, but it is counted differently.

Not possible

Ruled out by universal verifiability. See 6.3.2

7.6 False Count and False Returns

Falsifying the tallying while counting the votes, or tallying honestly and returning a false result.

Not possible

Ruled out by universal verifiability. See 6.3.2

7.7 Italian Attack [18]

Possible. Unlikely and low impact

The Italian Attack is an issue of coercion, not integrity. Nonetheless we chose to include this threat here. The Italian Attack is an unlikely threat because ballot size for question 2.2 allows for only 5 unordered (using choice order randomization) candidates as the signature. It is low impact because there are easier routes to coercion, see 6.6

7.8 Altering of results via Intrusion

Not possible

Ruled out by 5.1.3. Possible if intrusion is never detected. Intrusion is mitigated with Fail2Ban, see 3.1 and 3.2.

More analysis is required on the feasibility of intrusion and on intrusion detection.

Threats to privacy

7.9 Voter Exposure

This allows the attacker to know the choice of a specific voter, or to obtain some clues about it.

Not possible

Ruled out by the conjunction of 5.2.1 and 5.2.2. However, exposure is possible via shoulder surfing [16].

7.10 Encryption Compromise

Knowing or getting clues about the plaintext of an encryption regardless of who the voter was.

Not possible

Ruled out by 5.2.2.

7.11 Subliminal Channel [29]

Irrelevant

If the voting device is compromised, a subliminal channel is irrelevant.

7.12 Chosen Ciphertext Attack [17]

Not possible

Ruled out by 4.5

7.13 Decryption of Non-anonymized Ballots via Intrusion

Possible. Unlikely

Attackers would have to infiltrate all election authorities.

More analysis is required on the feasibility of intrusion.

Threats to availability

7.14 Denial of Service Attacks

Possible.

The threat is mitigated at both the Registry and the Polling Station with Cloudflare and Fail2Ban, see 3.1 and 3.2.

7.15 Intrusion Compromises Availability

Possible.

Intrusion could stop the election outright. Selective disenfranchisement effected from the Polling Station or Registry is ruled out by 5.1.3. Intrusion is mitigated with Fail2Ban, see 3.1 and 3.2.

More analysis is required on the feasibility of intrusion and on intrusion detection.

Other risks

7.16 Hardware / Software errors

General errors stemming from software or hardware faults

Possible. Unlikely

It is possible that software or hardware errors cause problems ranging in severity from minor defects to general system failures. However, redundancy and testing make these problems unlikely.

8. Alternative trade-offs

Building voting systems is, like any engineering discipline, an exercise in balancing opposing constraints and choosing optimal tradeoffs for the case at hand. We consider some alternatives.

8.1 Privacy and Eligible Verifiability

As described in 6.3.3, published election records contain no link between encrypted ballots and registry information. This precludes eligible verifiability and requires an additional item in the trust model, 5.1.2, which is required to rule out 7.2 (repeating) and 7.3 (ballot stuffing). A different tradeoff could be made, where some element of privacy is sacrificed to achieve full verifiability.

In this particular project such a tradeoff was impossible to make, as publishing census information was out of the question. If one removed this obstacle, publishing this data would achieve eligible verifiability, but would require the assumption of honesty (5.2) to be strengthened to “at least (t - 1) authorities”. (See also Long-term privacy in [19])

Other options are possible within the constraints of maintaining a private census:

- Ballot stuffing protection against the Polling Station via random voter ID’s issued by the Registry
- Ballot stuffing protection against the Polling Station and Registry via eligibility audits performed by trusted third parties (which are trusted with census data)
- Ballot stuffing protection against the Registry via Multi Party Registry (analogous to tally authorities)

8.2 Coercion and Integrity

We note that our system does not satisfy incoercibility (6.6), and although receipt-freeness (6.10) holds partially for the client software, the point may be mute as there may be other ways to provide a receipt. So we can ask, how much value are we getting out of attempting to maintain receipt-freeness in this case? And more importantly, what could be gained from sacrificing this feature of the client software? These ideas are echoed in [23], section 5.1. Revealing the randomness used for encryption would eliminate receipt-freeness completely, but would enable ballot auditing. And ballot auditing would allowing us to remove 5.1.1 from our trust model, a considerable improvement for integrity. Thus an opportunity for an alternative tradeoff, or perhaps even a net improvement, exists here.

9. Recent improvements

Following are some improvements that have been made with respect to the system used in the Podemos online primaries of March 2014.

- Mobile friendly Polling Station UI

  The Polling Station UI has been rewritten to support mobile devices and improve user experience while voting.

