Architektur Vorschlag für elektronische Umfragen / eVoting

Die Zeit ist reif über neue Methoden in unseren Demokratien nachzudenken. Mit heutiger Technologie wäre eVoting, oder sei es auch nur eMeineMeinung, absolut sicher umsetzbar.

Ich weiss, gewissen Querdenkern sträubt sich jetzt jedes Haar: eVoting will ich nicht, das kann noch viel mehr gefaked, beinflusst, manipuliert werden, bleib mir bloss weg damit.

Stimmt! Wenn es falsch angegangen wird bestehen diese Gefahren.

Ich glaube eine Architektur gefunden zu haben die 100% sicher ist und nicht manipuliert werden kann. Drum bitte, lieber Querdenker, sei kein Leerdenker, denke! Gib mir die Chance meine Idee darzulegen, lies alles durch, versuche es zu verstehen, oder Frage mich (Twitter https://twitter.com/CervelatFresser) bei Unklarheiten.

Mit so einer Lösung könnte man vor jedem politischen Entscheid, der grosse Teile der Bevölkerung betrifft, das Volk erst befragen, und erst dann eine Lösung ausarbeiten, auf Grund der Antworten der Bevölkerung. Zur Abstimmung bringen könnte man dann 2-3 Varianten und nicht nur ja/nein für einen einzigen Vorschlag. Und das alles ginge schnell. Man könnte das Volk innert Tagen, und nicht erst in 6 Monaten, befragen wenn eine radikale Situation eintritt.

In diesem Blog Beitrag möchte ich meinen Vorschlag zu einem sicheren eVoting System erklären. Technisch könnte ich es selber nicht umsetzen, dazu fehlt mir das KnowHow, aber ich verstehe genügend von Architekturen und der Funktionsweise der relevanten Themen (Verschlüsselung/ Signaturen / Blockchain).

Im weiteren nutze ich den Begriff **eVoting**, aber man könnte das System auch unverändert nutzen um einfach nur Umfragen zu machen, die Anforderungen an beide Anwendungen (eVoting/ Umfragen) sind fast dieselben. Dazu weiter unten mehr.

Mein Vorschlag für eine sichere Architektur, wo man Stimmen nicht fälschen kann, basiert auf **Blockchain** Technologie. Deshalb möchte ich kurz erklären worum es bei Blockchain geht.

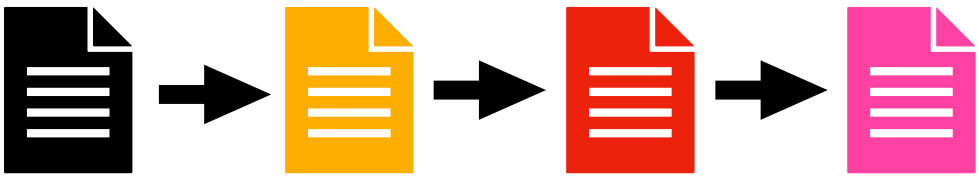
Was Blockchain ist und wie es funktioniert

Die meisten von Euch haben bestimmt schon von der Kryptowährung **Bitcoin** gehört. Bitcoin war die erste Anwendung der Blockchain Technologie. Kryptowährungen setzen fast ausschliesslich auf diese Architektur setzen. Wieso das so ist erkläre ich weiter unten.

Eine Blockchain ist vereinfacht gesagt ein **Datenspeicher**, genauer, ein verteilter und dezentraler Datenspeicher.

Herkömmliche Systeme haben meist zentrale Datenspeicher und sehr selten sind die verteilt. Zum Beispiel die Festplatte eines Notebooks ist extrem zentral und extrem nicht verteilt, nämlich im Gerät selbst. Eine Unternehmung kann sich entscheiden die Daten dezentral zu speichern, zum Beispiel in einem externen Rechenzentrum oder in der Cloud. Unternehmen die ihre Daten redundant halten wollen, um Ausfallsicherheit zu haben, speichern ihre Daten nicht nur dezentral, sondern auch verteilt, z.B. in 2 geographisch verteilten Rechenzentren. Dann sind die Daten, selbst wenn das eine Rechenzentrum abbrennt, nicht verloren.

In der Blockchain, das ist jetzt sehr generell formuliert, weil es verschiedene Typen von Blockchains gibt, werden Datenblocks gespeichert und aneinander gehängt. Drum der Name Blockchain, oder Kette von Blöcken mit Daten.



