Berner Fachhochschule (BFH), CH-2501 Biel, Switzerland

Examination of the Swiss Post Internet Voting System

Releases 1.3.2 (July 2023) and 1.3.3 (October 2023)

(with Addendum on Sub-Release 1.3.3.2)

Rolf Haenni, Reto E. Koenig, Philipp Locher

December 22, 2023

On behalf of the Federal Chancellery

1. Introduction

This examination report discusses some specific technical topics of the Swiss Post e-voting system that were affected by the changes included in the October 2023 release of both the documents and the source code. As such, it provides an addendum to previous reports that we submitted to the Federal Chancellery (FCh) earlier this year. These documents were published on the FCh's web site in March and July 2023, together with reports from other experts.[1] For readers of this report, it is important to consider it in the context of all previous reports, including the original ones from March 2022:

[1] Rolf Haenni, Reto E. Koenig, Philipp Locher, Eric Dubuis. *Examination of the Swiss Post Internet Voting System – Scope 1 (Cryptographic Protocol)*, March 28, 2022.

[2] Rolf Haenni, Reto E. Koenig, Philipp Locher, Eric Dubuis. *Examination of the Swiss Post Internet Voting System – Scope 2 (Software)*, March 28, 2022.

[3] Rolf Haenni, Reto E. Koenig, Philipp Locher, Eric Dubuis. *Re-Examination of the Swiss Post Internet Voting System – Scope 1 (Cryptographic Protocol) and Scope 2 (Software)*, Version 1.0.2, February 23, 2023

[4] Rolf Haenni, Reto E. Koenig, Philipp Locher, Eric Dubuis. *Re-Examination of the Swiss Post Internet Voting System — Releases 1.2.3 (February 2023) and 1.3 (April 2023), with Addendum on Version 1.3.1*, June 30, 2023.

Many of the findings and issues discussed in these reports have been addressed in corresponding updates released since the beginning of 2023. However, there is still a number of open issues and pending recommendations. By listing all previous examination reports, we want to make sure that the current report is understood as a followup to our previous reports.

1.1. Purpose and Goals of Mission

We have been assigned with this supplementary examination task in September 2023 by the Federal Chancellery, and we received more detailed information about our mission on October 12, 2023. The start of the mission was fixed to October 30, the announced release date for the new Version 1.3.3, and the deadline for submitting our report was fixed to November 17 (and later pushed backed to November 21). The relatively short examination period was justified by "*a very limited number of modifications related to write-ins, voter authentication, and [...] improvements of the Voter-Portal*". Two other minor modifications related to the maximum size of supported voting options and the alphabet used for the start voting key were announced a few days before the start of our mission.

The examination has been conducted jointly by the listed authors from the Bern University of Sciences and independently of any other group of people.

[1] See https://www.bk.admin.ch/bk/de/home/politische-rechte/e-voting/ueberpruefung_systeme.html

1.2. Overview of the Current Release

To conduct our examination of the latest release, we downloaded the following updated specification document from the public repositories on gitlab.com:[2]

- [SysSpec] *Swiss Post Voting System – System Specification*, Version 1.3.2, Swiss Post Ltd., October 23, 2023

As in earlier releases, a special version of this document was given to the examination experts with all the changes highlighted, and a summary of the changes is given in the document's revision chart and the repository's CHANGELOG.md file. Generally, the number of relevant changes in this document is very moderate.

As officially announced, all other specification documents remained unchanged since the last two releases in April and June 2023. Here is the complete list of the current versions of all other specification documents:

- [CryptPrim] *Cryptographic Primitives of the Swiss Post Voting System – Pseudocode Specification*, Version 1.3.1, Swiss Post Ltd., June 15, 2023
- [VerSpec] *Swiss Post Voting System – Verifier Specification*, Version 1.4.1, Swiss Post Ltd., June 16, 2023
- [ProtProofs] *Protocol of the Swiss Post Voting System – Computational Proof of Complete Verifiability and Privacy*, Version 1.2.0, Swiss Post Ltd., April 19, 2023
- [ArchDoc] *E-Voting Architecture Document*, Version 1.3.0, Swiss Post Ltd., April 14, 2023

The new software release was published on October 26, 2023. We received e-mail announcements from Swiss Post with pointers to corresponding repositories on October 27 and October 30, respectively. The new release was announced as *Release 1.3.3*, which apparently refers to the current versions of the main two components e-voting and crypto-primitives (see GitLab histories in Figure 1). The components e-voting-libraries, verifier and data-integration-service were released as Versions 1.3.4, 1.4.3 and 2.7.2, respectively. To avoid problems in this document related to this inconsistent version numbering policy, we will generally refer to the current software version as the *October release*. Similarly, we refer to the versions that we examined earlier this year as the *February release* [3] and the *June release* [4].

Notice that Swiss post referred to the June version as *Version 1.3.1*, which implies that an intermediate *Version 1.3.2* must have been released in the meantime. In fact, on July 22, we received from Swiss Post an update announcement for Version 1.3.2 (to which we will refer as the *July release*), but we were never asked to conduct an examination. Also, in the instructions that we received from the Federal Chancellery for conducting an examination of the October release, no complementary instructions were included to also examine the July release. However, we also inspected the changes introduced in the July release (see Subsection 2.4).

[2] See https://gitlab.com/swisspost-evoting

1.3. Changes Since the June Release

In Figure 1, we give an overview of the GitLab commit histories of corresponding project repositories' **master** branches. It shows that all components have been updated in the new release. It also shows that internal sub-versions were created regularly, approximately once every 1–2 weeks. To conduct our analysis, we mainly looked at the latest versions released on October 26 (red) and compared them to the versions released on June 15 (blue) and July 22 (green).

