

# Online verkiezingen in de praktijk: verbetering en toepassing van het Helios verkiezingssysteem

Pieter Maene

Thesis voorge dragen tot het behalen  
van de graad van Master of Science  
in de ingenieurswetenschappen:  
elektrotechniek, optie Ingeb edde  
systemen en multimedia

**Promotor:**

Prof. dr. ir. B. Preneel

**Assessoren:**

Prof. dr. ir. V. Rijmen

Prof. dr. ir. L. Van Eycken

**Begeleiders:**

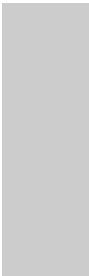
Dr. ir. J. Hermans

Dr. ir. F. Vercauteren

© Copyright KU Leuven

Zonder voorafgaande schriftelijke toestemming van zowel de promotor als de auteur is overnemen, kopiëren, gebruiken of realiseren van deze uitgave of gedeelten ervan verboden. Voor aanvragen tot of informatie i.v.m. het overnemen en/of gebruik en/of realisatie van gedeelten uit deze publicatie, wend u tot ESAT, Kasteelpark Arenberg 10 postbus 2440, B-3001 Heverlee, +32-16-321130 of via e-mail [info@esat.kuleuven.be](mailto:info@esat.kuleuven.be).

Voorafgaande schriftelijke toestemming van de promotor is eveneens vereist voor het aanwenden van de in deze masterproef beschreven (originele) methoden, producten, schakelingen en programma's voor industrieel of commercieel nut en voor de inzending van deze publicatie ter deelname aan wetenschappelijke prijzen of wedstrijden.



---

## Voorwoord

*Pieter Maene*

---

# Inhoudsopgave

Voorwoord . . . . .	i
Inhoudsopgave . . . . .	ii
Samenvatting . . . . .	iv
Lijst van figuren en tabellen . . . . .	v
Lijst van afkortingen en symbolen . . . . .	vii
1 Inleiding . . . . .	1
2 Literatuurstudie . . . . .	3
2.1 Geschiedenis [2] . . . . .	3
2.2 Vereisten . . . . .	4
<i>Vertrouwen</i> 4, <i>End-to-End Verifiability</i> 5	
2.3 Systemen zonder cryptografie . . . . .	5
<i>Open Counting</i> [1] 5, <i>Floating Receipts</i> [36] 6 , <i>ThreeBallot</i> [34] 7 , <i>Scratch-Card</i> 10	
2.4 Systemen met cryptografie . . . . .	11
<i>Secret-Ballot Receipts</i> [8] 12, <i>Scratch &amp; Vote</i> [5] 13	
2.5 Online stemmen [3] . . . . .	15
2.6 Conclusie . . . . .	16
3 Helios . . . . .	17
3.1 Cryptografische technieken . . . . .	17
<i>ElGamal</i> [15] 17, <i>Homomorfe encryptie</i> 18, <i>Threshold encryptie</i> 19	
3.2 Stemhokje . . . . .	20
3.3 Ballot Tracking Center . . . . .	21
3.4 Controleapplicatie . . . . .	21

<b>4</b>	<b>Procedure</b>	23
4.1	<i>Voorbereiding</i>	23
<i>Trustees</i>	23, <i>Vragen</i> 23, <i>Kiezers</i> 23	
4.2	<i>Helios</i>	24
<i>Aanmaken van de verkiezing</i>	24, <i>Trustees</i> 25, <i>Vragen</i> 26,	
<i>Kiezers</i> 26, <i>Bevriezen van het stembiljet</i> 28, <i>Telling</i> 28		
<b>5</b>	<b>Interface</b>	29
5.1	<i>Beheer</i>	29
5.2	<i>Trustee Dashboard</i>	31
5.3	<i>Controleapplicatie</i>	32
<b>6</b>	<b>Kringverkiezing</b>	37
6.1	<i>Vereisten [22][46]</i>	37
<i>Trustees</i> 37, <i>Vragen</i> 37, <i>Kiezers</i> 38		
6.2	<i>Aanpassingen</i>	38
<i>Shibboleth</i> 38, <i>Opkomst</i> 38, <i>Publicatie van het resultaat</i> 38		
6.3	<i>Testen</i>	38
<i>Beheer</i> 39, <i>Stemhokje</i> 40, <i>Stresstest</i> 41		
6.4	<i>Stemdag</i>	42
<b>7</b>	<b>Bewaren van sleutels en fingerprints</b>	43
7.1	<i>Sleutels</i>	43
<i>Disk</i> 43, <i>Web Storage [17][27]</i> 44		
7.2	<i>Fingerprints</i>	44
<b>8</b>	<b>Web Cryptography API</b>	47
8.1	<i>Web Cryptography API [39]</i>	47
8.2	<i>NfWebCrypto</i>	47
8.3	<i>Benchmarks</i>	48
<i>Modulaire exponentiatie</i> 48, <i>RSA</i> 49		
8.4	<i>Besluit</i>	50
<b>9</b>	<b>Besluit</b>	53
<b>A</b>	<b>English Paper</b>	55
	<b>Bibliografie</b>	61



---

## Samenvatting

Verkiezingen spelen een belangrijke rol in een democratische maatschappij. Kiezers moeten op een betrouwbare en gebruiksvriendelijke manier hun stem kunnen uitbrengen. Bij de huidige systemen moet de kiezer vertrouwen hebben in de organisatie die verantwoordelijk is voor de verkiezingen. Een voter-verifiable stemsysteem geeft de kiezer de mogelijkheid om na te gaan of zijn eigen stem correct geregistreerd is en of het resultaat correct is.

Het is mogelijk om een dergelijk systeem met papieren stembiljetten te implementeren, zowel met als zonder cryptografische technieken. Helios is een voter-verifiable stemsysteem voor online verkiezingen. De procedure die bij deze systemen gevuld moet worden, is echter vaak zeer complex.

In deze thesis wordt de procedure gegeven die in Helios gevuld moet worden om een verkiezing op te zetten. De interface van het systeem werd ook herwerkt om de beheerder beter te ondersteunen. Ook de applicatie die gebruikt wordt door de kiezers werd vereenvoudigd. Na het systeem uitgebreid getest te hebben, werd het in de praktijk gebruikt voor een reële verkiezing waarbij ongeveer 750 mensen hun stem uitbrachten.

Daarnaast wordt besproken hoe de trustees hun geheime sleutels kunnen bewaren. Er wordt ook onderzocht of de fingerprints waarvoor momenteel strings gebruikt worden niet op een betere manier weergegeven kunnen worden. Tot slot worden de prestaties van de Web Cryptography API vergeleken met deze van bestaande implementaties in JavaScript. Deze nieuwe specificatie geeft ontwikkelaars toegang tot cryptografische functies die in de browser ingebouwd zijn.

---

## Lijst van figuren en tabellen

### Lijst van figuren

2.1	Multi-Ballot[34] . . . . .	8
2.2	Scratch-Card biljet[30] . . . . .	10
2.3	Strookje met optische geëncrypteerde stem[8] . . . . .	12
2.4	Laatste stukje ticket met beide kanten nog samen[8] . . . . .	12
2.5	Scratch & Vote biljet[5] . . . . .	14
2.6	Scratch & Vote encryptie[5] . . . . .	15
3.1	Uitgebrachte stem . . . . .	21
4.1	Aanmaken van de verkiezing . . . . .	24
4.2	Trustees . . . . .	25
4.3	Threshold schema . . . . .	26
4.4	Vragen . . . . .	27
4.5	Uploaden van kiezers . . . . .	27
4.6	Admin . . . . .	28
5.1	Overzicht van de verkiezing (oud) . . . . .	30
5.2	Beheer van de verkiezing . . . . .	30
5.3	Voortgang procedure . . . . .	31
5.4	Trustee Dashboard . . . . .	32
5.5	Genereren van encrypted shares (oud) . . . . .	33
5.6	Genereren van encrypted shares . . . . .	33
5.7	Controleapplicatie (oud) . . . . .	34
5.8	Controleapplicatie . . . . .	35
6.1	Old (Left) and New (Right) Voting Booth Flows . . . . .	40
6.2	Bevestigen van de stem . . . . .	41
6.3	Resultaat van de stemming . . . . .	42
7.1	RoboHash van de fingerprint in Figuur 6.2 . . . . .	45

## LIJST VAN FIGUREN EN TABELLEN

---

8.1	Modulaire exponentiatie . . . . .	49
8.2	RSA . . . . .	51

## Lijst van tabellen

8.1	Modulaire exponentiatie . . . . .	50
8.2	RSA . . . . .	50



---

## Lijst van afkortingen en symbolen

### Lijst van afkortingen

DRE Direct-Recording Electronic  
VVPAT Voter-Verified Paper Audit Trail

### Lijst van symbolen

$pk$  Publieke sleutel  
 $sk$  Geheime sleutel  
 $df$  Decryptiefactor



## Hoofdstuk

### 1

---

## Inleiding

Verkiezingen zijn een essentieel onderdeel van het democratisch proces en spelen dus een zeer belangrijke rol in onze maatschappij. In 2014 vinden in 40 landen nationale verkiezingen plaats. 42% van de wereldbevolking zal dit jaar hun stem kunnen uitbrengen.<sup>[41]</sup> Ook in België is 2014 een groot verkiezingsjaar: op 25 mei werd tegelijk gestemd voor het Europees parlement, de Kamer van volksvertegenwoordigers en de parlementen van de deelstaten.

In 2012 werd tijdens de presidentiële verkiezingen in Amerika door beide kandidaten ongeveer één miljard dollar uitgegeven tijdens hun campagne.<sup>[43]</sup> In België werd tijdens de federale verkiezingen van 2010 door alle partijen samen 13,7 miljoen euro gespendeerd. Niet alleen de budgetten van de kandidaten zijn zo hoog: de kosten van de organisatie worden voor 25 mei op meer dan 10 miljoen euro geschat.<sup>[44]</sup>

Gezien het belang van de verkiezingen en de enorme bedragen die ermee gemoeid gaan, is het dus noodzakelijk dat er een betrouwbaar systeem is om het resultaat vast te leggen. Het moet bovendien eenvoudig te gebruiken zijn door de kiezers. Tijdens de Amerikaanse presidentsverkiezingen van 2000 werd in Palm Beach County, Florida een biljet gebruikt dat verwarring zou zijn. Dit kreeg des te meer aandacht omdat de uitslag erg nipt was.<sup>[53]</sup> In België stemt bijna de helft van de kiezers elektronisch, maar ook hier kan de interface voor problemen zorgen. Tijdens de gemeenteraadsverkiezingen van 2012 kon per ongeluk een voorkeurstem gegeven worden door te lang op het scherm te duwen.<sup>[23]</sup> In veel steden werden voor 25 mei stemcomputers ter beschikking gesteld om op te oefenen.<sup>[32]</sup>

Het is dus niet eenvoudig om een gebruiksvriendelijke oplossing te ontwikkelen, noch voor papieren biljetten, noch wanneer digitaal gestemd wordt. Daarnaast heeft de kiezer vandaag geen enkele manier om na te gaan of zijn stem juist meegeteld is en of het algemene resultaat correct is. Hij moet de instantie die de verkiezing organiseert dus volledig vertrouwen. Dit kan opgelost worden door gebruik te maken van een voter-verifiable systeem. Een groot nadeel is echter dat veel van deze systemen een complexe procedure hebben. Helios is een systeem dat het mogelijk maakt om online voter-verifiable verkiezingen te organiseren.

## 1. INLEIDING

---

In [Hoofdstuk 2](#) worden papieren systemen besproken die voter-verifiable zijn. Veel van de technieken die hier gebruikt worden, zullen ook terugkomen in Helios. [Hoofdstuk 3](#) tot [Hoofdstuk 5](#) behandelen de werking van dit systeem, de procedure die gevuld moet worden voor het opzetten van een verkiezing en de wijzigingen aan de interface. Het aangepaste systeem werd ook in de praktijk gebruikt om na te gaan hoe bruikbaar het is voor een echte verkiezing ([Hoofdstuk 6](#)).

In [Hoofdstuk 7](#) wordt kort gekeken naar methoden om de sleutels en fingerprints te bewaren. Tot slot worden de prestaties van de Web Cryptography API geëvalueerd ([Hoofdstuk 8](#)).

## Hoofdstuk

# 2

## Literatuurstudie

Deze thesis handelt over methoden voor online verkiezingen. Een interessant verwant probleem zijn systemen die gebruik maken van papier. In dit hoofdstuk wordt specifiek gekeken naar papieren voter-verifiable systemen.

Eerst wordt een kort overzicht van de geschiedenis van stemsystemen gegeven ([Sectie 2.1](#)). Vervolgens worden de voornaamste vereisten bekeken waaraan deze systemen moeten voldoen ([Sectie 2.2](#)). Belangrijk hierbij is de definitie van een voter-verifiable systeem. Tot slot worden zowel systemen onderzocht die geen gebruik maken van cryptografische methoden ([Sectie 2.3](#)) als deze die daar wel op steunen ([Sectie 2.4](#)).

### 2.1 Geschiedenis [2]

Onze samenleving heeft een rijke geschiedenis van stempelprocedures, die teruggaat tot Athene in het oude Griekenland. Hier bracht men een negatieve stem uit op een potscherf. In deze paragraaf bekijken we kort enkele methodes die bepalend zijn geweest voor de manier waarop we vandaag werken.[\[51\]](#)

Sinds de uitvinding van het geheime stembiljet in 1858 in Australië is er eigenlijk niet meer zoveel veranderd. In dit systeem worden de biljetten op voorhand gedrukt door de staat en veilig bewaard tot op de stemedag. Elke stemgerechtigde krijgt op de stemedag een biljet waarna hij zijn stem uitbrengt in een stembokje. Het grootste voordeel van deze methode ligt in het feit dat elke stem geheim is.

Deze stemmethode maakte het daarnaast ook mogelijk om mechanische (en later elektrische) machines te gebruiken. Mechanische systemen werden gebruikt in grotere gemeenschappen en waren gebaseerd op hendels en mechanische tellers. De eerste van deze machines werden in 1892 ingevoerd in New York. Rond 1960 werden de eerste elektrische machines ingevoerd. Deze maakten gebruik van optische scans. Bij deze systemen moest de stem meestal op een specifieke manier aangegeven worden, bijvoorbeeld door het inkleuren van bolletjes.

## 2. LITERATUURSTUDIE

---

Sinds 2000 worden DRE machines steeds vaker gebruikt. Hierop draait speciale stemsoftware die de keuze van de kiezer digitaal vastlegt. Deze machines maken het stemproces aanzienlijk eenvoudiger. Het grootste probleem is dat de kiezer geen enkele bevestiging krijgt en dat hij deze machines dus volledig moet vertrouwen.[48]

Het gebrek aan controle door de kiezer bij DRE vormde de aanleiding voor het ontwerpen van Voter-Verified Paper Audit Trails (VVPAT) machines. Hierbij toont de machine de kiezer een afgeschermde afdruk van zijn stem, waarna hij deze kan accepteren of weigeren. Op die manier kan de kiezer verifiëren dat zijn stem correct is. In principe zouden bij een hertelling dan ook deze papieren tickets en niet de digitale data gebruikt moeten worden.[54]

### 2.2 Vereisten

Bij het ontwerpen van een stemsysteem zijn er twee tegenstrijdige doelen. Enerzijds moet het mogelijk zijn dat de kiezer thuis kan controleren of zijn stem juist meegeteld is. Anderzijds mag diezelfde persoon niet kunnen bewijzen voor wie hij precies gestemd heeft, noch mag het mogelijk zijn dat anderen hierachter kunnen komen. Wanneer hij dit wel kan, zou hij zijn stem kunnen verkopen aan iemand die de verkiezing wil beïnvloeden of gedwongen kunnen worden om op een bepaalde kandidaat te stemmen.

Hoewel het vaak zeer moeilijk is om een grote verkiezing doorslaggevend te wijzigen, wordt stemfraude toch regelmatig geconstateerd.[2] Eén van de grote moeilijkheden is dat zowel kiezers als bijzitters corrupt kunnen zijn. Er kan dus van geen enkele deelnemer verwacht worden dat hij eerlijk is.

Het dus essentieel dat het stemsysteem vertrouwd wordt door de kiezers. Bij de huidige systemen moet hij dat echter meestal leggen bij de instantie die verkiezing organiseert ([Sectie 2.2.1](#)). Nieuwe technieken laten ons echter toe om end-to-end verifiable systemen te ontwikkelen, waar de kiezer zelf de correctheid van het resultaat kan nagaan ([Sectie 2.2.2](#)).

#### 2.2.1 Vertrouwen

De huidige manier van stemmen vereist dat de kiezer zeer veel vertrouwen legt in het gebruikte systeem. Zoals verder besproken wordt ([Sectie 2.2.2](#)), zijn er nieuwe ontwerpen waarbij de kiezer kan controleren of zijn stem correct meegeteld is. Deze systemen steunen vaak op moeilijke cryptografische technieken, die heel wat achtergrondkennis vragen om ze te begrijpen.

Een vereiste voor om het even welk stemsysteem is dat het vertrouwd wordt door een gemiddelde kiezer, de bijzitters, de publieke opinie en media. Opdat deze mensen een dergelijk systeem zouden vertrouwen, moeten de experts die het systeem goedkeuren dit op een eenvoudige manier aan hen kunnen uitleggen.[30] Daarom zullen eenvoudige

systemen die geen gebruik maken van cryptografie waarschijnlijk sneller aanvaard worden door een breed publiek.

#### 2.2.2 End-to-End Verifiability

In een end-to-end verifiably voting systeem wordt niet nagegaan of de code van de stemmachines volledig correct is. In plaats daarvan wordt wiskundig bewezen dat het resultaat correct is. Op die manier kan de moeilijke en vaak ondoorzichtige fysische chain-of-custody vermeden worden. Dit betekent ook dat iemand niet langer speciale toegang moet hebben om de resultaten te controleren. Om het even wie kan nagaan of de bewijzen correct zijn.

Dergelijke end-to-end verifiable systemen kunnen zowel met als zonder cryptografische technieken gerealiseerd worden. Cryptografie kan enerzijds gebruikt worden om stemmen te encrypteren, zodat ze zeker geheim blijven. Anderzijds geven sommige systemen een zero-knowledge bewijs dat aantoont dat de stemmen correct geteld zijn.

### 2.3 Systemen zonder cryptografie

In deze paragraaf worden enkele systemen besproken waarin geen gebruik gemaakt wordt van cryptografie. Open Counting ([Sectie 2.3.1](#)) is een techniek waarbij alleen de telfase aangepast wordt. Floating receipts ([Sectie 2.3.2](#)) kunnen de veiligheid van elk papieren stemsysteem sterk verbeteren. ThreeBallot ([Sectie 2.3.4](#)) en Scratch-Card zijn beiden voter-verifiable systemen die gebruik maken van papieren tickets. Twin ([Sectie 2.3.2](#)) bouwt verder op respectievelijk floating receipts. Vooral ThreeBallot wordt in detail besproken omdat de belangrijkste concepten van papier-gebaseerde voter-verifiable verkiezingen hierin aan bod komen.

#### 2.3.1 Open Counting [1]

Open counting vertrekt van de systemen zoals we ze vandaag kennen, maar de stemmen worden op een nieuwe manier geteld. Het stembiljet is aangepast om eenvoudig optisch geteld te worden. De stemmen worden nog steeds geteld door ambtenaren. Elke stem wordt op een scherm getoond aan verschillende telstations, elk met hun eigen hardware die het getoonde biljet filmt en analyseert. Ieder station geeft op regelmatige tijdstippen zijn huidig totaal en wanneer er onenigheid is, wordt het gedisputeerde biljet gezocht en het probleem opgelost.

Tijdens het tellen geeft ieder station ook een hash van hun opgenomen video. Hiervoor wordt een veilige hash-functie gebruikt. Deze hashes kunnen dan later gebruikt worden tijdens een geautomatiseerde audit om te controleren of er niet geknoeid is met de beelden. Dit proces is publiek en dus kan iedereen zijn eigen hardware meebrengen en de telling zelf uitvoeren. Het systeem wordt zo ontworpen dat een eenvoudige camera en computer volstaan. Ook deze waarnemers kunnen een hash van hun video publiceren om geloofwaardiger over te komen.

## 2. LITERATUURSTUDIE

---

De verschillende telstations controleren continu elkaar en ook de waarnemers kunnen achteraf onregelmatigheden melden. Omdat het systeem snel werkt, kan de telling in het stembureau zelf gehouden worden. Op die manier kunnen alle belanghebbenden aanwezig zijn en kan het transport van de ongetelde biljetten vermeden worden. Het transparante karakter en het gebruik van eenvoudige hardware kunnen het vertrouwen van kiezers in het systeem sterk vergroten.

Open counting is een relatief eenvoudige manier om de telprocedure transparanter te maken naar de kiezers. Elke stem moet echter afzonderlijk getoond worden, waardoor dit systeem alleen gebruikt kan worden wanneer het aantal biljetten beperkt is.