Wie funktioniert ein Hash Code?

Jeder neue Block enthält den **Hash Code** für den letzten Block. Damit können die Blöcke nicht neu sortiert werden.

Ein Hash Code ist das Resultat einer mathematischen Funktion über eine Menge Daten. Sind die Daten zweier Mengen identisch (auf das Bit genau), dann ist auch der Hash Code identisch. Ändert man aber auch nur eine Winzigkeit, wechselt zum Beispiel in einem 4 MB grossen Bild einen einzelnen Punkt von 50% grau auf 49% grau, dann ändert sich der Hash Code dieses Bildes und man weiss: es ist nicht mehr das Original. Man weiss zwar nicht unbedingt was geändert wurde, aber dass etwas geändert wurde.

Das ist ein zentrales Element der Blockchain: einmal gespeicherte Daten können nicht geändert werden, ohne dass es bemerkt wird.

Beim eVoting wären es die abgegebenen Stimmen die nicht verändert werden könnten.

Erklärung: Wird im schwarzen Block oben in der Grafik etwas geändert, ändert sich der Hash Code, damit stimmt der Hash Code im gelben Block nicht mehr mit dem schwarzen Block überein. Man könnte jetzt den Hash Code im gelben Block ändern, wodurch sich aber die Daten des gelben Blockes ändern womit der Hash Code für den gelben Block anders wäre und der rote Block nicht mehr auf den gelben referenzieren würde. Man müsste also die ganze Kette ausgehen vom schwarzen bis zum absolut neuesten neu “organisieren”.

Das ginge an einem zentralen Ort super leicht und niemand würde es bemerken. Aber die Blockchain hat die Daten ja dezentral, auf unzähligen Servern, verteilt.

Was heisst dezentral verteilt bei der Blockchain?

In der Blockchain gibt es unzählige Server, manchmal Nodes oder Validators genannt, welche helfen die neuen Datenblöcke zu “validieren”.

Vereinfacht gesagt: Wenn es 10 Nodes gibt, dann müssen die 10 Nodes alle derselben Meinung sein bevor sie einen neuen Block zulassen/schreiben.

Ein Beispiel (weshalb Kryptowährungen fast ausschliesslich auf Blockchain basieren)

Anna hat 10€. Die will sie mir schicken weil ich einen Kuchen für sie gebacken habe. Ich bin zufrieden wenn mein Konto nach der Transaktion 10€ mehr drauf hat. Aber es ist natürlich auch wichtig, dass Anna 10€ weniger auf ihrem Konto hat. Sonst könnte sie die 10€ ja unendlich viele Male Leuten schicken und quasi alle reich machen.

Das System muss also 2 Sachen sicherstellen: Anna hat nach der Transaktion 10€ weniger auf ihrem Konto, und ich habe nach der Transaktion 10€ mehr auf meinem Konto.

Sind sich nun alle 10 Nodes einig, dass im neuen Block diese neue Tatsache (Anna -10€, ich +10€) festgehalten wird, dann kann der Block geschrieben werden.

Findet einer der Nodes: Nö, ich bescheisse und sage: nach der Transaktion hat Anna nicht weniger € auf dem Konto, dann wird, je nach Implementation, entweder der Node ausgeschlossen von künftigen Validierungen, oder einfach die Transaktion zwischen Anna und mir nicht festgehalten.

Als Resultat haben wir jetzt 10 Nodes die alle über einen neuen Block verfügen wo Anna 10€ weniger auf dem Konto hat und ich 10€ mehr.

Leerdenker sagen jetzt:

Ein Hacker kann ja easy den Kontostand bei Anna ändern, also sogar 1000€ gutschreiben.

Stimmt! Liest bitte nochmals das Kapitel Wie funktioniert ein Hash Code? Und erkläre mir weshalb auf Kryptowährungen Blockchains seit mehr als 10 Jahren täglich Milliarden gehandelt werden ohne je gehackt worden zu sein.

Erklärung für einen Hack: Wir hätten jetzt 2 unterschiedliche Blockchains. Weil der Hacker nur auf einem der 9 Nodes den Kontostand ändern kann, die anderen 9 Nodes haben noch die Wahrheit

gespeichert. Weil sich die ganze Kette von Blocks beim gehackten Node jetzt von den Ketten welche die 9 anderen Nodes gespeichert haben unterscheiden, traut niemand mehr dem gehackten Node.