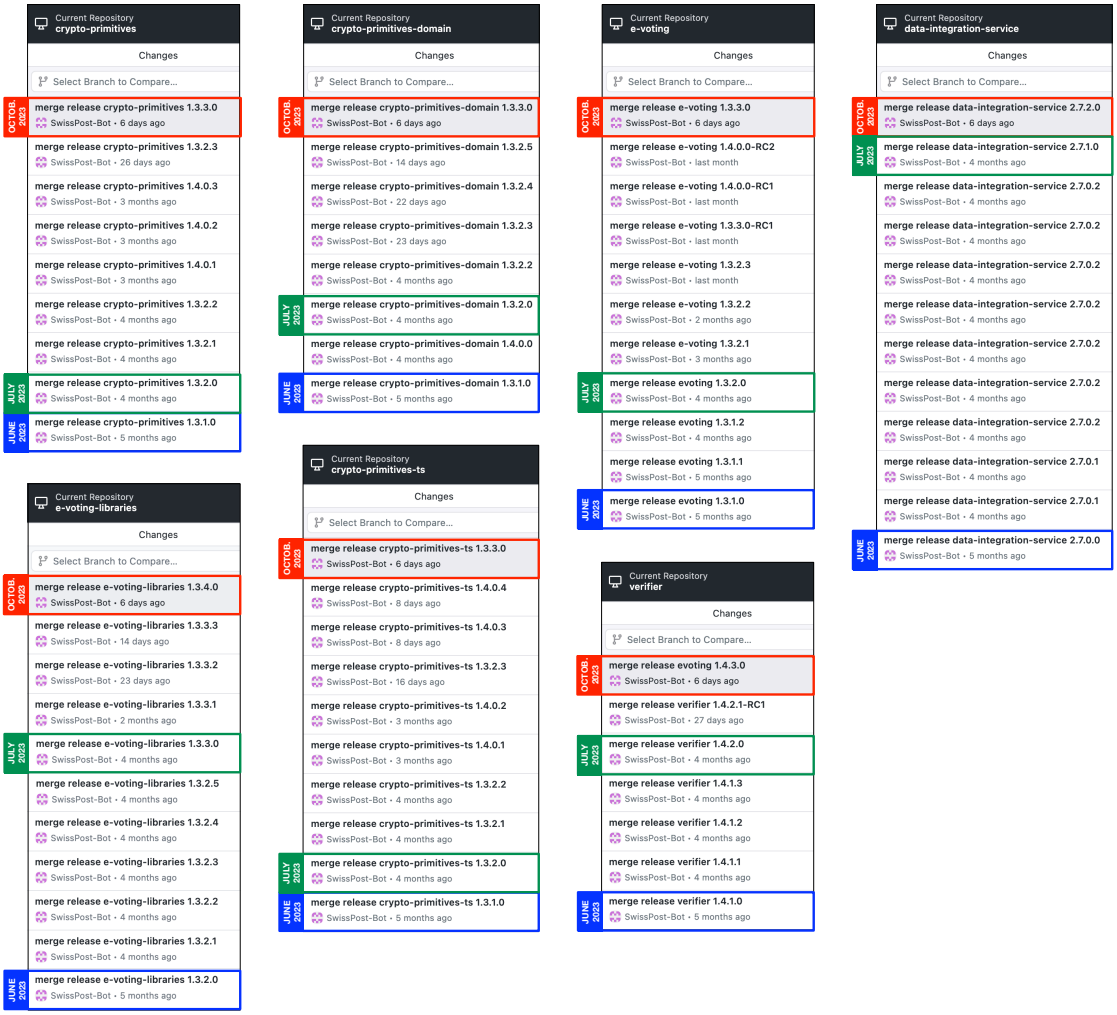

Figure 1: Commit histories of the GitLab projects since the June release.

1.3.1. July Release

According to the above-mentioned e-mail announcement on July 22, no changes were made to the specification documents of the July release. Swiss post called it a "*maintenance patch*" of the code base, that does not touch the cryptographic protocol:

> "*[...] the update contains no changes to the specification or proof documents. Version 1.3.2 is mainly a maintenance patch that fixes a few bugs and some isolated improvements that do not impact the cryptographic protocol.*"

As in earlier releases, relevant changes were listed for all components in corresponding CHANGELOG.md files. Table 1 shows a summary of the entries from these files. No relevant changes (except for updated dependencies) were listed for the crypto-primitives, crypto-primitives-ts, and verifier components. In the e-voting component, in addition to the modifications listed in the CHANGELOG.md file, we discovered a critical undocumented modification (see discussion in Subsection 2.4.2).

Component	Changes listed in the component's CHANGELOG.md file
crypto-primitives-domain	• Added the `electionEventId` instance field to the hashable form method in `SetupComponentPublicKeysPayload` (resolves issue [...] on GitLab).
e-voting	• Added an error message in the Voter-Portal in case of desynchronized system time. • Increased the minimum password size of the import/export password to 24 characters. • Added parallelization and a retry mechanism to the compute step in the configuration phase. • Introduced a transient CONFIRMING voting card state in the voting server. • Fixed a bug in the xml-signature tool when handling keystores without a signing key pair. • Strengthened the exactly-once processing in the control components. • Minor bug fixes and improvements in the Voter-Portal.
e-voting-libraries	• Improved the generation of the eCH-0110.xml tally file to account for invalid votes in case of votes for lists without candidates. • Added the field `isUnchangedBallot` to the produced eCH-0222.xml representing the raw votes. • Added a class for validating passwords to e-voting-libraries. • Fixed a bug that required all keystores to have a signing alias.
data-integration-service	• Reuse `electionIdentification` as `electionGroupIdentification` if the `electionGroup` has no identification and contains a single election. • Provide a warning if the `evotingTestBallotBoxesFromDate` is after the `evotingFromDate`. • Provide a splitter for generating multiple files per ballot box. • Replace incumbent text `"Bisher"` with `"bisher"`.
All components	• Updated dependencies and third-party libraries.

