An analysis of the security and privacy issues of the Neovote online voting system

Enka Blanchard[1,2,3][0000−0003−1511−8864], Antoine Gallais[2][0000−0002−9457−7112], Emmanuel Leblond[4], Djohar Sidhoum-Rahal[5], and Juliette Walter[6]

[1] CNRS, enka.blanchard@cnrs.fr, https://koliaza.com
[2] Laboratoire d'Automatique, de Mécanique et d'Informatique Industrielles et Humaines, UPHF
[3] Centre Internet et Société, CNRS
[4] Scille SAS
[5] Observatoire des Mutations Institutionelles et Juridiques, Université de Limoges
[6] Unite Live

Abstract. This article provides the first security and privacy analysis of the Neovote voting system, which was used for three of the five primaries in the French 2022 presidential election. We show that the demands of transparency, verifiability and security set by French governmental organisations were not met, and propose multiple attacks against the system targeting both the breach of voters' privacy and the manipulation of the tally. We also show how inconsistencies in the verification system allow the publication of erroneous tallies and document how this arrived in practice during one of the primary elections.

Keywords: Cybersecurity · Electronic Voting System · Privacy · Case Study · Online Voting.

1 Introduction

Voting is commonly associated with the sovereign expression of a voter's will — an expression which multiple theorems from voting theory have shown not to be necessarily aligned with their true desires [28]. We now understand that an important pre-condition for the expression of a voter's will is the privacy of the ballot, which was not always the case [22, 27, 6] and is still debated — e.g. for ballot selfies [20]. However, most individuals' voting experiences happen not in national elections but in small settings such as boardrooms or union meetings — which form the main client base of Neovote. This creates new issues when it comes to privacy, as authority figures can often easily exert direct coercive power if the vote goes against their wishes [4]. The coercive power of managers within a company — and the related potential for vote buying — should not be neglected, as they are central to large-scale voting fraud campaigns such as those in Russia [14, 18]. Thus, if we seek to change or enforce norms that guarantee voters' privacy, smaller-scale votes are an immediate target. The potential for coercion also means that such settings require even more stringent guarantees of

privacy, such as publishing not the full tally (from which one could infer some information) but only the result [15]. Before the Covid-19 pandemic, most votes of this kind still happened with paper ballots. The switch to remote work has made this impossible in many settings, leading to the development and partial adoption of many e-voting systems. Some focus on security and privacy [26], whereas others are purposefully designed with usability and understandability in mind, discarding entirely the question of coercion [7].

This article focuses on one such family of voting systems, created in 2007 by a company called Neovote. This company was relatively discreet until 2017, at which point they started establishing themselves as a market leader (they now announce handling upwards of 10 000 elections per year). Since then they have been used for internal elections by at least 245 companies and institutions[7]. They have also won[8] at least 21 competitive public markets — half of which come from academic institutions — valued up to 1.28M€ each for a total of more than 6.5M€ (some contracts do not specify amounts). More importantly, they were recently chosen to organise 3 of the 5 main primary elections for the French Presidential election of 2022. Those are the primary for Les Républicains (LR, two rounds in December 2021), the Primaire de l'Ecologie (PE, two rounds in September 2021), and the Primaire Populaire (PP, single round using majority judgement in January 2022).

As actors like Neovote take an increasingly central role in both democratic institutions and private organisations, it stands to reason that they should be scrutinised and audited by independent actors. They have in practice mostly received attention from the press, especially with the recent primaries [29]. This included some criticism, mostly focused on the fact that the votes they organised were happening online [30] and required some private information (phone and credit card information), as well as for the fact that they allowed certain people to vote twice by using multiple phones and credit cards [19].The problem of making sure that the remote voter's identity is correct is a central one, to which no solution has yet been found and accepted. However, beyond these issues which apply to all e-voting systems, there remains the question of whether Neovote attains its other security objectives — such as guaranteeing the integrity of the tally or the privacy of voters.

The only other academic work on Neovote was performed by researchers and students from Bordeaux University's computer science department and put online on February 18th, 2022 (as this article was being written). As their university was using Neovote for their internal elections, they had a unique opportunity to study it. They showed multiple vulnerabilities — including some affecting privacy, such as the use of ESMTPS (Extended Mail Transfer Protocol), which allows man-in-the-middle attacks during password recovery [5]. They criticised the weakness of some of the registration procedures, dependent on information considered "private" but that would be available to many colleagues (and triv-

[7] Based on the partial information available on Legifrance, the official French government website for legal information https://www.legifrance.gouv.fr/.

[8] This information was gathered using BOAMP, the French *Official Bulletin for Public Market Offers*, https://www.boamp.fr/.

ially obtainable by many human resources departments), such as full name, place and date of birth, or student/personnel number (shown on some access badges).

Unlike the Bordeaux team, we did not have access to a small-scale vote which we could participate in. Instead, we decided to run a purely observational study on the PP vote where some of us were legitimately registered. We voted as standard citizens without trying to artificially affect the outcome. All the while, we recorded both our actions in screen-capture and what happened in our browsers, keeping a copy of all html files and scripts that were downloaded[9]. One of the authors of this article was initially a whistle-blower who registered for the PE vote and tried to make public some of the information observed during this vote[10]. This author then shared the corresponding files for the verification process, which the rest of us then authenticated.

This article presents three results concerning Neovote's online voting system:

- The voting system does not respect French security and privacy regulations and does not fulfill its own claims, especially when it comes to vote verifiability.
- The lack of end-to-end verification creates an opportunity for both errors and attacks, leading to the temporary publication of an erroneous tally in the Primaire Populaire vote, which we document here.
- The verification process does not give voters any guarantees as to the integrity of the voting process. However, it allows multiple attacks that could deanonymise some or even all voters.

