WATERLOO
CHERITON SCHOOL OF
COMPUTER SCIENCE

Selections:

Internet voting with over-the-shoulder coercion-resistance

Jeremy Clark

Overview

- We consider the problem of over-the-shoulder adversaries in Internet voting
- We design a voting protocol resistant to coercion and vote selling attacks
- Selections uses a panic password system
- Tallying and revocation of voters are efficient in the number of voters

Internet Voting

- We are interested in voting systems with three properties:
  - Verifiable: the results are provably correct
  - Coercion-Resistant: an adversary cannot determine how a voter voted or force a voter to vote a certain way
  - Remote/Internet: vote casting is unsupervised; adversaries may be present
- VCRR (Verifiable/Coercion-Resistant/Remote)
- We do not deal with the untrusted platform issue (currently a separate line of research)

JCJ/Civitas

- Juels et al. [JCJ05] propose first VCRR voting system
- Clarkson et al. [CCM08] implement it as Civitas
- Main issue: tallying is roughly quadratic in the number of voters
- Voters are issued credentials; if coerced, they can generate a fake secret key and simulate a proof of correctness

JCJ/Civitas

- Selections improves the efficiency of tallying to linear
- Selections makes authentication password-based (“something you know”) and generating a fake password can be performed mentally
- Selections implements bare-handed registration
- JCJ also offers a game-based definition of coercion resistance that we use for our security proof

AFT

- Araujo et al. [AFT07] also provide a linear-time VCRR voting system
- Voters are issued signed credentials, however verification algorithm is private
- If coerced, voters submit an unsigned value and simulate proof of signature
- Revocation is difficult if voters lose credentials or voter list needs modification after election begins
- In Selections, revocation is efficient

Others

- Other VCRR voting systems (or subprotocols) have been proposed:
  - Smith [Smi05] and Weber et al. [WAB07] provide linear tally but are broken with respect to coercion resistance
  - Acquisti [Acq04] allows write-ins; also broken
  - Krivoruchko [Kri07] and Wen & Buckland [WB09] provide registration protocols with certain merits
  - Araujo et al [ARRTY10] improves AFT. Too recent for consideration in this work

Building Blocks

Exponential Elgamal

- We use the exponential variant of Elgamal [CGS96]
- Essentially, it is Enc(g^m) where Enc(m) is regular Elgamal
- Allows an additive homomorphism, however decryption is limited to small m
- In Selections, we never decrypt to recover m, only to perform a plaintext equality test between two ciphertexts

Threshold Elgamal

- We also use the threshold variant of Elgamal [Ped91]
- *n* trustees generate a public key such that *t* out of *n* can jointly decrypt a message with their private key shares

Plaintext Equality Test

- Due to Jakobsson & Juels [JJ00]

Given $c_1 = \mathsf{Enc}(m_1)$ and $c_2 = \mathsf{Enc}(m_2)$:

$$\mathsf{PTE}(c_1, c_2) = \begin{cases} 1 & m_1 = m_2 \\ \neg 1 & m_1 \neq m_2 \end{cases}$$

Jointly blind $\hat{c} = (((c_1/c_2)^{b_1})^{b_2} \ldots)^{b_t}$ and then jointly decrypt $\mathsf{Dec}(\hat{c})$.

Panic Passwords

- Due to Clark & Hengartner [CH08]
- System responds indistinguishably between real and fake passwords, but some hidden action is taken if fake password is used
- For voting, votes cast with fake passwords are discarded
- Hard part: making this verifiable while protecting the actions of voters

Panic Passwords

- Trivial solution: issue two passwords, one real and one fake
  - Does not work here: adversary will demand two password and vote with both
- Issue one password, everything other than it is a fake password
  - Usability issues: a typo would not be detected
- Want: arbitrarily large number of fake passwords distributed sparsely

Panic Passwords

- 5P system:
  - Passwords are 5 words from a dictionary
  - Any other 5 dictionary words will be a panic password
  - Any arbitrary string not in the dictionary will be invalid
- Users only memorize one password, rule for generating panic passwords can be done mentally, sufficient entropy with 5 words ($|D|^5$ ~ 70 bits for Unix dictionary), typos fairly likely to be invalid

Bare-Handed Proofs

- We want proofs that convince only the voter
- However, voter should be convinced without a computer
- Bare-handed: computers can only be used before to prepare values or afterward to verify aspects of the proof that do not reveal what was proven

