Formal Verification to Protect Democracy

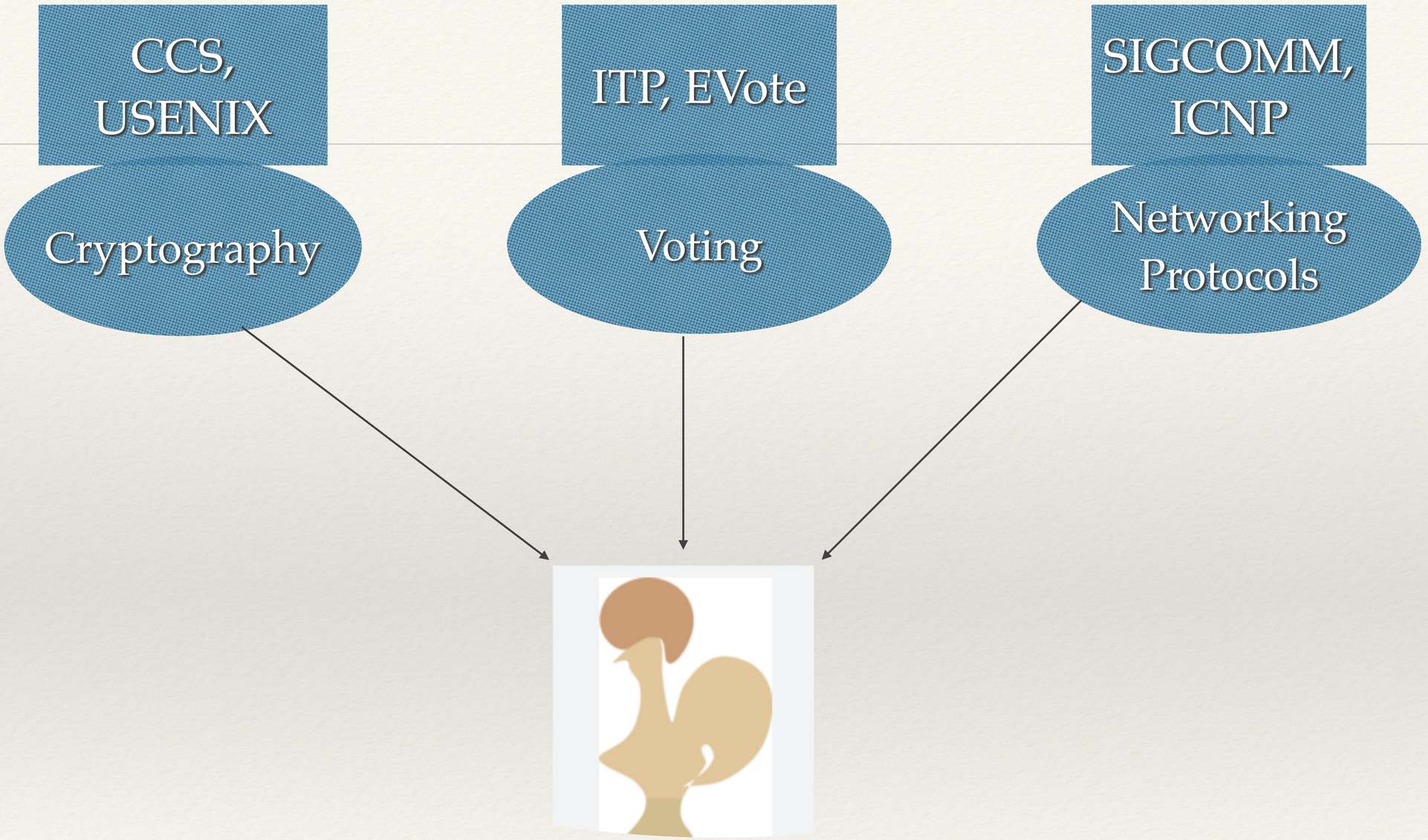
Mukesh Tiwari,
University of Cambridge,
Cambridge

UNIVERSITY OF CAMBRIDGE

>>whoami

Integrated Post-Graduate (Bachelors + Masters)

Doctor of Philosophy (Formally Verified Verifiable Electronic Voting Scheme)


CCS, USENIX
ITP, EVote
SIGCOMM, ICNP
Cryptography
Voting
Networking Protocols

Voting bug may have cost Rina Mercuri a seat on council

Stephen Mudd

Local News

VOTING BUG: Rina Mercuri was almost certain to win a seat on council in 2012 but a software glitch saw it handed to Alison Balind instead. Picture: Anthony Stipo.