This article starts by looking at the legal and regulatory constraints that apply to the French e-voting ecosystem, and follows with a first set of issues with Neovote concerning transparency and inconsistent claims. We then perform a code comparison between the public Neovote code and the obsolescent and unmaintained asmcrypto.js library, showing that some of the cryptographic primitives used in the first were directly taken from the latter. We follow with the errors we documented in the online tally published on the official website of the PP election. We then look at the verification procedures proposed by Neovote and show how they do not provide any guarantee on the integrity of the procedure, while still allowing an adversary to breach the voters' privacy in multiple configurations. Finally, we look at how the legal system has reacted to these issues (with a focus on France) and conclude on general recommendations to ensure the privacy of voters beyond the strict confines of national elections.

2 Legal and regulatory constraints in France

France strongly differentiates between state-organised votes (referenda and elections) and other kinds of votes (many of which concern internal representation in private companies, unions, and institutions). This also includes votes for primary elections, which are poorly covered by French law [24]. Besides some restricted

[9] The corresponding video, html and script files are available upon request.

[10] He also warned the company as well as relevant institutions (ANSSI and CNIL) before waiting the regulatory three months, talking to us and warning some journalists

experiments on electronic (offline) voting machines, state-organised votes are all paper-based for residents of mainland France [13]. The second kind of vote is thus a place for experimentation, socio-political as well as technological. The Primaire Populaire combined both by having online voting as well as a not-yet-standard evaluative voting system called majority judgement [3].

The main regulations that apply to votes handled by companies like Neovote then come from two sources. The first is the National Commission on Informatics and Liberty (Commission nationale de l'informatique et des libertés, CNIL) who established some guidelines on electronic and online voting, updated for the last time in 2019 [9]. These guidelines define three risk levels dependent on the importance and scale of the vote and the assumed capabilities of adversaries. Each level adds a supplementary set of constraints. We will focus here on five specific ones which are the most relevant to the problems we consider.

- Security objective n°1-07: Ensure the total separation of the voter's identity and the expression of their vote for the whole processing duration.
- Security objective n° 1-11: Ensure that the opening of the ballot box and the tallying of its contents can be verified a posteriori.
- Security objective n° 2-06: Use an information system that puts into place the physical and computational security measures recommended by the publishers of the information system and by the National Cybersecurity Agency of France (ANSSI).
- Security objective n° 2-07: Ensure the transparency of the ballot box for all voters.
- Security objective n° 3-02: Allow all voters to check the transparency of the ballot box using third-party tools.

Neovote states that they are homologated (i.e. have received government certification) to organise votes at the highest risk level. This means that they also have to follow the ANSSI regulations, especially the Selection Guide for Cryptographic Algorithms [1]. This guide restates two common cryptographic statements — derived from Kerckoffs' principle — by discouraging from creating new protocols (2.2.6), but most importantly not trying to reimplement standard tools (2.2.5): "This is why it is imperative to only use libraries which have been tested and which benefit from regular maintenance on their security protocols for any use of cryptographic mechanisms." [1].

If we want to restate in more standard academic language the properties required by the CNIL regulations, we can assimilate n° 1-07 to ensuring the privacy of the vote [2]. Properties n° 2-07 and n° 3-02 correspond to a mix between *cast as intended* and *recorded as cast* (as it implies that voters should be able to check that their vote was indeed recorded, although this is not explicitly stated) [32, 2]. Property n° 1-11 would then be closest to *tallied as recorded*.

One curious detail that we discovered during this investigation is that the constraint of code transparency is often left unsaid in academic verifiable voting, especially when proposing new systems — except for a few recent exceptions [16, 17]. Most E2E-verifiable voting systems are open-source [10, 2], and the availability of both the code and the protocol are often considered a given. However, it

is at best rarely explicitly stated that vote verifiability depends on the openness of the source-code (or at least the openness on the protocols used). However, one can wonder how a system could be verifiable if it integrates some black-box components — or at least, if those components do not have strong constraints on all inputs and outputs. This problem and its implications for the online voting technology ecosystem will be explored in more details in Section 7. Before we get to this, we need to analyse how the different claims made by Neovote relate to these regulations and how the voting system developed fails to comply with them on multiple fronts.

3 Claims and transparency

We can start our analysis of Neovote by its public-facing website. It features a number of claims which are either hard to make sense of or inconsistent, as seen on the two following examples.

- They claim to be homologated by many top-level French institutions (Senate, National Assembly, Ministry of the Interior, Council of State), none of which generally practice homologation. When pressed to document or explain these claims by colleagues, they gave no response [5].
- Some of their technical vocabulary is far from standard and is not defined (geometric models for ballot boxes, random ballot boxes). They also reject certain standard tools and claim not to use any kind of database, as well as no "mélangeurs" (mixers or mix-nets). They also indicate being deployed on SecNumCloud servers (a French certification for cloud security) while also saying they do not use any cloud resources.

One central claim they make is that they develop all their code internally, including a modified Debian[11] OS and a full cryptographic toolset, which goes against the ANSSI guidelines. However, as the next section will show, it also is not correct, and some of their code is taken from external libraries. However, before proving this element, we must address a first issue with the Neovote system: it offers no information on its inner workings. This means that we had no source code, no documentation, no white paper or technical paper on the protocol they use, or even any information on who designed the system or implemented it (besides that some of them were trained in a French engineering school). Neovote claims that this lack of transparency — for example, on how they deal with cyberattacks — is due to legal obligations as they handle "top-secret" information [19] even as their CEO claims that e-voting is legitimate for national elections "as long as the solution is ethical and transparent" [29]. This limits the analyses and security tests that can be made ethically, and as such the following sections only feature what could be gleaned from client-side information.

