CHvote – towards 2.0

Evolution of the electronic voting system of Canton Geneva

Bio

- Thomas Hofer
- Java Dev
- OWASP Geneva, *co-chapter leader*

- State of Geneva

- @thhofer / thomas.hofer@owasp.org

REPUBLIQUE ET CANTON DE GENEVE

POST TENEBRAS LUX

OWASP
Open Web Application Security Project

Outline

- Context
- Updated requirements
- Core protocol concepts
- Implementation overview
- Current results
- Ongoing work

Context

- Stable democracy
- 4 referendum & initiatives rounds / year
- municipal, cantonal and federal elections
  - Usually 2-4 rounds / year
- currently used by several cantons
- Developed, hosted & maintained by Geneva

Preliminary project – Goals

- New voting protocol (BFH)
- PoC Implementation (State of Geneva)
  - Feasibility
  - Performance
  - Constraints and limitations
- Publication of specification and code

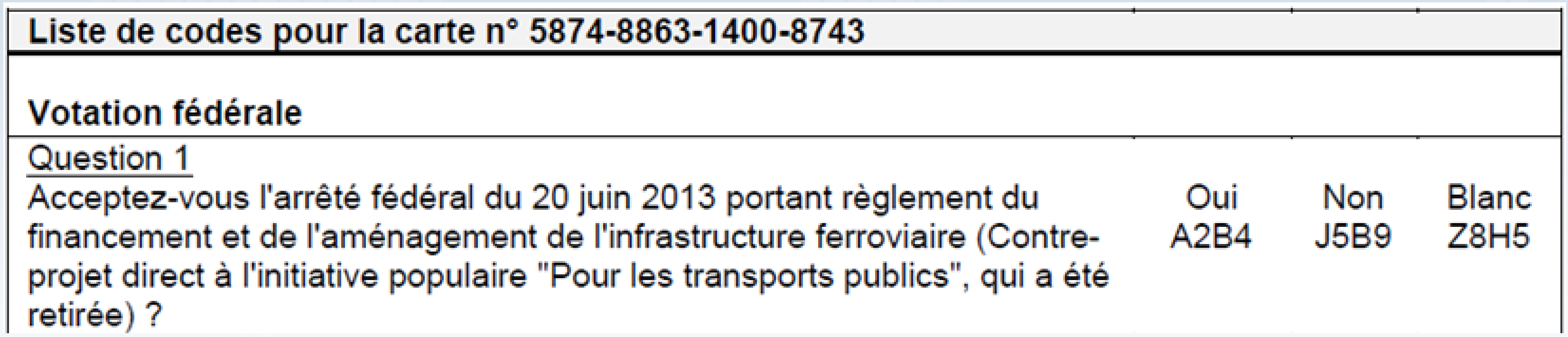
Updated requirements > Intro

- New ordinance on Electronic Voting (2013)
  - Technical & admin requirements
- Compliance levels -> allowed percentage of electorate
  - 30% / 50% / 100%
- https://www.bk.admin.ch/themen/pore/evoting/07979/index.html

OWASP
Open Web Application Security Project

Upd. Req. > Individual verifiability

voters must receive proof that the server system has registered the vote as it was entered by the voter on the user platform – *VEleS, art. 4*



Liste de codes pour la carte n° 5874-8863-1400-8743

Votation fédérale

Question 1 Acceptez-vous l'arrêté fédéral du 20 juin 2013 portant règlement du financement et de l'aménagement de l'infrastructure ferroviaire (Contre-projet direct à l'initiative populaire "Pour les transports publics", qui a été retirée) ?	Oui A2B4	Non J5B9	Blanc Z8H5

- In current version: random codes per voter / response

Upd. Req. > End-to-End encryption

Votes must not be stored or transmitted in unencrypted form at any time from being entered to tallying. – *Technical and administrative requirements, section 3.3.4*

- In current version:
  - Incompatible with individual verifiability implementation
  - Server needs to know vote to return the matching verification code

OWASP
Open Web Application Security Project

Upd. Req. > Universal verifiability

For universal verification, the auditors receive proof that the result has been ascertained correctly. They must evaluate the proof in a observable procedure. – *VEleS, art. 5 paragraph 4*

- In current version:
  - Not available; external supervision by party representatives holding the private decryption key