### 2.3.2 Floating Receipts [36]

Floating receipts zijn een waardevolle toevoeging voor elk voter-verifiable papieren telsysteem. Een doos met stembiljetten wordt aan de uitgang van het stembureau geplaatst. De kiezer maakt bij het buitengaan een kopie van een stembiljet dat hij hieruit trekt, vooraleer hij zijn eigen erbij legt. Hij neemt dus een willekeurig ticket mee dat niet het zijne is, maar hij kan dit ticket toch later gebruiken om te controleren of de stemprocedure correct verlopen is. Omdat de doos initieel leeg is, krijgen de eerste  $T$  kiezers geen ticket mee naar huis. Hierbij is  $T$  een constante die veel kleiner is dan het aantal kiezers, maar voldoende groot zodat  $1/T$  klein is.

Niemand weet dus met grote waarschijnlijkheid van wie hij het biljet gekopieerd heeft. Omdat de aanvaller geen betrouwbare methode heeft om alle kopieën van een ticket te bemachtigen, is het systeem bestendig tegen het vervangen van biljetten of het verkopen van een stem. Een nadeel is dat kiezer niet langer zijn eigen ticket heeft en dus misschien minder gemotiveerd is om te controleren of dit correct meegeteld is. Er wordt echter verondersteld dat een groot aantal kiezers dat toch nog steeds zal doen.

Om floating receipts te gebruiken, moeten de kiezers een extra stap volgen in de stemprocedure. De stemprocedure wordt dus complexer, maar dit kan verantwoord worden door de voordelen die deze techniek met zich meebrengt.

#### *Short Ballot Assumption*

Bij de *Short Ballot Assumption* (SBA) moet het aantal kandidaten op het biljet beperkt blijven. Wanneer er minder mogelijkheden zijn om een biljet in te vullen, wordt de kans kleiner dat iemand anders zijn biljet op identiek dezelfde manier invult. Het wordt dan voor een aanvaller moeilijker om een biljet aan een specifieke kiezer te koppelen.[9]

#### *Twin* [36]

Twin is een voter-verifiable uitbreiding van het klassieke systeem die gebruik maakt van floating receipts. Een traditioneel stembiljet wordt door elke kiezer individueel

ingevuld. Onderaan het stembiljet wordt een ID geplaatst, maar dit wordt verborgen door een kraslaag. Na het invullen wordt het biljet gecontroleerd door een machine die deze laag eraf haalt en het biljet in een doos deponeert. Alle kiezers na de  $T^{de}$  krijgen een ticket van een willekeurig biljet mee naar huis. Wanneer de stemming afgelopen is, worden alle verzamelde biljetten gepubliceerd op een bulletin board.

Twin is een heel eenvoudig systeem, zonder ingewikkelde wiskunde of specifieke regels voor het correct invullen van het stembiljet. Aangezien het ticket een kopie is van het biljet van iemand anders, kan een kiezer zijn stem niet verkopen. Daarnaast is het zowel voor de tellers als een aanvaller moeilijk om het resultaat ingrijpend te veranderen zonder gedetecteerd te worden. Er wordt immers verondersteld dat een groot aantal kiezers nagaat of zijn ticket correct op het bulletin board staan.

Het voordeel bij dit systeem is dat het gekende biljet behouden blijft. De kiezer moet dus geen nieuwe regels volgen bij het invullen ervan. Het is echter wel belangrijk dat de kiezer de procedure volgen. Dit moet duidelijk aangegeven worden door de bijzitters.

### 2.3.3 ThreeBallot [34]

ThreeBallot is een stemsysteem dat ontworpen werd door Ronald Rivest in 2006. Het systeem maakt gebruik van een speciaal stembiljet dat bestaat uit drie identieke delen. Er wordt gestemd door het invullen van rijen en het deponeren van kolommen. Door alle stembiljetten samen met een lijst van de kiezers op een publieke website (het bulletin-board) te plaatsen, wordt het systeem end-to-end verifiabel.

#### *Multi-Ballot*

De drie delen waaruit het stembiljet bestaat, kunnen ofwel op één blad geprint worden ofwel op meerdere met perforaties ertussen. De drie delen zijn identiek, op een willekeurige identifier na. De drie IDs op een multi-ballot hebben bovendien geen enkel verband met elkaar of met deze op de andere. In [Figuur 2.1](#) wordt een voorbeeld van een multi-ballot getoond.

Bij het invullen van het multi-ballot gelden de volgende regels. Elke rij van drie bolletjes komt overeen met één kandidaat. Om voor een kandidaat te stemmen, moet de kiezer exact twee bolletjes inkleuren. Om tegen te stemmen, moet er één bolletje aangeduid worden. In elke rij moet exact één of twee bolletjes ingevuld zijn, anders is het biljet ongeldig. Hoe de verschillende delen ingevuld zijn, maakt hierbij niet uit.

Omdat het belangrijk is voor de telling dat de kiezer deze regels juist volgt, moet hij na het invullen van het biljet dit invoeren in een controlesmachine. Wanneer het biljet niet correct ingevuld is, dan geeft de machine aan waar de kiezer een fout gemaakt heeft. Indien alles wel juist is aangeduid, dan wordt een rode streep geprint op het biljet waarna hij de aparte delen indient. Deze controlesmachine mag geen enkele opname maken van de ingevoerde biljetten.

## 2. LITERATUURSTUDIE

---

BALLOT		BALLOT		BALLOT	
President		President		President	
Alex Jones	<input type="radio"/>	Alex Jones	<input type="radio"/>	Alex Jones	<input type="radio"/>
Bob Smith	<input type="radio"/>	Bob Smith	<input type="radio"/>	Bob Smith	<input type="radio"/>
Carol Wu	<input type="radio"/>	Carol Wu	<input type="radio"/>	Carol Wu	<input type="radio"/>
Senator		Senator		Senator	
Dave Yip	<input type="radio"/>	Dave Yip	<input type="radio"/>	Dave Yip	<input type="radio"/>
Ed Zinn	<input type="radio"/>	Ed Zinn	<input type="radio"/>	Ed Zinn	<input type="radio"/>
3147524		7523416		5530219	

FIGUUR 2.1: Multi-Ballot[34]

Voordat hij de drie aparte biljetten afgeeft, moet de kiezer er willekeurig één uitkiezen waarvan hij een kopie meekrijgt als ticket. Het is het veiligste om dit te implementeren in de controlesmachine. Door de manier waarop het biljet ingevuld is, geeft het ticket geen informatie over hoe de kiezer gestemd heeft.

### *Tellen van de stemmen*

Wanneer de verkiezing afgelopen is, worden alle biljetten gescand en de gegevens op het bulletin board gepost. Merk op dat het eigenlijke biljet niet online gezet wordt, omdat de kiezer hier iets op zou kunnen schrijven. Ook een lijst van iedereen die deelgenomen heeft aan de stemming wordt geüpload. Een kiezer kan nu nagaan of zijn ticket ook op het bulletin board staat.

Omdat alle stemmen op het bulletin board staan, kan iedereen zelf de telling verifiëren. De stemmen kunnen zoals anders geteld worden, zij het met een kleine aanpassing. Omdat er twee bolletjes gekleurd zijn bij een voorstem en maar één bij een tegenstem, is het resultaat voor elke kandidaat vermeerderd met het aantal kiezers.

### *Integriteit*

Door het toevoegen van een ticket en het bulletin board kan de kiezer nagaan dat zijn stembiljet gepubliceerd is en dat het totale aantal geregistreerde biljetten klopt. Deze nieuwe controles zullen ons toelaten om verschillende vormen van fraude eenvoudig te detecteren. Het is ook belangrijk te kijken of zij zelf geen nieuwe zwakheden introduceren.

Het toevoegen van nieuwe stemmen is onmogelijk zonder ook de lijst met kiezers aan te passen. Daarnaast kunnen er ook geen stemmen bijgewerkt of verwijderd worden zonder dat er mogelijk een kiezer komt klagen dat zijn stem niet correct online staat. Grootschalige fraude wordt op deze manier onmogelijk.

Bij de **Three-Pattern** aanval, vraagt de koper aan de kiezer om alle drie de delen in een bepaald patroon in te vullen. Wanneer hij dit patroon dan niet terugvindt

op het bulletin board, wordt de kiezer niet betaald. Een mogelijke oplossing is het gebruik van een DRE machine. Deze print zelf de deelbiljetten in een willekeurig patroon nadat de kiezer zijn keuze heeft op een scherm.

Bemerk tot slot dat de controlesmachine die de geldigheid van de tickets nagaat zeer goed getest moet worden. Wanneer deze aangepast zou worden, kan ze bijvoorbeeld kiezers toelaten om voor een bepaalde kandidaat drie bolletjes te kleuren en voor een andere geen. Zo zouden die stemmen veel meer gewicht krijgen dan deze die zich wel aan de regels houden. Het is bovendien onmogelijk om dergelijke ongeldige patronen achteraf terug te vinden omdat de verschillende biljetten los van elkaar worden ingediend.

Tot slot zou een aanvaller kunnen betalen voor het ticket van de kiezer. Zo kan deze de correctheid van zijn stem niet meer nagaan. De aanvaller zou dan in theorie het biljet kunnen aanpassen dat op het bulletin board geplaatst werd. De kiezers moeten dus aangemoedigd worden om hun ticket niet af te geven. Deze aanval kan eenvoudig tegengegaan worden wanneer de kiezer zonder medeweten van de aanvaller een kopie maakt van het ticket. Voor een digitaal getekend ticket (bv. met een barcode) volstaat dit namelijk ook om klacht neer te leggen.

#### *Stemgeheim*

Zoals eerder aangehaald, bevat het ticket zelf geen informatie over hoe de kiezer gestemd heeft. Het mag echter ook niet mogelijk zijn om de drie deelbiljetten aan elkaar te linken. Het ID op het ticket zou anders gebruikt kunnen worden om uit te zoeken welk tripel van een bepaalde kiezer is. Een vereiste voor het systeem is ook dat niemand vooraf weet welke drie deelbiljetten zullen samenhoren. Een mogelijke oplossing hiervoor is om de delen apart te houden en er willekeurig drie te laten trekken door de kiezer.

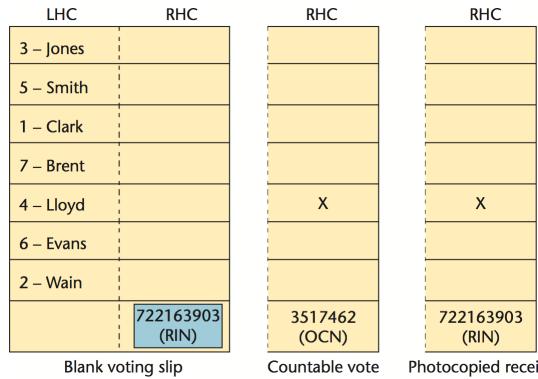
De kiezer mag zijn eigen multi-ballot niet kunnen reconstrueren op basis van de biljetten die op het bulletin board gepost worden. Dit kan opgelost worden door het ID te printen in de vorm van een 1D of 2D barcode, wat moeilijk te onthouden is. Het is ook verstandig om de kiezer geen vrije toegang te geven tot een kopieermachine bij het maken van het ticket. Dit is de reden dat het ticket best geprint wordt door de controlesmachine.

Het moet ook onmogelijk zijn voor de kiezer om zijn stem nog te wijzigen nadat ze aanvaard is door de controlesmachine. Een eerste oplossing is om hem geen fysische toegang meer te geven tot de biljetten nadat ze gecontroleerd zijn. Een tweede is om samen met de rode streep ([Sectie 2.3.3](#)) een checksum op het biljet te printen die moeilijk veranderd kan worden door de kiezer.

Tot slot moet er opgepast worden dat een **Reconstructie** aanval niet mogelijk is. Hierbij haalt een aanvaller alle mogelijke geldige multi-ballots uit de biljetten die op het bulletin board geplaatst werden. Samen met het ticket van de kiezer zou hij dan

## 2. LITERATUURSTUDIE

---



FIGUUR 2.2: Scratch-Card biljet[30]

in sommige gevallen kunnen achterhalen hoe deze gestemd heeft. Om de integriteit van de stemming te kunnen controleren, heeft de kiezer alleen het ticket van een geldig biljet nodig. Het is dus niet noodzakelijk dat hij het ticket van zijn eigen biljet mee naar huis neemt. Een mogelijke manier om dit te implementeren is door gebruik te maken van floating receipts ([Sectie 2.3.2](#)). Deze wordt niet expliciet vermeld in de paper van Rivest, maar hij bespreekt wel gelijkaardige methoden.[\[34\]](#)

### *Bruikbaarheid*

Het ThreeBallot systeem is veel complexer dan de manier waarop nu gestemd wordt. De belangrijkste manier om ervoor te zorgen dat het systeem goed werkt is dan ook het opleiden van de kiezer. Het is ook moeilijker om het biljet te corrigeren wanneer er een fout gemaakt wordt: meestal is de enige optie om opnieuw te beginnen met een blanco biljet. Het gebruik van DRE machine zou het stemmen ook sterk vereenvoudigen. De kiezer moet dan wel controleren dat het geprinte biljet correct is. In [Sectie 2.3.3](#) werd reeds aangegeven dat dit ook de ThreePattern aanval onmogelijk maakt.

Tot slot merken we nog op dat het tellen van de stemmen wel meer werk vraagt, aangezien er drie keer zoveel biljetten geteld moeten worden. ThreeBallot vergroot het vertrouwen van de kiezer in de integriteit van de verkiezing, ten koste van een moeilijker stemproces en meer werk bij het tellen.

### 2.3.4 Scratch-Card

Scratch-Card[\[30\]](#) maakt gebruik van een speciaal biljet dat makkelijk in twee gedeeld kan worden ([Figuur 2.2](#)). Belangrijk is dat de kandidaten op elk biljet in een willekeurige volgorde moeten staan. Een kiezer moet een willekeurig biljet trekken. Na het stemmen moet de kiezer het linkerdeel vernietigen. Hij kan een kopie van het rechterdeel als ticket mee naar huis nemen.

Op het rechterdeel van het biljet is onderaan een kraslaag aangebracht. Bovenop deze laag is het unieke ID (RIN) van het biljet geprint, dat de kiezer later kan gebruiken om zijn stem op het bulletin board terug te controleren. Bovendien verbergt deze laag een vooraf geprinte code die de volgorde van de kandidaten aangeeft (OCN). Bij het tellen wordt deze laag verwijderd en tegelijk verdwijnt ook het RIN van het biljet. Het is zeer belangrijk dat de RIN en OCN volledig ongecorreleerd zijn, want anders zou achterhaald kunnen worden van wie de stem is.

Omdat het ticket een kopie is van het biljet met het kraslaagje nog intact, kan de OCN nooit meer achterhaald worden. Het is dus belangrijk goed te controleren dat de tellers niet proberen om RIN/OCN-combinaties neer te schrijven tijdens het verwijderen van het laagje.

Een nadeel aan het voorgestelde systeem is dat iedereen die een origineel rechterdeel bemachtigt, kan achterhalen op wie dat biljet gestemd heeft. Er is gelukkig wel geen rechtstreekse link tussen de RIN en de persoon die gestemd heeft. Een alternatief systeem print daarom een ID op het linker- en rechterdeel (CIN). Op het rechterdeel zit deze CIN opnieuw onder een kraslaag. In deze variant moet de kiezer na het stemmen ook zijn linkerdeel in een doos deponeren. Als ticket krijgt hij opnieuw een kopie mee van het rechterkant, waarop de kraslaag nog intact was.

Om de stemmen te tellen, wordt opnieuw de kraslaag verwijderd zodat de CIN gelezen kan worden. Vervolgens moet de linkerkant met dezelfde CIN gevonden worden om te achterhalen op wie gestemd is. Het grootste probleem is dat het zoeken naar de juiste paren heel veel werk zou vragen bij grote verkiezingen, tenzij dit geautomatiseerd zou worden.

Net zoals bij Twin ([Sectie 2.3.2](#)) is het invullen van het biljet zeer eenvoudig voor de kiezer. Ook hier is het echter belangrijk dat de kiezers de juiste procedure volgen en een kopie nemen van hun biljet voordat de kraslaag verwijderd is. Ook hier moeten de bijzitters dus toezien op het correct verloop van de verkiezing.

## 2.4 Systemen met cryptografie

De systemen in de vorige sectie maakten geen gebruik van ingewikkelde cryptografische technieken. Omdat het hierdoor eenvoudiger te begrijpen is, zal zo'n systeem sneller vertrouwd worden door de kiezer ([Sectie 2.2.1](#)). In deze sectie worden toch enkele systemen besproken die hier wel op steunen. Door gebruik te maken van zero-knowledge bewijzen, homomorfe cryptografie en mixnets kunnen immers veilige end-to-end verifieerbare systemen ontworpen worden.

Bij Secret-Ballot Receipts ([Sectie 2.4.1](#)) wordt optische cryptografie toegepast om de stem op het ticket te encrypteren. Scratch & Vote ([Sectie 2.4.2](#)) bouwt verder op de principes die geïntroduceerd werden bij Scratch-Card ([Sectie 2.3.4](#)).

## 2. LITERATUURSTUDIE

---



FIGUUR 2.3: Strookje met optische geëncrypteerde stem[8]



FIGUUR 2.4: Laatste stukje ticket met beide kanten nog samen[8]

### 2.4.1 Secret-Ballot Receipts [8]

Secret-Ballot Receipts werden in 2004 gepubliceerd door David Chaum. De kiezer ziet zijn stem geprint worden in het stemhokje en kan zijn ticket gebruiken om nadien te controleren of ze correct meegeteld is. Omdat zijn keuzes geëncrypteerd worden tijdens het printproces kan hij het ticket niet gebruiken om te bewijzen hoe hij gestemd heeft. Bovendien is het niet nodig om vertrouwde hardware te gebruiken aangezien de publieke code op relatief eenvoudige systemen gedraaid kan worden.

Nadat de kiezer zijn keuzes aangegeven heeft, worden deze door een speciale printer afgedrukt. De printer drukt tegelijk op beide kanten van het strookje afzonderlijke, maar uitgelijnde afbeeldingen. De kiezer wordt gevraagd om de afdruk te controleren en kan zijn stem eventueel nog aanpassen. Wanneer hij tevreden is, kan hij kiezen of hij de boven- of onderkant wil meenemen. Pas dan wordt het laatste stukje van het ticket afgedrukt en kan hij de twee delen uit de printer nemen, terwijl ze nog aan elkaar vastzitten.

Door de twee kanten van elkaar los te maken, wordt de afbeelding op het strookje schijnbaar willekeurig. Het doorgelaten licht op de plaatsen waar geen van beide kanten bedrukt is, maakte de stem zichtbaar. Geen van beide lagen bevat dus informatie over hoe gestemd is. Het laatst geprinte stukje is verschillend omdat daar wel tekst opstaat die ook na het scheiden van de twee lagen nog gelezen kan worden. Op de ene kant wordt duidelijk aangegeven dat deze bijgehouden moet worden en op de andere dat hij afgegeven moet worden. Deze laatste wordt duidelijk zichtbaar voor de kiezer vernietigd.

De computer houdt zelf een digitale versie van het volledige ticket bij en verwijdert ook de data van de andere kant. Deze data wordt na het aflopen van de stemming geüpload naar een online bulletin board. Omdat het ticket geen informatie bevat over de stem van de kiezer, kan hij dit aan iedereen tonen zonder zijn stem openbaar te maken. Door het ticket te scannen kan eenvoudig vastgesteld worden of het authentiek is. Bij een ongeldige controle is men dus zeker dat de apparatuur niet correct gewerkt heeft.

De kiezer kan na de stemming nagaan of zijn ticket juist op het bulletin board staat. Hij kan dit eenvoudig doen door te kijken of de versie die daar staat volledig overeenkomt met zijn eigen ticket. Na het afsluiten van de stemming wordt de uiteindelijke verzameling van stemmen die geteld moeten worden, online gezet. Er worden ook digitale handtekeningen van de set gepubliceerd die gebruikt kunnen worden om de echtheid ervan te controleren. Wanneer de stemmen geteld zijn, wordt een nieuwe set online geplaatst. Deze bevat even veel biljetten, maar nu zijn afbeeldingen gedecrypteerd en is elke stem leesbaar. Om de privacy van de kiezer te bewaren, zijn de biljetten willekeurig geordend.

Er wordt gebruik gemaakt van een audit proces om te controleren of beide sets identiek dezelfde biljetten bevatten. Het telproces verloopt in verschillende stappen en na elke stap wordt een klein aantal willekeurig gekozen biljetten gedecrypteerd van de set tussen twee stappen in het telproces. Deze biljetten worden zo gekozen dat ze niet voldoende informatie bevatten zodat een kiezer geïdentificeerd kan worden, maar wel gebruikt kunnen worden om na te gaan of er geen biljetten toegevoegd, verwijderd of gewijzigd werden.