A second point that compounds with the first issue is that multiple elements make this analysis harder. First, the available code — two Javascript files called

[11] One peculiar element regarding this is that some of the voting archives available from their servers are encrypted with utf8 passwords, and others in iso-8859-1, generally only used on Windows.

by the html of the voting interface — was partially obfuscated. Second, despite our team using multiple browsers and operating systems, no-one managed to directly download the .har files of the information exchanged by the browsers during the experiment (the contents then had to be manually copied and saved in text files). This does not directly indicate malicious intent against external observations and was initially thought to be a bug, but still merits a mention. Finally, both the voting website (with a neovote.fr domain name) and the Primaire Populaire website refused to interact with the wayback machine (archive.org), limiting the ability to get a "neutral" external copy of our observations.

4 Code re-use from asmcrypto

As stated previously and despite their claims to having all their code developed internally, we found multiple examples of code re-use by Neovote. This is not by itself a problem and we are entirely in favour of code re-use, especially in security contexts, when the code is open-source, maintained and frequently audited. The code we initially downloaded from Neovote was not fully minified but partially obfuscated, with variable names and function names replaced by arbitrary strings (with some structure in the name format). This is apparently done each time the scripts are downloaded from the server. However, certain function names (such as *aes_init*) were not fully replaced, probably to address dependencies, and neither were strings shown to the client. This is what allowed the initial discovery of the asmcrypto.js library, available on Github under an MIT license. From then, we could find a certain number of functions copied and transpiled from one project to the other (from typescript to javascript). We'll look at two main examples and why they are relevant to our considerations.

4.1 General copy

The first example we look at is linked to the AES encryption of the ballots. Starting with aes.ts of the asmcrypto.js library[12], we can find on lines 55-96 the code shown on Figure 1b. On the Neovote script n°2 downloaded during the study, lines 542-546 correspond (once expanded) to the code shown on Figure 1c. The three main variable names are colour-coded (in red, purple and blue) to facilitate recognition. Some keywords and structures do not match due to differences between Javascript and Typescript.

The code shown on Figure 1c could have been imported directly from the asmcrypto.js library at compilation time, integrated and transpiled. This would have contradicted some of Neovote's claims — having all code produced internally — but would have followed national security recommendations. However, two problems remain, the main one being the inadequacy of the library used to start with, which is not maintained anymore: the last changes to its cryptographic source files were done in 2018 (a few files for testing and node support were added in 2020 but did not warrant a new release). According to its own

[12] https://github.com/asmcrypto/asmcrypto.js/blob/master/src/aes/aes.ts

```
AES_Encrypt_process
  (data: Uint8Array):Uint8Array
  {if (!is_bytes(data)) throw new
    TypeError("data isn't of
      expected type");
    let asm = this.asm;
    let heap = this.heap;
    let amode = AES_asm.ENC[this.
      mode];
    let hpos = AES_asm.HEAP_DATA;
    let pos = this.pos;
    let len = this.len;
    let dpos = 0;
    let dlen = data.length || 0;
    let rpos = 0;
    let rlen =
       (len + dlen) & -16;
    let wlen = 0;
    let result = new Uint8Array(
      rlen);
    while (dlen > 0) {
      wlen = _heap_write(heap,
        pos + len,
        data, dpos, dlen);
      len += wlen;
      dpos += wlen;
      dlen -= wlen;
      wlen = asm.cipher(
        amode, hpos + pos,
        len);
      if (wlen)
      result.set(heap.subarray(
        pos, pos + wlen),
        rpos);
      rpos += wlen;
      if (wlen < len) {
        pos += wlen;
        len -= wlen;}
      else {
        pos = 0;
        len = 0;}}
    this.pos = pos;
    this.len = len;
    return result;}
```

(a) AES encryption code from asmcrypto.js.

```
export function
  getNonZeroRandomValues
  (buf: Uint8Array)
{getRandomValues(buf);
  for (let i = 0;
    i < buf.length;
    i++) {
    let byte = buf[i];
    while (!byte) {
      const octet =
      new Uint8Array(1);
      getRandomValues(octet);
      byte = octet[0];}
    buf[i] = byte;}}
```