LOCAL NEWS

1 Grants of up to $50,000 available for farmers hit by floods

2 MLHD ends year with 131 new COVID-19 cases, 'close contact' redefined

3 Six die as NSW records over 21,000 new cases in a day

4 PM announces major COVID-19 testing changes

Moscow's blockchain voting system cracked a month before election

French researcher nets $15,000 prize for finding bugs in Moscow's Ethereum-based voting system.

MOTHERBOARD
TECH BY VICE

Experts Find Serious Problems With Switzerland's Online Voting System Before Public Penetration Test Even Begins

The public penetration test doesn't begin until next week, but experts who examined leaked code for the Swiss internet voting system say it's poorly designed and makes it difficult to audit the code for security and configure it to operate securely.

KZ By Kim Zetter

Federal review says Dominion software flaws haven't been exploited in elections


By Sean Lyngaas, Evan Perez and Whitney Wild, CNN
Updated 12:18 PM EDT, Fri June 3, 2022

Flaws found in NSW iVote system yet again

Analysis of source code published at the request of the NSW Electoral Commission shows that the state's election system software was still vulnerable to attack.

I hear you: paper ballots are great

AEC ✏️
@AusElectoralCom

The Senate count is one of the most complex upper house counts in the world - it's so complex that we needed to write a program to distribute your preferences, as doing it by hand would mean we couldn't provide elected Senators in time to take their seat.

12:02 AM · Jun 14, 2022

13 Retweets **2** Quote Tweets **62** Likes

Blind advocates allege NSW's removal of online voting system is a breach of human rights

State electoral commission accused of discrimination for suspending iVoting platform as Blind Citizens Australia takes case to watchdog

Home › The Ministry and its Network › News › **2020**

A+ A–

French citizens abroad – Approval of electronic voting for consular elections (15 January 2020)

Claim

Formal verification can help us in protecting democracy

Theorem Provers (Coq) can help us in implementing bug-free vote-counting methods

This petition was submitted during the 2010–2015 Conservative – Liberal Democrat coalition government

View other petitions from this government

Petition

Schulze method voting system

More details

Most voting systems are flawed due to their methodology (see Arrow's Impossibility Theorem for more details) but some are more 'fair' than others. The Schulze method, based on the Schwartz criterion is a voting system which is mathematically 'fairer' and will generate voting outcomes which are a better representation than most voting systems. This petition aims to start a debate into the practicalities of introducing this system into UK public voting systems.

This petition is closed

This petition ran for 6 months

3 signatures

10,000

Schulze Method

$$\prod_{k=0}^{n}\prod_{i=0}^{n}\prod_{j=0}^{n} dist[i][j] = max(dist[i][j], min(dist[i][k], dist[k][j]))$$

Schulze Method

$$\prod_{k=0}^{n}\prod_{i=0}^{n}\prod_{j=0}^{n} dist[i][j] = min(dist[i][j], dist[i][k] + dist[k][j])$$

Floyd-Warshall Algorithm

Schulze Method

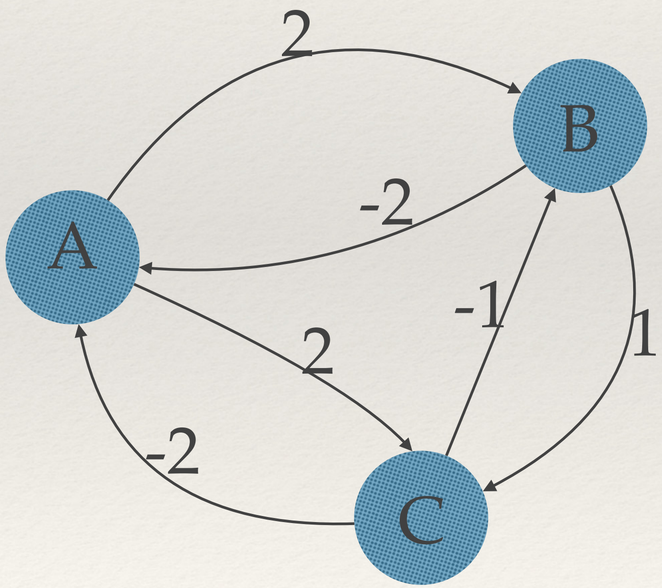
	C < B < A	C = B < A	C = B = A
A	1	1	1
B	2	2	1
C	3	2	1
	Ballot 1	Ballot 2	Ballot 3