Omdat de optische encryptie neerkomt op een one-time pad, kan zelfs een aanvaller met ongelimiteerde rekenkracht de stem niet achterhalen. De gebruikte sleutels zijn dus de pixels van één van beide kanten. Deze zijn niet willekeurig, maar in de praktijk kunnen ze hiervan niet onderscheiden worden tenzij door de personen die de decryptie zullen uitvoeren.

Aangezien alles digitaal opgeslagen wordt, kan de telling ook voor grote verkiezingen efficiënt uitgevoerd worden. Een bijkomend voordeel is dat kiezers via een computer moeten stemmen, wat ze vaak reeds gewoon zijn. Hoewel de software op eenvoudige machines kan draaien, zijn er nog steeds speciale printers nodig om de tickets af te drukken.

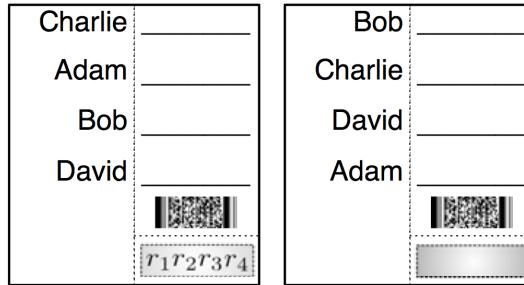
### 2.4.2 Scratch & Vote [5]

Scratch & Vote werd in 2006 ontworpen door Ben Adida en Ronald L. Rivest. Het is een variatie op Scratch-Card ([Sectie 2.3.4](#)) waarbij gebruik gemaakt wordt van homomorfe cryptografie en zero-knowledge correctheidsbewijzen. Iedereen kan het uiteindelijke resultaat verifiëren en alleen de cijfertekst van de uitslag moet gedecrypteerd worden door de verantwoordelijken van de verkiezing. Het grote verschil met Scratch-Card ([Sectie 2.3.4](#)) is dat er niet langer met een RIN/CIN gewerkt moet worden, net omdat de stemmen nu geëncrypteerd worden.

Bij het aanmelden ontvangt de kiezer een biljet dat uit twee delen bestaat. Op de linkerzijde staan de kandidaten in een willekeurige volgorde, die alleen door de kiezer gezien mag worden. Op de rechterkant kan de kiezer zijn stem uitbrengen. Onderaan dit deel staan verder een 2D barcode en een kraslaag. Net zoals bij Scratch-Card wordt de linkerkant na het invullen van het biljet in het stemhokje afgescheurd en in een doos gedeponeerd. Een bijzitter controleert of de kraslaag op

## 2. LITERATUURSTUDIE

---



FIGUUR 2.5: Scratch & Vote biljet[5]

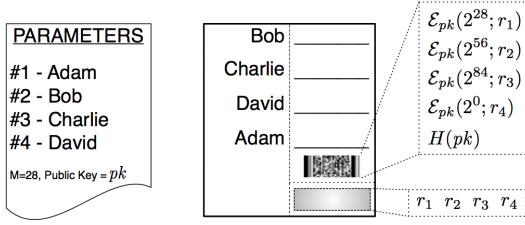
de rechterkant nog intact is en verwijderd deze daarna. Vervolgens wordt dit stukje zichtbaar voor de kiezer vernietigd. Tot slot laat de kiezer de eigenlijke stem en barcode scannen. Wanneer het stukje met de kraslaag verwijderd is van het biljet, bevat het rechterdeel geen informatie meer die gebruikt kan worden om de stem van de kiezer te achterhalen. Het gescande deel kan dus als ticket meegenomen worden.

Door aan te melden op het bulletin board, kan de kiezer controleren of zijn biljet correct gescand werd. Omdat alle gescande biljetten online geplaatst worden, kan iedereen nagaan of de cijfertekst van de eindtelling correct is.

Voor de encryptie wordt gebruik gemaakt van het Paillier public key cryptosysteem. Dit systeem heeft een additief homomorfisme door het vermenigvuldigen van de cijferteksten. Het tellen van de stemmen bij een verkiezing met meerdere kandidaten zou echter niet mogelijk zijn zonder gebruik te maken van een multi-counter. Het aantal beschikbare bits voor de leesbare tekst wordt onderverdeeld in verschillende tellers. Hierbij worden voldoende bits beschikbaar gemaakt voor elke teller zodat ze niet in elkaar kunnen overlopen.

Het systeem maakt daarnaast gebruik van zero-knowledge bewijzen. Deze worden gebruikt om aan te tonen dat een set cijferteksten  $c_1, c_2, \dots, c_l$  de encryptie is van de permutatie van  $m_1, m_2, \dots, m_k$  (ervan uitgaande dat geen twee subsets van  $m_i$  dezelfde som hebben). De verschillende  $m_i$  zijn de tellers voor de kandidaten.

Daarnaast worden ook bewijzen opgesteld die aantonen dat de biljetten zelf correct zijn. Omdat deze bewijzen te lang zijn om op de biljetten te printen, worden ze voor de start van de verkiezing geüpload naar het bulletin board. Ze worden tijdens het tellen van de stemmen gebruikt om te verzekeren dat elk biljet maar één stem uitbrengt per verkiezing. Om de kiezer te garanderen dat zijn biljet tijdens het tellen niet ongeldig verklaard zal worden, wordt ook een officiële lijst van alle geldige biljetten voorzien. De kiezer kan dan eenvoudig nagaan of zijn biljet hierop voorkomt.



FIGUUR 2.6: Scratch &amp; Vote encryptie[5]

De 2D barcode op elk biljet encodeert de willekeurige volgorde van de cijferteksten voor de verschillende kandidaten samen met een hash van de publieke sleutel (Figuur 2.6). De startwaarde van de teller van elke kandidaat wordt samen met een willekeurige waarde geëncrypteerd. Zo heeft elk biljet een unieke cijfertekst voor elke kandidaat. Deze willekeurige waarden worden verborgen door de kraslaag. De startwaarden van de tellers vormen samen met de publieke sleutel de gepubliceerde parameters. Samen met de willekeurige waarden kunnen ze dus gebruikt worden om de volgorde van de kandidaten op het biljet te achterhalen. Daarom is het belangrijk dat dit stukje van het biljet vernietigd wordt.

Deze informatie wordt toch op de biljetten geprint omdat ze nodig zijn voor een audit. Dit wordt gedaan door de kiezer twee biljetten te laten kiezen. Door de kraslaag weg te halen, worden de willekeurige waarden zichtbaar en kan nagegaan worden of het biljet correct is. Door het verwijderen van de kraslaag wordt het biljet ongeldig, wat de reden is dat de kiezer twee biljetten moet nemen. Op deze manier wordt de helft van de biljetten getest en dus is de kans groot dat foutieve biljetten gedetecteerd worden.

Zowel naar de kiezer als de bijzitters is dit systeem zeer gebruiksvriendelijk. Het standaard stembiljet is slechts licht gewijzigd en de kiezer zou er dus vertrouwd mee moeten zijn. Ook de stemprocedure is niet radicaal gewijzigd. Aangezien de telprocedure ook geautomatiseerd is, kan deze methode ook voor grote verkiezingen gebruikt worden.

## 2.5 Online stemmen [3]

Een mogelijkheid om mensen van op afstand te laten stemmen, is door een online verkiezingssysteem te gebruiken. Studies hebben aangetoond dat technologieën die het eenvoudiger maken om te stemmen, de opkomst kunnen verbeteren.[42] Het voornaamste probleem bij dit soort systemen is dat de omgeving niet langer gecontroleerd kan worden door de organisator van de verkiezing. Bij een online verkiezing kan de kiezer echter gedwongen worden om op een bepaalde manier te stemmen terwijl de aanvaller over zijn schouder meekijkt. Het is voor de kiezer ook eenvoudiger zijn stem te verkopen aangezien hij op dezelfde manier kan bewijzen aan de koper dat hij correct gestemd heeft.

## 2. LITERATUURSTUDIE

---

Helios lost dit probleem niet op, maar stelt dat er situaties zijn waar het gevaar op een aanval kleiner is omdat de inzet er minder groot is. Voorbeelden zijn verkiezingen bij een lokale sportclub of studentenorganisaties. Hier is echter wel nog steeds nood aan een betrouwbaar en geheim verkiezingssysteem. Een online verkiezingssysteem is hier dan ook ideaal, omdat de organisatie van de verkiezing anders veel meer werk zou vragen.

### 2.6 Conclusie

Bij open counting ([Sectie 2.3.1](#)) wordt een transparante manier van tellen gebruikt om het vertrouwen van de kiezer in het resultaat te vergroten. In tegenstelling tot de andere systemen kan hier door de kiezer wel niet nagegaan worden of zijn stem meegeteld is. Voter-verifiability wordt daar bekomen door gebruik te maken van een ander type biljet samen met een aangepaste stemprocedure.

Deze aanpassingen maken het stemmen voor de kiezer ingewikkelder. In tegenstelling tot bij een klassiek biljet, moeten nu verschillende regels gevuld worden om het biljet correct in te vullen. De stemprocedures zijn vaak ook ingewikkelder dan voordien, met nieuwe regels voor zowel de kiezers als de bijzitters.

Door cryptografische technieken te gebruiken, kan de gebruiksvriendelijkheid van papieren end-to-end verifiable systemen sterk verbeterd worden. Een nadeel is hier dan weer dat de meeste mensen nog steeds zullen moeten vertrouwen op het oordeel van een expert over de correctheid van het systeem.

Papieren voter-verifiable systemen hebben dus enkele grote nadelen die hun praktisch nut sterk beperken. Ze zijn vaak zeer complex en niet bruikbaar voor grote verkiezingen. Cryptografische technieken lossen deze problemen wel deels op, maar worden dan weer moeilijker vertrouwd door de kiezer.

Helios is in de eerste plaats een systeem dat toelaat om online verkiezingen te organiseren. Deze systemen kunnen echter alleen gebruikt worden wanneer het gevaar op dwang klein is. Het is hier immers niet mogelijk om de omgeving te controleren waarin de kiezer zijn stem uitbrengt, waardoor het voor een aanvaller eenvoudiger is om invloed uit te oefenen.

Het Helios verkiezingssysteem is een open-source project dat geleid wordt door Ben Adida.<sup>[3]</sup> Het laat toe om online voter-verifiable verkiezingen te organiseren. Online verkiezingen kunnen wel alleen gebruikt worden in een context waar dwang geen grote bedreiging is. Dit systeem werd vorig jaar door Robbert Coeckelbergh uitgebreid met threshold encryptie en gerangschikte verkiezingen.<sup>[10]</sup>

In dit hoofdstuk worden de belangrijkste cryptografische technieken uitgelegd die Helios gebruikt ([Sectie 3.1](#)). Daarna wordt de functionaliteit van de publieke delen van Helios besproken: het stemhokje ([Sectie 3.2](#)), het ballot tracking center ([Sectie 3.3](#)) en de controleapplicatie ([Sectie 3.4](#)).

### 3.1 Cryptografische technieken

Op het vlak van cryptografische technieken leunt Helios het dichtste aan bij Scratch & Vote ([Sectie 2.4.2](#)). Het belangrijkste verschil is dat ElGamal gebruikt wordt in plaats van Paillier voor de homomorfe encryptie van de stemmen. Tot slot wordt de methode besproken waarop de sleutel verdeeld wordt tussen de trustees ([Sectie 3.1.3](#)).

#### 3.1.1 ElGamal [15]

Het ElGamal cryptosysteem is een asymmetrisch schema dat gebaseerd is op het Diffie-Hellman protocol. Dit betekent dat een sleutelpaar met zowel een geheime als publieke sleutel nodig is. De publieke sleutel wordt gebruikt om de klaartekst te encrypteren. De decryptie kan alleen uitgevoerd worden met de geheime sleutel.

Er wordt gewerkt in de groep  $\mathbb{Z}_p$  waar  $g$  de generator is. De geheime sleutel  $sk$  wordt willekeurig gekozen binnen  $\mathbb{Z}_{p-1}$ . De publieke sleutel is dan  $pk = g^{sk} \bmod p$ . De cijfertekst van een ElGamal encryptie bestaat uit twee delen:  $c_1$  ([Vergelijking 3.1](#)) en  $c_2$  ([Vergelijking 3.2](#)). In deze vergelijkingen is  $m$  de klaartekst en  $r$  opnieuw een willekeurig getal binnen  $\mathbb{Z}_{p-1}$ .  $c_1$  en  $r$  hebben respectievelijk de functie van tijdelijke publieke en private sleutel.<sup>[29]</sup>

$$c_1 = g^r \mod p \quad (3.1)$$

$$c_2 = m \cdot pk^r \mod p \quad (3.2)$$

De cijfertekst kan dan gedecrypteerd worden volgens [Vergelijking 3.3](#).

$$m = \frac{c_2}{c_1^{sk}} \mod p \quad (3.3)$$

### 3.1.2 Homomorfe encryptie

Bij homomorfe encryptie kan een specifieke operatie met de cijfertekst uitgevoerd worden. De resulterende cijfertekst is de encryptie van een bepaalde bewerking op de klaarteksten.[\[49\]](#) Zo is het Paillier cryptosysteem dat gebruikt wordt in Scratch & Vote ([Sectie 2.4.2](#)) homomorf onder  $(\times, +)$ . Dit betekent dat een vermenigvuldiging van de cijferteksten resulteert in een optelling van de klaarteksten.

Het homomorfisme  $(\times, +)$  kan in een verkiezingssysteem gebruikt worden om efficiënt de stemmen op te tellen. De berekening van het resultaat kan immers gebeuren aan de hand van de cijferteksten. Er moet nu alleen een decryptie gebeuren om het uiteindelijke resultaat vrij te geven.

Aan de hand van [Vergelijking 3.2](#) kan gezien worden dat ElGamal standaard homomorf is onder  $(\times, \times)$ . Zoals hiervoor besproken, is voor een verkiezingssysteem echter het homomorfisme  $(\times, +)$  nodig. Dit kan gerealiseerd worden door de klaartekst ook in de exponent te plaatsen ([Vergelijking 3.4](#)).

$$c_2 = g^m \cdot pk^r \mod p \quad (3.4)$$

[Vergelijking 3.5](#) geeft het homomorfisme dat zo bekomen wordt.

$$\begin{aligned} \mathcal{E}(m_1) \cdot \mathcal{E}(m_2) &= (g^{r_1}, g^{m_1} \cdot pk^{r_1}) \cdot (g^{r_2}, g^{m_2} \cdot pk^{r_2}) \\ &= (g^{r_1+r_2}, g^{m_1+m_2} \cdot pk^{r_1+r_2}) \\ &= \mathcal{E}(m_1 + m_2) \end{aligned} \quad (3.5)$$

Omwille van het discreet logaritme probleem is het terugvinden van  $m$  echter niet meer zo vanzelfsprekend.[\[24\]](#) Dit kan alleen gedaan worden door  $g^m \mod p$  te berekenen voor elke  $m$  en vervolgens te zoeken welke hetzelfde is als de klaartekst van de decryptie.

Scratch & Vote ([Sectie 2.4.2](#)) gebruikt multi-counters voor een stemming met meerdere kandidaten. Helios daarentegen encrypteert ieder mogelijk antwoord op een vraag afzonderlijk. Wanneer de optie gekozen wordt, is  $m = 1$ ; anders wordt  $m = 0$  gesteld.

### 3.1.3 Threshold encryptie

Oorspronkelijk kon de publieke sleutel voor de verkiezing alleen berekend worden als het product van de afzonderlijke publieke sleutels van de trustees ([Vergelijking 3.6](#)). Voor de decryptie moet iedere trustee zijn factor uit de noemer van [Vergelijking 3.7](#) berekenen. Deze factoren worden in Helios de decryptiefactoren genoemd.

$$PK = \prod_{i=1}^n pk_i \mod p \quad (3.6)$$

$$g^m = \frac{c_2}{\prod_{i=1}^n c_1^{sk_i}} = \frac{c_2}{\prod_{i=1}^n df_i} \mod p \quad (3.7)$$

Een groot probleem hierbij is dat wanneer één trustee zijn geheime sleutel verliest, het resultaat niet meer gedecrypteerd kan worden. Daarom werd threshold encryptie toegevoegd door Robbert Coeckelbergh.[\[10\]](#) Er kan nu een threshold schema gedefinieerd worden zodat slechts  $k$  van de  $n$  trustees hun decryptiefactor moeten berekenen.

#### *Secret Sharing*

De methode die geïmplementeerd werd is gebaseerd op Shamir's secret sharing.[\[38\]](#) Iedere trustee genereert eerst een veelterm van graad  $k - 1$ . Vervolgens stuurt hij elke trustee (ook zichzelf) een zogeheten *share* van deze veelterm. Deze share is de waarde van de veelterm voor een punt  $x$ . Deze  $x$ -coördinaat moet door iedereen gekend zijn, omdat het vereist is dat de shares die een trustee ontvangt van de anderen voor dezelfde waarde werden aangemaakt. Daarom wordt hiervoor binnen Helios het database ID van de trustee gebruikt. Dit is een uniek natuurlijk getal dat wordt opgehoogd telkens een nieuwe trustee aangemaakt wordt.

Vervolgens moet de trustee de  $n$  shares die hij zo ontvangt, optellen. Zo bekomt hij de  $y$ -coördinaat die hoort bij zijn ID op een nieuwe veelterm, die de som is van de  $n$  veeltermen die door de trustees gegenereerd werden. Deze waarde kan hij nu gebruiken als zijn geheime sleutel  $sk$ . Omdat ElGamal gebruikt wordt als encryptieschema, wordt zijn publieke sleutel  $pk = g^{sk} \mod p$ .

Als geheime sleutel voor de verkiezing wordt nu de waarde voor  $x = 0$  op de gemeenschappelijke veelterm genomen. Deze veelterm kan door Lagrange-interpolatie gereconstrueerd worden uit  $k$  punten. Hiervoor worden de eerste  $k$  trustees gebruikt (dat zijn deze met het laagste ID). Omdat alleen de publieke sleutels van de trustees beschikbaar zijn, wordt echter onmiddellijk de publieke sleutel voor de verkiezing berekend ([Vergelijking 3.11](#)).

$$\lambda_i(x) = \prod_{j=1, j \neq i}^k \frac{x - x_j}{x_i - x_j} \quad (3.8)$$

$$V(x) = \sum_{i=1}^k sk_i \lambda_i(x) \quad (3.9)$$

$$SK = V(0) = \sum_{i=1}^k sk_i \lambda_i(0) \quad (3.10)$$

$$PK = g^X = \prod_{i=1}^k pk_i^{\lambda_i(0)} \mod p \quad (3.11)$$

Om het resultaat te decrypteren moet de geheime sleutel voor de verkiezing gebruikt worden ([Vergelijking 3.10](#)). Het grote voordeel van threshold encryptie is dat het hier niet belangrijk is van welke  $k$  trustees de decryptiefactoren en bijhorende Lagrange-interpolatie gebruikt worden.

$$g^m = \frac{c_2}{\prod_{i=1}^k c_1^{sk_i \lambda_i(0)}} = \frac{c_2}{\prod_{i=1}^k df_i^{\lambda_i(0)}} \mod p \quad (3.12)$$

### *Communicatiesleutels*

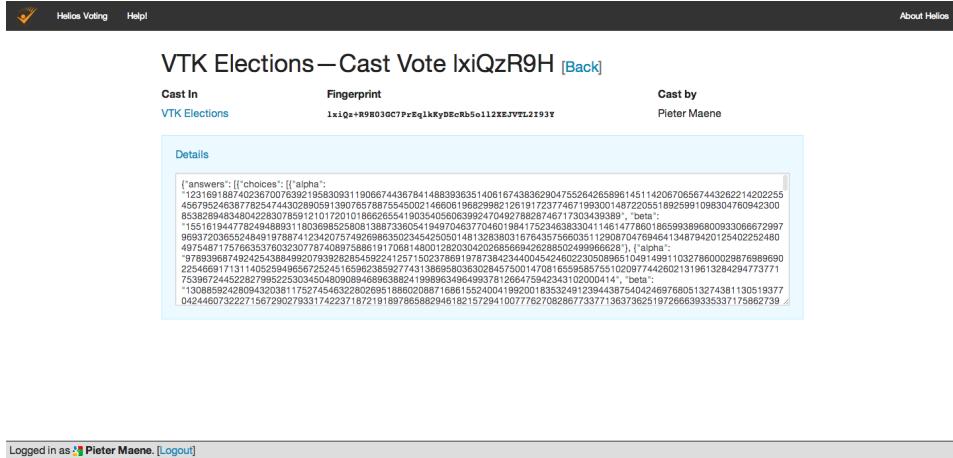
Voordat iedere trustee zijn gegenereerde shares doorstuurt naar de andere trustees, worden deze geëncrypteerd en getekend. Hiervoor wordt respectievelijk de publieke sleutel voor encryptie en voor tekenen van de andere trustee gebruikt. Dit geeft aanleiding tot twee nieuwe sleutelparen die de communicatiesleutels genoemd worden.