Bare-Handed Proofs

- Election with 3 candidates: Alice, Bob, and Carol
- Voter votes for Bob and machine prints encryption of Bob's name
- With an interactive sigma protocol, the machine prints:
  - A simulated proof that the ciphertext encrypts Alice
  - Transcript of a real proof that the ciphertext encrypts Bob
  - A simulated proof that the ciphertext encrypts Carol
- For the real proof pertaining to Bob, the voter only verifies the *order* is correct (commitment before voter's challenge). This part is bare-handed
- The voter retains the ciphertext and 3 proof transcripts. The validity of all three proofs are verified with a computer afterward

Untappable Channels

- It seems we cannot completely eliminate the need for an untappable channel and maintain coercion-resistance
- The next-best thing is to use it only once and bootstrap that interaction into an arbitrary number of future interactions that are coercion-resistant

Registration

Registration

- We want an encryption of the voter's password posted on a public list such that no one knows the password except the voter
- Additionally, the voter should not be able to prove knowledge of the password
- Basic approach: voter encrypts, registrant rerandomizes, and registrant provides bare-handed proof of correct rerandomization

Registration

- The approach of issuing a real proof alongside simulated proofs won’t work: adversary will try all passwords
- We use a simple cut-and-choose (adapted from Benaloh [Ben06])
- Voter prepares encryptions for e.g., 10 passwords
- The registrant will rerandomize the ciphertexts and print proofs

Registration

- The voter discloses which it wants to register and destroys the accompanying proof
- The registrant is a machine, input values can be barcodes, and proofs are printed onto scratch-off cells
- To erase a proof, the voter scratches off the cell
- A confirmation code could be under the scratch-off to demonstrate the information was destroyed

c_1
c_2
c_3
c_4
c_5
c_6
c_7
c_8
c_9
c_{10}

c_1
c_2
c_3
c_4
c_5
c_6
c_7
c_8
c_9
c_{10}

c_1	r_1'	c_1'
c_2	r_2'	c_2'
c_3	r_3'	c_3'
c_4	r_4'	c_4'
c_5	r_5'	c_5'
c_6	r_6'	c_6'
c_7	r_7'	c_7'
c_8	r_8'	c_8'
c_9	r_9'	c_9'
c_{10}	r_{10}'	c_{10}'

c_1
c_2
c_3
c_4
c_5
c_6
c_7
c_8
c_9
c_{10}

c_1	r_1'	c_1'
c_2	r_2'	c_2'
c_3	r_3'	c_3'
c_4	r_4'	c_4'
c_5	r_5'	c_5'
c_6	r_6'	c_6'
c_7	r_7'	c_7'
c_8	r_8'	c_8'
c_9	r_9'	c_9'
c_{10}	r_{10}'	c_{10}'

"Use 6th"

c_1
c_2
c_3
c_4
c_5
c_6
c_7
c_8
c_9
c_{10}

c_1	r_1'	c_1'
c_2	r_2'	c_2'
c_3	r_3'	c_3'
c_4	r_4'	c_4'
c_5	r_5'	c_5'
		c_6'
c_7	r_7'	c_7'
c_8	r_8'	c_8'
c_9	r_9'	c_9'
c_{10}	r_{10}'	c_{10}'

"Use 6th"

c_1	r_1'	c_1'
c_2	r_2'	c_2'
c_3	r_3'	c_3'
c_4	r_4'	c_4'
c_5	r_5'	c_5'
		c_6'
c_7	r_7'	c_7'
c_8	r_8'	c_8'
c_9	r_9'	c_9'
c_{10}	r_{10}'	c_{10}'

"Use 6th"

c_1	r_1'	c_1'
c_2	r_2'	c_2'
c_3	r_3'	c_3'
c_4	r_4'	c_4'
c_5	r_5'	c_5'
		c_6'
c_7	r_7'	c_7'
c_8	r_8'	c_8'
c_9	r_9'	c_9'
c_{10}	r_{10}'	c_{10}'

Public Roster	
Voter 1	
Voter 2	$c=c_6'$
Voter 3	

"Use 6th"

Registration

- Soundness is 1-1/L, where L is 10 in the example
- Ideally, soundness would be $1-1/2^L$
- Improving soundness: open problem
- Improving usability: open problem