Schulze Method

	A	B	C
A	0	1	1
B	-1	0	0
C	-1	0	0

$$C = B < A$$

Schulze Method

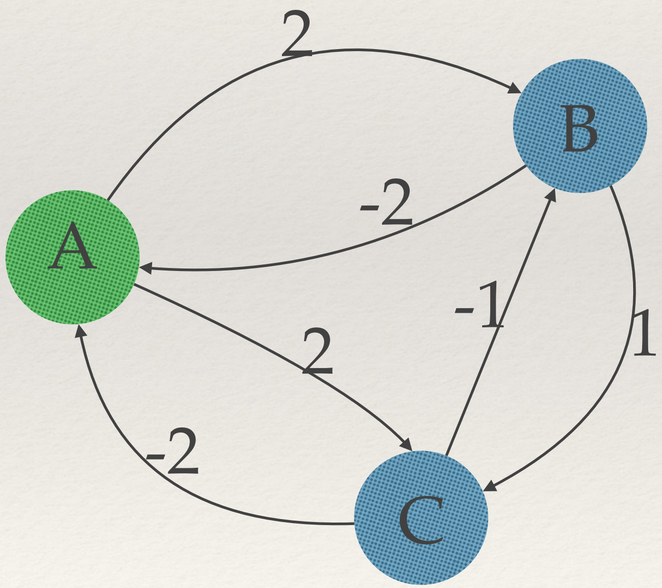


	A	B	C
A	0	2	2
B	-2	0	1
C	-2	-1	0

Margin Matrix (m)

Schulze Method

A wins



	A	B	C
A	0	2	2
B	-2	0	1
C	-2	-1	0

Generalised Margin Matrix (M)

Electronic Voting Requirements

Correctness

Verifiability

Schulze Voting as Evidence Carrying Computation

Dirk Pattinson & Mukesh Tiwari

Conference paper

948 Accesses | 6 Citations

Part of the Lecture Notes in Computer Science book series (LNTCS,volume 10499)

No More Excuses: Automated Synthesis of Practical and Verifiable Vote-Counting Programs for Complex Voting Schemes

Lyria Bennett Moses, Rajeev Goré, Ron Levy, Dirk Pattinson & Mukesh Tiwari

Conference paper | First Online: 06 October 2017

Modular Formalisation and Verification of STV Algorithms

Milad K. Ghale, Rajeev Goré, Dirk Pattinson & Mukesh Tiwari

The Ballot Identification Problem

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
3	2	1	4	6	7	5	11	9	10	15	14	8	13	12

Verifiable Homomorphic Tallying for the Schulze Vote Counting Scheme

Thomas Haines, Dirk Pattinson & Mukesh Tiwari

Conference paper | First Online: 14 March 2020

Part of the Lecture Notes in Computer Science book series (LNPSE,volume 12031)

Encrypted Ballot

$$C < B < A$$

	A	B	C
A	E(0, r1)	E(1,r2)	E(1, r3)
B	E(-1, r4)	E(0, r5)	E(1, r6)
C	E(-1, r7)	E(-1, r8)	E(0, r9)

$$E(m, r) = (g^r, g^m \cdot h^r)$$

	A	B	C
A	E(0, r1)	E(1,r2)	E(1, r3)
B	E(-1, r4)	E(0, r5)	E(1, r6)
C	E(-1, r7)	E(-1, r8)	E(0, r9)

Ballot 1: C < B < A

	A	B	C
A	E(0, r11)	E(-1, r12)	E(-1, r13)
B	E(1, r14)	E(0, r15)	E(-1, r16)
C	E(1, r17)	E(1, r18)	E(0, r19)

Ballot 2: A < B < C

$$Enc(m_1, r_1) = (g^{r_1}, g^{m_1} \cdot h^{r_1})$$
$$Enc(m_2, r_2) = (g^{r_2}, g^{m_2} \cdot h^{r_2})$$
$$Enc(m_1, r_1) \cdot Enc(m_2, r_2) = (g^{r_1+r_2}, g^{m_1+m_2} \cdot h^{r_1+r_2})$$

Challenge

What is the ordering of candidates in this ballot?