(b) Function from a pull-request adding RSAES-PKCS#1v1.5 to asmcrypto.js.

```
.xTpmDHxL=function
($xTpmppgF)
{if(!xTpmppDL($xTpmppgF)){throw new
TypeError("data isn't of expected
  type");}
var $xTpmpDmV=this.$xTpmpDmV;
var $xTpmppgm=this.$xTpmppgm;
var $xTpmpDLT=xTpmpDNN.xTpmDHNL[this.
  $xTpmpDYs];
var $xTpmppgg=xTpmpDNN.xTpmDHxH;
var $xTpmpDYV=this.$xTpmpDYV;
var $xTpmprNV=this.$xTpmprNV;
var $xTpmppHx=0;
var $xTpmppHs=$xTpmppgF.length||0;
var $xTpmpDLY=0;
var $xTpmpDLL=
   ($xTpmprNV+$xTpmppHs)&-16;
var $xTpmppHg=0;
var $xTpmpDLD=new Uint8Array(
  $xTpmpDLL);
while($xTpmppHs>0){
$xTpmppHg=xTpmppgk($xTpmppgm,
  $xTpmpDYV+$xTpmprNV,
  $xTpmppgF,$xTpmppHx,$xTpmppHs);
$xTpmprNV+=$xTpmppHg;
$xTpmppHx+=$xTpmppHg;
$xTpmppHs-=$xTpmppHg;
$xTpmppHg=$xTpmpDmV.xTpmpDkr(
  $xTpmpDLT,$xTpmppgg+$xTpmpDYV,
  $xTpmprNV);
if($xTpmppHg)
{$xTpmpDLD.set($xTpmppgm.subarray(
  $xTpmpDYV,$xTpmpDYV+$xTpmppHg),
  $xTpmpDLY);}
$xTpmpDLY+=$xTpmppHg;
if($xTpmppHg<$xTpmprNV){
  $xTpmpDYV+=$xTpmppHg;
  $xTpmprNV-=$xTpmppHg;}
else{
  $xTpmpDYV=0;
  $xTpmprNV=0;}}
this.$xTpmpDYV=$xTpmpDYV;
this.$xTpmprNV=$xTpmprNV;
return $xTpmpDLD;}
```

(c) AES encryption code from the Neovote script n°2.

```
var xTpmpDDx=
   function
   (xTpmpDpH)
{xTpmpDpD(xTpmpDpH);
for(var $xTpmprNs=0;
  $xTpmprNs<xTpmpDpH.length;
  $xTpmprNs++){
 var $xTpmpDDW=xTpmpDpH[$xTpmprNs];
 while(!$xTpmpDDW){
  var $xTpmpDDV=
  new Uint8Array(1);
  xTpmpDpD($xTpmpDDV);
  $xTpmpDDW=|\$|xTpmpDDV[0];}
xTpmpDpH[$xTpmprNs]=$xTpmpDDW;}}
```

(d) Equivalent function adding RSAES-PKCS#1v1.5 found in Neovote script n°2.

Fig. 1: Two pairs of matching code fragments from asmcrypto.js (left) and Neovote (right). Within each fragment pair, the main matching variables are consistently coloured.

readme, the library is also optimised for speed instead of security or resistance to side-channel attacks. Finally, it cannot be considered standard by any metric (rigorous testing, maintenance, or even number and size of projects that use it).

4.2 RSAES-PKCS

Let's now look at the second problem, with a second function in the Neovote script that allows the computation of RSAES-PKCS. This function is not found in the asmcrypto.js library but is present in a pull request from 2019, which was never approved or merged with the main branch[13]. This pull request adds support for the protocol RSAES-PKCS#1v1.5, which is now considered obsolescent by ANSSI [1], and for which certain attacks were found as early as 2006 [8]. The code for this protocol can be found in the PE scripts and isn't present in the PP scripts. However, other elements of this pull request also remain in the scripts we observed for the PP. The code on Figure 1b is found in this pull request, whereas the one shown on Figure 1d is found in Neovote script n°2.

The most natural interpretation is that the Neovote developers copied the code from this pull request manually, combining the disadvantages of 1) having external, non-standard code, and 2) a priori not having a method to keep this code up to date despite using obsolescent cryptographic primitives[14]. As stated by the ANSSI regulations, unless one has the highest level of cryptographic competence, one should not develop one's own protocols. Neovote's internal development of its security primitives would only be justified by such an expertise, which is incompatible with the observed ignorance of standard security practices.

5 Publishing the tally

The second central issue we found with the Neovote system is that there is no end-to-end aspect to check either the integrity of each ballot or that of the tally. After the ballot box is opened (with auditors present), the results are apparently given by Neovote to the client who is tasked with publishing them (although Neovote proposes to handle the publication of the vote results on its website). This creates an opportunity not only for errors but for attacks from an adversary willing to discredit the election by having false results published.

We did observe what we believe was just an error[15] for the Primaire Populaire vote which temporarily indicated wrong results, as shown on Figure 2. Because of how hard it was to access the website (with frequent errors), it is hard to know how long that information was shown for. Our observations started after the results had already been published, so we do not know how early this version was present online. However, we measured it to be at least 15 minutes (in the two hours after the official results were published).

[13] https://github.com/asmcrypto/asmcrypto.js/pull/172

[14] Another option would be for them to have a package manager that allows imports from arbitrary pull requests, which they would maintain manually (which is inconsistent with their use of obsolescent primitives).

[15] Since the initial write-up of this article, we were contacted by the election officer for the PP vote who stated that Neovote transferred the results and that it was their responsibility and mistake. It still remains that the lack of verifiability on Neovote's end allowed for this mistake to happen.

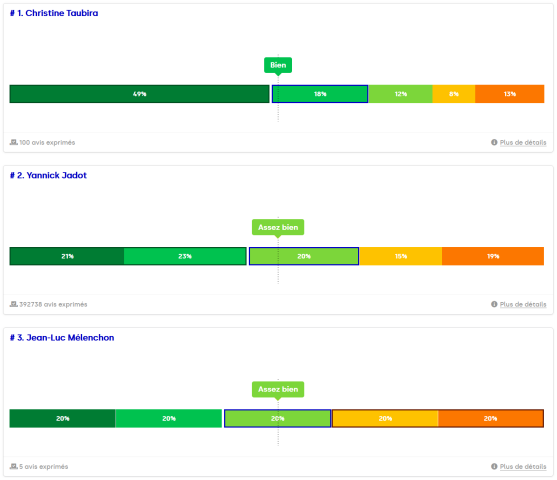

Fig. 2: A screen capture from the Primaire Populaire official results page taken slightly before 8 pm (2 hours after the official results were due to be released, with the website evolving significantly during and after this time).