Das dieses System funktioniert zeigen diverse Kryptowährungen die auf diesem oder einem angelehnten Prinzip beruhen. Bitcoin gibt's seit dem 3. Januar 2009 und es werden täglich ca. 50 Milliarden Dollar Transaktionen darauf verschoben. Wäre das System hackbar hätte es nie diese Grösse erreicht.

Elektronische Signatur

Ein weiteres zentrales Element bei der Blockchain ist die elektronische Signatur von Transaktionen.

Im Beispiel oben mit Anna und mir: Wie verhindert man, dass nicht ich eine Transaktion absetzen kann, wo mir Anna mir 10€ zahlt? Oder: Wenn die Transaktion sagt: Anna -10€ und ich +10€ und das alles ist, dann könnten wir ja von Millionären Millionen abbuchen.

Stimmt! Aber nicht wenn die Transaktionen elektronisch signiert sind.

Was dies bedeutet und wie es funktioniert erkläre ich in diesem Kapitel.

Kurz gesagt kann ein Empfänger einer **signierten Nachricht** (oder Transaktion) zweifelsfrei sagen wer die Nachricht geschickt hat.

Im Beispiel würden die 10 Nodes KEINE Transaktion akzeptieren die NICHT von Anna signiert ist. Schliesslich sendet sie ja Geld, also muss die Nachricht von ihr kommen.

Das ist, nur so nebenbei, im übrigen ein weiterer Faktor weshalb ein Hacker nicht einfach Anna's Kontostand ändern kann. Das bedürfte einer Transaktion die dies entsprechend auslöst, und so eine Transaktion müsste von Anna signiert sein, was nicht geht wie ich in diesem Kapitel zu erklären versuche.

Wie geht elektronische Signatur?

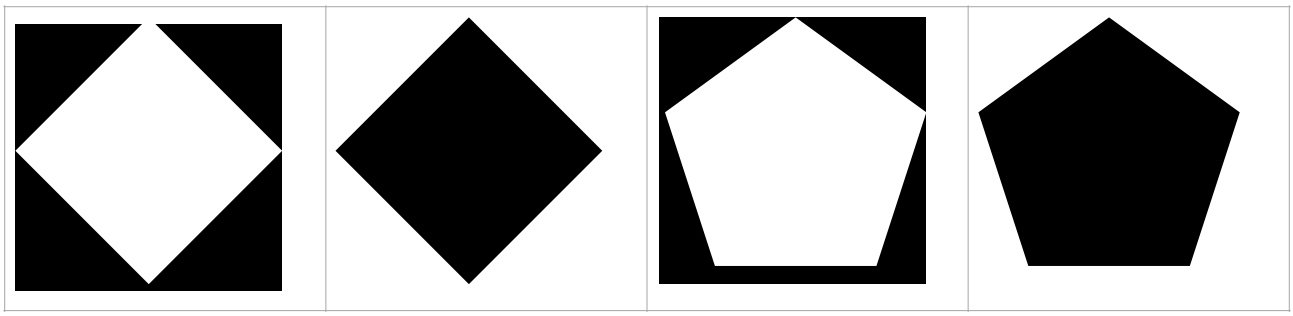
Systeme die elektronische Signaturen, oder Verschlüsselung, brauchen, setzen heute fast ausschliesslich auf das Public/Private Key Verfahren.

Jeder Teilnehmer am System, Anna, ich, die 10 Nodes, verfügen über je einen privaten Schlüssel, den nur der Teilnehmer selber kennt, und über einen öffentlichen Schlüssel, den jeder kennt.

Public und Private (oder öffentlicher und privater) Schlüssel werden, vereinfacht gesagt, mit derselben mathematischen Funktion errechnet. Diese Funktion basiert auf Primzahlen (sehr grossen Primzahlen). Deshalb kann man selbst wenn man den einen (öffentlichen) Schlüssel besitzt, den anderen (private) Schlüssel nicht berechnen, weil die dazu verwendete Primzahl nicht bekannt ist.

Wir wissen jetzt also: Private und Public Key (Schlüssel) ergänzen sich immer und jeder Teilnehmer am System verfügt über beide, wobei alle "öffentlichen" Schlüssel für jeden Teilnehmer ersichtlich sind.

Anna		**Ich / Node 1…9 etc.**	
Private	Public	Private	Public



Das spannende ist nun, dass die beiden Schlüssel (public/private) quasi mathematische Zwillinge sind.