Table 1: Overview of the changes announced for the July release (Version 1.3.2).

1.3.2. October Release

The notice given to the experts on October 27 states that there have only been minor modifications made to the code and documentation. As for the July release, Swiss Post calls it a "*maintenance patch*":

> "*Release 1.3.3 is mainly a maintenance patch and incorporates some lessons from the two first productive election events. Notably, some voters had issues with authentication due to unsynchronized system clocks. To mitigate this, we've expanded the allowable time step and added a permissible forward time step. Additionally, to reduce voter errors when entering the start voting key, we refined the start voting key alphabet, substituting* `"l"` *and* `"o"`*. We also enhanced our security advices [...], offering clearer instructions for verifying the voting client's instruction, including the verification of the voting website's HTML file.*"

In a separate communication, a bug fix related to managing the configuration of several simultaneous write-in elections is explained as follows (translated from French):

> "*As far as write-ins are concerned, no changes to the protocol or specifications are required. However, we fixed a bug that prevented us from managing configurations involving several write-in elections. A targeted correction in a single class of our* e-voting-libraries *component was enough to resolve this problem.*"

Again, relevant changes were accurately listed in corresponding CHANGELOG.md files. Table 2 shows a summary of the relevant entries from these files. No relevant changes (except for updated dependencies) were listed for the crypto-primitives, crypto-primitives-ts, and verifier components. However, in the case of the crypto-primitives and crypto-primitives-ts components, we realized that quite some changes were made to the code since the releases in June and July. This can be seen in the GitLab history of Figure 1, which shows the existence of several sub-versions between Version 1.3.2 (July) and Version 1.3.3 (October). Surprisingly, some of these sub-version refer to "merge commits" of Versions 1.4.0.1 through 1.4.0.4 (presumably developer builds of the next major release).

By inspecting the modifications made in these sub-versions, we discovered that 15 Java and 8 TypeScript files (plus a greater number of files with test code) had been modified since the July release, but that these changes have been reverted in later sub-versions. In other words, we can confirm that the current versions of crypto-primitives and crypto-primitives-ts have not been changed since the June release. However, we found it quite confusing seeing so many temporary changes in the Git history of the master branch for no obvious reason. In this regard, it is unclear whether these merge commits were intentional or not, and if yes, what Swiss Post wanted to achieve with them.

Component	Changes listed in the the component's CHANGELOG.md file
crypto-primitives-domain	• Increased the maximum number of supported voting options from 3,000 to 5,000. • Adapted the validation to the new Start Voting Key alphabet.
e-voting	• Increased the voter authentication time step to 300 seconds, added a forward time step, and improved the error message for desynchronized clients. • Replaced two characters in the alphabet of the start voting key for increased usability by using the new GenRandomSVK algorithm. • Implemented a mechanism to recover gracefully from a partially failed compute operation in the configuration phase. • Enhanced the validation process for inaccurately addressed messages within the control components' mixing operations. • Fixed a bug that prevented the mixing of very large ballot boxes. • Reduced the default value for cache entries in the fixed-base exponentiation table (performance optimization) for better stability. • Voter-Portal: Extended the configurability of the legal terms page and the help menu. • Voter-Portal: Added the possibility to use hyperlinks in the support address. • Voter-Portal: Display the question number next to the question and improve the display of variant ballots. • Voter-Portal: Resolved two browser-specific issues that prevented the proper display of the Start Voting Key.
e-voting-libraries	• Fixed a bug in the correct creation of the tally files when having multiple elections with write-ins. • Defined the usability-optimized alphabet for the Start Voting Key.
data-integration-service	• Enforce the presence of the `referenceOnPosition` field for each candidate. • Add a check that each election contains an empty list. • Prevent configurations where two different counting circles have the same ID.
All components	• Updated dependencies and third-party libraries.

Table 2: Overview of changes announced for the October release (Version 1.3.3).

2. Summary of Findings

Given the short time frame for this assessment, we restricted our analysis to a few specific subjects for which we noticed modifications in the source code. One of them is the recurrent examination of the voter authentication process (Subsection 2.1). Another subject is the Voter-Portal (Subsection 2.2), which usually is not our primary concern, but which may also have an impact from a security standpoint. A third subject is the "Exactly-Once Processing" of the submitted ballots (Subsection 2.4.1), which contained a flaw that we overlooked in our analysis of an earlier mission. Another subject that came up while looking at the July release is the JavaScript configuration of the voting client (Subsection 2.4.2), which used to execute the critical code in a Web Worker, but this configuration was subsequently changed twice without announcement.

2.1. Voter Authentication

Voter authentication has been the subject of extensive discourse over an extended period. In early system specification documents, information about the authentication method has been missing entirely, until the December 2022 release added a technical description of the process to Section 5.1. In Appendix B.3.3 of our report from Febrary 2023 [3], we condemned the approach as overly convoluted for no apparent benefit, and we noted that the implementation differed from the specification. We advised removing the implementation because it appeared like an unnecessary Scytl remnant.