OWASP
Open Web Application Security Project

Upd. Req. > Control components

The trustworthy part of the system includes either one or a small number of groups of independent components secured by special measures (control components). Their use must also make any abuse recognisable if per group only one of the control components works correctly and in particular is not manipulated unnoticed. – *VEleS, art. 5, par. 6*

- In current version:
  - Application server protected by organisational measures and enforced policies

Core protocol concepts

- El Gamal homomorphic encryption
- Oblivious Transfer for individual verifiability
  - Cast-as-Intended Verification in Electronic Elections Based on Oblivious Transfer
- Pedersen Commitments
- Non-interactive Zero-Knowledge Proofs
- Wikström's Proof of a Shuffle

Homomorphic encryption

- Allows re-encryptions
  - Useful for anonymizing when shuffling
- Allows for key sharing
  - Control components each hold a key share

Oblivious Transfer

- In short
  - Server knows n secret messages
  - Client allowed to retrieve k secret messages
  - Server cannot know which messages the client asked for
  - *Perfect match for the verification codes issue!*
- In detail
  - Cast-as-Intended Verification in Electronic Elections Based on Oblivious Transfer

OWASP
Open Web Application Security Project

Commitments and ZKPs

- "public" commitments for the secrets
- ZKPs relative to those commitments
  - Chain of truth from key generation to ballot decryption

- Combination yields Universal verifiability

OWASP
Open Web Application Security Project

Wikström's Proof of a Shuffle

- Re-encrypting mix-net

- Since shuffled, simple pre-image proofs would not work
- Since re-encrypted, ciphertexts are not equal

- Need for a specific profo that the cryptographic shuffle is valid

OWASP
Open Web Application Security Project

Implementation

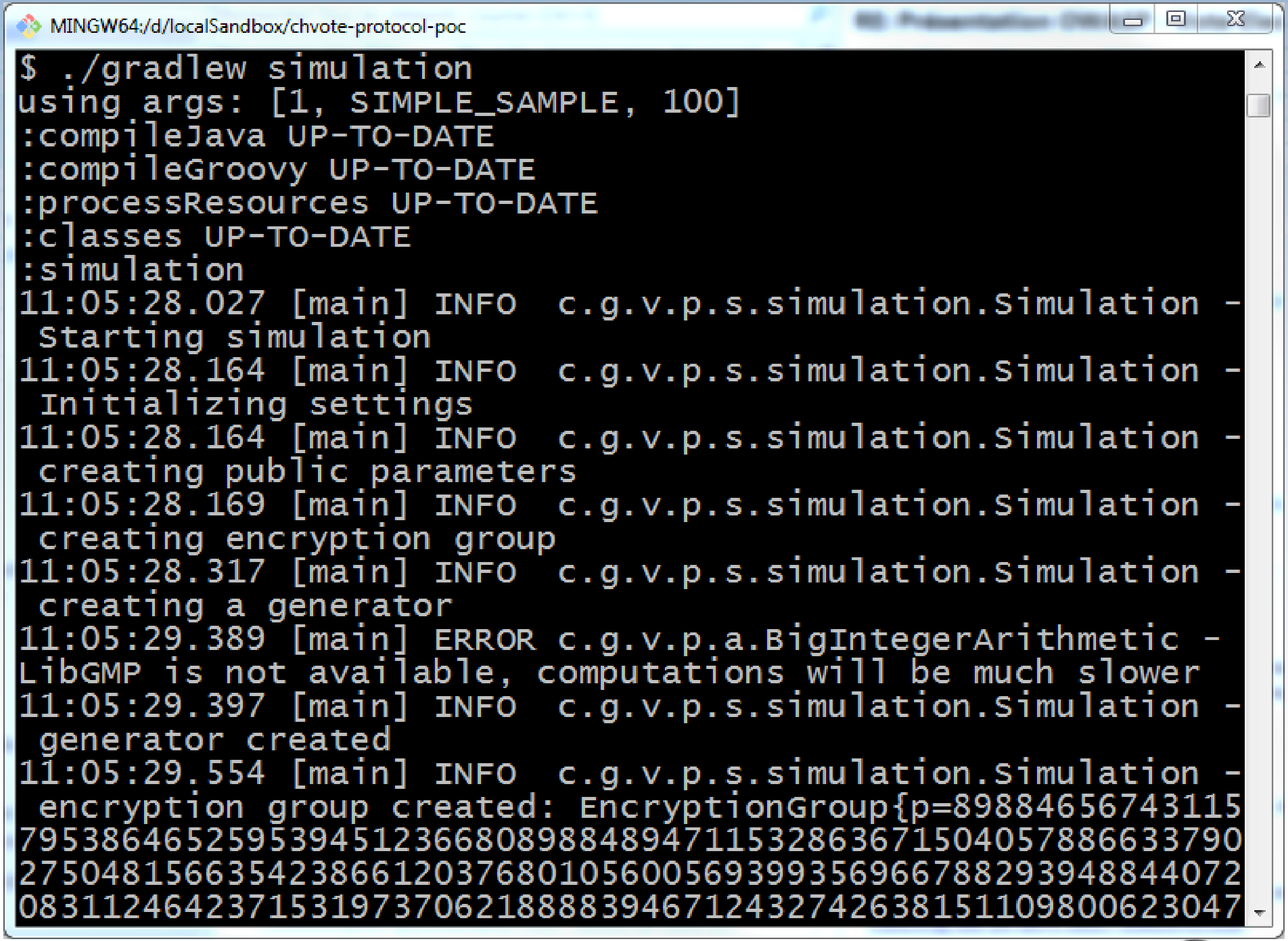
- Algorithms
  - ch.ge.ve.protopoc.service.algorithm
- Utilities defined in specification
  - ch.ge.ve.protopoc.service.support
- Simulation-related classes
  - ch.ge.ve.protopoc.service.simulation
- Run simulation
  - `./gradlew simulation`