## 3.2 Stemhokje

Het stemhokje staat volledig los van de rest van het systeem. Het is een applicatie die de kiezer kan gebruiken om zijn stem uit te brengen. Nadat hij zijn keuze gemaakt heeft, wordt deze geëncrypteerd met de publieke sleutel voor de verkiezing. Zoals besproken in [Sectie 3.1](#), zal de cijfertekst bestaan uit twee delen ([Vergelijking 3.1](#) en [Vergelijking 3.4](#)). Wanneer threshold encryptie aanstaat, wordt de publieke sleutel zoals gedefinieerd door [Vergelijking 3.11](#) gebruikt; anders is het deze van [Vergelijking 3.6](#).

Het encryptieproces is in JavaScript geïmplementeerd en vindt dus volledig plaats in de browser van de kiezer. De stem van de kiezer zal dan nooit ongeëncrypteerd op de server toekomen. Alle informatie over de verkiezing kan eenvoudig opgevraagd worden. De kiezer zou dus ook zijn eigen software kunnen gebruiken om zijn stem te encrypteren.

### 3.3. Ballot Tracking Center



FIGUUR 3.1: Uitgebrachte stem

Aan elke geëncrypteerde stem wordt een *Smart Ballot Tracker* toegekend. Dit is een fingerprint die de stem identificeert ([Sectie 7.2](#)). Aan de hand daarvan kan de kiezer zijn stem volgen doorheen het systeem en kan hij verifiëren dat zijn stem niet aangepast is.

## 3.3 Ballot Tracking Center

Net zoals bij Scratch & Vote ([Sectie 2.4.2](#)) worden de uitgebrachte stemmen gepubliceerd. In het ballot tracking center kan een lijst met alle kiezers teruggevonden worden. Bij een publieke verkiezing kan iedereen deze opvragen, bij een private alleen de geregistreerde kiezers. Standaard staan hier gewoon de namen van de kiezers, maar de beheerder kan ervoor kiezen om aliasen te gebruiken in de plaats.

Van zodra een kiezer zijn stem uitgebracht heeft, kan deze bekijken worden ([Figuur 3.1](#)). Dit is nodig omdat zo het resultaat van de verkiezing gecontroleerd kan worden ([Sectie 3.4](#)). Al deze informatie kan ook opgevraagd worden als JSON, zodat ze eenvoudiger verwerkt kan worden door een applicatie.

## 3.4 Controleapplicatie

Net zoals het stembokje is de controleapplicatie onafhankelijk en draait ze lokaal in de browser van de gebruiker. Ze kan gebruikt worden om het resultaat van de verkiezing te controleren. Alle ruwe data wordt gedownload van de server, waarna de nodige berekeningen lokaal worden uitgevoerd.



Dit hoofdstuk geeft de procedure die in de aangepaste versie van Helios ([Hoofdstuk 5](#)) gevuld moet worden om een verkiezing op te zetten. In [Sectie 4.1](#) wordt bekeken welke informatie gekend moet zijn voordat met het aanmaken van de verkiezing aangevat wordt. De procedure zelf wordt in [Sectie 4.2](#) beschreven.

### 4.1 Voorbereiding

#### 4.1.1 Trustees

De trustees zijn de personen die verantwoordelijk zullen zijn voor het decrypteren van het resultaat. Aangezien zijn een belangrijke rol spelen in het verkiezingsproces, is het belangrijk vooraf vast te leggen wie dit zal doen. Helios heeft ondersteuning voor threshold encryptie ([Sectie 3.1.3](#)). Er moet dus ook nagedacht worden over het threshold schema: hoeveel trustees nodig zullen zijn om het resultaat van de verkiezing te decrypteren. Om een trustee aan te maken, moeten zijn naam en een geldig e-mailadres opgegeven worden.

#### 4.1.2 Vragen

Het belangrijkste van de verkiezing zijn uiteraard de vragen die de kiezers zullen moeten beantwoorden. Bij elke vraag kan in Helios aangegeven worden hoeveel antwoorden elke kiezer kan aanduiden, dus ook dit moet op voorhand bepaald worden. Het resultaat kan ofwel absoluut ofwel relatief weergegeven worden. In het eerste geval zal bij elke optie het aantal stemmen voor deze optie getoond worden, in het tweede alleen de relatieve plaatsen van de opties onderling.

#### 4.1.3 Kiezers

Een verkiezing in Helios kan opengesteld worden voor iedereen of voor specifieke kiezers. Bij een gesloten verkiezing moet op voorhand een lijst van geldige kiezers opgesteld worden.

## 4. PROCEDURE

---

The screenshot shows the 'Create a New Election' page of the Helios Voting software. At the top, there are links for 'Helios Voting' and 'Help!', and a 'About Helios' link in the top right. Below that is a 'CREATE' button. The main form has a title 'Create a New Election'. It contains several input fields and dropdown menus:

- 'Name' input field
- 'Short Name' input field
- 'Election Type' dropdown menu set to 'Election'
- 'Use Voter Aliases' checkbox
- 'Randomize Answer Order' checkbox
- 'Private' checkbox
- 'Use Threshold Encryption' checkbox
- 'Voting Starts at' input field
- 'Voting Ends at' input field
- 'Publish Tally at' input field
- 'Help E-mail Address' input field
- 'Description' input field

At the bottom right of the form is a 'Next' button.

At the very bottom of the page, a status bar says 'Logged in as Pieter Maene. [Logout]'

FIGUUR 4.1: Aanmaken van de verkiezing

Deze kiezers kunnen in Helios geïmporteerd worden door een CSV-bestand aan te maken. Helios heeft van elke kiezer de volledige naam en een uniek ID nodig. Het e-mailadres van de kiezer kan opgegeven worden, maar dit is niet vereist. Er kan ook aangegeven worden welk authenticatiesysteem (bv. Google of Shibboleth) gebruikt wordt door de kiezer. Wanneer dit laatste veld leeg is, zal een gegenereerd wachtwoord naar hem opgestuurd worden.

## 4.2 Helios

### 4.2.1 Aanmaken van de verkiezing

De eerste stap is het aanmaken van de verkiezing. Hier moet eerst de basisinformatie van de verkiezing opgegeven worden zoals de naam en het type verkiezing. Ofwel kan er een verkiezing aangemaakt worden, ofwel een referendum. Voor beide types is de procedure echter identiek.

Vervolgens kunnen een aantal belangrijke functies aan- of uitgezet worden. Wanneer aliasen gebruikt worden, dan worden de namen van de kiezers verborgen in het publieke ballot tracking center ([Sectie 3.3](#)). Het is ook mogelijk om de antwoorden in een random volgorde op het biljet te laten zetten. Een geheime verkiezing kan alleen bekeken worden door de geregistreerde kiezers. Hier moet ook aangegeven worden of threshold encryptie ([Sectie 3.1.3](#)) gebruikt zal worden.

## 4.2. Helios

The screenshot shows the Helios Voting interface with the following details:

- Header:** Helios Voting Help
- Navigation:** CREATE ADMIN TRUSTEES About Helios
- Title:** VTK Elections—Trustees [Back]
- Text (above table):** Trustees are responsible for decrypting the election result. Since this election uses threshold encryption, all trustees will have to go through a key ceremony. During this ceremony, each trustee receives part of the key that will be used to encrypt the ballots. When freezing the election, you will have to define the threshold scheme. This controls the number of trustees that will have to provide their secret key before the results can be decrypted.
- Text (below table):** Each trustee will be given a link to his/her personal dashboard. If a trustee lost his link, you can resend it by hitting [Send Link](#) below.
- Text (below table):** Helios is automatically your first trustee and will handle its key pair generation and decryption automatically. You may add additional trustees if you want, and you can even remove the Helios trustee. However, we recommend you do this only if you have a solid understanding of the trustee's role.
- Text (info bar):** Next: After freezing the trustee list, the threshold scheme can be defined.
- Buttons:** Add a Trustee > Freeze Trustee List <
- Table:** Public Keys

Trustee	Public Key for Signing	Public Key for Encryption
[x] [Send Login] Robin Ska	Not uploaded yet.	Not uploaded yet.
[x] [Send Login] Jeroen Van Hemelen	Not uploaded yet.	Not uploaded yet.
[x] [Send Login] Pieter Maene	Not uploaded yet.	Not uploaded yet.
[x] [Send Login] Daan Wendelen	Not uploaded yet.	Not uploaded yet.
[x] [Send Login] Daniel Slenders	Not uploaded yet.	Not uploaded yet.
[x] [Send Login] Tom Van der Voorde	Not uploaded yet.	Not uploaded yet.

- User Info:** Logged in as Pieter Maene. [Logout]

FIGUUR 4.2: Trustees

Tot slot is het mogelijk om aan te geven wanneer de verkiezing moet starten en sluiten. Wanneer het resultaat niet onmiddellijk na het aflopen van de verkiezing gepubliceerd mag worden, kan een alternatief tijdstip hiervoor opgegeven worden.

### 4.2.2 Trustees

Eens de nieuwe verkiezing aangemaakt is, moeten de trustees toegevoegd worden. Deze stap is eerst gezet in de procedure omdat hij veruit de meeste tijd in beslag neemt. De beheerder van de verkiezing moet eerst alle trustees aanmaken. De trustee krijgt dan ook meteen een link toegestuurd via e-mail naar zijn trustee dashboard ([Sectie 5.2](#)) waar hij al zijn acties zal moeten uitvoeren.

#### *Verkiezing zonder threshold encryptie*

Wanneer geen threshold encryptie gebruikt wordt, is dit proces eenvoudig. Van zodra een trustee aangemaakt is, kan hij het sleutelpaar genereren dat gebruikt zal worden als deel van de sleutel voor de verkiezing. Alle trustees moeten dit gedaan hebben voordat de beheerder de verkiezing kan bevriezen en openstellen voor de kiezers. Hij kan ondertussen echter wel verdergaan met de procedure.

#### *Verkiezing met threshold encryptie*

Wanneer alle trustees aangemaakt zijn, kan deze lijst bevroren worden. Op dit moment moet ook het threshold schema ingegeven worden. Wanneer dit gebeurd is,

## 4. PROCEDURE

---

The screenshot shows a web-based voting application interface. At the top, there's a navigation bar with icons for Helios Voting and Help, and a link to About Helios. Below this, the title "VTK Elections—Threshold Scheme" is displayed, along with a "[Back]" link. A message indicates that there are 6 trustees for the election and that the user can now specify the number of trustees required to decrypt the result. It specifies that any number between 1 and 6 can be submitted. A text input field labeled "Trustees for Decryption" contains the value "3". Below this is a blue "Submit" button. At the bottom of the page, a status bar shows "Logged in as: Pieter Maene" and a "[Logout]" link.

FIGUUR 4.3: Threshold schema

begint voor de trustees de sleutelceremonie. Deze bestaat uit drie stappen.

1. De trustees moeten eerst de twee sleutelparen genereren die gebruikt zullen worden om hun onderlinge communicatie te beveiligen. Pas wanneer alle trustees deze stap uitgevoerd hebben, kan verdergaan worden.
2. Elke trustee maakt vervolgens zijn shares aan en encrypteert deze voor de andere trustees. Hier moet opnieuw gewacht worden totdat alle trustees deze stap voltooid hebben.
3. De trustees kunnen nu de shares die ze van de andere gekregen hebben decrypteren en optellen. Zo bekomen ze het sleutelpaar dat later gebruikt zal kunnen worden om de encryptiesleutel van de verkiezing de reconstrueren.

Pas wanneer alle trustees de volledige sleutelceremonie doorlopen hebben, kan de verkiezing door de beheerder bevoren worden. Omdat de verschillende trustees dus vaak op elkaar moeten wachten, krijgen ze een e-mail elke keer de volgende stap uit de ceremonie aangevat kan worden. Tijdens het wachten op de trustees, kan de beheerder verdergaan met de procedure.

### 4.2.3 Vragen

Terwijl de trustees bezig zijn met het uitvoeren van de sleutelceremonie, kan het stembiljet wel aangemaakt worden. Alle informatie die hiervoor nodig is, werd reeds voorbereid ([Sectie 4.1.2](#)).

### 4.2.4 Kiezers

Ook deze stap werd reeds volledig voorbereid ([Sectie 4.1.2](#)). Hier kunnen de CSV-bestanden geüpload worden, waarna het systeem ze verwerkt en alle kiezers toevoegt aan de verkiezing.

## 4.2. Helios

The screenshot shows the Helios Voting interface. At the top, there is a navigation bar with links for 'CREATE', 'ADMIN', 'TRUSTEES', 'ADMIN', and 'QUESTIONS'. On the right side of the top bar, there is a link 'About Helios'. Below the navigation bar, the page title is 'VTK Elections—Questions [Back]'. A question is listed: '1. Would you vote for or against the Pixel team as board candidates for VTK in the year 2014-2015? [x] [Edit]'. Below the question, it says 'Voters can select between 1 and 1 answers.' and 'Result Type: Absolute'. There are three options: '1. For', '2. Against', and '3. Abstain'. Below this, there is a section titled 'Add a Question' with a placeholder 'Question' field containing 'Which fun team would you prefer to win?'. Underneath, there are five 'Answer' fields with the values 'Nautilus', 'Rebellion', an empty field, an empty field, and an empty field. At the bottom of this section are two buttons: 'Add Question' and 'Add 5 More Answers'. At the very bottom of the page, there is a status bar with the text 'Logged in as Pieter Maene. [Logout]'.

FIGUUR 4.4: Vragen

The screenshot shows the Helios Voting interface. At the top, there is a navigation bar with links for 'CREATE', 'ADMIN', 'TRUSTEES', 'ADMIN', and 'QUESTIONS'. On the right side of the top bar, there is a link 'About Helios'. Below the navigation bar, the page title is 'VTK Elections—Bulk Upload Voters [Back]'. A note says: 'If you would like to specify your list of voters by name and e-mail address, you can bulk upload a list of such voters here. Please prepare a text file of comma-separated values with the fields: <voter\_id>[,<email>],<full\_name>[,<user\_type>]'. It also states: 'If the system cannot validate the second field as an e-mail address, it will assume that it specifies the user's full name instead. The user type field is optional and can be used to specify the type of user account that should be created for the voter. If left empty, a password will be generated for the voter.' Below this, there is a section for examples with the text: 'For example:  
bobsmith,bob@acme.org,Bob Smith  
...'. A note says: 'The easiest way to prepare such a file is to use a spreadsheet program and export as "CSV".'. There is a 'List of Voters' input field with the placeholder 'Choose File' and a note 'No file chosen'. Below this is a large gray area for file upload, with a 'Upload' button at the bottom. At the very bottom of the page, there is a status bar with the text 'Logged in as Pieter Maene. [Logout]'.

FIGUUR 4.5: Uploaden van kiezers

## 4. PROCEDURE

The screenshot shows the Helios Voting Admin interface. At the top, there's a navigation bar with links for CREATE, ADMIN, TRUSTEES, ADMIN, QUESTIONS, VOTERS, ADMIN, and VIEW. The title 'VTK Elections—Admin [Archive]' is displayed. Below the title, a paragraph of text provides instructions for setting up the election. A large numbered list follows, detailing the steps from creating the election to releasing the tally. A blue banner at the bottom of the list states 'This election is complete.' Below this, there's a section titled 'Embed an Election Badge' with instructions and a code snippet for adding an election badge to a website. A footer bar at the bottom shows the user is logged in as Pieter Maene and includes a logout link.

FIGUUR 4.6: Admin

### 4.2.5 Bevriezen van het stembiljet

Wanneer alle voorgaande informatie ingegeven is en de sleutelceremonie afgerond, kan het stembiljet bevroren worden. Wanneer er geen tijdsstip gegeven is waarop de verkiezing moet starten, kunnen de kiezers onmiddellijk beginnen stemmen. Wanneer geen einddatum opgegeven is, blijft ze open totdat de beheerder ze sluit.

### 4.2.6 Telling

De stemming kan steeds manueel afgesloten worden door het telproces te starten. De stemmen worden dan homomorf samengegeteld in een geëncrypteerde resultaat ([Sectie 3.1.2](#)). Vooraleer dit gedecrypteerd kan worden, moeten we opnieuw wachten op de trustees.

Wanneer geen threshold encryptie gebruikt wordt, moeten alle trustees nu hun decryptiefactor berekenen en uploaden. Wordt dit wel gebruikt, dan moeten slechts  $k$  van de  $n$  trustees dit doen. Zodra alle nodige decryptiefactoren beschikbaar zijn, kan de beheerder het resultaat vrijgeven. Indien geen publicatiedatum opgegeven werd, is het resultaat dan voor iedereen beschikbaar. Anders kan de beheerder het resultaat wel reeds bekijken, maar wordt het pas gepubliceerd op die datum.

Het verbeteren van de gebruiksvriendelijkheid van het Helios verkiezingssysteem was het doel van deze thesis. Er zijn op vier belangrijke plaatsen wijzigingen aangebracht. Veruit het meeste werk was nodig om de procedure van [Hoofdstuk 4](#) duidelijker te maken in het beheer ([Sectie 5.1](#)). Ook de gebruiksvriendelijkheid van het trustee dashboard werd sterk verbeterd ([Sectie 5.2](#)). Tot slot werd ook de output van de applicatie die gebruikt kan worden om het resultaat te controleren, verduidelijkt ([Sectie 5.3](#)). Hoewel in dit hoofdstuk alleen de grootste aanpassingen besproken worden, werd de layout doorheen het hele systeem aangepast.

### 5.1 Beheer

In de oude interface zaten het beheer en het bekijken van de verkiezing op dezelfde pagina ([Figuur 5.1](#)). De gewone kiezers zagen de beheersfuncties uiteraard niet, maar dit maakte het wel verwarring voor de beheerder. Bovendien was het niet duidelijk wat de volgende stap was of waar de beheerder juist in de procedure zat. Daarom werd besloten om deze twee functies uit elkaar te halen en een aparte pagina te voorzien voor het beheer ([Figuur 5.2](#)).

Op deze nieuwe pagina wordt de volledige procedure weergegeven. Er wordt ook aangegeven wat de volgende stap is en welke stappen reeds voltooid zijn. Tot slot staan onderaan alle problemen die nog opgelost moeten worden voordat het stembiljet bevroren kan worden. Bovendien wordt bovenaan elke pagina ook getoond waar de beheerder zit en wat de volgende stap is ([Figuur 5.3](#)). Zo kan hij dit ook volgen wanneer hij niet op de speciale beheerpagina zit.

Voor een verkiezing met threshold encryptie ([Sectie 3.1.3](#)) zijn er belangrijke veranderingen in de workflow. Robbert Coeckelbergh had een publiek bulletin board geïntroduceerd waar iedereen sleutelparen voor communicatie kon aanmaken en uploaden. De beheerder kon dan trustees toevoegen aan een verkiezing door ze te selecteren uit een lijst. Dit betekent ook dat dezelfde sleutelparen voor communicatie gebruikt werden bij elke verkiezing waar deze persoon trustee was.

## 5. INTERFACE

---

The screenshot shows the Helios Voting interface. At the top, there is a navigation bar with links for 'Helios Voting', 'Help!', and 'About Helios'. Below the navigation bar, the title 'VTK Elections' is displayed, followed by a blue 'edit' button. A message indicates that this is a 'public election created by Pieter Maene' and provides a link to 'archive it'. It also states that the election is not featured on the front page. Below this, there are links for 'questions (0)', 'voters & ballots', and 'trustees (1)'. A note says 'Next Step: add questions to the ballot, and enter your voter list (or open registration to the public)'. A message box informs that 'voting is not yet open' and that the user is 'not eligible' to vote in this election. A yellow box labeled 'Audit Info' is present. At the bottom, a status bar shows the user is 'logged in as Pieter Maene' with a 'logout' link.

FIGUUR 5.1: Overzicht van de verkiezing (oud)

The screenshot shows the 'Admin' section of the Helios Voting interface. At the top, there is a navigation bar with links for 'CREATE', 'ADMIN', and 'TRUSTEES'. Below the navigation bar, the title 'VTK Elections—Admin' is displayed, followed by blue '[Edit]' and '[Archive]' buttons. A note says 'On this page you will find the procedure that has to be followed to get this election up and running. You'll notice that there are quite a few steps, but all of them are well-explained once you get there. The blue box that is shown at the top, gives you a reminder of which task is next. Below all steps you will find a button that takes you to its page.' Below this, a 'Procedure' section lists the following steps:

1. Create-a-new-election.
2. Manage the election's trustees.
  - a. Enter their names and e-mail addresses and send them the link to their dashboard.
  - b. Freeze the trustee list and define the threshold scheme. Once the list is frozen, it can no longer be modified. The threshold scheme determines how many trustees will need to present their keys in order to decrypt the result.
  - c. Wait for the key ceremony to be performed. In the meantime, you can continue with the next step in the procedure.
    - i. All trustees should upload their communication keys. When the last has been added, they will all receive an e-mail to start the second step.
    - ii. Each trustee should generate *encrypted shares* for the others. These shares will be used in the last step to determine the part of the election key that each trustee will receive. Again, an e-mail will be sent to all trustees when the last share has been uploaded.
    - iii. Finally, all trustees have to decrypt their shares and retrieve their part of the key that will be used to encrypt the ballots.
3. Add questions to the ballot.
4. Enter your voter list (or open registration to the public).
5. Freeze ballot and open election. Once you do this, the election will be immediately open for voting.
6. Wait for the voters to cast their ballots. The election will end at your discretion.
7. Compute encrypted tally. The encrypted votes will be combined into an encrypted tally.
8. Wait for the trustees to provide their share of the key used to encrypt the ballots.
9. Release tally and notify all voters.