Swiss Post completely redesigned and re-implemented the authentication process. The current version uses time-based one-time passwords (TOTP) as specified in RFC6238. TOTP's major objective is to mitigate the risk of replay attacks. For this, it defines a time interval of usually $\mathtt{T} = 30$ seconds, after which the one-time passwords are updated. The current version of the protocol increases this interval from $\mathtt{T} = 30$ seconds to $\mathtt{T} = 300$ seconds, by changing Line 5 of Algorithm 5.1 (GetAuthenticationChallenge) and Line 2 of Algorithm 5.2 (VerifyAuthenticationChallenge) accordingly. Furthermore, the time window for accepting a password has been extended from $[\mathtt{TS} - \mathtt{T}, \mathtt{TS}]$ to $[\mathtt{TS} - \mathtt{T}, \mathtt{TS} + \mathtt{T}]$, i.e., it now tolerates one backward and one forward out-of-sync time interval ($\mathtt{TS}$ denotes the current system time). Algorithm 5.2 has been extended accordingly. We confirm that these changes have been incorporated into the source code, see Lines 80–82 of the Java class `VerifyAuthenticationChallengeAlgorithm` and Line 59 of the JavaScript file get-authentication-challenge-algorithm.js. This results in a total time interval of 10 minutes instead of the usual 30 seconds.

To the best of our knowledge, extending the TOTP time interval to 10 minutes is a result of a small percentage of voters who were unable to cast their vote in the October 2023 parliamentary election due to out-of-sync system clocks on their computers. While modern computers synchronize their system clocks quite regularly, for example by automatically connecting to a Network Time Protocol (NTP) server every 20 minutes, it is still a matter of having the right system settings to do so properly. Therefore, the occurrence of timing problems resulting from unsynchronized system clocks is not unexpected, given the diverse range of machines and systems employed by voters in a real-world election. It is evident that augmenting the TOTP time interval by a factor of

20 diminishes the likelihood of encountering such issues. However, it remains uncertain if this adjustment is adequate for entirely mitigating the problem.

To prevent further cases of voters unable to cast a vote, one could enlarge the TOTP time interval even further, to one hour, multiple hours, one day, etc., but this would entirely undermine TOTP's general purpose of offering a counter-measure to replay attacks. As a result, we reiterate our earlier proposal to call the approach itself into doubt. In our report from June 2023, we concluded our analysis of this subject by the following comment [4, Section 3.1]: "*TOTP is a technique for generating password that expire quickly, which prevents attackers to use stolen password over a long time period. We do not see such a scenario in the given application context. From our perspective, it seems that a technology is used without having a good reason to do so.*" Note that other auditors reached similar conclusions in their reports.

Swiss Post defended their decision to preserve the present TOTP-based authentication method in response. In our understanding, their argument consists of four different points. First, they refer to potential future enhancements that are not specified in detail. Second, they explain that the voting server should block invalid requests instead of forwarding them to the control components (referring to "good design principles"). Third, they state that the system needs to be robust against network man-in-the-middle attackers. Fourth, they underline the importance of time stamps to prevent replay attacks.

> "*[. . .] proposed simplifying the voter authentication protocol by removing the time stamp element. In response to these suggestions, Swiss Post plans to explore potential enhancements for the voter authentication protocol, such as incorporating the message in the authentication challenge, for e-voting release 1.4 scheduled in 2024. However, we still believe that it is a good design principle to let the voting server validate requests before forwarding them to the control components. The system needs to be especially robust against network attackers sitting in between the voting client and the voting server. Removing the time stamp aspect of the voter authentication protocol would facilitate replay attacks.*"

None of these points adequately addresses our concerns. For example, given that the voting client communicates over HTTPS with the voting server, man-in-the-middle and replay attacks are already excluded with very high probability once the TLS channel is established. It would therefore be sufficient to verify the presence of a legitimate start voting key $\texttt{SVK}_{\texttt{id}}$ at the beginning of the vote casting process, i.e., right after the HTTPS connection has been established. This verification needs to be conducted exactly once to ensure the legitimacy of the established session between the voting client and voting server and to protect the control components from receiving invalid requests from someone else not knowing the start voting key. In such a setting, TOTP does not offer any further benefits, because it does not prevent the untrusted voting client from sharing the start voting key (and other secrets) with a potential adversary, who could then establish its own connection to the voting server.

We conclude this discussion with two remarks. First, we repeat our standpoint that a proper threat model seems to be missing to justify the use of TOTP-based authentication, i.e., the attack scenarios that this method should prevent in the given context are still very unclear. We also have the impression that security considerations are being

mixed up across different communication layers, which are given by the protocol messages defined by the cryptographic protocol and the underlying network infrastructure needed to execute the protocol in practice. To optimize conceptual clarity, we recommend to maintain the separation of these layers as strictly as possible.

Second, we want to point out that the usability problem of voters with out-of-sync system clocks can also be seen as an additional attack vector, that could popentially lead to a situation where a much larger group of voters is affected by this problem. The details of the attack are described at https://www.scip.ch/?labs.20221117. The attack is based on the fact that system clock synchronization is typically accomplished by connecting to a Network Time Protocol (NTP) server on a regular basis (for example every 20 minutes), which responds with the current time. However, using an ARP spoofing attack, an attacker may try to broadcast incorrect NTP packets to the local network with the goal of manipulating the system clocks of unconscious voters. In the success case, voters within the compromised network will no longer be able to pass TOTP-based authentication. Note that such an attack can easily be adjusted to the actual time interval, i.e., the proposed increase to 10 minutes has no positive impact.

2.2. Voter-Portal

The reported changes within the Voter-Portal have been listed in Table 2. In essence, these changes allow more configurations of the content of the Voter Portal and provide the possibility to introduce hyperlinks for the support address. While these modifications were likely intended to enhance user experience or system functionality, their security implications warrant closer examination, particularly in the light of prevailing academic research and expert opinion in the field of e-voting security.