Vote Casting

Vote: $\{ B , c', \pi_1 , g^p , \pi_2 \}$

Selections is designed to be versatile with common ballot types from E2E systems

Generally, B will be an encryption of a candidate with a validity proof

Required to be submittable to a mix-network

Vote: $\{ B , c', \pi_1 , g^p , \pi_2 \}$

Roster

c_1=Enc(g^{p1})

c_2=Enc(g^{p2})

c_m=Enc(g^{pm})

c'=ReRand(g^{pi})

Vote: { B , c', π_1 , g^p , π_2 }

Roster

$c_1 = \mathrm{Enc}(g^{p1})$

$c_2 = \mathrm{Enc}(g^{p2})$

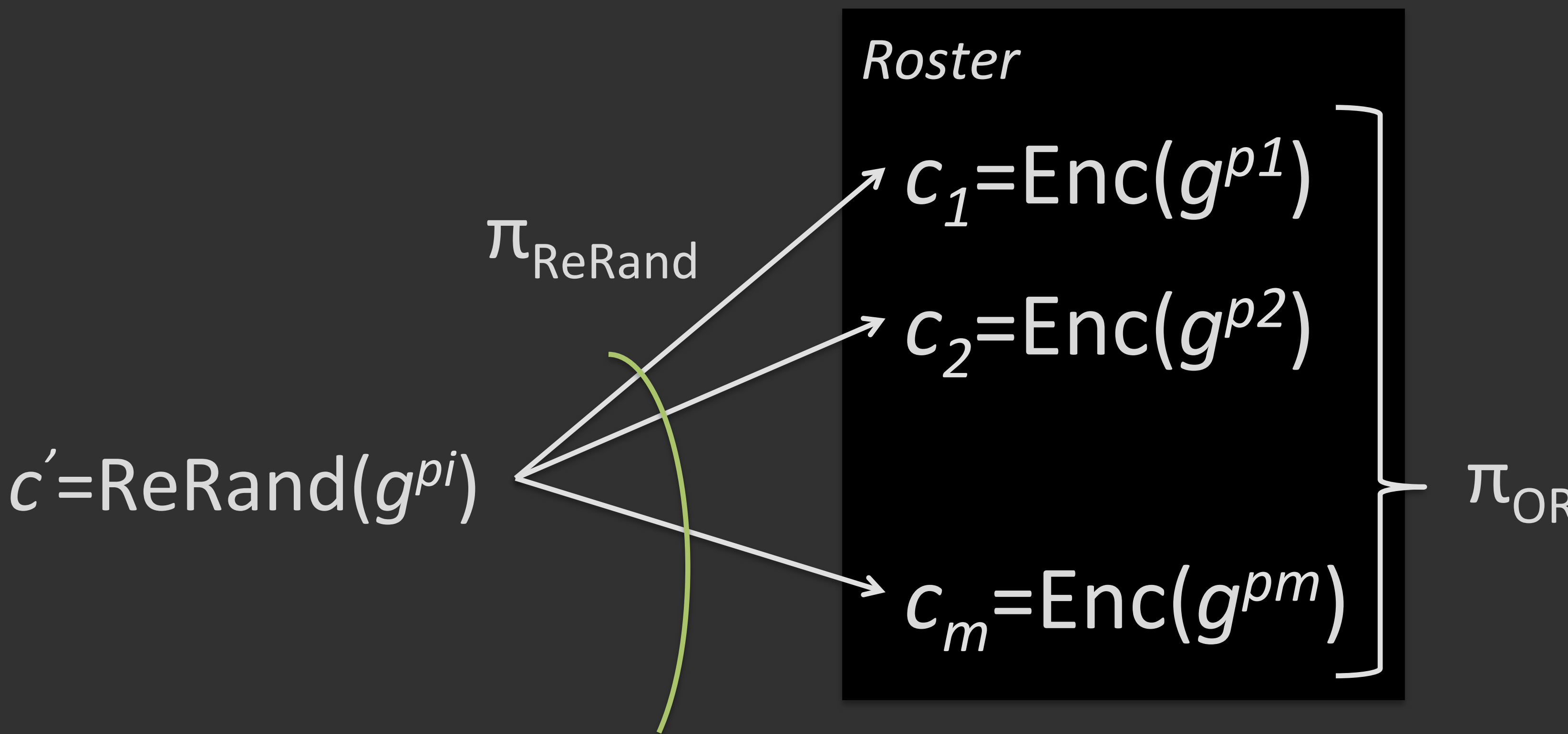
$c_m = \mathrm{Enc}(g^{pm})$

π_{ReRand}

$c' = \mathrm{ReRand}(g^{pi})$

π_{OR}

Vote: $\{\ B\ ,\ c',\ \pi_1\ ,\ g^p\ ,\ \pi_2\ \}$



β : number of included entries, creates an anonymity set

The size of β impacts coercion resistance

Roster

$$\pi_{\mathrm{ReRand}}$$

$$c' = \mathrm{ReRand}(g^{pi}) \quad \left.\begin{array}{l} c_1 = \mathrm{Enc}(g^{p1}) \\ c_2 = \mathrm{Enc}(g^{p2}) \\ \\ c_m = \mathrm{Enc}(g^{pm}) \end{array}\right\} \pi_{\mathrm{OR}}$$

Vote: { B , c’, π_1 , g^p , π_2 }

The voter asserts their password and encodes it as g^p

It may or may not match the password encrypted in c'

Vote: $\{ B , c', \pi_1 , g^p , \pi_2 \}$

Intuition: eventually (after anonymization), the trustees will be able to compare these values with a plaintext equality test (PTE)

PTE (recall $c' = Enc(g^p)$ from Roster)

Vote: $\{ B , c', \pi_1 , g^p , \pi_2 \}$

The voter proves knowledge of *p* in a way that is simultaneous to other values in the tuple (*e.g.*, inclusion in RO query via Fiat-Shamir)

This prevents an adversary from replaying a $\{g^p, \pi_2\}$ pair alongside a modified *B* or *c'*