The screen capture shows three distinct problems. First, the winning candidate's name is misspelled (Christine instead of Christiane Taubira). Second, the tally should be the same for all candidates (as a version of evaluative voting was used), but Christiane Taubira has 100 votes expressed and Jean-Luc Mélenchon has 5. Third, whereas Christiane Taubira's results are in agreement with the official tally, Jean-Luc Mélenchon's results are exactly one vote in each category. Combined with the observed database errors when trying to access the website, this suggests that someone manually fixed the buggy database query by going back to initial placeholders and fixing them by manually adding the votes (out of 100 instead of the real total), but forgot to do so for Jean-Luc Mélenchon.

6 Vote verification

The next attacks we consider focus on the verification system for ballots. Following CNIL recommendation 2-07, Neovote for a certain time had an online system (hosted on verifier-mon-vote.fr/) to verify one's ballot. The public interface allowed one to download the source code, and also to download the ballot box for a given election. Following CNIL recommendation 3-02, it was also possible (although non-trivial) to create one's own script to verify the vote using the ballot box. However, not all votes featured this verification system.

6.1 Availability of the verification process

Although it was used for the PE election, and apparently for the Bordeaux University vote, the source code disappeared at some point before the PP election[16], making the system incompatible with CNIL recommendation 3-02. More importantly, the verification step disappeared entirely for the PP election, against

[16] This was also commented upon in [5].

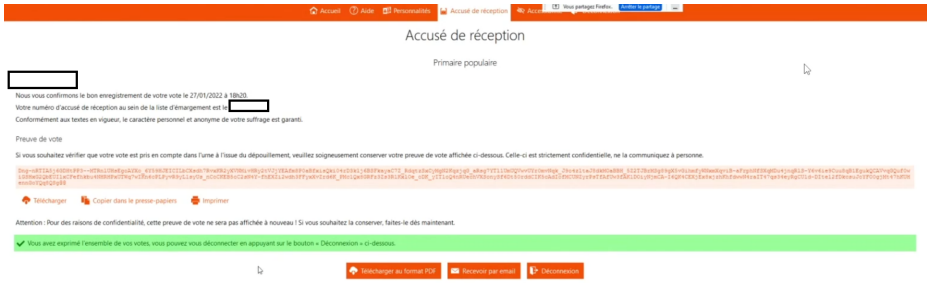

Fig. 3: The interface featuring the "proof of vote", with personal identification removed (this was recorded while in a screenshare with the rest of the team). All the big buttons on the bottom close the page, and downloading the receipt requires clicking on the text below the "proof of vote".

CNIL recommendation 2-07. The voting phase still featured a "proof of vote" as a receipt and stated that this receipt was only going to be shown once for confidentiality reasons. As shown on Figure 3, the system also had some design flaws. The three buttons at the bottom of the page state "download in pdf", "receive by email", and "log out". Clicking on them gives only a partial receipt that shows that one voted but does not indicate the "proof of vote", and it is by then too late to get the full receipt[17]. Even if one managed to download the full receipt, the question remained of what to do with it: no information was ever given (either by email or on any of the voting websites) with instructions on how to verify, and the verification website used for previous votes did not accept any of our receipt/login/password combinations.

As we did not have direct access to the Neovote code, the analyses in the rest of this section and the next section are based on code received from one of the authors who participated in the PE election (and sent us the code before the PP election and before we discovered that Neovote had removed the vote verification system). We have since received other copies of the code downloaded by colleagues, but we initially ascertained the authenticity of this code by :

- checking that the code structure and obfuscation process were compatible with our more recent observations;
- checking that some functions were still present in an identical form except for the obfuscation (such as some of the ones taken from asmcrypto.js);
- checking the old code was compatible with information from [5].

6.2 Attacking the ballot box

We can now consider how the system was deployed for the PE vote — using the code we received. This election had 5 candidates in its first round and each receipt was composed of 5 hashes. Each of those potentially corresponds to a different vote/candidate, which we understand as a privacy measure meant to guarantee that one can't link a receipt to any specific ballot. One central question

[17] Two members of our team made this mistake despite being warned, and only retrieved the full receipt by having screen-recording software active at the time.

is then how those extra hashes are handled, which could be done by adding other ballots' hashes, or by creating decoy hashes not in the ballot box. The hashes were computed by taking one candidate's identifier, appending a random string, encrypting in with the server's public key and then hashing the result using SHA-512. The receipt is then encrypted using AES, but the key is a publicly available constant, so the usefulness of this step is still mysterious[18]. At least one hash was computed locally but the server returned 5 hashes.

When a voter started the verification process, the first step was to download the ballot box from the server. This contained multiple files and metadata, with two files being of particular interest to us. First, *ballot_ data.csv* which contained one encrypted ballot per line, followed by its hash. Then, *extra_ hashes.csv*, which contained a list of hashes, presumably there to protect the privacy of the first voters — who can't rely on other voters' hashes — but which could also contain decoy hashes. The first step when verifying a receipt was to remove from the 5 hashes all hashes present in *extra_ hashes.csv*. The next step was then to check that the remaining hashes were all present in *ballot_ data.csv* (and that they didn't correspond to different election rounds). The next step was to ask the server to decrypt all the ballots (which were encrypted using RSA-3072 on its public key), which took a long time and was failure-prone — and also made the system vulnerable to DDOS attacks. This step then allowed tallying the votes.