Even though we have not been able to verify this explicitly, the mentioned changes underscore a significant concern: these changes appear to contribute to a negative perception of security, especially within the academic community. This sentiment is echoed in the realm of academic research focusing on the vulnerabilities of e-voting systems. In particular, the study cited in [KVMR20] offers an in-depth analysis of the insecurities associated with insecure displays in e-voting. This research unequivocally establishes that providing instructive information on an insecure voting interface introduces potential avenues for exploitation: "*Our study has shown that participants struggle to detect manipulations if an adversary manages to manipulate the voting interface*".

The Swiss Post system, used as a model in these user studies, exemplifies the risks identified. A critical takeaway from the research is the insufficiency of merely enabling users to detect manipulations. If the response to such detection is misguided—such as contacting a malicious support line or retrying without addressing the underlying security breach—the system's integrity remains compromised. "*One additional finding - which is likely to hold for any verifiable voting system - is that it is not enough to make people detect a manipulation if they then call the malicious support hotline or simply try again. This needs to be addressed as future work*".

The essence of this issue lies in the new dynamic established by the Voter-Portal's updates. Voters are now conditioned to expect additional information on the Voter-Portal,

an expectation that can be exploited by malicious entities. By introducing more interactive or informative elements on the interface, the system inadvertently educates users to seek out and interact with these new features. This change in user behavior creates new potential pathways for attackers to manipulate user interactions or divert users to compromised resources.

To counterbalance this expanded threat landscape, voter education and vigilance are crucial. For instance, encouraging voters to contact the official hotline, as listed in the voting materials, instead of engaging with suspicious links on a potentially compromised webpage, could serve as a mitigating measure. However, the effectiveness of this strategy is heavily contingent upon the voters' ability to recognize and appropriately respond to security threats. This dependency introduces an element of uncertainty, as it assumes a level of alertness and knowledge among voters that may not consistently be present.

In the light of these considerations, we see it as imperative to align system updates and modifications with the advice and findings of leading academics and experts in e-voting security. The recommendations from the study in [KVMR20], particularly those pertaining to the Swiss Post system, highlight the consequences of overlooking such expert guidance. This has also been addressed by Andreas Kuster's blog [kuster23] and similar critiques [Sch23].

2.3. Other Modifications (October Release)

We conclude our review of the October release by looking at some modifications, that are less essential in terms of security implications. All modifications are listed in corresponding CHANGELOG.md files (see Table 2).

2.3.1. New Alphabet for the Start Voting Key

A subtle change has been implemented for improving the usability of entering the start voting key `SVK`. In the alphabet from which the characters are selected, the Base32 lowercase characters `'l'` and `'o'` have been replaced by `'8'` and `'9'`, respectively. The replaced characters are so-called *homoglyphs* (characters with shapes that appear identical or very similar), which are known to increase the chance of misspellings in text-based user interfaces. This change is therefore meaningful.

The new alphabet $\mathbb{A}_{SVK}$ is specified in [SysSpec, Section 3.6] and a new algorithm GenRandomSVK has been introduced as Algorithm 4.4. By selecting the desired number of random characters from $\mathbb{A}_{SVK}$ and concatenating them into a string, the new algorithm is simple and straightforward. It is called in Line 4 of Algorithm 4.3 (GenVerDat). In the code, we observed the creation of a new class `StartVotingKeyAlphabet` (inheriting from `Alphabet`), which correctly lists the characters from $\mathbb{A}_{SVK}$ in the right order. A singleton instance of this class is created in the method `GenVerDatAlgorithm::genVerDat`, which passes it as an argument to `GenRandomSVKAlgorithm::genRandomSVK`. By doing so, the implementation of the updated specification has been done correctly (the new algorithm is implemented using Java streams, so the matching is not one-to-one, but the correctness is obvious).

Independently of its correctness, we made some observations regarding the specification and implementation of the new algorithm:

- In Algorithm 4.4, string concatenation is described implicitly using vector notation, whereas in algorithm IntegerToWriteIn [SysSpec, Algorithm 3.13] and algorithm LeftPad [CryptPrim, Algorithm 3.15] string concatenation is done explicitly using the common double-pipe operator ||. We recommend using a consistent notation throughout all documents.
- In Algorithm 4.3, the new alphabet $\mathbb{A}_{SVK}$ is part of the context, whereas in Algorithm 4.4 it is an input argument. By creating a `StartVotingKeyAlphabet` instance in `genVerDat` and passing it as input to `genRandomSVK`, the implementation corresponds to the pseudo-code. However, we do not understand why $\mathbb{A}_{SVK}$ is not part of the context of Algorithm 4.4 itself, because by its name, the new algorithm is very specific to that particular alphabet. In other words, passing $\mathbb{A}_{SVK}$ from Algorithm 4.3 to Algorithm 4.4 seems unnecessarily complicated (or unnecessarily restrictive, see next remark).
- In our February 2023 report, we recommended a simplification for the algorithms GenRandomBase16String, GenRandomBase32String, and GenRandomBase64String [3, Pages 53–54]. So far, our recommendation has not been taken into account. However, exactly our proposed simplification has now been used in the new algorithm GenRandomSVK, but without making the obvious generalization to arbitrary alphabets. We assume that this happened unconsciously. As a consequence, four different algorithms are currently included in the specification for exactly the same general purpose, namely for selecting a certain number of random characters from a given alphabet. Therefore, we repeat here our recommendation to implement a single generic algorithm GenRandomString$(\ell, \mathbb{A})$, where ℓ denotes the length of the random string and $\mathbb{A}$ the given alphabet. This algorithm can then be called with corresponding alphabets, depending on the context. Note that the generic algorithm would be identical to GenRandomSVK, except for its generic name and the neutral name of the alphabet parameter.