- Use of Cloudflare against DOS attacks

  One of the biggest threats to availability is the possibility of a denial of service attacks. These attacks can have adverse effects on participation or stop an election altogether. Responding to the threat of DOS attacks is a very difficult technical challenge that requires sophisticated hardware and software resources. Agora Voting has outsourced this service to Cloudflare[30].

Appenix A. Revisions of this document

March 2014

First version.

June 2014

Updated to reflect recent improvements (section 9).

July 2014

Fixed incorrect references to 5.3.1 (now 5.1.3) in 7.8 and 7.1.5.
Updated section 6.1, 8.2 (census)
Updated sections 3,7 and 9 (cloudflare)

October 2014

Updated section 5.2
Updated section 6.1, removed 8.3 (Eligibility and Practicality)
Updated section 8.1
Updated section 9
Fixed incorrect reference to 5.2.1, now 5.1.2 in 8.1
Removed 7.31 (SMS key theft)

Appenix B. References

[1] http://podemos.info/

[2]
http://www.eldiario.es/politica/primarias-dirigira-Podemos-congreso-fundacional_0_267723491.html

[3] http://en.wikipedia.org/wiki/Plurality-at-large_voting

[5] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In CRYPTO, 1985.

[6] Torben P. Pedersen. A Threshold Cryptosystem without a Trusted Party (Extended Abstract). In EUROCRYPT, 1991.

[7] Torben P. Pedersen. Distributed Provers and Verifiable Secret Sharing Based on the Discrete Logarithm Problem. PhD thesis, 1992

[8] Verificatum -- An efficient and provably secure mix-net
http://www.newton.ac.uk/programmes/SAS/seminars/2012020211451.html

[9] David Chaum, Untracable electronic mail, return addresses, and digital pseudonyms, Comm. ACM, 24, 2 (Feb. 1981); 84-90

[10] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In CRYPTO, 1986

[11] SCHNORR, C.-P. Efficient identification and signatures for smart cards. In Advances in Cryptology - CRYPTO ’89 (1989), G. Brassard, Ed., vol. 435 of Lecture Notes in Computer Science, Springer, pp. 239–252.

[12] A New Implementation of a Dual (Paper and Cryptographic) Voting System - http://www.e-voting.cc/wp-content/uploads/downloads/2012/10/315-329_Farhi_New_Implementation_of_Dual_Voting_System_corrected.pdf

[13] Andrew Regenscheid and Nelson Hastings. A Threat Analysis on UOCAVA Voting Systems Section 5.5 - http://www.nist.gov/itl/vote/upload/uocava-threatanalysis-final.pdf

[14] Eitan Grundland. An Analysis of the Wombat Voting System Model - http://www.grundland.org/An%20Analysis%20of%20the%20Wombat%20Voting%20System%20Model.pdf

[15] Douglas Jones. Threats to Voting Systems - http://homepage.cs.uiowa.edu/~jones/voting/nist2005.shtml

[16] http://en.wikipedia.org/wiki/Shoulder_surfing_(computer_security)

[17] Birgit Pfitzmann and Andreas Pfitzmann. How To Break The Direct RSA-Implementation Of Mixes (1990

[18] Vanessa Teague, Kim Ramchen and Lee Naish. Coercion-Resistant tallying for STV voting - http://static.usenix.org/events/evt08/tech/full_papers/teague/teague_html/

[19] Krishna Sampigethaya, Radha Poovendran. A framework and taxonomy for comparison of electronic voting schemes http://www.ee.washington.edu/research/nsl/papers/JCS-05.pdf

[20] Orhan Cetinkaya. Analysis of Security Requirements for Cryptographic Voting Protocols - http://www.ceng.metu.edu.tr/~corhan/Papers/sreis08.pdf

[21] Steve Kremer, Mark Ryan and Ben Smyth. Election verifiability in electronic voting protocols. - http://www.bensmyth.com/files/Smyth10-definition-verifiability.LNCS.pdf

[22] Michael I. Shamos. Cellphone Vote-Buying. In Developing an Analysis of Threats to Voting Systems: Preliminary Workshop Summary, pp 130 - http://www.nist.gov/itl/vote/upload/threatworksummary.pdf

[23] Ben Adida. Helios: Web-based Open-Audit Voting - http://static.usenix.org/event/sec08/tech/full_papers/adida/adida.pdf

[24] Running mixnet-based elections with Helios - https://www.usenix.org/legacy/event/evtwote11/tech/final_files/Bulens.pdf

[25] Niko Farhi. An Implementation of Dual (Paper and Cryptograhic) Voting System -

http://courses.cs.tau.ac.il/~amnon/Students/niko.farhi.pdf

[26] http://openkratio.org/

[28] https://agoravoting.com/

[29] Ariel J. Feldman and Josh Benaloh. On Subliminal Channels in Encrypt-on-Cast Voting Systems

[30] https://www.cloudflare.com/features-security