OWASP
Open Web Application Security Project

Implementation – Snippet

```
/**
 * Algorithm 7.4: GetNIZKPChallenge
 *
 * @param y     the public values vector (domain unspecified)
 * @param t     the commitments vector (domain unspecified)
 * @param kappa the soundness strength of the challenge
 * @return the computed challenge
 */
public BigInteger getNIZKPChallenge(Object[] y, Object[] t,
                                    int kappa) {
    return conversion.toInteger(
        hash.recHash_L(y, t)).mod(BigIntegers.TWO.pow(kappa));
}
```

Implementation – Demo


MINGW64:/d/localSandbox/chvote-protocol-poc

```
$ ./gradlew simulation
using args: [1, SIMPLE_SAMPLE, 100]
:compileJava UP-TO-DATE
:compileGroovy UP-TO-DATE
:processResources UP-TO-DATE
:classes UP-TO-DATE
:simulation
11:05:28.027 [main] INFO  c.g.v.p.s.simulation.Simulation -
 Starting simulation
11:05:28.164 [main] INFO  c.g.v.p.s.simulation.Simulation -
 Initializing settings
11:05:28.164 [main] INFO  c.g.v.p.s.simulation.Simulation -
 creating public parameters
11:05:28.169 [main] INFO  c.g.v.p.s.simulation.Simulation -
 creating encryption group
11:05:28.317 [main] INFO  c.g.v.p.s.simulation.Simulation -
 creating a generator
11:05:29.389 [main] ERROR c.g.v.p.a.BigIntegerArithmetic -
LibGMP is not available, computations will be much slower
11:05:29.397 [main] INFO  c.g.v.p.s.simulation.Simulation -
 generator created
11:05:29.554 [main] INFO  c.g.v.p.s.simulation.Simulation -
 encryption group created: EncryptionGroup{p=8988465674311S
7953864652595394512366808988489471153286367150405788663379O
27504815663542386612037680105600569399356966788293948844072
08311246423715319737062188883946712432742638151109800623047
```

Results: Specification

- https://ia.cr/2017/325
- Written by team at BFH

OWASP
Open Web Application Security Project

Results: PoC implementation

- Covers complete protocol (incl. proofs)
- Available on GitHub
  - https://github.com/republique-et-canton-de-geneve/chvote-protocol-poc
- Issues & PRs welcome!

Result: Performance estimates

- Also available on GitHub
  - Much better than initially feared
  - 100k ballots could be
    - Shuffled,
    - Decrypted,
    - & Verified;
    - Using “standard” hardware
    - Within operational time constraints

Soooo… what’s left then?

- GUI ☺
- Distribution
  - Real infrastructure for *Control Components*
- Resilience
- Custom rules for layout, specific elections, …
- Back-office, test zone, …
- Cantonal interoperability

Q&A

Open Web Application Security Project