	A	B	C
A	E(0, r1)	E(**1**, r2)	E(**1**, r3)
B	E(**1**, r4)	E(0, r5)	E(-1, r6)
C	E(**1**, r7)	E(-1, r8)	E(0, r9)

Challenge

What about this one?

	A	B	C
A	E(0, r1)	E(100, r2)	E(100, r3)
B	E(-100, r4)	E(0, r5)	E(1, r6)
C	E(-100, r7)	E(-1, r8)	E(0, r9)

Challenge

Deciding if a ballot is valid or not

For **every ballot**, the counting authority generates a **secret permutation**, but it is **publicly verifiable** that it is a valid permutation (Zero-Knowledge Proof)

[A -> C, C -> B, B -> A]

	A	B	C
A	E(0, r1)	E(100, r2)	E(100, r3)
B	E(-1, r4)	E(0, r5)	E(1, r6)
C	E(-1, r7)	E(-1, r8)	E(0, r9)

Original Ballot

	A	B	C
A	**E(1, r10)** * E(100, r2)	**E(1, r11)** * E(100, r3)	**E(1, r12)** * E(0, r1)
B			
C			

Row-Shuffled Ballot

+ Zero-Knowledge Proof

[A -> C, C -> B, B -> A]

	A	B	C
A	E(1, r10) * E(100, r2)	E(1, r11) * E(100, r3)	E(1, r12) * E (0, r1)
B			
C			