Below the procedure, there is a blue button labeled 'Trustees »'. At the bottom, there is a section titled 'Issues' with a note: '1. Add questions to the ballot. 2. Add at least one trustee. 3. Enter your voter list (or open registration to the public)'.

FIGUUR 5.2: Beheer van de verkiezing

## 5.2. Trustee Dashboard



FIGUUR 5.3: Voortgang procedure

Voordat alle trustees toegevoegd konden worden aan een verkiezing, moest de beheerder dus wachten totdat iedereen zijn sleutelparen voor communicatie geüpload had. Omdat iedereen hiervoor buiten het systeem om nog eens gecontacteerd moet worden, maakte dit de sleutelceremonie nog ingewikkelder. Een bijkomend nadeel was dat om het even wie een sleutelpaar zou kunnen aanmaken, omdat het bulletin board publiek toegankelijk was. Omdat er ook geen controle was op de identiteit van de uploader, was het mogelijk om je als iemand anders voor te doen. Ervan uitgaande dat deze persoon geen toegang heeft tot de e-mailaccount van zijn slachtoffer, zou hij wel nooit de link krijgen naar het trustee dashboard ([Sectie 5.2](#)). Hij kan dus niet actief de rol van trustee spelen, maar kan het opzetten van de verkiezing wel tijdelijk blokkeren.

Daarom werd besloten om het bulletin board weg te halen. De trustees worden nu eerst toegevoegd aan de verkiezing, waarna ze een link krijgen toegestuurd naar hun trustee dashboard. De eerst stap van de sleutelceremonie die ze daar moeten uitvoeren, is nu het uploaden van de sleutelparen voor communicatie. Het proces dat een trustee doorloopt, wordt zo ook beter vergelijkbaar met dat wanneer geen threshold encryptie gebruikt wordt.

## 5.2 Trustee Dashboard

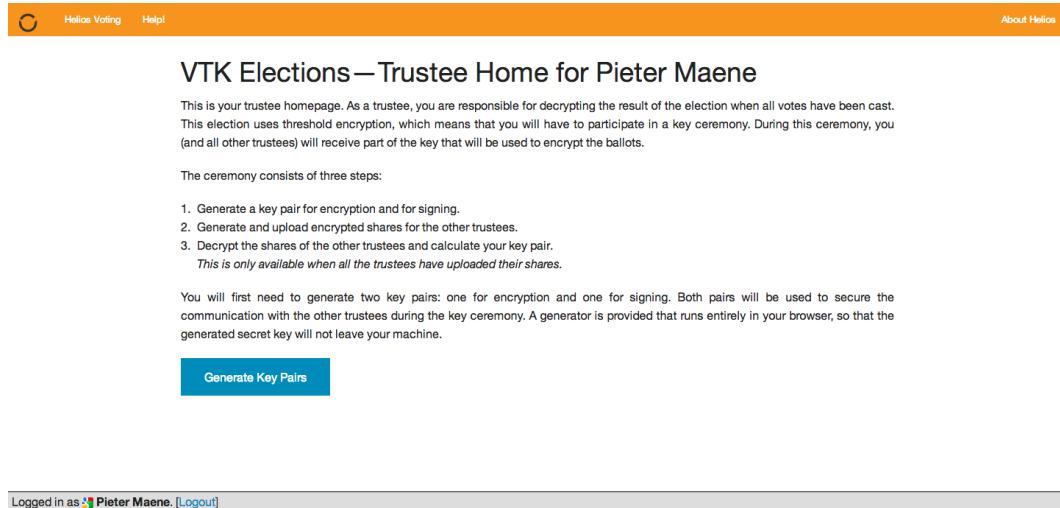
Iedere trustee krijgt een unieke link naar zijn trustee dashboard, dat wordt getoond in [Figuur 5.4](#). Deze link bevat naast zijn e-mailadres ook een geheime code. Dit is de plaats waar hij al de acties van de sleutelceremonie zal moeten uitvoeren. De beheerder van de verkiezing is ook vaak een trustee. De balk bovenaan werd oranje gemaakt zodat er voor hem een duidelijker verschil is tussen beide rollen.

Zoals vermeld in [Sectie 5.1](#) werd het aanmaken van de sleutelparen voor communicatie naar hier verplaatst. De volledige sleutelceremonie wordt hier op dezelfde manier getoond als de procedure voor de verkiezing op de pagina van de beheerder ([Figuur 5.2](#)). Op die manier ziet de trustee duidelijk welke stappen hij nog moet uitvoeren. Daarnaast wordt ook de rol van een trustee hier nog eens uitgelegd.

De trustee zal in totaal drie geheime sleutels genereren tijdens de sleutelceremonie ([Sectie 4.2.2](#)). Elke geheime sleutel moet lokaal bewaard worden in het JSON formaat. Oorspronkelijk moest de trustee deze zelf kopiëren naar een bestand. Om zijn sleutel te gebruiken, moest hij de inhoud daarvan opnieuw kopiëren naar een tekstvak.

## 5. INTERFACE

---



FIGUUR 5.4: Trustee Dashboard

Door gebruik te maken van de HTML5 File API kon de omgang met de sleutels veel gebruiksvriendelijker gemaakt worden.<sup>[31]</sup> Enerzijds voorziet deze API de `Blob` interface om in JavaScript bestanden te genereren en aan te bieden als download in de browser. Anderzijds kan de `FileReader` interface gebruikt worden om objecten die voldoen aan de `Blob` of `File` interface uit te lezen. Een voorbeeld van deze laatste zijn bestanden die geselecteerd worden in een file input element.

In het trustee dashboard wordt de eerste API interface gebruikt om de geheime sleutels onmiddelijk in een JSON bestand te plaatsen dat gedownload kan worden. Zo krijgen de trustees direct een bestand en worden de cryptografische details van de sleutel voor hen verborgen. De tweede interface wordt dan weer gebruikt om de trustee deze bestanden te laten selecteren om vervolgens de geheime sleutel uit te lezen naar het tekstvak. Het verschil tussen de oude en nieuwe manier van werken kan gezien worden in [Figuur 5.5](#) en [Figuur 5.6](#) respectievelijk.

Om de communicatie te beveiligen worden twee sleutelparen gebruikt ([Sectie 3.1.3](#)). Oorspronkelijk moesten deze apart gedownload worden. Omdat het voor de gebruiker eenvoudiger is om maar met één bestand te moeten werken, werden deze samengevoegd.

### 5.3 Controleapplicatie

De controleapplicatie werd besproken in [Sectie 3.4](#). Het resultaat van de verificatie werd oorspronkelijk onduidelijk weergegeven ([Figuur 5.7](#)). Zoals gezien kan worden in [Figuur 5.8](#), werd er meer structuur gebracht in de output.

### 5.3. Controleapplicatie

Welcome, this is your trustee homepage.  
As a trustee you are responsible for decrypting the result of the election when all the votes are casted.  
Therefore you will have to agree on a secret with the other trustees and you will have to complete 4 steps:

First Step: Generate encrypted shares for the other trustees.  
Second Step: Upload the shares.  
Third Step: Decrypt the shares of the other trustees. (This is only available when all the trustees uploaded their shares)  
Fourth Step: Save your secret key and upload your public key.

**FIRST STEP: Generate your encrypted shares**

To generate your encrypted shares enter your secret key for signatures:

```
161714543628581683103489191575377694476894884548217384400096093935661614228265453402256600432618573167330958374128199700532702877341579268889238802849257347418178338081663851254  
11769928178256157437052613324420768068784813251264870197763776547098473637098158475394578642891388486224580689576831701469
```

**Generate encrypted shares**

[skip to the second step](#)  
(you need to have already computed your encrypted shares.)

Logged in as Pieter Maene. [\[Logout\]](#)

FIGUUR 5.5: Genereren van encrypted shares (oud)

This is your trustee homepage. As a trustee, you are responsible for decrypting the result of the election when all votes have been cast. This election uses threshold encryption, which means that you will have to participate in a key ceremony. During this ceremony, you (and all other trustees) will receive part of the key that will be used to encrypt the ballots.

The ceremony consists of three steps:

1. Generate a key pair for encryption and for signing.
2. Generate and upload encrypted shares for the other trustees.
3. Decrypt the shares of the other trustees and calculate your key pair.  
*This is only available when all the trustees have uploaded their shares.*

**Generate Your Encrypted Shares**

To generate your encrypted shares, you first have to enter your secret key for signing. If you generated your key pair with our generator, it was stored in a file called `communication_keys.json`. This file contains your private keys, which are unknown to the server. This is the reason you need it here again.

Secret Key for Signing  communication\_keys.json

```
{"encryption": "public_key": "g":  
    "45315181387564396468333986216932223943710120431435472618409227525885020319627224726363  
    701529735668903577409091069810980332284525393196586705603070395613062360391093726191146  
    121415195311289299204727460560843415628717898690939082513769340997469477437688161938290  
    49261126632430956422637754104221208257819369317, "p":  
    "148236482780273000683277092471145624741R603251541210622310208588823QR753Q16927422177QRRA13Q}
```

**Generate Your Encrypted Shares**

Logged in as Pieter Maene. [\[Logout\]](#)

FIGUUR 5.6: Genereren van encrypted shares

## 5. INTERFACE

---

The screenshot shows the 'Election Verifier' application interface. At the top, there is a dark header bar with the 'Helios Voting' logo. Below it, the main title 'Election Verifier' is displayed. A text input field contains the URL '/helios/elections/6627c0d1-db50-11e3-a89a-040cced8b6a0'. A large blue button labeled 'Start' is centered below the input field. The main content area is titled 'Election' and displays a log of election loading and verification steps:

```
Loading election...
Loaded election: VTK Elections
Election fingerprint: GNkpQI2abBbXe73RVb7UGFvoxbkERUm4a8vxfnx4vA
Loading list of voters...
Loaded voter list, now loading ballots for each...
Loading ballot for voter #1
FOUND a ballot for voter #1
```

Below this, a section titled 'Ballots' shows a detailed log of ballot verification for Voter #1:

```
Voter #1
-- UUID: d8db6495-6002-4725-9e0f-7a42fa1a47f7
-- Ballot Tracking Number: xjQz+r9H03GC7PrEqkKyDEcRb5o12XEJVTL2i93Y
Question #1, Option #1 -- VERIFIED
Question #1, Option #2 -- VERIFIED
Question #1, Option #3 -- VERIFIED
Question #2, Option #1 -- VERIFIED
Question #2, Option #2 -- VERIFIED
Question #2, Option #3 -- VERIFIED
Question #2 OVERALL -- VERIFIED
```

Next, a section titled 'Trustees' shows trustee information:

```
Trustee #1: pieter.maene@vtk.be
-- PK xIMrTb9m2d2pWfRzS/AvvrsRbjgBWgQfzVB4hcdHpN4 -- VERIFIED.
```

Finally, a section titled 'Tally' shows the final tally results:

```
Question #1: Would you vote for or against the Pixel team as board of the student union VTK in the year 2014-2015?
Answer #1: Pro - COUNT = 1
-- Trustee pieter.maene@vtk.be: decryption factor verifies
-VERIFIED
Question #2: Which ineligible would you prefer to win?
Answer #1: Rebellion - COUNT = 1
-- Trustee pieter.maene@vtk.be: decryption factor verifies
-VERIFIED
```

A summary section titled 'FINAL RESULT' concludes with the message:

```
ELECTION FULLY VERIFIED -- SUCCESS!
verification took 2101 ms
```

FIGUUR 5.7: Controleapplicatie (oud)

### 5.3. Controleapplicatie

The screenshot shows the Helios Voting Election Verifier interface. It consists of several sections:

- Election Verifier**: A header bar with the Helios Voting logo. Below it, a message states: "This tool will verify the tally of the specified election. The required data is retrieved from the server, but all calculations are done locally in your browser."
- Election URL**: An input field containing the URL: /helios/elections/6627c0d1-db50-11e3-a89a-040cced8b6a0.
- Start**: A blue button to initiate the verification process.
- Election**: A section titled "Election" showing the progress: "Loading election...".
- Election Name**: VTK Elections
- Election Fingerprint**: GNkpgQf2abBbXe73RVb17UGFvoxbkERUUm4a8vxfnx4vk
- Verification Status**: A green bar indicating "✓ The election was verified."
- Ballots**: A section titled "Ballots" showing the progress: "Loading voters...", "Loading ballots...", "Verifying ballots...". It details the voter's UUID and ballot tracker, and lists questions and their options. For Question #1, all three options are checked. For Question #2, all two options are checked.
- Verification Status**: A green bar indicating "✓ All ballots were verified."
- Trustees**: A section titled "Trustees" showing the progress: "Loading trustees...". It lists the trustee's email and public key.
- Tally**: A section titled "Tally" showing the breakdown of votes for the election. It includes counts for Pro, Contra, Abstention, and Nautillus factors across various categories.
- Verification Status**: A green bar indicating "✓ The tally was verified."

FIGUUR 5.8: Controleapplicatie



Het Helios verkiezingssysteem werd in de praktijk gebruikt tijdens de kringverkiezing van de Vlaamse Technische Kring. Dit is de faculteitskring van de studenten burgerlijk ingenieur aan de KU Leuven. Uit de kiesreglementen van VTK zelf en deze van LOKO volgen de vereisten waaraan het systeem moet voldoen ([Sectie 6.1](#)). Om het systeem hiervoor te kunnen gebruiken, waren enkele aanpassingen nodig die overlopen worden in [Sectie 6.2](#). Gezien het cruciale karakter van een verkiezing, werd het systeem vervolgens uitgebreid getest ([Sectie 6.3](#)). Tot slot wordt het verloop van de stemdag zelf besproken in [Sectie 6.4](#).

### 6.1 Vereisten [22][46]

#### 6.1.1 Trustees

Het kiesreglement van LOKO schrijft voor dat over de verkiezingen gewaakt moet worden door een Neutraal Comité. Dit wordt samengesteld door de huidige studentenvertegenwoordigers. Het Neutraal Comité wordt bij VTK aangeduid met VKK. De huidige praeses is hier steeds de voorzitter van. Hij wordt geholpen door minstens vijf andere mensen uit het huidige praesidium die in een masteropleiding zitten en zelf geen kandidaat zijn in de verkiezing.

De voorzitter van VKK moet de verkiezing zelf kunnen beheren. Daarnaast moet het mogelijk zijn om alle leden van VKK aan te duiden als trustees van de verkiezing. Het mag ook niet mogelijk zijn dat één trustee de decryptie van het resultaat kan blokkeren.

#### 6.1.2 Vragen

Er zijn twee verschillende types kiesploegen bij VTK. Enerzijds zijn er de serieuze ploegen die opkomen als praesidium voor het volgende jaar en anderzijds de lolploegen die niet als vertegenwoordigers verkiesbaar zijn. Beide verkiezingen moeten als aparte vragen op hetzelfde biljet gesteld kunnen worden.

## 6. KRINGVERKIEZING

---

### 6.1.3 Kiezers

## 6.2 Aanpassingen

### 6.2.1 Shibboleth

Shibboleth is het authenticatie mechanisme dat gebruikt wordt voor de centrale login van de KU Leuven. Iedere gebruiker kan uniek geïdentificeerd worden aan de hand van zijn studentennummer. Alle stemgerechtigde kiezers voor een kringverkiezing hebben een dergelijke account. Helios had hier echter nog geen ondersteuning voor. Gezien het modulaire ontwerp van het authenticatiesysteem, was dit relatief eenvoudig toe te voegen.

De kieslijsten zelf werden opgemaakt door LOKO, de Leuvense studentenkoepel. Deze moesten wel nog omgezet worden naar een bestand dat uitgelezen kon worden door Helios. Het studentennummer werd hierbij gebruikt als het unieke ID voor elke kiezer. Een probleem was hier wel dat de e-mailadressen van de kiezers niet mee opgenomen waren in deze lijsten. Deze kunnen wel mee doorgegeven worden met het antwoord dat de server ontvangt nadat de student zich aangemeld heeft via de centrale login.

### 6.2.2 Opkomst

Bij VTK vertegenwoordigt het praesidium de studenten ook op onderwijsvlak. Om dit te kunnen doen, moet er volgens het reglement van LOKO een meerderheid behaald worden bij een opkomst van minstens 10%.<sup>[22]</sup> De berekening van dit percentage werd dan ook toegevoegd zodat dit samen met de resultaten bekend gemaakt kon worden.

### 6.2.3 Publicatie van het resultaat

Helios publiceert het resultaat van de verkiezing standaard van zodra het bekend is. Het is echter traditie om deze pas om middernacht bekend te maken. Het systeem moet hier dus licht voor aangepast worden.

## 6.3 Testen

Om er zeker van te zijn dat het systeem op de stemdag zelf goed zou functioneren, werd het na de installatie op de server getest. Hierbij werd niet alleen nagegaan of alles technisch in orde was, maar werd ook feedback gevraagd over de gebruiksvriendelijkheid. De testen van het beheer ([Sectie 6.3.1](#)) en het stemhokje ([Sectie 6.3.2](#)) werden uitgevoerd in de context van de vereisten voor VTK ([Sectie 6.1](#)).

### 6.3.1 Beheer

Aangezien hier de meeste veranderingen gebeurd zijn ([Sectie 5.1](#)), moest zeker dit deel goed getest worden. Hiervoor werd een testverkiezing aangemaakt door de voorzitter van VKK, met de echte trustees en vragen. De echte kieslijsten werden vervangen door een lijst met de huidige praesidiumleden.

De trustees voor de verkiezing zijn de zes leden van VKK. Als threshold schema ([Sectie 3.1.3](#)) werd ervoor gekozen dat drie van hen nodig zijn om het resultaat te decrypteren. Tijdens deze test kwamen echter twee fouten naar boven. Ten eerste werd de Lagrange-interpolatie niet over het eindig veld berekend, maar met een rationale breuk. Wanneer slechts twee trustees gebruikt werden, was dit geen probleem omdat de noemer dan gelijk is aan één. Dit kon eenvoudig opgelost worden door de teller en noemer voor het hele product apart te berekenen ([Vergelijking 6.1](#) en [Vergelijking 6.2](#)). Vervolgens wordt de teller vermenigvuldigd met de modulaire inverse van de noemer ([Vergelijking 6.3](#)).

$$N_i = \prod_{j=1, j \neq i}^k -x_j \mod q \quad (6.1)$$

$$D_i = \prod_{j=1, j \neq i}^k (x_i - x_j) \mod q \quad (6.2)$$

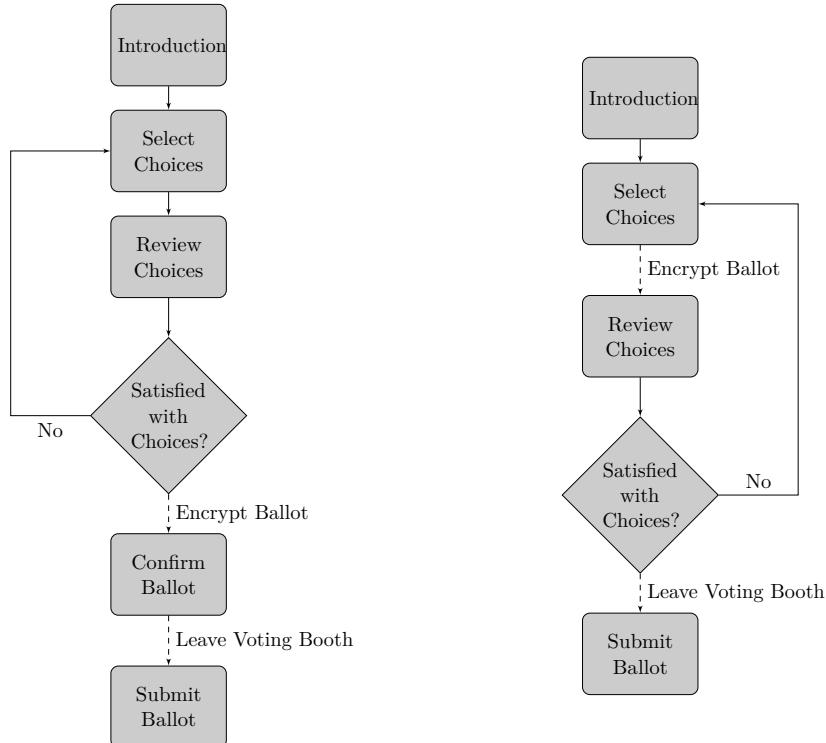
$$\lambda_i(0) = N_i * (D_i^{-1} \mod q) \mod q \quad (6.3)$$

Ten tweede werd de Helios trustee automatisch toegevoegd als trustee aan de verkiezing. Alle acties die een trustee moet uitvoeren, waren hiervoor geautomatiseerd. Na het uitvoeren van een aantal tests bleek dat de shares van deze trustee niet compatibel waren met deze van de anderen. Het grote verschil tussen beide is dat deze van Helios in Python en niet in JavaScript gegenereerd werden. Omdat er niet direct een fout gevonden kon worden, werd ervoor gekozen om Helios niet langer als trustee toe te voegen. Omdat alles hiervoor geautomatiseerd was, had deze toch niet zo'n belangrijke rol binnen het proces. Hij werd voornamelijk toegevoegd omdat dan een verkiezing zonder eigen trustees opgezet kon worden, wat leidt tot een eenvoudigere procedure.