Vote: $\{ B , c', \pi_1 , g^p , \pi_2 \}$

Vote Processing

Vote	Ballot	Roster Entry	Proof off of Roster	Asserted Password	PoK of Password
1	B	c’	π_1	g^p	π_2
2	B	c’	π_1	g^p	π_2
3	B	c’	π_1	g^p	π_2
4	B	c’	π_1	g^p	π_2
5	B	c’	π_1	g^p	π_2
6	B	c’	π_1	g^p	π_2
7	B	c’	π_1	g^p	π_2
8	B	c’	π_1	g^p	π_2
9	B	c’	π_1	g^p	π_2

Vote	Ballot	Roster Entry	Proof off of Roster	Asserted Password	PoK of Password
1	B	c’	π_1	g^p	π_2
2	B	c’	π_1	g^p	π_2
3	B	c’	π_1	g^p	π_2
4	B	c’	π_1	g^p	π_2
5	B	c’	π_1	g^p	π_2
6	B	c’	π_1	g^p	π_2
7	B	c’	π_1	g^p	π_2
8	B	c’	π_1	g^p	π_2
9	B	c’	π_1	g^p	π_2

Step 1: Check Proofs

Vote	Ballot	Roster Entry	Proof off of Roster	Asserted Password	PoK of Password
1	B	c’	π_1	g^p	π_2
2	B	c’	π_1	g^p	π_2
3	B	c’	π_1	g^p	π_2
4	B	c’	π_1	g^p	π_2
5	B	c’	π_1	g^p	π_2
6	B	c’	π_1	g^p	π_2
7	B	c’	π_1	g^p	π_2
8	B	c’	π_1	g^p	π_2
9	B	c’	π_1	g^p	π_2

Step 1: Check Proofs

Vote	Ballot	Roster Entry	Proof off of Roster	Asserted Password	PoK of Password
1	B	c’	π_1	g^p	π_2
2	B	c’	π_1	g^p	π_2
3	B	c’	π_1	g^p	π_2
4	B	c’	π_1	g^p	π_2
6	B	c’	π_1	g^p	π_2
7	B	c’	π_1	g^p	π_2
8	B	c’	π_1	g^p	π_2
9	B	c’	π_1	g^p	π_2

Step 1: Check Proofs

Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
3	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p

Step 1: Check Proofs

Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
3	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p

Step 2: Check for Duplicates

Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
3	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p

Step 2: Check for Duplicates

Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
3	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p