Das heisst, dass man eine Nachricht die mit einem Schlüssel verschlüsselt wurde, mit dem anderen entschlüsselt werden kann.

Diese Tatsache ermöglicht es Nachrichten zu verschlüsseln und/oder zu signieren.

Wenn Anna mir eine verschlüsselte Nachricht schickt, muss sichergestellt werden, dass nur ich diese Nachricht entschlüsseln kann. Wenn Anna also zum Verschlüsseln meinen öffentlichen Schlüssel verwendet (darauf hat sie ja Zugriff, weil er eben öffentlich ist), dann kann nur der “Zwilling” dieses Schlüssels, mein privater Schlüssel, auf den nur ich Zugriff habe, die Nachricht entschlüsseln.

Schicke ich der Anna eine verschlüsselte Nachricht, dann nutze ich Anna’s öffentlichen Schlüssel zum verschlüsseln und nur sie kann mit ihrem privaten Schlüssel die Nachricht entschlüsseln.

Eine Nachricht signieren funktioniert genau gleich, aber mit anderen Schlüsseln.

Schickt mir Anna eine signierte Nachricht, also eine Nachricht, wo sie sicher sein will, dass ich weiss dass nur Anna diese Nachricht hätte schicken können, dann verschlüsselt sie die Nachricht mit ihrem privaten Schlüssel. Jeder der Zugriff auf den öffentlichen Schlüssel von Anna hat, jeder, weil der Schlüssel ja öffentlich ist, kann also verifizieren (die Nachricht entschlüsseln) dass nur Anna die Nachricht hat schicken können.

Bei Kryptowährungen arbeitet man deshalb mit der elektronischen Signatur. Wenn Anna mir 10€ schickt, muss sie diese Transaktion mit ihrem privaten Schlüssel signieren (verschlüsseln) und die 10 Nodes können mit dem öffentlichen Schlüssel von Anna verifizieren: Anna hat die Transaktion geschickt, und nicht jemand der der Anna 10€ stehlen wollte.

Codius Server - App Hosting

Ein weiteres wichtiges Element meiner vorgeschlagenen Architektur ist der Codius Server von Ripple. Auf diesem Server kann man Programme ausführen lassen. Hat man mehrere Codius Server kann man auf jedem dieser Server dieselben Programme ausführen lassen. Das coole dabei ist die Tatsache, dass die Programme gehasht werden. Das heisst es gibt wie oben beschrieben für die Programme einen Hash Code. Damit kann man 100% sicherstellen, dass die Programme nicht gehackt/verändert wurden.

Jedes Programm in meiner Architektur würde auf Codius laufen. Damit wäre sichergestellt, dass die Programme nicht verändert wurden.

Man könnte also die Programme ausgiebig testen, dass sie keine offenen Türen haben, dann kompilieren und den erstellten Hash Code nutzen um zu beweisen, dass die Programme nach dem Testen nicht gehackt wurden.

Was hat das ganze jetzt mit eVoting zu tun?

Wir wissen jetzt wie Blockchain, die Basis meiner eVoting Architektur, funktioniert. Es werden Daten (Transaktionen) unveränderbar auf unzähligen Nodes gesichert.

Nochmals: Tausende Kryptowährungen zeigen seit Jahren die Technologie ist nicht hackbar.

Wie hilft uns das jetzt bei einem sicheren eVoting System?

Schauen wir uns erst die wichtigsten Anforderungen an ein eVoting System an.

Anforderungen eVoting System

Ein Teil der Anforderungen findet man z.B. hier: https://aceproject.org/ace-en/focus/e-voting/countries

Irgendwo habe ich mal die Tabelle unten gefunden.

Requirement	Description
Authenticity	Only users with the right to vote should be able to cast a vote
Singularity	Each voter should be able to vote only once
Anonymity	It should not be possible to associate a vote to a voter
Integrity	Votes should not be able to be modified or destroyed
Uncoercability	No voter should be able to prove the vote that he/she has casted
Verifiability	Anyone should be able to independently verify that all votes have been correctly counted
Auditability and Certifiability	Voting systems should be able to be tested, audited and certifiable by independent agents
Mobility	Voting systems should not restrict the voting place
Transparency	Voting systems should be clear and transmit accuracy, precision, and security to voter
Availability	Voting systems should be always available during the voting period
Accessibility and Convenience	Voting systems should be accessible by people with special needs and without requiring specific equipment or abilities
Detectability and Recoverability	Voting systems should detect errors, faults and attacks and recover voting information to the point of failure

Was bedeuten die Begriffe?