Verder verliep deze test zeer goed. De voorzitter kon zonder hulp de volledige procedure ([Hoofdstuk 4](#)) voor het aanmaken van een verkiezing met threshold encryptie doorlopen. Er werd verder geen formele analyse van de nieuwe interface gemaakt, aangezien deze slechts door een beperkt aantal mensen gebruikt zal worden. Ook de trustees hadden weinig moeite om hun geheime sleutels te bekomen. Ondanks de e-mails die verstuurd worden telkens een actie van hen nodig is, duurde het nog steeds even voordat dit in orde was.

## 6. KRINGVERKIEZING

---



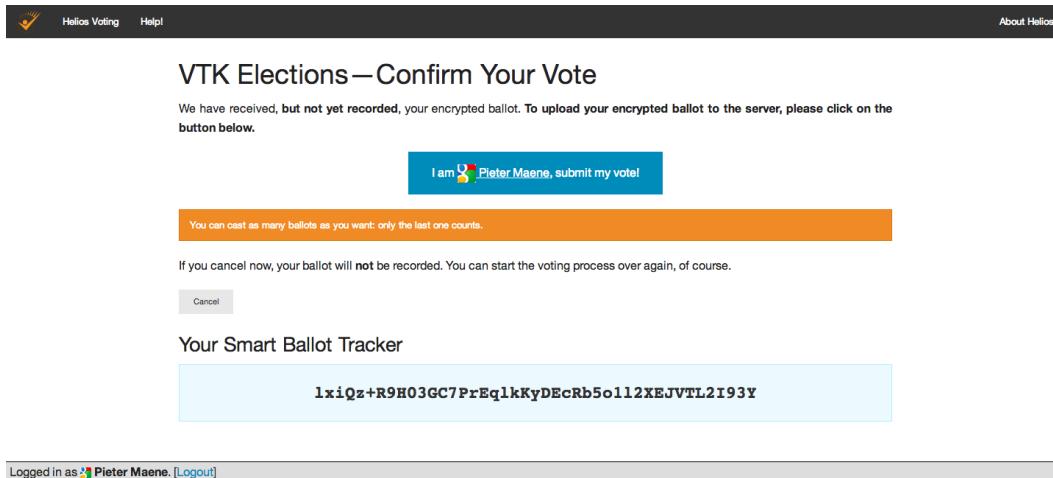
FIGUUR 6.1: Old (Left) and New (Right) Voting Booth Flows

### 6.3.2 Stemhokje

Tijdens het testen kwamen geen functionele problemen naar voor met het stemhokje. Toch kwam ook hier veel feedback op van het praesidium. Veel mensen vonden het stemmen niet intuïtief. Er waren twee belangrijke bemerkingen. Ten eerste waren sommige bewoordingen te ingewikkeld. De interface is volledig in het Engels en er kwamen veel onbekende vaktermen in voor. Dit kon eenvoudig opgelost worden door eenvoudigere terminologie in de plaats te gebruiken.

Ten tweede vonden velen het proces te moeilijk. Oorspronkelijk waren er vier stappen ([Figuur 6.1](#)). Gebruikers kregen eerst een korte uitleg, gevolgd door het aanduiden van hun keuzes. Hierna konden deze gecontroleerd worden, waarna de stem geëncrypteerd werd. Daarna volgde nog een scherm (??) waar gekozen kon worden om een audit te doen van de stem of ze te uploaden. In het laatste geval werd het stemhokje verlaten, maar moest de gebruiker nog eens bevestigen dat hij zijn stem wilde uitbrengen. Vooral deze twee laatste stappen vonden veel mensen verwarrend.

Om dit proces te vereenvoudigen, werd de laatste stap van het stemhokje verwijderd. Na het bevestigen van zijn keuzes, wordt de kiezer onmiddellijk doorgestuurd naar het uploaden buiten het stemhokje. De audit van een stem werd verplaatst naar het scherm waar de keuzes bevestigd moet worden. Omdat de stem hiervoor reeds



FIGUUR 6.2: Bevestigen van de stem

geëncrypteerd moet zijn, wordt dit nu gedaan na het beantwoorden van de laatste vraag. Het grootste nadeel hierbij is dat de encryptie opnieuw uitgevoerd moet worden wanneer de kiezer zijn stem wijzigt. In de huidige implementatie gaat dit echter al relatief vlot in moderne browsers. In de toekomst zal dit waarschijnlijk nog sneller kunnen door gebruik te maken van de Web Cryptography API ([Hoofdstuk 8](#)).

Na het aanpassen van het stemhokje werd deze test herhaald om zeker te zijn dat alles nog correct werkte. Ook tijdens deze test kwamen nog vragen om enkele kleine dingen te wijzigen voor de echte verkiezing. Zo wordt er een waarschuwing gegeven wanneer de kiezer het laatste scherm sluit voordat hij zijn stem geüpload heeft.

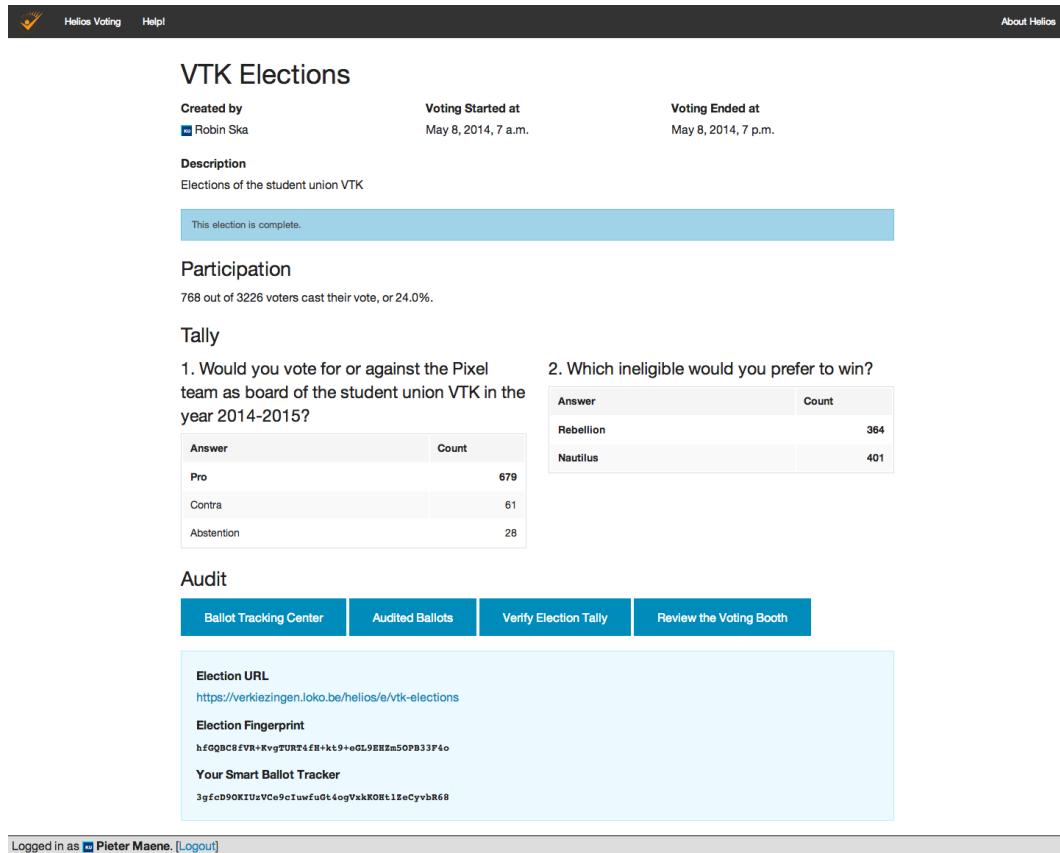
### 6.3.3 Stresstest

Er is elk jaar een opkomst van ongeveer 25% bij de kringverkiezing van VTK, wat iets minder dan 1 000 kiezers zijn. Daarom werd nagegaan of Helios ook een groot aantal stemmen nog correct kon verwerken. Hiervoor werd een aparte verkiezing aangemaakt met twee trustees en drie vragen. Vervolgens werden 1 000 stemmen uitgebracht. Het resultaat kon van de eerste keer correct gedecrypteerd worden, dus hier was geen verdere actie nodig.

Deze stemmen werden gegenereerd door een server-side actie. De kiezers stemmen normaal via het stemhokje, maar dat proces is moeilijk te automatiseren. Het grote verschil is dat de encryptie in Python in plaats van JavaScript gebeurt.

*Deze test werd niet op de server maar lokaal uitgevoerd. Het doel was hierbij te kijken of decryptie ook bij een groot aantal stemmen correct functioneerde. De belasting op de server van een groot aantal kiezers werd niet onderzocht omdat iedere kiezer slechts een beperkt aantal lichte requests uivoert.*

## 6. KRINGVERKIEZING



FIGUUR 6.3: Resultaat van de stemming

## 6.4 Stemdag

De stemming zelf vond plaats op donderdag 8 mei 2014. Oorspronkelijk stond ze open van 7u00 tot 19u00. Omdat de communicatie 's ochtends traag op gang gekomen was, werd dit verlengd tot 20u00, wat ondersteund wordt door Helios. De trustees en het threshold schema waren dezelfde als tijdens de test (Sectie 6.3.1). De vragen en het resultaat worden getoond in [Figuur 6.3](#). In totaal hebben 768 mensen hun stem uitgebracht, wat niet significant minder is dan in andere jaren. Dit is dus een sterke indicatie dat het proces voldoende eenvoudig is. Zowel bij het stemmen als het decrypteren van het resultaat waren er geen problemen.

## Bewaren van sleutels en fingerprints

Tijdens de sleutelceremonie generen de trustees sleutelparen, waarvan de sleutels uiteraard veilig bewaard moeten kunnen worden ([Sectie 7.1](#)). Daarnaast werkt Helios op verschillende plaatsen met fingerprints. In [Sectie 7.2](#) wordt eerst hun doel besproken, waarna gekeken wordt naar alternatieve methoden om ze weer te geven. [Voor beide werden deze oplossingen theoretisch onderzocht, maar niet geïmplementeerd.](#)

### 7.1 Sleutels

#### 7.1.1 Disk

De trustees downloaden hun geheime sleutels als JSON bestanden ([Sectie 5.2](#)). Deze zullen dus eerst bewaard worden op de harde schijf van de computer die op dat moment gebruikt wordt. Hier zijn twee belangrijke nadelen aan. Ten eerste zou iedereen die toegang heeft tot die machine de sleutel kunnen bemachtigen. Ten tweede kan de sleutel alleen vanaf deze machine gebruikt worden.

Een eenvoudige oplossing voor het eerste probleem is de trustees vragen om hun account zeker te beveiligen met een wachtwoord. De sleutel zou ook op een beveiligde partitie geplaatst kunnen worden. Dit kan bijvoorbeeld gedaan worden door gebruik te maken van TrueCrypt.[\[45\]](#)

Wanneer de sleutel op een andere machine ingevoerd moet worden, kan deze op een USB-stick opgeslagen worden. Hier is het zeker aangeraden om de sleutel op een geëncrypteerde partitie te plaatsen. Daarnaast zou de sleutel ook geüpload kunnen worden naar een private server of een cloud opslagdienst. In dit geval moet hij zeker eerst geëncrypteerd worden.

Het zou echter veel gebruiksvriendelijker zijn om dit in de generator in te bouwen. Voordat de sleutel gedownload wordt, kan deze eerst symmetrisch geëncrypteerd worden met een wachtwoord. Hiervoor zou bijvoorbeeld AES-CTR gebruikt kunnen worden, waarbij de sleutel afgeleid wordt van het opgegeven wachtwoord.[\[20\]](#) Er zijn

verschillende JavaScript libraries die hiervoor ondersteuning hebben.[12][7] Bovendien wordt deze mode ook ondersteund door de Web Cryptography API ([Hoofdstuk 8](#)).[39]

### 7.1.2 Web Storage [17][27]

Door gebruik te maken van de HTML5 Web Storage specificatie, zouden de sleutels ook op een alternatieve manier bewaard kunnen worden. Deze specificatie geeft een ontwikkelaar onder andere toegang tot een persistente key/value store waar tot 5MB aan data in opgeslagen kan worden. Er wordt ook voldaan aan de same-origin policy. Dit wil zeggen dat de data alleen toegankelijk zijn van op hetzelfde domein.[16]

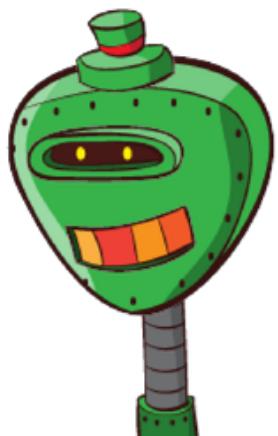
Deze functionaliteit zou toelaten om de geheime sleutels volledig te verbergen voor de trustees. In plaats van hen een bestand te laten downloaden met de sleutels, worden deze in de `localStorage` opgeslagen. Aangezien er naast de same-origin policy geen beveiligingsmechanismen ingebouwd zijn in de specificatie, is het beter om de sleutel eerst te encrypteren ([Sectie 7.1.1](#)). Een `secureStorage` binnen de browser met ingebouwde encryptie zou dit nog eenvoudiger maken voor een ontwikkelaar.[56] Een groot nadeel aan deze oplossing is dat de sleutel vast zit in de machine die gebruikt is om hem aan te maken.

## 7.2 Fingerprints

Op verschillende plaatsen binnen Helios worden fingerprints gebruikt. Dit zijn base64-geencodeerde SHA-256 hashes van specifieke data. Zo is de *Smart Ballot Tracker* ([Figuur 6.2](#)) een fingerprint van de geëncrypteerde stem. Aan de trustee wordt ook een fingerprint van de gegenereerde publieke sleutels getoond. Dit zijn echter lange strings zijn die moeilijk te onthouden kunnen worden. Bovendien kan niet in één oogopslag gezien worden of twee fingerprints hetzelfde zijn.

Daarom kunnen hiervoor beter visuele hashes gebruikt worden. Dit zijn unieke afbeeldingen op basis van een bepaalde string. Afbeeldingen kunnen niet alleen sneller herkend worden, het is ook veel eenvoudiger om ze te bewaren. Wanneer de fingerprint een string is, moet deze neergeschreven worden of gekopieerd worden naar een bestand. Een afbeelding daarentegen kan eenvoudig gedownload worden. Voorbeelden van dergelijke visuele hashsystemen zijn RoboHash ([Figuur 7.1](#)) en Identicon.[37][50]

Een belangrijke voorwaarde voor het stemhokje is dat er geen netwerk requests meer mogen gebeuren voordat de stem volledig geëncrypteerd is.[3] In het aangepaste stemhokje wordt het biljet geëncrypteerd nadat alle keuzes gemaakt zijn. Hoewel deze voorwaarde dus voldaan is wanneer de fingerprint getoond wordt, moet toch de voorkeur gegeven worden aan een systeem dat de figuren lokaal kan genereren. Indien een JavaScript implementatie niet mogelijk is, zouden ze gedownload kunnen worden van de server over een beveiligde verbinding.



FIGUUR 7.1: RoboHash van de fingerprint in [Figuur 6.2](#)



---

## Web Cryptography API

Wanneer webontwikkelaars vandaag cryptografische functies nodig hebben in hun toepassingen, moeten ze bijna JavaScript gebruiken omwille van compatibiliteit. Hoewel er de laatste jaren zeer grote vooruitgang geboekt is, zullen de meeste JavaScript engines nog steeds minder goed presteren dan native code.[33][11][40] De W3C startte in 2012 een working group op om een nieuwe browser API te definiëren: de Web Cryptography API.[47]

In [Sectie 8.1](#) wordt deze nieuwe API kort besproken. Daarna wordt in [Sectie 8.2](#) gekeken naar de NfWebCrypto polyfill, die de nieuwe functionaliteit implementeert in een plugin voor Google Chrome. Tot slot wordt deze implementatie in [Sectie 8.3](#) vergeleken met bestaande cryptografische libraries.

### 8.1 Web Cryptography API [39]

De Web Cryptography API definieert cryptografische operaties die gebruik maken van sleutels die beheerd worden door de browser. Al deze methodes zitten in de `SubtleCrypto` interface. Er zijn zowel methodes voor het beheren van het sleutelmateriaal als het encrypteren van data.

De API is ontworpen vanuit het idee om alleen standaardfunctionaliteit aan te bieden. Er wordt dus maar een beperkt aantal algoritmes ondersteund, die bovendien elk nog vaak hun eigen beperkingen hebben.[13] Naargelang de functionaliteit van het algoritme, ondersteunt ook niet elk algoritme alle methodes van de interface.

### 8.2 NfWebCrypto

Aangezien de standaard nog niet voltooid is, wordt deze nauwelijks ondersteund door de grote browsers.[18] Alleen Internet Explorer 11 heeft reeds een implementatie, maar hierin is slechts een beperkt aantal algoritmes aanwezig.[25] Om de vergelijking met de andere JavaScript libraries toch te kunnen uitvoeren, was dus een alternatieve implementatie nodig van deze API.

## 8. WEB CRYPTOGRAPHY API

---

PolyCrypt is een JavaScript polyfill ontwikkeld door BBN Technologies.[28] Een polyfill implementeert een browser API die (nog) niet native ondersteund wordt. Omdat deze gebaseerd is op een oudere draft van de API, miste ook hierin functionaliteit die nodig was voor de tests. Een bijkomend nadeel aan deze implementatie is dat ze in JavaScript geschreven is, waardoor een eventueel snelheidsvoordeel ten opzichte van de andere libraries waarschijnlijk niet naar voor zou komen.

NfWebCrypto daarentegen is een C++ polyfill ontwikkeld door Netflix.[26] Intern wordt de bekende OpenSSL library gebruikt voor de cryptografische functionaliteit. Het grote voordeel is hier dus dat de cryptografische code nu native is, waardoor de prestaties vergelijkbaar zouden moeten zijn met die van een echte implementatie in de browser. Het grootste nadeel is hier wel dat het een plugin is voor Chrome. Bovendien moet deze handmatig gecompileerd worden en moet de browser op een speciale manier gestart worden, zodat het gebruik ervan niet zo vanzelfsprekend is. Hoewel dit dus gebruikt kan worden voor de tests, is dit niet direct bruikbaar voor praktische applicaties.

### 8.3 Benchmarks

#### 8.3.1 Modulaire exponentiatie

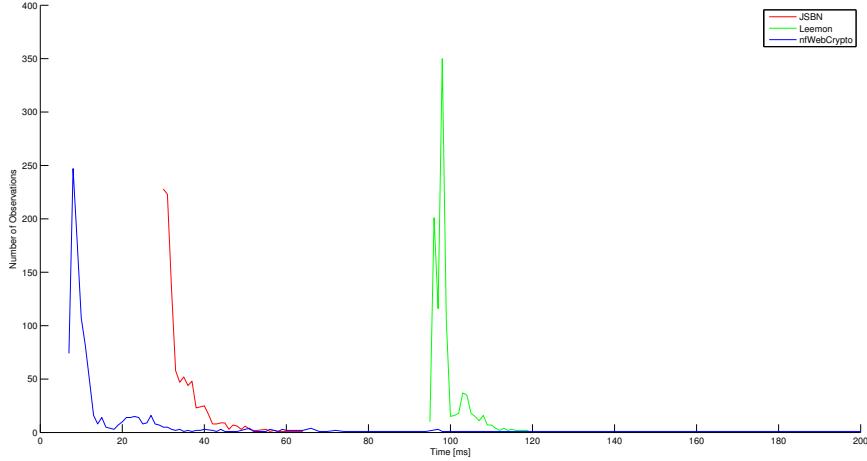
Helios maakt gebruik van ElGamal ([Sectie 3.1.1](#)) om de stemmen te encrypteren. Dit algoritme wordt echter niet ondersteund door de Web Cryptography API ([Sectie 8.1](#)). Deze kan dus niet onmiddellijk gebruikt worden om de encryptie te versnellen. De modulaire exponentiaties vragen veruit de meeste rekentijd. Een verbetering daar zou dus ook positief zijn voor de algemene prestaties.