The introduction of this particular new algorithm is exemplary for the lack of simultaneous accuracy, consistency, and generality that we sometimes encounter in algorithms designed by Swiss Post. It appears that there is still considerable room for improvement in this area, for instance by addressing open recommendations from previous examination reports more systematically.

A final remark on this topic is our observation that $\mathbb{A}_{SVK}$, which in [SysSpec, Section 3.6] is called "*lowercase Base32 alphabet*", differs from the Base32 standard as defined by RFC4648 in multiple ways. First, lowercase characters are used instead of uppercase characters. Second, digits precede letters in the character ordering. Third, different homoglyphs are excluded. We are not sure, whether departing from the given standard is intentional or not, but at least some explanations should be given.

2.3.2. Number of Voting Options Increased

Without giving any justifications, an increase of the maximum number of voting options has been announced in the CHANGELOG.md file of the crypto-primitives-domain

component (while the number of selectable voting options remains unchanged at 120). We assume that this is a consequence of increasing the security level from 112 to 128 bits, which results in 50% larger group parameters. By inspecting the source code, we can confirm that the constant `MAXIMUM_NUMBER_OF_VOTING_OPTIONS` in the Java class `VotingOptionsConstants` has been increased from 3000 to 5000 (without changing any of the other parameters).

An obvious problem that we encountered while examining this change can be found in the JavaScript test file gq-group-generator.js of the e-voting component. This file also defines a constant `MAXIMUM_NUMBER_OF_VOTING_OPTIONS`, but we observed that this constant has not been adjusted accordingly, i.e., its current value is still at 3000. Since the current release is inconsistent in that matter across two different files at different locations, it is obviously a bug that needs to be fixed (even if it only affects a test file).

2.3.3. Bug Fixes

a) Mixing of Very Large Ballot Boxes

The CHANGELOG.md file of the e-voting component announced the fixing of "*a bug that prevented the mixing of very large ballot boxes*", without given any further information about the bug itself, the location of the bug in the source code, or the proposed solution (see Table 2). At the beginning of our examination, just by looking at the commit history of the master branch, we were unable to locate corresponding modifications in the source code. In the commit history of the develop branch, however, we found the following two recent commit messages containing the related terms like "mix", "shuffle", "bug", or similar. However, neither of them is pointing directly to a bug fix related to the mixing of large ballot boxes:

- 25/10/23: "Improve the MixDecryptProcessor." (1 file modified)
- 20/07/23: "Add possibility to resume mixing at anytime." (7 files modified)

The *improvement* of the Java class `MixDecryptProcessor` consists of an added statement for checking the correctness of the `nodeId` when receiving a `mixDecryptOnlineRequest`. Without understanding the full purpose, context, and impact of this class, we assume that the absence of this check in prior releases corresponds to the bug that has been fixed in the current release (the added *possibility* to resume mixing seems to be an added feature, not a bug fix). However, we were unable to conclusively reconstruct the circumstances leading to the mixing-related problem for large ballot boxes mentioned in the CHANGELOG.md file. We would have expected Swiss Post to provide more detailed information.

b) Multiple Elections With Write-Ins

Another bug fix was reported in the CHANGELOG.md file of the e-voting-libraries component: "*Fixed a bug in the correct creation of the tally files when having multiple elections with write-ins*" (see Table 2). As already stated in Subsection 1.3.2, we received from Swiss Post an additional comment that "*a single class of our* e-voting-libraries *component was enough to resolve this problem*", but without indicating which one. Here again, we would have expected more accurate information about the pre-conditions, location in

the code, and impact of the problem, together with a description of the implemented solution.

Based on our present comprehension of the bug in question and its corresponding resolution, it is reasonable to assume that its influence is confined exclusively to the generation process of the final eCH-documents. Even without understanding all modifications in detail, we expect that there is no negative impact for elections without write-in candidates. However, we cannot confirm that the changes fully resolve the reported bug.

2.4. Other Modifications (July Release)

By looking at the changes reported for the July release, for which we have not received an explicit examination mission from the FCh, we decided to slightly enhance our current mission of examining the October release. Even though the time for this enhancement was very limited, we made some interesting observations in two distinct subject areas.

2.4.1. Exactly-Once Processing of Ballots

In our examination report from March 2022, we identified the lack of synchronization between parallel message processing in the control components [2, Section 2.6]. Swiss Post addressed the issue by introducing a so-called `ExactlyOnceCommandExecutor`, which we reviewed in the first re-examination. In our report from February 2023, we concluded that the problem had been resolved and that the implementation had been carried out correctly [3, Sections 2.2.6 and 3.3.3]. We thoroughly analyzed the Java source code, paying particular attention to the crucial function `findSemanticallyIdenticalCommand`, which ensures that semantically identical messages are identified and rejected based on certain supplied context parameters. However, we overlooked that in the database the composite primary key consists not only of these context values but includes an additional `correlationId`. If multiple semantically identical messages are concurrently processed, a race condition should arise during the final insertion into the database. This is due to a violation of the primary key constraint, resulting in the rejection of all but one of the messages. In contrast to the context values, the `correlationId` is assigned by the untrusted voting server and is irrelevant for the cryptographic protocol. As a result, the voting server colluding with a voter can infiltrate several semantically identical messages by altering the `correlationId`. This problem has been identified by Florian Moser.[3]

Swiss Post has resolved the issue in the July release by defining an additional unique constraint on the context values. In consequence, the aforementioned attack results in a unique constraint violation and all messages except one will be appropriately rejected. The rationale for introducing an additional unique constraint and not simply removing the `correlationId` from the composite primary key remains somewhat unclear. The proposed solution appears to be more complicated than necessary.