Begriff	Übersetzung	Erlärung
Authenticity	Authentizität	Nur wer wählen darf soll wählen können. Also in Deutschland nur Deutsche, in Bayern nur bayrische Bürger und alle die wählen müssen im Wahlrechtsalter sein
Singularity	Singularität	Jeder Wähler darf nur 1x wählen
Anonymity	Anonymität	Kein Stimme darf auf einen bestimmten Wähler zurückgeführt werden können. Man darf also nichts drüber wissen wenn der Michel gewählt hat.
Integrity	Integrität	Keine Stimmen dürfen zerstört oder verändert werden
Uncoercability	Unverwechselbarkeit	Kein Wähler darf beweisen können was er gewählt hat
Verifiability	Nachvollziehbarkeit	Man muss unabhängig überprüfen können, dass alle Stimmen korrekt gezählt wurden
Auditability and Certifiability	Auditierbarkeit Zertifizierbar sein	Wahlsysteme müssen unabhängig getested, auditiert und zertifiziert werden können

Mobility	Mobilität	In meiner Architektur wählt man via App auf dem SmartPhone. Dass die App nicht gehackt wurde, könnte man wiederum mit einem Hash Code prüfen
Transparency	Transparenz	Ein eVoting System muss genau sein und dem Wähler Sicherheit vermitteln
Availability	Verfügbarkeit	Durch die Natur der Blockchain, mit verschiedenen Servern, ist dies sichergestellt
Accessibility and Convenience	Barrierefreiheit Einfache Verwendung	SmartPhone App
Detectability and Recoverability	Erkennen und Beheben von Fehlern	Das System soll Fehler, Hackerangriffe und dergleichen erkennen und eliminieren. Blockchain funktioniert seit Jahrzehnten ohne Fehler und konnte bisher, wenn richtig umgesetzt, nicht gehackt werden

Schauen wir uns nun an wie jede dieser Anforderungen umgesetzt werden kann.

Die einfachen zuerst, die schwierigen am Ende.

Authentizität

Das hat nichts mit dem eVoting System an sich zu tun. Da geht's mehr um Sicherheit beim Login. Jede Bank bietet heute 2FA Autorisierung an. Das eVoting System könnte auch darauf, 2FA, basieren. Man könnte sich zum Beispiel auf der Gemeinde verifizieren lassen und damit auf der installierten SmartPhone App einen Code den die Gemeinde generiert hat eingeben, plus gleichzeitiges Aktivieren der 2FA.

Wer Kryptowährungen kauft, hat meist ein "Wallet". Dieses Wallet enthält den private Key. Bei der Erstellung wird aber auch der Public Key erstellt. Erinner dich, public private Key sind quasi Zwillinge. Beim Erstellen der Wallet werden Seed Words generiert (zum Beispiel 24). Diese Seed Words, quasi die Primzahl die für die Berechnung der public / private Keys genutzt wird, sind mit heutigen Computern nicht möglich zu hacken. Mehr dazu hier: https://datarecovery.com/rd/can-brute-force-attacks-crack-bitcoin-private-keys/

Wechsle ich mein SmartPhone, habe ich 2 Möglichkeiten: mit den Seed Words auf dem neuen SmartPhone die App entsprechend neu zu installieren, oder mich bei der Gemeinde erneut verifizieren zu lassen.

Singularität

Vor jeder Wahl würde jeder potentielle Wähler einen "Wahltoken" in seinem Konto erhalten. Einen Token pro Abstimmung. Stimmt man über 3 Gesetze ab, wären es 3 Token.

Gebe ich meine Stimme zum 1. Gesetz ab, wird der entsprechende Token "abgebucht", wie im Beispiel wenn Anna mir 10€ schickt. Sie hat dann 10€ weniger, beim eVoting hat der Wähler nachdem er gewählt hat keinen entsprechenden Token mehr.

Integrität

Wer daran zweifelt, soll nochmals die Kapitel über Blockchain Technologie lesen

Nachvollziehbarkeit

Siehe Erklärung wie Blockchain funktioniert.

Audierbarkeit / Zertifizierbar sein

Welches System kann man nicht auditieren? Mit Hash Codes könnte man die Unveränderheit der Programme gewähren.