Before going into details on how the hashes are treated, we can already find two problems with this verification process. First, the ballot box is not digitally signed by Neovote, and neither are the receipts. This means that we have a simple attack which fools the Neovote verifier, which goes as follows. One could create a fake ballot box and spread it. This box would be like the original with an arbitrary number of hashes moved from *ballot_ data.csv* to *extra_ hashes.csv*, while adding new votes and hashes for any chosen candidate into *ballot_ data.csv*. As the tallying function ignores all the ballots in *extra_ hashes.csv*, this procedure preserves the number of voters, and changes the results arbitrarily. Let's suppose that the receipt has hashes for all candidates and that there is a low number of extra hashes — which is the most reasonable option, as shown in the next subsection. Then by keeping intact all the hashes for the weakest candidate and only discarding other hashes, we can ensure that almost all ballots[19] will have at least one correct hash in the fake ballot box. As at least one hash from the receipt is still present, the verifier will validate the vote.

The second problem is more serious as it casts doubt on the whole verification mechanism. As the receipts are not digitally signed and have a simple structure, one can create a fake receipt that has arbitrary hashes in it. This receipt will show an error with the ballot box that gets detected by the verification algorithm. Thus, there is no way to distinguish an honest voter who detects an error (or fraud) in the system from a dishonest adversary that tries to cast doubt on the integrity of the election. In the absence of unforgeable proofs of malfeasance,

[18] This was also commented upon by [5], who also noticed that there was a single salt used for all voters and publicly available, further reducing its usefulness.

[19] If the receipt hashes are chosen from the last votes cast, there is exactly one ballot that would show wrongdoing. If they are random, the exact probability depends on many factors.

any whistle-blower is then seen as suspicious, and there are no simple ways to resolve the deadlocks within the bounds of the *unaccountable* system [23].

6.3 Deanonymising the votes

As we have seen, one can fake receipts and ballot boxes, minimising any potential usefulness of the verification procedure. However, despite their near-uselessness, the receipts can still be dangerous. This brings us to our main attack, which seeks to destroy any privacy by deanonymising the votes.

Let's now consider how the hashes are generated and which ones go in *extra_hashes.csv*. We have two main possibilities. First, let's suppose that all the extra hashes from the receipt are absent from the ballot box (which would only feature the hash of the real vote). The decoys would then all be in *extra_hashes.csv* and, by communicating their receipt before the opening of the ballot box (which is only done after the voting period ends), the voter could prove how they voted, as only one ballot from the ballot box matches a hash on their receipt — the decryption keys being available after the election.

Let's now suppose that some but not all hashes were included in the *extra_hashes.csv* list, which would mean that at least some of the hashes in the receipt correspond to other voters' ballots. The voter might not be able to prove how they voted, but could at least prove how they did not vote. Thus, unless one wants to allow some coercion attacks, we can now suppose that the other hashes are generally not included in *extra_hashes.csv* — which is consistent with our observations for the EELV vote. The voter's anonymity would then supposedly be protected by the fact that they have the hashes of five real ballots[20]. Let's show two problems with this approach.

First, if all (or nearly all) receipts are kept private, the organisers can commit a clash attack and assign n ballots for a candidate to kn voters. Most of the voters from one candidate will then have duplicates, but none of them will be able to notice unless they share their hashes with other voters. With $k = 2$, supposing all voters decide to check with one other trusted voter, the probability of one of them finding a duplicate tends towards $1 - e^{-1/2}$ [25]. However, this assumes that all voters check with one other person. If only a fraction c checks, the probability[21] of finding a duplicate drops to at most $\frac{c}{2} + O(\frac{1}{n})$. This is rare for a "verifiable" voting system, where the norm is that attacks that modify a constant proportion of ballots are discovered with probability $1 - C^{-k}$, where k is the number of ballots checked and C's value depends on the protocol and the proportion of modified ballots. Neovote also goes against usual verifiable voting norms by dissuading voters from sharing their receipts (even after the ballot box is opened), which makes this attack possible in practice.

Let us now suppose that, despite the organisers' recommendations, most or even all receipts become public. The question that matters is then how the hashes are distributed when voters obtain their receipts. Similarly to what happens with the extra hashes above, if the receipt's hashes do not show at least one vote for

[20] Except potentially for the first few voters, who have the hashes of the initial extra ballots.

[21] As the probability for each couple is $\frac{1}{2n-1}$, by union bound with $c \times n$ couples, we obtain $\frac{c}{2} + O(\frac{1}{n})$.

each candidate, they can be enough to partially uncover how someone voted[22]). Thus, to avoid the attacks above, we can assume that each receipt has n hashes, the correct one plus one from each other candidate. We will assume here that we have $n = 2$ candidates[23]. We can then consider two main cases:

1. The hashes are taken from the last set of votes cast for each candidate.
2. The other hashes are taken randomly, guaranteeing a hash per candidate.

Let's look at the first case. If we receive a set of receipts, we will observe sequences of receipts of the form $(x_1, y), (x_2, y), ...(x_k, y)$, where x_i are distinct hashes for candidate x and y is a constant hash for y. By finding the receipts (x_1, y') and (x_k, y''), we can know that the first sequence of receipts corresponds exclusively to votes in favour of x (and the two extra receipts are for y).