[3] See https://gitlab.com/swisspost-evoting/e-voting/e-voting/-/issues/9

2.4.2. Web Worker

Running cryptographic algorithms in JavaScript on the client side is a critical security factor because voter's privacy is at stake. We have previously emphasized the importance of reducing the number of dependencies and third-party libraries. Additionally, we recommended running the cryptographic core in a dedicated Web Worker to reduce the risk of generic attacks. For more details on this subject, we refer to our reports from March 2022 and February 2023 [2, 3].

As part of our investigation of the July release, we noticed that Swiss Post silently removed the Web Worker from api.js in Version 1.3.0, i.e., instead of running the cryptographic core in a dedicated context, it was executed in the same environment as the rest of the Angluar application, leaving it vulnerable to the aforementioned generic attacks. This was possible from Version 1.3.0 (April release) to Version 1.3.1 (June release). Only in Version 1.3.2 (July release) the Web Worker execution environment was re-introduced with the inclusion of the files worker-api.js and worker-wrapper.js. To the best of our knowledge, the April release was used in production for the national referendums in June 2023, either the June or July release was used for the municipal referendum in Rapperswil/SG in September 2023, and the July release was used for the parliamentary elections in October 2023.

We value the new Web Worker integration and the new API, as the previous API was legacy code from Scytl and its implementation was somewhat unclear and complicated. Nevertheless, the fact that the cryptographic core was removed from a Web Worker execution environment for two official releases—without any notification or entry in the CHANGELOG.md files—is incomprehensible. It gives the impression that the Web Workers were mistakenly removed and later silently re-introduced.

Unfortunately, we have overlooked the missing Web Workers in our examination of the April and June releases, which is a mistake on our part. However, in the instructions received for these examinations, we were only told to verify the changes listed in the CHANGELOG.md files. Hence, we verified the removal of legacy libraries and the reduction of third-party libraries of the JavaScript cryptographic core, but we did not check its integration again. We did that in detail in the February 2023 report [3, Addendum-1], and given our positive verdict, we assumed that the awareness for this topic had reached Swiss Post, i.e., we could not imagine them to revert the improvement they had just implemented.

3. Conclusion

Upon thorough examination, we have not identified any new, significant security vulnerabilities within the requested scope. It is important to note, however, that several security concerns identified in previous assessments remain unresolved and have not been incorporated into the current implementation of the system.

Our analysis suggests that the current approach of conducting partial examinations based on recent changelogs has become insufficient. There is a clear need for a comprehensive re-examination of the system to ensure a thorough evaluation of all relevant security aspects.

We recommend an earlier and more integrated involvement in the process of ensuring and simplifying the system in use, in line with the guidelines outlined in A.25 of the the "*Massnahmenkatalog*" [BK23]. This integration aims to align our research findings more effectively with the requirements of cantons and municipalities. Through this enhanced collaboration, we anticipate a greater mutual understanding of the specifics of all stakeholders involved in this process, ultimately leading to improved system security, functionality and usability.

A. Addendum: November Release

Minor updates of the software components e-voting (Version 1.3.3.2) and data-integration-service (Version 2.7.2.1) have been released on November 16 and 29, respectively. In both cases, we received corresponding announcements shortly after the release date. In their first announcement, Swiss Post called the new e-voting sub-release a "*small corrective patch*" for the following issues:

> "*It mainly corrects issues that we had with previous versions when multiple ballot boxes with a large number of votes were mixed concurrently. Additionally, the patch enhances the configurability of canton-specific text options within the voter portal.*"

In their second announcement, Swiss Post explained that "*two technical problems*" related to the data-integration-service component have been addressed in the new sub-release. In the sequel, we will refer to these new versions as the *November release*.

We agreed with the Federal Chancellery to inspect the implemented changes until December 22. The purpose of this addendum section is to give an overview of the changes made in the November release and to discuss these changes from the perspective of the cryptographic protocol. The results of our analysis are presented in Appendix A.2. This subsection also contains a few other remarks about minor issues that came up while conducting our analysis.

A.1. Overview of Changes

In Figure 2, we give an overview of the GitLab commit histories of the two updated project repositories. It shows that an additional Version 1.3.3.1 of the e-voting component has been released between the October (red) and the November (blue) releases. As announced, all other project repositories remained unchanged. This implies that no changes have been made to the specification documents.

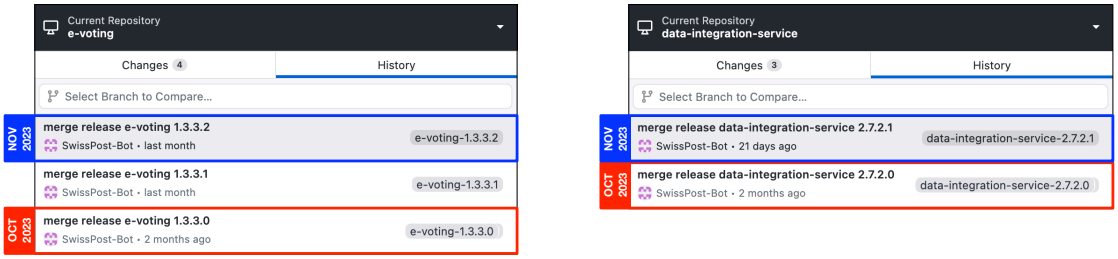

Figure 2: Commit histories of the GitLab projects since the October release.

In both cases, additional items with pointers to the updated code were added to the CHANGELOG.md files. A summary of these items is given in Table 3. All items (except the third one) address issues that have already been a subject in the October release, namely the increased configurability of the voter portal to accommodate canton-specific needs (see discussion in Subsection 2.2), the problems encountered when mixing very large ballot boxes (see discussion in Subsection 2.3.3), and the missing consistency checks with respect to a given election definition. An additional small bug related to the voting server

database definition has been corrected (third item in Table 3). We successfully identified the relevant modifications in the code without any difficulty for all items.