Aangezien de enige publieke methodes van de API op hoog niveau werken, is er geen functie beschikbaar om dit te doen. Diffie-Hellman wordt echter wel ondersteund voor het genereren van een asymmetrisch sleutelpaar. De `deriveKey` methode van de API berekent de gedeelde geheime sleutel volgens [Vergelijking 8.1](#).[14]

$$K = (g^a)^b \mod p \quad (8.1)$$

Hier is  $g^a$  de publieke sleutel van A en  $b$  de geheime sleutel van B. Om de modulaire exponentiatie  $x^y \mod p$  uit te rekenen, moet het dus mogelijk zijn om de publieke sleutel  $g^a$  gelijk te stellen aan  $x$  en de geheime sleutel aan  $y$ .

Om een specifieke geheime sleutel te kunnen gebruiken in de `deriveKey` methode, moet deze eerst geïmporteerd worden in de key store. Hiervoor kan de `importKey` methode gebruikt worden. Samen met de naam van het algoritme (DH) worden als parameters de geheime sleutel, het priemgetal  $p$  en de generator  $g$  meegegeven. De standaard laat het importeren van de geheime sleutel alleen toe wanneer deze het PKCS8 formaat heeft.[21] Om de testen te vereenvoudigen, werd de NfWebCrypto



FIGUUR 8.1: Modulaire exponentiatie

plugin aangepast om het importeren van ruwe geheime sleutels toch mogelijk te maken.

Daarna kan de modulaire exponentiatie uitgerekend worden door `deriveKey` aan te roepen met  $x$  als argument voor de publieke sleutel. Na het afleiden wordt de gemeenschappelijke sleutel opnieuw opgeslagen in de key store. Ook hier waren enkele kleine aanpassingen nodig aan de code van de plugin om het berekenen van de geheime sleutel met onze eigen waarden mogelijk te maken. De `exportKey` methode kan tot slot gebruikt worden om het ruwe resultaat te exporteren.

De prestaties van NfWebCrypto worden vergeleken met deze van JSBN en Leemon, twee big number libraries die geschreven zijn in JavaScript.[55][6] Er wordt telkens een exponent van 1 024 bit gebruikt, wat ook de lengte van de parameters in Helios is. De berekening werd voor elke library 1 000 keer uitgevoerd om een statistisch relevant resultaat te bekomen. De resultaten worden weergegeven in [Figuur 8.1](#) en [Tabel 8.1](#). De berekening duurt bij Leemon veruit het langste. De JSBN library is sterk geoptimaliseerd. De NfWebCrypto plugin is daarentegen gemiddeld nog eens dubbel zo snel. De variantie is hier zo groot omdat de eerste exponentiatie veel langer duurt.

Het resultaat dat bekomen wordt na de `exportKey` is wel niet correct. In de communicatie met de plugin moeten de parameters `base64` geëncodeerd worden. Vermoedelijk loopt hier nog iets mis tussen de conversie in C++ en JavaScript.

### 8.3.2 RSA

Een tweede vergelijking werd gemaakt tussen JSBN en NfWebCrypto voor een RSA encryptie. Dit wordt niet gebruikt in Helios, maar ook deze resultaten zijn

## 8. WEB CRYPTOGRAPHY API

---

Library	Gemiddelde [ms]	Variantie [ms]
JSBN	33,9250	26,5019
Leemon	99,1470	15,0785
NfWebCrypto	16,0670	470,6251

TABEL 8.1: Modulaire exponentiatie

Library	Gemiddelde [ms]	Variantie [ms]
JSBN	0,6310	0,3632
NfWebCrypto	2,1360	0,4219

TABEL 8.2: RSA

zeer interessant. De berekening van de RSA cijfertekst is opnieuw een modulaire exponentiatie ([Vergelijking 8.2](#)).[\[35\]](#)

$$c = m^e \mod n \quad (8.2)$$

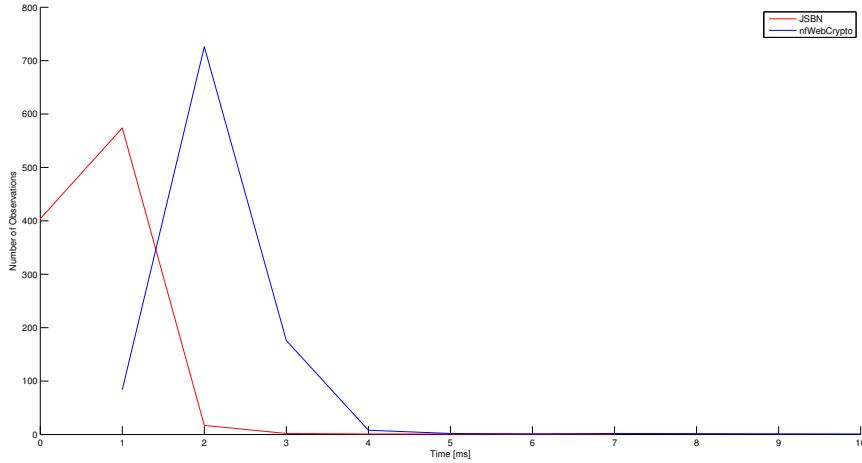
Hier is het paar  $(n, e)$  de publieke sleutel van de partij waarvoor het bericht geëncrypteerd wordt. Voor de publieke exponent  $e$  werd de standaardwaarde 65 537 genomen. Merk op dat deze heel wat korter is dan de 1 024-bit exponenten die in de tests van [Sectie 8.3.1](#) gebruikt werden. De lengte van de modulus is 1 024 bit.

De JSBN library voorziet in enkele klassen specifiek voor RSA encryptie en decryptie. Het is dus zeer eenvoudig om een bepaalde klaartekst te encrypteren met een gegeven publieke sleutel. De Web Cryptography API ondersteunt hier het **RSAES-PKCS1-v1\_5** algoritme voor.[\[19\]](#) Hoewel het mogelijk zou moeten zijn om publieke sleutels in het SPKI-formaat te importeren, bleek het opnieuw moeilijk om de sleutel die gebruikt werd voor de JSBN encryptie om te zetten. Daarom werd besloten om voor elke encryptie een nieuwe sleutel te genereren. De tijd die hiervoor nodig is, wordt niet meegerekend in het eindresultaat.

Ook hier werd de encryptie 1 000 keer uitgevoerd. De resultaten van deze test zijn terug te vinden in [Figuur 8.2](#) en [Tabel 8.2](#). We zien dat de JavaScript implementatie voor deze veel kortere exponent sneller is dan de berekening door de NfWebCrypto plugin. De communicatie met de plugin weegt hier duidelijk zwaarder door.

## 8.4 Besluit

Dankzij de Web Cryptography API zal het eenvoudiger worden om cryptografie te gebruiken in de browser. Bovendien is er een merkbare snelheidswinst ten opzichte van de huidige JavaScript implementaties, wat belangrijk is voor de gebruiksvriendelijk.



FIGUUR 8.2: RSA

Dat slechts een beperkt aantal algoritmes ondersteund wordt, is een belangrijk nadeel van de API. Omdat de methodes die publiek beschikbaar zijn op een hoog niveau werken, is het zeer moeilijk om deze te gebruiken voor het versnellen van alternatieve algoritmes.

Daarnaast was het ook ingewikkeld om sleutels die buiten de plugin gegenereerd waren, correct te importeren. Dit werd echter voornamelijk veroorzaakt door de encoding van de communicatie en is dus niet noodzakelijk een inherent probleem van de API.



---

## Besluit

Om het vertrouwen van kiezers in het resultaat te vergroten, kan een voter-verifiable verkiezingssysteem gebruikt worden. Er zijn verschillende papieren systemen beschikbaar die dit realiseren, maar deze hebben zeer complexe procedures. Hierdoor kunnen ze vaak niet op grote schaal gebruikt worden in de praktijk ([Sectie 2.6](#)). Helios is een voter-verifiable stemsysteem voor online verkiezingen. Dit betekent wel dat het alleen gebruikt kan worden wanneer dwang geen grote bedreiging vormt.

De procedure die gevuld moet worden om een verkiezing op te zetten in Helios werd besproken in [Hoofdstuk 4](#). De interface werd ook zo aangepast dat deze procedure er veel duidelijker in terug te vinden is ([Sectie 5.1](#)). In de praktische tests bleek dat deze nieuwe interface een grote hulp was. Beheerders van de verkiezing konden zonder veel aanwijzingen een verkiezing met threshold encryptie opzetten ([Sectie 6.3.1](#)).

Om Helios te gebruiken in de kringverkiezing van VTK, werden eerst nog enkele aanpassingen doorgevoerd. Vervolgens werd alles grondig getest en op basis van de gegeven feedback nog verder gewijzigd. Vooral de procedure in het stemhokje werd tijdens deze tests als te complex ervaren ([Sectie 6.3.2](#)). In [Sectie 6.4](#) werd de stemdag zelf besproken, waarbij geen grote problemen vastgesteld werden.

In [Sectie 7.1.1](#) werd gezien dat de geheime sleutels van de trustees bewaard worden in een bestand. Ook wanneer ze deze ergens anders nodig hebben, zullen deze vaak op een USB-stick geplaatst worden. Om de sleutels te beveiligen kan ofwel de partitie geëncrypteerd worden, ofwel de bestanden zelf. Als alternatief zouden deze ook in de browser zelf opgeslagen kunnen worden ([Sectie 7.1.2](#)). Visuele hashes zouden het bewaren en vergelijken van de fingerprints sterk vergemakkelijken ([Sectie 7.2](#)).

De Web Cryptography API geeft webontwikkelaars eenvoudig toegang tot cryptografische functies. Uit de vergelijkingen met bestaande JavaScript implementaties bleek dat deze ook een grote snelheidsverbetering opleveren. Er wordt wel slechts een beperkt aantal algoritmes ondersteund en de API biedt alleen high-level functies aan, waardoor het zeer moeilijk is om alternatieven te implementeren ([Sectie 8.3.1](#)).



Bijlage

A

---

English Paper

# Online Elections in Practice: Improvement and Application of the Helios Voting System

Pieter Maene

**Abstract**—Helios is a voter-verifiable online voting system. Such a system allows the voter to verify whether his ballot was registered correctly and check the election result. The procedure for these systems is typically very complicated. The system's usability is determined by its interface which should support the procedure. Therefore, the procedure to manage an election with Helios was written out. The system's interface was modified to better support this procedure. Additionally, the Web Cryptography API adds native cryptographic functionality to user agents. Its faster implementation could speed up encryption operations and improve the usability. The modified system was deployed for a real election where the election administrator and voters used it successfully.

## I. INTRODUCTION

Elections are an important part of our democratic society. In 2014, national elections will be held in 40 countries and 42% of the world population will be able to cast its vote.<sup>[1]</sup> It is therefore necessary to have a trustworthy voting system. In current systems, voters usually have to trust the election organiser. Voter-verifiable systems allow them to verify whether their ballot was registered correctly and check the election result. Helios is such a voter-verifiable system, that can be used for online elections. It is an open-source project developed by Ben Adida.<sup>[2]</sup> Robbert Coeckelbergh added threshold encryption to it, so that one trustee can no longer stall the election.<sup>[3]</sup> Online elections can only be used in situations where coercion is not a big threat, though.

The Web Cryptography API is a new W3C specification that will add native cryptographic functionality to user agents.<sup>[4]</sup> It could improve the usability of web applications that need user-level cryptography. The performance of this new API is compared to that of existing JavaScript cryptography libraries.

This paper will discuss the procedure that should be followed to organise an election in Helios ([Section II](#)). The interface of the system was also modified to better support this procedure and improve its usability ([Section III](#)). In [Section IV](#) the performance of the new Web Cryptography API is analysed. Finally, the system was deployed in a production environment and used for a real election ([Section V](#)).

## II. PROCEDURE

This section gives the procedure that should be followed to organise an election in the improved version of Helios ([Section III](#)). Before he can start the procedure ([Section II-B](#)), the administrator should prepare some information ([Section II-A](#)).

### A. Preparation

The trustees will be responsible for the decryption of the result. It is therefore important to give some thought to who this role will be given. Since using threshold encryption results in a more robust decryption process, it is advisable to use this. In that case, the election administrator should also determine the threshold scheme.

For each question, the number of answers that can be selected by the voter can be specified. The result for each answer can be shown either as the absolute number of votes it received, or its relative position with respect to the other answers.

Helios supports both open and closed elections. Anybody can vote in the former, while only specific voters are allowed to cast their ballot in the latter. A closed election should therefore be used when voter lists are available.

### B. Helios

The first step is the creation of the election. Here, all basic information like its name should be given. It is also possible to enable options here, like threshold encryption or whether the election is private or not. The start and end date of the election can be controlled here as well.

After creating the election, the trustees have to be added. This step was placed first in the procedure because it is the longest to complete. After being added, the trustee receives an e-mail with the link to his trustee dashboard.

If threshold encryption is not being used, each trustee simply has to generate a key pair. Once all trustees have done this, the election administrator can freeze the election. If threshold encryption was enabled, the key ceremony becomes more complicated. After adding all trustees, the election administrator can define the threshold scheme. The trustees can then start the following ceremony.

- 1) The trustee has to generate his communication keys and upload the public keys to the server. They will be used to encrypt and sign the communication between them.
- 2) Once all trustees have done this, each of them can generate his encrypted shares and send them to the others.
- 3) After receiving a share from all other trustees, the trustee can decrypt and add them to obtain his share of the key that will be used to encrypt the ballots. It will serve as his private key for the election and he has to upload the corresponding public key.

When all trustees have done this, the election can be frozen. Since the trustees will have to wait for the others, e-mails are sent to all of them when action is required.

Once the election is frozen, voters will be able to cast their ballot, unless a specific starting time was entered. If no end time was given, the election will be closed at the administrator's discretion.

After the election has been closed, the homomorphic tally can be decrypted. If threshold encryption is not enabled, all trustees will have to calculate their decryption factor and submit it. Otherwise, only the number of trustees defined in the threshold scheme will have to do this.

### III. USER INTERFACE IMPROVEMENTS

Most of our work on the interface to improve the usability concentrated on two big parts of Helios. First, the admin was modified to better reflect the procedure discussed in [Section II](#). This was done by creating a separate admin page where it is clearly shown to the election administrator. Second, the trustee dashboard was also simplified. Here, the HTML5 File API was used to make it easier for the trustee to manage his keys. Additionally, the layout was streamlined throughout the application.

#### A. Admin

The old interface integrated the administrative functions in the election view. This made it hard for the election admin to keep track of his progress in the procedure. It was therefore decided to move this functionality to a separate page. It shows the entire procedure and clearly indicates which steps have been completed. Since the actions lead away from this page, a bar was put on top of all pages where the progress is indicated as well.

The workflow for an election with threshold encryption was thoroughly changed. Originally, a public bulletin board was available where anybody could upload a pair of communication keys. The admin could only add trustees to an election from a list of people who had done this. This meant that he had to wait for all trustees to do this before he could define the threshold scheme and continue setting up the election. This required a secondary path of communication, outside of Helios. Because the bulletin board was publicly accessible, an attacker could impersonate a trustee. Unless he has access to the trustee's e-mail account, he would not receive the URL to the trustee dashboard. In that case, he would only have been able to stall the key ceremony.

The bulletin board was removed completely and trustees are immediately added to the election. They then receive an e-mail with a unique link to their trustee dashboard where they can generate their communication keys.

#### B. Trustee Dashboard

The trustees will perform all actions of the key ceremony in the trustee dashboard. The trustee's role is explained and an overview of all steps in the ceremony is shown in the same way as the election procedure on the admin page.

The trustee will generate three key pairs during the ceremony. The private keys are stored locally in JSON files. Originally, the JSON was simply shown to the user and he

had to manually save it to a file. When the key was needed, the file's contents needed to be copied again to a text area.

The HTML5 File API offers two API interfaces that can be used to ease this process.[\[5\]](#) The first is the `Blob` interface that can be used to generate files in JavaScript. They can then be offered as a download in the browser. The second is the `FileReader` interface which allows a web developer to read files that conform to the `Blob` or `File` interfaces.

They are used in the trustee dashboard to offer the secret keys as a download. The details of the key are now hidden from the trustee. When his key is needed, he has to select this file in an input element. The file's contents are then read to the text area. These adjustments make it a lot easier for the trustee to manage his keys.

In addition to using this API, the two secret keys of the communication pair are now stored in a single file. Since the distinction between two was only confusing to the trustee, this simplifies the key management for him.

### IV. WEB CRYPTOGRAPHY API

Currently, web developers implement cryptographic functionality in JavaScript. Despite great improvements in recent years, it is still slower than native code.[\[6\]](#)[\[7\]](#)[\[8\]](#) In 2012 a working group was started by W3C to write the Web Cryptography API specification, which would add cryptographic functionality to user agents.[\[9\]](#) The specification defines the `SubtleCrypto` interface which has several high-level functions.[\[4\]](#) There are methods to manage the keys and to encrypt data. The API only supports a specific set of algorithms, nor does each algorithm support all functions of the interface.

Since the specification has not been completed yet, most user agents do not support it. Internet Explorer 11 is the only browser that already has an implementation, but it misses some algorithms.[\[10\]](#) Therefore, the `NfWebCrypto` polyfill was used for the benchmarks. This is a C++ plugin for Chrome, so its performance should be comparable to a real implementation. Since a polyfill had to be used, the API's functionality was not used in production.

#### A. Modular Exponentiation

Helios uses ElGamal to encrypt the ballots, which is an algorithm not supported by the API. The modular exponentiations are responsible for the majority of the calculation time. An improvement there would impact overall performance.

Since the API only offers high-level functions, this had to be implemented separately. Diffie-Hellman is one of the supported algorithms. The `deriveKey` method uses [Eq. 1](#) to calculate the shared secret.[\[11\]](#)

$$K = (g^a)^b \mod p \quad (1)$$

In this equation,  $g^a$  is the public key of A and  $b$  the private key of B. If the public and private key can be specified, it can be used to calculate the modular exponentiation  $x^y \mod p$ . It is possible to import a specific private key into the user agent's key store. For the Diffie-Hellman algorithm, the

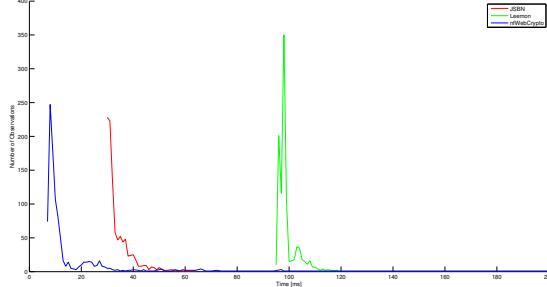


Fig. 1: Modular Exponentiation

Library	Gemiddelde [ms]	Variantie [ms]
JSBN	33,9250	26,5019
Leemon	99,1470	15,0785
NfWebCrypto	16,0670	470,6251

TABLE I: Modular Exponentiation

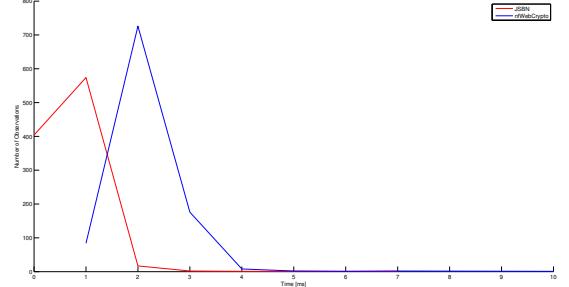


Fig. 2: RSA

Library	Gemiddelde [ms]	Variantie [ms]
JSBN	0,6310	0,3632
NfWebCrypto	2,1360	0,4219

TABLE II: RSA

specification only allows private keys in the PCKS8 format to be imported. To make testing easier, the NfWebCrypto plugin was modified to allow the import of raw keys. After calculating the shared secret, the result can be obtained with the `exportKey` function.

The performance of the NfWebCrypto library is compared to that of JSBN and Leemon, two JavaScript big number libraries.[12][13] A 1,024-bit exponent is used, which is the size of the keys used by Helios. The calculation is repeated 1,000 times for each library. The results are shown in Fig. 1 and Table I. The first exponentiation of the NfWebCrypto plugin took a lot longer, which explains the large variance. The NfWebCrypto plugin result was not correct. The communication with the plugin has to be base64 encoded, which resulted in conversion issues between C++ and JavaScript.