Let's now suppose that the hashes are taken randomly. Then certain hashes will be present in a single receipt and will correspond to that receipt's real ballot. If one manages to have access to all receipts, they will then manage to deanonymise a constant proportion of ballots for each candidate. This proportion can be shown to be in expectation $e^{\theta(-B/A)}$ where A is the number of votes for the candidate and B the number of votes for other candidates[24]. Moreover, a coercer for candidate A just has to look at the receipts of people who say that they voted for A. If k of them share the same hash for A, all but one of them are lying. Even assuming that the coercer can only get half of all receipts for A, a voter willing to vote for another candidate will get caught with probability at least 0.5, in which case the coercer will know that there is at least an 0.5 probability that the voter did not follow the voting orders.

7 Discussion

The findings in the previous sections show a non-negligible number of vulnerabilities and failures to respect cryptographic best — or even common — practices. In other contexts, such failings could potentially be handled by competition between service providers. However, the national aspect of the votes handled and the fact that the practices observed could lead to the privatisation of some democratic practices means that they fall under the public purview. In such a situation, we should expect the state to provide regulation and enforcement.

However, an analysis of the few legal decisions concerning Neovote shows that French courts do not fully address the weaknesses of electronic voting systems. The Administrative Court of Appeal of Marseille cancelled in 2019 a vote organised by Neovote, criticising the "réassort" protocol that allowed voters to renew their secure identification, as it "did not offer a protection of the voter's privacy at a level equivalent to other voting methods" [11]. The other problems shown above were not mentioned (not only the technical ones which were not necessarily known, but also the lack of transparency).

[22] We are assuming that the system is candidate-agnostic — which Neovote apparently is — in that we don't consider the possibility that it could force the presence of a given candidate (who could have more coercive power) on the receipt.

[23] The attacks below can also work for $n > 2$ albeit with smaller sequences and more ambiguity. Statistically, they still allow the deanonymisation of at least a constant fraction of voters.

[24] The proof goes by linearity of expectation, using the fact that the expected number of single-hash receipts for A is $A(1-\frac{1}{A})^B$. We obtain $A \times e^{B \times log(1-\frac{1}{A})}$, which is $A \times e^{\theta(-B/A)}$ using the Taylor expansion for the logarithm.

A more worrying case stems from a decision taken by the Cour de cassation (one of the French supreme courts) [12]. Multiple employees were seeking the cancellation of a vote under the reasoning that the independent auditing did not focus on the vote itself or the source code that was used, but on a theoretical version of the protocol. The Court considered that the expertise *in abstracto* made before the vote satisfied the legal rules concerning the necessity of independent auditing. A single audit was then considered sufficient for all votes organised with the same voting system and only a substantial modification of the system would warrant a new audit. However, this ignores the crucial fact that only systematic auditing — which would at the very least check digital signatures — would allow the discovery of any modifications in the system (whether they are substantial or not). This also ignores the fact that errors can happen (as with the publication of a wrong tally) and that certain attacks do not require any code modification to take place, only a change of context.

Finally, there is a very real concern that the privacy requirements considered by the Court are not realistic in a world where most citizens use and share part of their lives on social media. In its decision from 2021, the Court did not reject the framework of a simple 2-factor authentication system to send voters their voting information, even with a centralised system that could potentially deanonymise the voters. One element of the decision reads as follows: "It was also pointed out that [a voter] could only obtain the new password of another employee of the company as the latter had given him his birth place, meaning that this process, originating in an identity theft with [the accomplice's] consent, did not demonstrate a failure in the protocol". Thus, the Court did not denounce the reliance on the supposed secrecy of information that are easily obtainable by one's superiors and colleagues. This interpretation is increasingly at odds with best practice, especially in the wake of scandals like Cambridge Analytica's [21].

A last element is that, as pointed by our colleagues [5], there is little existing regulation for verifiable voting systems, and few recourse methods. 392738 people voted in the Primaire Populaire, and all were offered the possibility of verifying their vote. The fact that there was no scandal when this turned out to be impossible has two alternative explanations. First, the people trying to verify their vote might have been so few that they had no visible impact (which is worrisome for all verifiable voting systems). Second, the people might have been numerous but unable to trigger a political response.

8 Conclusion

We have used two of the latest high-profile electronic votes in France as a case study to show not only the weaknesses in the system used but most importantly the inadequacy of the regulatory system when faced with highly technical systems, even as those systems threaten to encroach on the roles traditionally played by institutions directly accountable to the public. To be sure, these problems are neither new nor restricted to France, and bring to mind previous experiments such as the Estonian e-voting experiment [31]. We must state clearly that we have no evidence of any actual wrongdoing that sought to affect the results of any of the primary elections we analysed. However, the current conditions cannot

guarantee the integrity of the votes, and we cannot have any confidence that if there had been an attack, evidence would have been available (independently of whether it would have been made public). Indeed, despite claiming they strived for transparency, the fact that wrong results were temporarily available online was not made public by the Primaire Populaire. The PP official responsible for the vote at the time contacted us to state that the *official results* were made public through a press conference, which they considered sufficient as the online version was for convenience and had no legal value.

We are not showing these vulnerabilities to advocate for a ban on all electronic voting systems, as some of them can be tailored to certain use-cases where security or privacy are secondary concerns. They can also be useful as teaching tools, especially when it comes to introducing voters to verifiable voting. However, we are concerned by the use of unregulated and unsecure voting tools in public institutions, which if any scandal were to happen could further reduce public trust not only in electronic voting — which might well be deserved — but in democratic practices in general. It then appears necessary to develop better guidelines — at national and supranational levels — on how public institutions should use such technologies. These should address both verifiability and transparency — including the openness of the protocol and the source code. We should also explore how to level the playing field between systems developed for free by public institutions (such as Helios or Belenios), which are often poorly marketed, and products sold by for-profit companies (whose marketing claims might not be checkable as anyone trying such endeavour faces IP violation lawsuits).