Component	Changes listed in the component's CHANGELOG.md file
e-voting	• Reduced the concurrency of the mixing process in the control components and added a retry mechanism to mitigate against out-of-thread problems when mixing large ballot boxes. • Fixed a bug that prevented the mixing of large ballot boxes due to a message size constraint. • Fixed a small bug in the voting server database table definition. • Voter Portal: Increased the configurability of the texts to accommodate canton-specific needs. • Voter Portal: Fixed a bug with incomplete translations on certain browsers.
data-integration-service	• Add a check that each election contains an empty list and no duplicate list identifiers.

Table 3: Overview of the changes announced for the November release.

A.2. Discussion of Changes

We found no code modifications in Table 3 that have a direct impact on the cryptographic techniques used after reviewing the changes. From this we conclude that both individual and universal verifiability remain unaltered, upholding the given integrity of the system. As a result, no negative consequences on the system's security features are expected within the established trust assumptions.

A.2.1. Mixing of Very Large Ballot Boxes

Following a discussion with representatives from Swiss Post regarding the aforementioned system alterations in Table 3, it has been clarified that the change in question exclusively pertains to the operational efficiency, specifically impacting the message transfer time. The revised process mandates a sequential delivery of messages, as opposed to the previous partially simultaneous delivery method. This adjustment results in a reduction of the memory requirements for the transport service, albeit potentially extending the duration of the message delivery process. Nevertheless, we are currently unable to determine the precise magnitude of the effect on the delivery performance.

A.2.2. Configurability of the Voter Portal

All updates to the voter portal solely affect the graphical user interface and have no impact on the underlying cryptography. This is clearly evident as the changes are limited to the voter-portal component note that the voting-client-js component does not show any modifications). Therefore, the changes have no immediate security implications. However, the indirect security risks of a malicious user interface misleading the voter (see our concerns expressed in Subsection 2.2) remain unresolved, as the changes rather expand

the user interface instead of restricting it. In addition to extending the configurability, a new optional first step was added, in which the voter is asked to accept the canton-specific legal terms.

A.2.3. Other Remarks

- We were able to locate the announced bug fix in the voting server database table definition in the file V2.001_processor_baseline.sql of the voting-server sub-component, where in the table `VERIFICATION_CARD_STATE` the type of the column `STATE` has been changed from `VARCHAR2(10)` to `NUMBER(1)`. Related changes have been made to the Java class `VerificationCardRepository` (Line 32). In both cases, however, we were unable to understand the full context and the problem that was caused by the previous version. Better explanations in the CHANGELOG.md file would have been appreciated.
- The purpose of the modification in the class `ContestService` is to add a "*check to avoid list identification duplicates*" (see last entry in Table 3). From an algorithmic point of view, the problem is simply to check if a nested unordered list contains any duplicates. For this simple problem, the implemented stream pipeline over 10 lines is extremely complicated for no obvious reason. As far as we were able to verify, the stream pipeline is correct, but it is clearly not a good example of using stream programming efficiently and of clean coding in general.
- In the Java class `VerifyTallyControlComponentAlgorithm` of the verifier component, we probably encountered a minor bug in the Lines 147–148, in which a stream pipeline of type `Boolean` terminates using a `reduce`-statement. The idea is to compute the logical AND of the results obtained from performing a check to each ballot box ID. In the borderline case of an empty ballot box, we believe that the result of this whole expression should be `true` in a trivial sense, because `true` is the identity element of the logical AND operator. In Line 148, however, `false` is returned in that case. Note that `Boolean` streams can be reduced more easily using the terminal operations `allMatch()`, `anyMatch()`, or `noneMatch()`.
- There is an inconsistency in the write-ins system parameters. The alphabet as defined in [SysSpec, Section 3.7.1] as well as in the Java class `WriteInAlphabet` from the e-voting-libraries component consists of 142 characters, which means that $\log_2 142 = 7.15$ bits are required for each character. Since the upper limit for the length of the write-ins is set to 500 (see variable `MAXIMUM_WRITE_IN_OPTION_LENGTH` in the Java class `VotingOptionsConstants`), we get a total of 3575 bits for representing a write-in candidate of maximal length. This number is in conflict with the 3071-bits group size defined for the extended security level in [CryptPrim, Section 2], i.e., such very long write-ins cannot be encrypted unambiguously using the ElGamal encryption system. For this reason, we recommend setting the maximal length to $\lfloor 3071/7.15 \rfloor = 429$ or less.

References

[KVMR20] O. Kulyk, M. Volkamer, M. Müller, and K. Renaud. Towards improving the efficacy of code-based verification in Internet voting. In J. Bonneau and N. Heninger, editors, *FC'20, 23rd International Conference on Financial Cryptography*, Kota Kinabalu, Malaysia, 2020.

[kuster23] A. Kuster. Know the Protocol! – How to Prevent Undetected Vote Manipulation on the Verified Swiss Post E-Voting System. https://andreaskuster.ch/blog/2023/CVD-EVoting-Swiss-Post, 2023.

[BK23] Die Schweizerische Bundeskanzlei (BK). Vote électronique, Massnahmenkatalog von Bund und Kantonen. 4. August 2023,

[Sch23] B. Schneier. Schneier on Security, Security Vulnerability of Switzerland's E-Voting System. 17. October 2023, https://www.schneier.com/blog/archives/2023/10/security-vulnerability-of-switzerlands-e-voting-system.html