Same

Step 2: Check for Duplicates

Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p

Delete Oldest

Step 2: Check for Duplicates

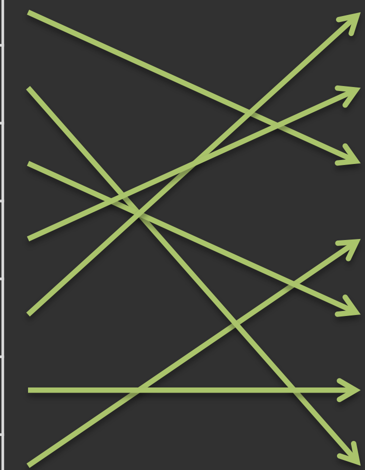
Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p

Step 3: Mix

Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p

Step 3: Mix

Vote	Ballot	Roster Entry	Asserted Password
1	B	c’	g^p
2	B	c’	g^p
4	B	c’	g^p
6	B	c’	g^p
7	B	c’	g^p
8	B	c’	g^p
9	B	c’	g^p



Ballot	Roster Entry	Asserted Password
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c

Each trustee:

Shuffle & Rerandomize & Prove [JJR02]

Step 3: Mix

Ballot	Roster Entry	Asserted Password
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c

Ballot	Roster Entry	Asserted Password
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c

Step 4: Check Passwords

Ballot	Roster Entry	Asserted Password
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c

PTE for each pair

Step 4: Check Passwords

Ballot	Roster Entry	Asserted Password
B	c’	c
B	c’	c
B	c’	c
B	c’	c
B	c’	c

PTE for each pair

Step 4: Check Passwords

Ballot
B
B
B
B
B

Step 4: Check Passwords

Ballot
B
B
B
B
B

Output: Eligible & Valid Ballots

Coercion-Resistance

Overview

- Register once and in-person
- If coerced, use panic password then later cast second vote with real password
- If selling, no guarantee password is real

Security Game

- System is set-up with honest voters, corrupted voters (non-adaptive adversary), and a voter specified for coercion
- A coin is flipped
- Upon heads, the voter complies fully with the adversary
- Upon tails, the voter deceives the adversary and achieves its original goal

Security Game

- A system is said to be coercion resistant if:
  - The voter can actually achieve its original goal with certainty when deceiving
  - The adversary cannot distinguish a compliant voter from a deceptive one (non-negligibly) better than it could with an ideal voting system
- Ideal voting system: voters give votes to trusted party and party outputs a tally

Security Game

- Ideal voting system comparison is important because an adversary can distinguish compliant voters from deceptive voters with just a tally!
- Example: adversary buys a vote for Alice and:
  - Final tally has no votes for Alice (deceived)
  - Only one voter votes a final tally shows one vote for Alice (complied)
  - Probabilistic tests comparing expected votes for a candidate to actual