Row-Shuffled Ballot

	A	B	C
A			
B			
C			

Row-Column-Shuffled Ballot

Zero-Knowledge Proof

Decryption with Zero-Knowledge Proof

	A	B	C
A			
B			
C			

Row-Column-Shuffled Ballot

	A	B	C
A	100	0	100
B	1	1	1
C	-1	-1	0

Decrypted Row-Column-Shuffled Ballot

Zero-Knowledge Proof

Auditing Cryptographic Elections

V: [AA(42928668587819269721303639304427843491, 15743475461916944313294560156147884544 2) AB(631293801065516615816918331994272542 45, 32763269503158514632245943680017270963) AC(70797962417159070468624651915445206901, 44851320559946088690317856734158588876) BA(47416906860069020194818826349401271735, 34723311193461870772237782478013838120) BB(16231383417204232646882226249643084828 2, 28876716000341063398184185368063253071) BC(39939643785190842340347924505399512 18, 16461639287523452172029446278051349424 3) CA(19626097863209944018821197417361884786, 13625069875902636519186533780072762183 8) CB(57509612685432357877489746887710619653, 12170130884507209922345170323063482920 1) CC(19567121790148851030250797881576449952, 89662921375692732648792606491779411997),..], I: [], M: AA(12290478120498989658984921093912932 2326, 11693031680256763098817195579370523988 7) AB(13926097040230515925628907693334634093, 66295795011071066818847417613417087689) AC(16798526665325261214200527126802519604 6, 14882012748689636599245881702481709488) BA(48416931874364056378776256658271698240, 31523941407575957765923588912374345476) BB(15048529392962149383209225103202236978 3, 52856518181869001681848458752677037859) BC(15809498198989339562596456795907541336, 68499163198285646686117318500275517752) CA(39984543038148551463974374548696167991, 69971605318730146660509911752539977872) CB(12776283670275428648209195090629372 5725, 78068383207197270616091945431021157786) CC(10066986377287274916518247743782414 5930, 40762096777064122641927297935184703530), Row Permuted Ballot: AA(53412624706778753501724234693119101183, 16217719416561790784951515674374 2279978) AB(23422537342414087623552660294202804245, 44825524871673235534268937953271384259) AC(72759165689087063250616570841918034101, 47017767514322875659715782012974319885) BA(10826829779202303152966676187149977632, 19657397496703518202798716058224167365) BB(74452133218641902091931698419218892 60, 16264757598693456694458288461331175 3232) BC(20693335079978602930175875924659688507, 60177103228559459463342207838329805614) CA(44501600492469550100725167457120496429, 10058847943906479659114834023818773461 2) CB(12513487068970069342495992555635323 0650, 16010051915126792662092457682327733 7416) CC(59447812662884810834426152069043271766, 98598544526853683350085145251897134557), Column Permuted Ballot: AA(81897112332739457883132019434192167291, 41881053639685806540442005114498285645) AB(17906397596232029936968360268629109460, 14038428094435033913792296468376651178 2) AC(10879893911286545624298365779962844 0647, 14045495647929700413416601757574714 4747) BA(37448939236570114311233606999337576263, 12937693582888945948795178891954567 6840) BB(14503672282789805504068315189430086 5637, 66407832034374248299481283873378557620) BC(10196840289674481405359301776282069960, 13909105758155252276966129938303818 5612) CA(12426928326126240781907089956698213614, 13684309877644337702969747218644693455 8) CB(14638536522729782866981437838346365 9578, 16472204893923681040231207677607781 2657) CC(12357263887065488024072762947910042 0412, 10251153278512312303372740667651229 0401),
Decryption of Permuted Ballot: AA0 AB-1 AC1 BA1 BB0 BC1 CA-1 CB-1 CC0, Zero Knowledge Proof of Row Permutation: [Tuple[Triple[ZModElement[31968434625790105317308409205174964218],
ZModElement[90914039542338682840821682762815886 17], ZModElement[18822457666923254076545558935866204 11]], Pair[GStarModElement[14299079770429258348658117290623563 3356],
Pair[GStarModElement[51971895714242613986924466966227924571], GStarModElement[47554889428172170177484485352803457790]]], ZModElement[27441632724372672019221689845569366377],
Triple[ZModElement[10068043074256531113834418449500621350], ZModElement[24550308266791839946407935571521888 18], Triple[ZModElement[58400559735077862465659159155072176 99],
ZModElement[99256654458219527714611323103151207 15], ZModElement[11839334976269933380004626847991090029]]]]; Tuple[Triple[ZModElement[5631612329965126663199697690466218 2596],
ZModElement[46771638583897386007416832884699081239], ZModElement[79425786387654051381940095476247501724]], Pair[GStarModElement[74899909380444346285670137664849172869],
Pair[GStarModElement[12162375208756535898564676011798 4099254], GStarModElement[12882184048719372906508035660673795 6024]]], ZModElement[29522913915789839486553175106746157 45],
Triple[ZModElement[69850736158747719436830094443438960892 9], ZModElement[41893277228118034703173299418443674300], Triple[ZModElement[75658628603606378941166362451129957 30],
ZModElement[40943225489480601901732102716860640502], ZModElement[37520613677535659786025948592590564651]]]]; Tuple[Triple[ZModElement[68444195985070955262641210850413688 11],
ZModElement[74738586484162419699318702138909108448], ZModElement[67198666917708769371857062017029194]], Pair[GStarModElement[10021806624363574569279187615581906860 3],
Pair[GStarModElement[93773605190217308607835003689913844181], GStarModElement[23792176457766587571084575009217658729]]], ZModElement[35345163014819480622430274366999389 92],