Mobilität

SmartPhone App

Transparenz

Bitte lies nochmal das ganze Dokument

Verfügbarkeit

Lies nochmals den Teil über Blockchains, dass unzählige Server beteiligt sind. Man könnte 100 Server installieren und 30 sind nötig um eine Stimme zu akzeptieren. Damit könnten 70 ausfallen und das System würde noch funktionieren.

Barrierefreiheit

Eine SmartPhone App kann dies sicherstellen. Man könnte selbst Blinden die Möglichkeit geben abzustimmen.
Ich frage mich wie man dies in heutigen System sicherstellt.
Wenn ich der Betreuer bin von einem Blinden und er über 2 Vorlagen abstimmen will, einmal mit Ja und einmal mit Nein, ich aber andersrum abstimmen würde, dann setze ich ihm einfach die verwechselten Zettel vor.

Erkennen und Beheben von Fehlern

In der Blockchain gibt es keine Fehler, bzw. die Technologie stellt sicher, dass es keine gibt, das ist quasi die Grundidee der Blockchain.

Fehlen noch 2 wichtige Kriterien.

Unverwechselbarkeit und Anonymität

Um diese beiden Anforderungen sicherstellen zu können nutzen wir Codius und 2 Blockchain Systeme.

Die Idee ist folgende (der **Hauptteil meiner vorgeschlagenen Architektur**)

Eine abgegebene Stimme wird an eine definierte Anzahl Nodes (x) verschlüsselt verschickt. Hat das System 30 Nodes, könnten x = 10 sein.

Das heisst wenn ich abstimme generiert meine App x verschlüsselte Transaktionen. Die x Server können meine Stimme entschlüsseln und sich (wie oben beschrieben bei Blockchain) darüber einigen, dass meine Stimme korrekt ist. Ich also ich bin, ich noch keine Stimme abgegeben habe und sie müssen sich auch einig sein was ich meine Stimme war (ja/nein).

Dabei weiss keiner der Server meinen Namen, nicht mal die Gemeinde die mich verifiziert hat, könnte sagen WER hinter einer abgegebenen Stimme steckt. Es ist nur sichergestellt dass ich stimmen darf.

Finden die x Server dass alles korrekt ist, wird meine Stimme mit demographischen Daten (Alter/ Geschlecht/Wohnsitz ect.) an das 2. System übermittelt. Im 2. System kann kein Bezug mehr zu mir hergestellt werden. Da existiert nur noch die Stimme mit den demografischen Daten. Auf dem 1. System wird meine Transaktion gelöscht, mein Token für die Stimme abgebucht.

Damit in diesem Schritt, wo meine Stimme ans 2. System übergeben wird, nichts schief läuft wird der verifizierte, zertifizierte Code (Codius) ausgeführt. Es handelt sich also um 100% nicht gehackten Code. Im 2. System gibt's wieder x Server die sich wieder einigen müssen ob die erhaltenen Daten okay sind. Also x Server müssen sagen ja (oder nein) wurde gestimmt.

Das heisst wir haben im 2. System jetzt eine Stimme (ja/nein) und demographische Daten. Wie man mit diesen demografischen Daten umgehen will, kann man vorher im System definieren.

Lebt in einem Dorf nur 1 Frau, darf man das Geschlecht nicht verwenden um keine Rückschlüsse auf die einzige Frau ziehen zu können. Stimmt 1 Frau nein, weiss jeder was Anna gewählt hat. Leben im Dorf aber 100 Frauen und 30 haben nein gestimmt hat man keine Ahnung wer jetzt ja wer jetzt nein gestimmt hat.

Solche demografischen Aussagen wären erst möglich sobald es statistisch nicht mehr zu Rückschlüssen führen könnte. Das kann man statistisch vorhersagen, welche Kriterien welchen Anforderungen genügen müssen.

Damit hätte man die fehlenden Anforderungen auch entsprochen.

Selbst wenn ich einen Screenshot mache bevor ich meine Stimme abgebe, wäre das kein Beweis. Ich hätte ja vor Abschicken nochmals ändern können.

Und da meine Stimme nach dem Übertragen ins 2. System gelöscht wird im 1. System, könnte niemand mehr, egal mit wieviel Aufwand, sagen welche Stimme ich abgab.

Meine Architektur wurde von einer Frau, welche den Vorschlag für das Schweizer eVoting System in der Luft zerpflückt hat, ebenfalls auseinander genommen.

Lest selbst was sie zu meiner Architektur meinte:

https://twitter.com/search?q=from%3A%40SarahJamieLewis%20to%3A%40ipinky77&src=typed_query