Selections

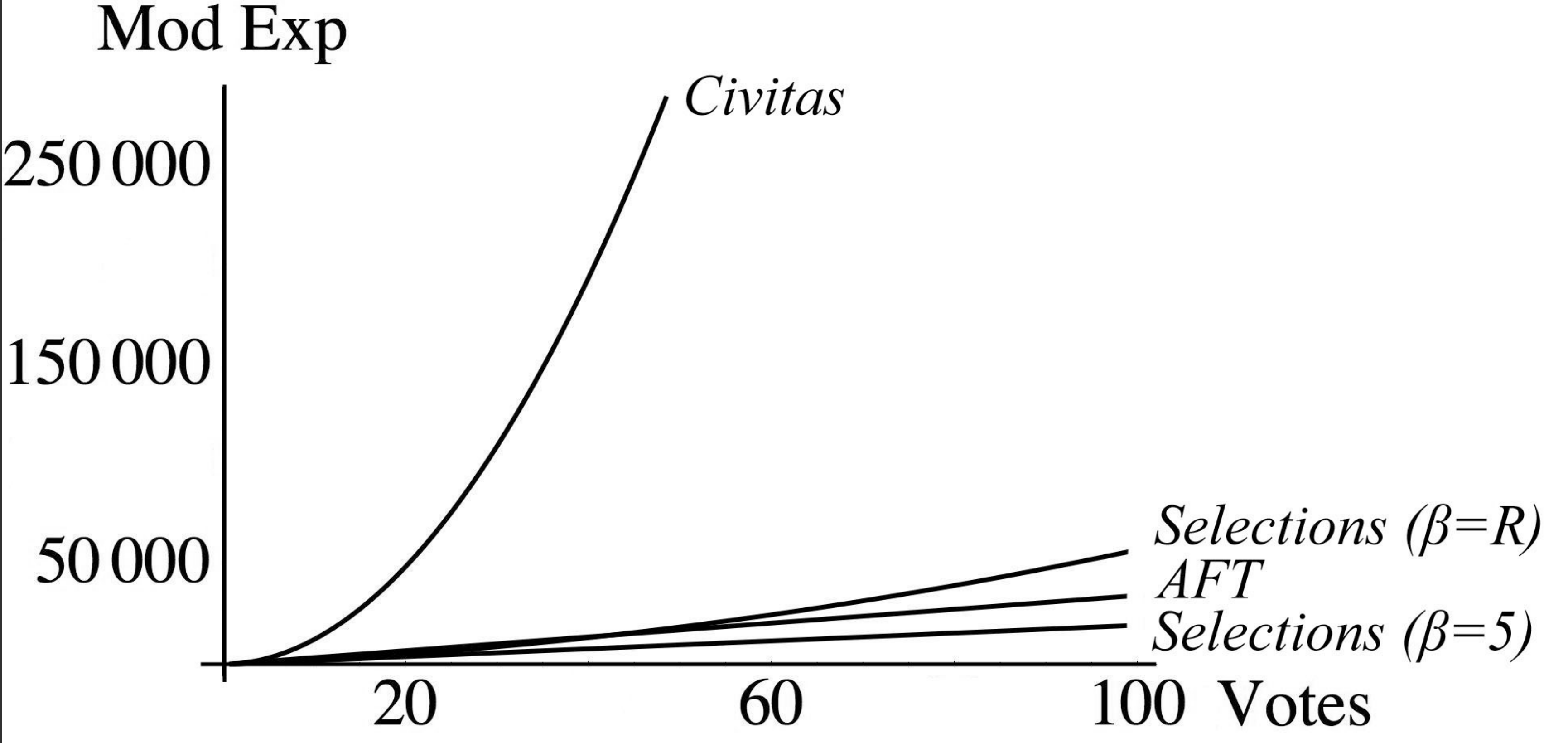
- In Selections, compliant voters give the adversary their real password
- Deceptive voters give the adversary a fake password and covertly cast a second vote with their real password
- Coercion resistance of Selections is based on DL-problem and CPA-security of Elgamal