Triple[ZModElement[37984478843103226661566979739786139 3], ZModElement[13798140787804288507342046832102472717], Triple[ZModElement[22794491719240761847552280450517633084],
ZModElement[14047864017471877462622635040970872612], ZModElement[71323139219272515199039205383715229556]]]]], Zero Knowledge Proof of Column Permutation:
[Tuple[Triple[ZModElement[47258223535294714327603182205766620520], ZModElement[51014452219536988420510529763076434 04], ZModElement[45376475102068876223527368065794441739]],
Pair[GStarModElement[62554691807476166976234211217108703053], Pair[GStarModElement[11655664215516008938339047752165421 5452], GStarModElement[11070312794960386728329201217833607 3637]]],
ZModElement[25855182474195412011703031731964470492], Triple[ZModElement[75735944307073157004525474112884481650], ZModElement[57734565352417940471364990482669964 73],
Triple[ZModElement[36661311711273341435106738169173029783], ZModElement[61694725547043688646474636349997494789], ZModElement[83353761070117991688542442496860634003]]]];
Tuple[Triple[ZModElement[67717774757322218905941440729430275044], ZModElement[47588951993696952842511881989894364 85], ZModElement[50652579195583978936337521781725546772]],
Pair[GStarModElement[50862655762759097690627415976512373722], Pair[GStarModElement[16748851379185470918694273525361178053 2], GStarModElement[65645649640596572360315122254599229 00]]],
ZModElement[30736797458434678016159157502182269362], Triple[ZModElement[39920185838646295267415843136645203 94], ZModElement[47439072215303152810024052086027501 81],
Triple[ZModElement[68091076375476219178234251994933983525], ZModElement[25451000004463165377104341871250847228], ZModElement[65774131621579883850539469053743022913]]]];
Tuple[Triple[ZModElement[13852557157818347937192852034240972635], ZModElement[38187242824159603365948078539829369 23], ZModElement[84442698181989374411229671388260775036]],
Pair[GStarModElement[51695159854615807945684566565949735916], Pair[GStarModElement[12965173297426374108488158485255788984 6], GStarModElement[57484066030163226053910957607720608763]]],
ZModElement[46260583246662963979630357227342425864], Triple[ZModElement[83347143319488920409944070774231865933], ZModElement[11641586680698176312901212770999845763],
Triple[ZModElement[15671396369987487104072672165830301 05], ZModElement[33714947378227442090889514561950742575], ZModElement[44578770697130634673687505939329786286]]]]], Zero Knowledge Proof of Decryption:
[Triple[Pair[GStarModElement[43561432915304444518083539460410208 12], GStarModElement[13479060889852891201587975012223763802 6]], ZModElement[15054343775183293657759741258189634472],
ZModElement[78462672003747872157282068562995869345]]; Triple[Pair[GStarModElement[99187265121091587883874331507159259184], GStarModElement[13931727357169015082787207694776 1944921]],
ZModElement[41210731301857756811265423124034455122], ZModElement[39603359097257057092072276161317907473]]; Triple[Pair[GStarModElement[85530403342365972144404795230697825178],
GStarModElement[15935447183899959735396021894613975909]], ZModElement[48710416412324962373324948249188158011], ZModElement[53506690221720868312062522839387616154]]], Permutation commitment:
Triple[GStarModElement[15248545316390671033073909314819466307 0], GStarModElement[29523442668474248947227922275796563365], GStarModElement[13070311312263622368929173166806418585 0]], Zero Knowledge Proof of commitment:
Tuple[Triple[ZModElement[30496233531000788386223096342107092718], ZModElement[65601380177914031085254386530980181750], ZModElement[75956078106460021439188012242290726261]],
Triple[GStarModElement[53848200136924947578935181350074431637], GStarModElement[25864911063678150567044115699075777441], GStarModElement[16783274825763146949593531267332168263 7]],
Tuple[GStarModElement[78732063467630215555582500593481455398], GStarModElement[89068631409607048015168698512821595874], GStarModElement[70638108326972021902986584411337150003],
GStarModElement[14332760054790926036351398012688691177], GStarModElement[36922434616664431920358457300829293435], GStarModElement[10375723757619048802484019209319155935]],
ZModElement[38510749350776483009209925870220536060], Tuple[ZModElement[64228926163594004923860808074744104973], ZModElement[42134683875741912600850253785764499 15],
Triple[ZModElement[69350627855380840740835314045798488 43], ZModElement[30156600121931236994458514919091065 21], ZModElement[74378651412666438952903024117135262788]], ZModElement[15211314027711782441674685161999630 36],
Triple[ZModElement[55970119344921807416005474333220351354], ZModElement[36119357575876843953211978338379631473], ZModElement[11421678842563484045461879662668905496]]]

RESEARCH-ARTICLE

Verified Verifiers for Verifying Elections

Authors: Thomas Haines, Rajeev Goré, Mukesh Tiwari Authors Info & Claims

Machine-checking Multi-Round Proofs of Shuffle: Terelius-Wikstrom and Bayer-Groth

Authors:
Thomas Haines, *Australian National University;* Rajeev Gore, *Polish Academy of Science;* Mukesh Tiwari, *University of Cambridge*

Integration Project

- Formal Verification of Software/Hardware Interface (FreeSpec, SpecCert, etc.)
- Formal Verification of Embedded System

Questions?