References

1. Guide de sélection d'algorithmes cryptographiques. Tech. rep., Agence Nationale de la Sécurité des Systèmes d'Information (2021)
2. Ali, S.T., Murray, J.: An overview of end-to-end verifiable voting systems. Real-World Electronic Voting pp. 189–234 (2016)
3. Balinski, M., Laraki, R.: Instaurons le «jugement majoritaire». Commentaire (2), 413–415 (2016)
4. Barrett, R.W.: Elephant in the boardroom: Counting the vote in corporate elections. Valparaiso University Law Review **44**, 125 (2009)
5. de Barros, F., Gergouil, T., Grelard, R., Thibault, S.: Analyse de systèmes de vote électronique. Master's thesis, Université de Bordeaux (Feb 2022)
6. Blanchard, E.: Usability: low tech, high security. Ph.D. thesis, Institut de Recherche en Informatique Fondamentale, Université Sorbonne Paris Cité (2019)
7. Blanchard, E., Robucci, R., Selker, T., Sherman, A.T.: Phrase-verified voting: Verifiable low-tech remote boardroom voting: How we voted on tenure & promotion cases during the pandemic. Cryptologia pp. 1–35 (2021)
8. Bleichenbacher, D., May, A.: New attacks on rsa with small secret crt-exponents. In: International Workshop on Public Key Cryptography. pp. 1–13. Springer (2006)
9. CNIL: Délibération n°2019-053 du 25 avril 2019 portant adoption d'une recommandation relative à la sécurité des systèmes de vote par correspondance électronique, notamment via internet. Tech. rep., JORF (2019)
10. Cortier, V., Gaudry, P., Glondu, S.: Belenios: a simple private and verifiable electronic voting system. In: Foundations of Security, Protocols, and Equational Reasoning, pp. 214–238. Springer (2019)

11. Cour Administrative d'Appel de Marseille, 5ème chambre: Décision n°19ma03754 du 16/12/2019
12. Cour de cassation civil, Chambre sociale: Décisionn°20-17.073 du 24/11/2021
13. Enguehard, C., Graton, J.D.: Machines à voter et élections politiques en France: étude quantitative de la précision des bureaux de vote. Cahiers Droit, Sciences & Technologies **4**(4), 159–198 (2014)
14. Frye, T., Reuter, O.J., Szakonyi, D.: Hitting them with carrots: Voter intimidation and vote buying in Russia. British Journal of Political Science pp. 1–25 (2018)
15. Groth, J.: Efficient maximal privacy in boardroom voting and anonymous broadcast. In: Juels, A. (ed.) Financial Cryptography. pp. 90–104. Springer (2004)
16. Haenni, R., Dubuis, E., Koenig, R.E., Locher, P.: Chvote: sixteen best practices and lessons learned. In: International Joint Conference on Electronic Voting. pp. 95–111. Springer (2020)
17. Haines, T., Roenne, P.: New standards for e-voting systems: Reflections on source code examinations. In: International Conference on Financial Cryptography and Data Security. pp. 279–289. Springer (2021)
18. Harvey, C.J.: Changes in the menu of manipulation: Electoral fraud, ballot stuffing, and voter pressure in the 2011 russian election. Electoral Studies **41**, 105–117 (2016)
19. Horn, A.: Le vote en ligne de la primaire populaire est-il sécurisé ? Numerama (2022)
20. Horwitz, D.A.: A picture's worth a thousand words: Why ballot selfies are protected by the first amendment. SMU Science and Technology Law Review **18**, 247 (2015)
21. Isaak, J., Hanna, M.J.: User data privacy: Facebook, cambridge analytica, and privacy protection. Computer **51**(8), 56–59 (2018)
22. Kinzer, B.L.: The un-englishness of the secret ballot. Albion **10**(3), 237–256 (1978)
23. Küsters, R., Truderung, T., Vogt, A.: Accountability: definition and relationship to verifiability. In: Proceedings of the 17th ACM conference on Computer and communications security. pp. 526–535 (2010)
24. Levade, A.: Le droit des primaires: règles, contrôle, finances, sanctions. Pouvoirs (3), 99–109 (2015)
25. Margolius, B.H.: Avoiding your spouse at a bridge party. Mathematics Magazine **74**(1), 33–41 (2001)
26. McCorry, P., Shahandashti, S.F., Hao, F.: A smart contract for boardroom voting with maximum voter privacy. In: Kiayias, A. (ed.) Financial Cryptography and Data Security. pp. 357–375. Springer International Publishing, Cham (2017)
27. McKenna, M.: Building "a closet of prayer" in the new world: the story of the "Australian ballot". Menzies Centre for Australian Studies (2001)
28. Reny, P.J.: Arrow's theorem and the gibbard-satterthwaite theorem: a unified approach. Economics letters **70**(1), 99–105 (2001)
29. Souffi, E.: La primaire écologiste, vote chez les Républicains... Neovote, l'entreprise championne des scrutins virtuels. Le Journal du Dimanche (2021)
30. Vasseur, V.: Primaire écolo, congrès LR : dans les entrailles de Neovote, spécialiste du vote électronique. France Inter (2021)
31. Vinkel, P.: Remote electronic voting in Estonia: legality, impact and confidence. TUT Press (2015)
32. Volkamer, M., Spycher, O., Dubuis, E.: Measures to establish trust in internet voting. In: Proceedings of the 5th International Conference on Theory and Practice of Electronic Governance. pp. 1–10 (2011)