Additional Notes

Efficiency

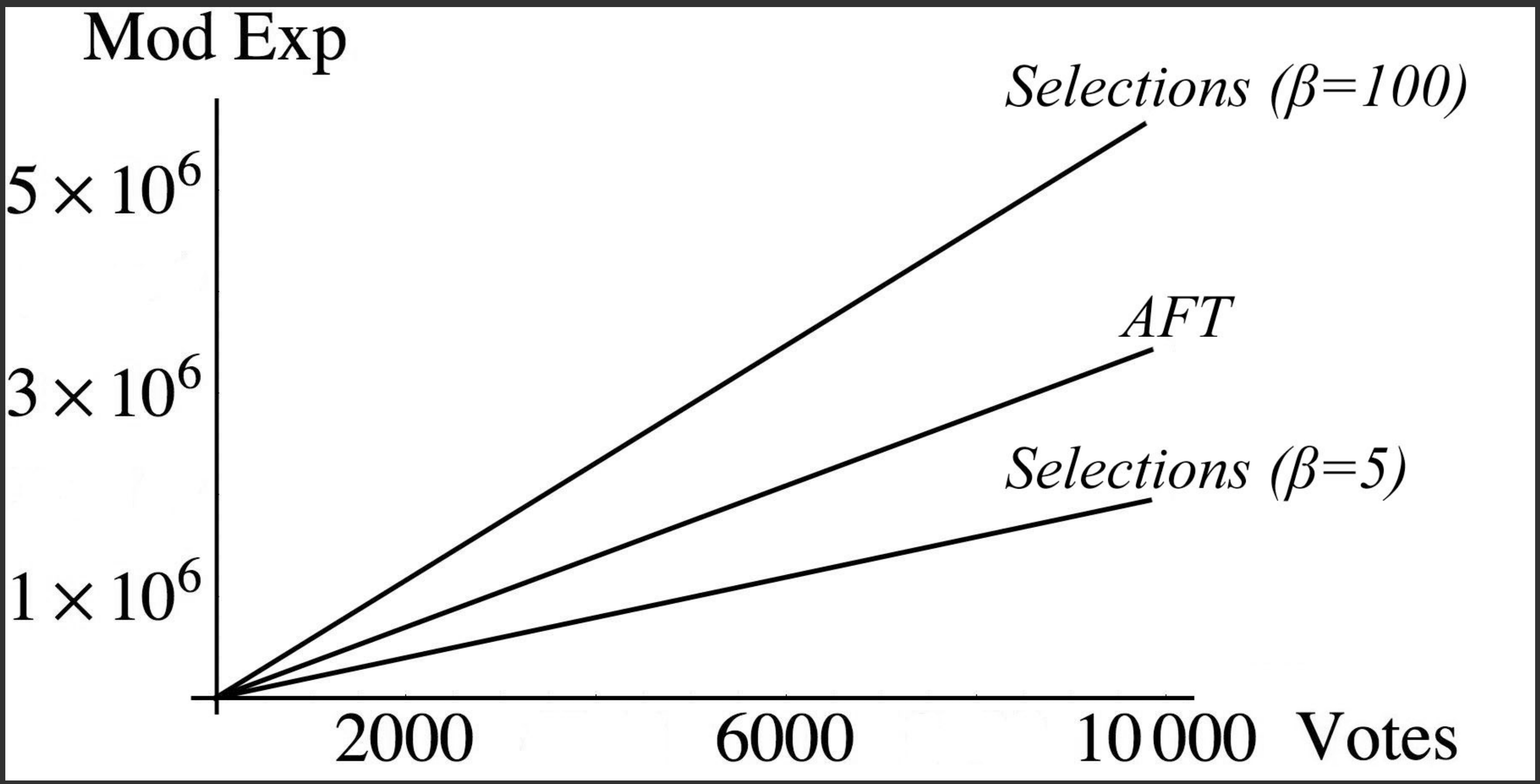
		Civitas	AFT	Selections
Registration	Registrar	7	9	2α
	Voter	11	10	4α-1
Casting	Voter	10	24	$(2\beta + 9)$
Pre-Tally	Check Proofs	$4V_0$	$20V_0$	$(4\beta + 6)V_0$
	Remove Duplicates	$(1/2)(V_1^2 - V_1)(8T + 1)$	—	—
	Check Removal	$(1/2)(V_1^2 - V_1)(8T + 1)$	—	—
	Mix	$8V_2T + 4RT$	$20V_2T$	$12V_2T$
	Check Mix	$4V_2T + 2RT$	$10V_2T$	$6V_2T$
	Remove Unregistered	$(8A + 1)V_2R$	$(16T + 8)V_2$	$(8T + 1)V_2$
	Check Removal	$(8A + 1)V_2R$	$(16T + 10)V_2$	$(8T + 1)V_2$

Efficiency



Mod Exp
250 000
150 000
50 000
Civitas
Selections (β=R)
AFT
Selections (β=5)
20
60
100
Votes

Efficiency



Mod Exp
Selections (β=100)
AFT
Selections (β=5)
5 × 10^6
3 × 10^6
1 × 10^6
2000
6000
10 000
Votes

Election-Specific Values

- Voters submit g^p
- Values could be matched across elections to identify voters
- Use a fresh generator for each election
- Trustees modify g with exponential blinding factors, likewise modify the Roster values, and publish new generator

Revocation

- Voters names can be crossed off the Roster (or moved)
- If other voters have begun voting, we need to ensure the revoked voter has not cast a ballot yet
- We only need to look a votes that include the revoked voter in the anonymity set and then we can use a PTE
- Coercion resistance does not extend to revoked voters!

Time-Consuming Step

Roster

π_{ReRand}

$$c^* = \mathrm{ReRand}(g^{pi}) \qquad \left.\begin{array}{l} c_1 = \mathrm{Enc}(g^{p1}) \\ c_2 = \mathrm{Enc}(g^{p2}) \\ \\ c_n = \mathrm{Enc}(g^{pn}) \end{array}\right\} \pi_{\mathrm{OR}}$$

π_{ReRand}: Chaum-Pedersen

π_{OR}: Cramer-Damgard-Schoenmakers

Conclusions

- Over the shoulder coercion and vote selling can be solved
- Still requires an in-person interaction
- Easy to transition to: voters voting in current election can register to vote online in the next
- Open problem: the voter’s untrusted computer

Questions?