### B. RSA

RSA was used to make a second comparison between JSBN and NfWebCrypto. This cipher isn't used by Helios, but the results are very interesting. A standard value of 65,537 was used for the public exponent of the encryption. Notice that this is a lot shorter than the exponents used in Section IV-A. The modulus is 1,024 bits large.

JSBN has specific functions for RSA encryption and decryption. The Web Cryptography API supports the RSAES-PKCS1-v1\_5 algorithm.[14] Although public keys in the SPKI format can be imported, this proved difficult. Therefore, a new key was generated for each encryption. The time required for this is not taken into account. The results are shown in Fig. 2 and Table II. Surprisingly, the JavaScript implementation is faster than the NfWebCrypto plugin. This is caused by communication overhead.

## V. BOARD ELECTION

The modified Helios voting system was used in the elections for the new board of VTK, the official student organisation for engineering students at the KU Leuven. Support for some

specific features was added. The system was then thoroughly tested before the election on May 8<sup>th</sup> 2014.

### A. Election Requirements

#### B. Specific Adjustments

Shibboleth is the single sign-on system used by the KU Leuven. Students have an account with a unique identification number. All voters in the election have such an account, so it could be used to authenticate them. Since Helios' authentication system is modular and already has support for other federated identity systems, adding Shibboleth was relatively straightforward.

The list of voters was provided by an external organisation. This included the student's identification number and name, but the e-mail address was missing. This is needed by Helios to send a confirmation message after casting a ballot. A voter's e-mail address is retrieved from the Shibboleth response and added to the account to fix this.

Helios did not yet show the election's participation percentage. This is the percentage of eligible voters that cast their vote. The system also immediately published the election result. They are traditionally announced to the public at midnight, but the election administrator does need earlier access to them. These two smaller features were therefore implemented as well.

### C. Tests

After installing the system on a server, it was thoroughly tested by the current board of VTK. In addition to verifying all technical aspects, the testers were also asked for feedback on the user interfaces. The election admin and booth were tested in the context of the actual election. The real voter list was replaced by the current board of the organisation, though.

While testing the threshold encryption process, a bug was found. Real fractions were used for the calculation of the Lagrange interpolation, instead of finite modular arithmetic. After solving this, the result still couldn't be decrypted correctly. By default, Helios was added as a trustee to the election.

It completed all steps in the ceremony automatically. During testing, the shares of this trustee turned out to be incompatible with these generated by the other trustees. The only difference between the two is that the former are generated in the Python application and the latter in the JavaScript trustee dashboard. Because the source of the error origin could not be found, Helios no longer could be an election trustee. Since its role in the whole process was already limited, this was not a major change. The main advantage of adding it, was that it allowed users to create an election without real trustees, which resulted in a simpler procedure.

Apart from these bugs, the test of the election admin went very well. The election administrator managed to go through the entire procedure independently. The trustees didn't have much difficulty generating their keys either.

Although its functionality worked perfectly, a lot of feedback was given on the voting booth. First, a lot of technical terms were used. This was easily solved by replacing them with more common words. Second, most testers found the voting process to be too complicated. The booth is an independent application written in JavaScript, so that only the encrypted vote is sent to the server.

There were two places where the user had to submit his vote originally (Fig. 3). After confirming his encrypted vote in the booth, it was sent to the server where it could finally be cast. The last screen of the voting booth was therefore removed. Since it still had to be possible to audit the encrypted vote, this also required the encryption to be moved. This is now done after the voter has made all his choices. As a result, the vote has to be encrypted again when these are changed. The current implementation is already reasonably fast and this will only improve once the Web Cryptography API is available (Section IV). The test was repeated after making these adjustments to verify their impact on both functionality and usability.

Finally, a stress test was run as well. This was done for a separate election with two trustees and three questions. 1,000 votes were inserted into the database. To do this, the encrypted votes were generated directly in Python instead of using the JavaScript voting booth. The result was decrypted successfully, indicating that the system can manage an election of that size.

#### D. Election Day

The election itself took place on May 8<sup>th</sup> 2014 from 7 am till 8 pm. During this time, 768 people cast a ballot which isn't significantly less than during previous years. This is a strong indication that the voting process was clear.

Since everything had been tested in advance, no problems were encountered during the voting or when decrypting the result.

## VI. CONCLUSION

Voter-verifiable systems can be used to improve voter confidence. Helios is such a system that can be used to organise online elections. Section II gave the procedure as well as some recommendations that should be followed to setup an election.

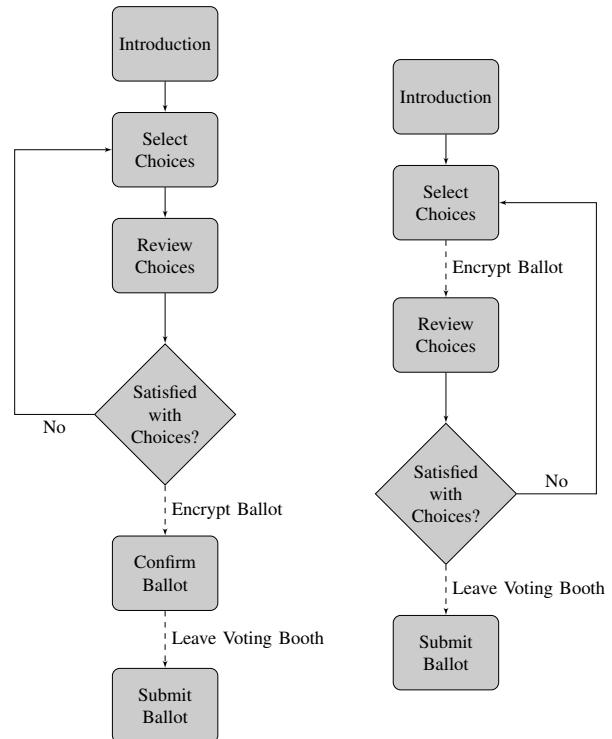


Fig. 3: Old (Left) and New (Right) Voting Booth Flows

The Web Cryptography API will make high-level cryptographic functions available to web developers. It also offers a considerable performance increase, compared to existing JavaScript libraries. Its biggest disadvantage is the small set of supported algorithms, which limits the possible applications. It also proved hard to import existing keys into the user agent's store.

By modifying Helios' interface to better support the procedure, its usability was greatly improved. Where the system was hard to use before, the election administrator didn't need any support in setting up the election. The trustees were also able to manage their keys without any problems. Using the simplified voting booth, a normal number of voters cast their vote in a real election.

## REFERENCES

- [1] The Economist, "The 2014 ballot boxes," <http://www.economist.com/node/21592755>, 2014, [Online; Accessed 17-May-2014].
- [2] B. Adida, "Helios Documentation," <http://documentation.heliosvoting.org>, 2014, [Online; Accessed 8-May-2014].
- [3] R. Coeckelbergh, "Application and extention of the Helios online voting system," 2013, Master's Thesis.
- [4] R. Sleevi and M. Watson, "Web Cryptography API," W3C, W3C Last Call Working Draft 12 September 2013, 2014. [Online]. Available: <http://www.w3.org/TR/2013/WD-FileAPI-20130912/>
- [5] A. Ranganathan and J. Sicking, "File API," <http://www.w3.org/TR/2014/WD-WebCryptoAPI-20140325/>, W3C, W3C Last Call Working Draft 25 March 2014, 2014.
- [6] J. Resig, "JavaScript Performance Rundown," <http://ejohn.org/blog/javascript-performance-rundown/>, 2008, [Online; Accessed 5-May-2014].
- [7] C. Cois, "JavaScript Performance Rundown, 2012," <http://codehenge.net/blog/2012/08/javascript-performance-rundown-2012/>, 2012, [Online; Accessed 5-May-2014].
- [8] F. Smedberg, "Performance Analysis of JavaScript," 2010, Master's Thesis.

- [9] W3C, “Web Cryptography Working Group Wiki,” [http://www.w3.org/2012/webcrypto/wiki/index.php?title=Main\\_Page&oldid=483](http://www.w3.org/2012/webcrypto/wiki/index.php?title=Main_Page&oldid=483), 2014, [Online; Accessed 5-May-2014].
- [10] Microsoft, “Web Cryptography,” [http://msdn.microsoft.com/en-us/library/ie/dn302338\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ie/dn302338(v=vs.85).aspx), 2014, [Online; Accessed 6-May-2014].
- [11] W. Diffie and M. Hellman, “New Directions in Cryptography,” *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.
- [12] T. Wu, “RSA and ECC in JavaScript,” <http://www-cs-students.stanford.edu/~tjw/jsbn/>, 2009, [Online; Accessed 6-May-2014].
- [13] L. Baird, “Big Integers in JavaScript,” <http://leemon.com/crypto/BigInt.html>, 2009, [Online; Accessed 6-May-2014].
- [14] J. Jonsson and B. Kaliski, “Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1,” <http://www.ietf.org/rfc/rfc3447.txt>, Internet Engineering Task Force, 2003.



**Pieter Maene** was born in Duffel in the year 1991. Because of an interest in science and technology, he started engineering studies at the KU Leuven in 2009. He enrolled in the electrical engineering master in 2011 where he took courses on signal processing, embedded systems and cryptography.



---

## Bibliografie

- [1] A. Adi, W. Adi, M. Schüler, and P. Fröhlich. Demonstration of “Open Counting”: A Paper-Assisted Voting System with Public OMR-At-A-Distance Counting.
- [2] B. Adida. *Advances in Cryptographic Voting Systems*. PhD thesis, MIT, 2006.
- [3] B. Adida. Helios: Web-based Open-Audit Voting. In *Proceedings of the 17th Conference on Security Symposium*, pages 335–348, 2008.
- [4] B. Adida. Helios Documentation. <http://documentation.heliosvoting.org>, 2014. [Online; Accessed 8-May-2014].
- [5] B. Adida and R. L. Rivest. Scratch & Vote: Self-Contained Paper-Based Cryptographic Voting. In *WPES*, pages 29–40, 2006.
- [6] L. Baird. Big Integers in JavaScript. <http://leemon.com/crypto/BigInt.html>, 2009. [Online; Accessed 6-May-2014].
- [7] bitwiseshiftleft. sjcl. <https://github.com/bitwiseshiftleft/sjcl>, 2014. [Online; Accessed 19-May-2014].
- [8] D. Chaum. Secret-Ballot Receipts: True Voter-Verifiable Elections. *Security Privacy, IEEE*, 2(1):38–47, 2004.
- [9] J. Cichoń, M. Kutyłowski, and B. Węglorz. Short Ballot Assumption and Three-ballot Voting Protocol. In *SOFSEM 2008: Theory and Practice of Computer Science*, volume 4910, pages 585–598. 2008.
- [10] R. Coeckelbergh. Toepassing en uitbreiding van het Helios online verkiezings-systeem, 2013. Master’s Thesis.
- [11] C.A. Cois. JavaScript Performance Rundown, 2012. <http://codehenge.net/blog/2012/08/javascript-performance-rundown-2012/>, 2012. [Online; Accessed 5-May-2014].
- [12] cryodex. aes-js. <https://github.com/cryodex/aes-js>, 2014. [Online; Accessed 19-May-2014].

## BIBLIOGRAFIE

---

- [13] A. Di Federico and R. Sleevi. Algorithms and referenced documents. <http://lists.w3.org/Archives/Public/public-webcrypto-comments/2013Jun/0004.html>, 2013. [Mailing List Archive; Online; Accessed 6-May-2014].
- [14] W. Diffie and M.E. Hellman. New Directions in Cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.
- [15] T. Elgamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
- [16] D. Gollmann. *Computer Security (Third Edition)*. Wiley, 2011.
- [17] I. Hickson. Web Storage. W3C Recommendation 30 July 2013, W3C, 2013.
- [18] HTML5test. Web Cryptography API. <http://html5test.com/compare/feature/security-crypto.html>, 2014. [Online; Accessed 6-May-2014].
- [19] J. Jonsson and B. Kaliski. Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. <http://www.ietf.org/rfc/rfc3447.txt>, 2003.
- [20] B. Kaliski. PKCS #5: Password-Based Cryptography Specification Version 2.0. <http://www.ietf.org/rfc/rfc2898.txt>, 2000.
- [21] B. Kaliski. Public-Key Cryptography Standards (PKCS) #8: Private-Key Information Syntax Specification Version 1.2. <http://www.ietf.org/rfc/rfc5208.txt>, 2008.
- [22] LOKO. Kiesreglement verkiezingen, 2014.
- [23] B. Maddens. Zijn de stemcomputers wel te vertrouwen? <http://www.knack.be/nieuws/belgie/zijn-de-stemcomputers-wel-te-vertrouwen/article-opinion-143099.html>, 2014. [Online; Accessed 17-May-2014].
- [24] A.J. Menezes, S.A. Vanstone, and P.C. Van Oorschot. *Handbook of Applied Cryptography*. 1996.
- [25] Microsoft. Web Cryptography. [http://msdn.microsoft.com/en-us/library/ie/dn302338\(v=vs.85\).aspx](http://msdn.microsoft.com/en-us/library/ie/dn302338(v=vs.85).aspx), 2014. [Online; Accessed 6-May-2014].
- [26] Netflix. Netflix WebCrypto (NfWebCrypto). <https://github.com/Netflix/NfWebCrypto>, 2014. [Online; Accessed 6-May-2014].
- [27] M. Pilgrim. Local Storage - Dive Into HTML5. <http://diveintohtml5.info/storage.html>, 2011. [Online; Accessed 19-May-2014].
- [28] Polycrypt. PolyCrypt: A WebCrypto Polyfill. <http://polycrypt.net>, 2014. [Online; Accessed 6-May-2014].

- [29] B. Preneel. Cryptography and Network Security. 2013.
- [30] B. Randell and P. Y.A. Ryan. Voting Technologies and Trust. *Security Privacy, IEEE*, 4(5):50–56, 2006.
- [31] A. Ranganathan and J. Sicking. File API. W3C Last Call Working Draft 25 March 2014, W3C, 2014.
- [32] De Redactie. Ga eens oefenen op een stemcomputer. <http://www.deredactie.be/cm/vrtnieuws/VK14/1.1937511>, 2014. [Online; Accessed 17-May-2014].
- [33] J. Resig. JavaScript Performance Rundown. <http://ejohn.org/blog/javascript-performance-rundown/>, 2008. [Online; Accessed 5-May-2014].
- [34] R. L. Rivest. The ThreeBallot Voting System. 2006.
- [35] R. L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-key Cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.
- [36] R. L. Rivest and W. D. Smith. Three Voting Protocols: ThreeBallot, VAV, and Twin. In *Proceedings of the USENIX Workshop on Accurate Electronic Voting Technology*, pages 16–16, 2007.
- [37] RoboHash. RoboHash. <http://robohash.org>, 2014. [Online; Accessed 19-May-2014].
- [38] A. Shamir. How to Share a Secret. *Communications of the ACM*, 22(11):612–613, 1979.
- [39] R. Sleevi and M. Watson. Web Cryptography API. W3C Last Call Working Draft 12 September 2013, W3C, 2014.
- [40] F. Smedberg. Performance Analysis of JavaScript, 2010. Master’s Thesis.
- [41] The Economist. The 2014 ballot boxes. <http://www.economist.com/node/21592755>, 2014. [Online; Accessed 17-May-2014].
- [42] The Guardian. Shaking up voter apathy with IT. <http://www.theguardian.com/society/2001/dec/11/localgovernment.politics>, 2001. [Online; Accessed 27-May-2014].
- [43] The New York Times. The 2012 Money Race: Compare the Candidates. <http://elections.nytimes.com/2012/campaign-finance>, 2012. [Online; Accessed 17-May-2014].
- [44] De Tijd. ‘Moeder aller verkiezingen’ kost meer dan 10 miljoen. <http://www.tijd.be/r/t/1/id/9388719>, 2013. [Online; Accessed 17-May-2014].
- [45] TrueCrypt. TrueCrypt - Free Open-Source On-The-Fly Encryption. <http://www.truecrypt.org>, 2014. [Online; Accessed 19-May-2014].

## BIBLIOGRAFIE

---

- [46] VTK. Verkiezingsreglement VTK vzw en VTK Ondersteuning vzw, 2014.
- [47] W3C. Web Cryptography Working Group Wiki. [http://www.w3.org/2012/webcrypto/wiki/index.php?title=Main\\_Page&oldid=483](http://www.w3.org/2012/webcrypto/wiki/index.php?title=Main_Page&oldid=483), 2014. [Online; Accessed 5-May-2014].
- [48] Wikipedia. DRE voting machine — Wikipedia. [http://en.wikipedia.org/w/index.php?title=DRE\\_voting\\_machine&oldid=555933682](http://en.wikipedia.org/w/index.php?title=DRE_voting_machine&oldid=555933682), 2013. [Online; Accessed 11-Apr-2014].
- [49] Wikipedia. Homomorphic encryption — Wikipedia. [http://en.wikipedia.org/w/index.php?title=Homomorphic\\_encryption&oldid=608522158](http://en.wikipedia.org/w/index.php?title=Homomorphic_encryption&oldid=608522158), 2014. [Online; Accessed 15-May-2014].
- [50] Wikipedia. Identicon — Wikipedia. <http://en.wikipedia.org/w/index.php?title=Identicon&oldid=590908926>, 2014. [Online; Accessed 19-May-2014].
- [51] Wikipedia. Ostracon — Wikipedia. <http://en.wikipedia.org/w/index.php?title=Ostracon&oldid=603659772>, 2014. [Online; Accessed 11-Apr-2014].
- [52] Wikipedia. Pretty Good Privacy — Wikipedia. [http://en.wikipedia.org/w/index.php?title=Pretty\\_Good\\_Privacy&oldid=606033746](http://en.wikipedia.org/w/index.php?title=Pretty_Good_Privacy&oldid=606033746), 2014. [Online; Accessed 19-May-2014].
- [53] Wikipedia. United States presidential election in Florida, 2000 — Wikipedia. [http://en.wikipedia.org/w/index.php?title=United\\_States\\_presidential\\_election\\_in\\_Florida,\\_2000&oldid=604767954](http://en.wikipedia.org/w/index.php?title=United_States_presidential_election_in_Florida,_2000&oldid=604767954), 2014. [Online; Accessed 17-May-2014].
- [54] Wikipedia. Voter-verified paper audit trail — Wikipedia. [http://en.wikipedia.org/w/index.php?title=Voter-verified\\_paper\\_audit\\_trail&oldid=599852086](http://en.wikipedia.org/w/index.php?title=Voter-verified_paper_audit_trail&oldid=599852086), 2014. [Online; Accessed 17-Apr-2014].
- [55] T. Wu. RSA and ECC in JavaScript. <http://www-cs-students.stanford.edu/~tjw/jsbn/>, 2009. [Online; Accessed 6-May-2014].
- [56] N.C. Zakas. SecureStore 1.0 Proposal. <http://www.nczonline.net/blog/securestore-proposal/>, 2011. [Online; Accessed 20-May-2014].

## Fiche masterproef

*Student:* Pieter Maene

*Titel:* Online verkiezingen in de praktijk: verbetering en toepassing van het Helios verkiezingssysteem

*Engelse titel:* Online Elections in Practice: Improvement and Application of the Helios Voting System

*UDC:* 621.3

*Korte inhoud:*

Een voter-verifiable stemsysteem geeft de kiezer de mogelijkheid om na te gaan of zijn eigen stem correct geregistreerd is en dat het resultaat correct is. Het is mogelijk om een dergelijk systeem met papieren stembiljetten te implementeren, zowel met als zonder cryptografische technieken. Helios is een voter-verifiable stemsysteem voor online verkiezingen. De procedure die bij deze systemen gevuld moet worden, is echter vaak zeer complex. In deze thesis wordt de procedure gegeven die in Helios gevuld moet worden om een verkiezing op te zetten. De interface van het systeem werd ook herwerkt om de beheerder beter te ondersteunen. Ook de applicatie die gebruikt wordt door de kiezers werd vereenvoudigd. Na het systeem uitgebreid getest te hebben, werd het in de praktijk gebruikt voor een reële verkiezing waarbij ongeveer 750 mensen hun stem uitbrachten. Daarnaast wordt besproken hoe de trustees hun geheime sleutels kunnen bewaren. Er wordt ook onderzocht of de fingerprints waarvoor momenteel strings gebruikt worden niet op een betere manier weergegeven kunnen worden. Tot slot worden de prestaties van de Web Cryptography API vergeleken met deze van bestaande implementaties in JavaScript. Deze nieuwe specificatie geeft ontwikkelaars toegang tot cryptografische functies die in de browser ingebouwd zijn.

Thesis voorgedragen tot het behalen van de graad van Master of Science in de ingenieurswetenschappen: elektrotechniek, optie Ingeboude systemen en multimedia

*Promotor:* Prof. dr. ir. B. Preneel

*Assessoren:* Prof. dr. ir. V. Rijmen

Prof. dr. ir. L. Van Eycken

*Begeleiders:* Dr. ir. J. Hermans

Dr. ir. F. Vercauterden