Esecurity: secure internet & evoting, summer 2010

MICHAEL NÜSKEN, KONSTANTIN ZIEGLER

12. Exercise sheet
Hand in solutions until Sunday, 18 July 2010, 23.59 h

We take a look at the remaining seven protocols not discussed so far: REENCPF, VOTEPF, PET, MIXNET, REGISTER, VOTE, TABULATE (see the appendix). The aim of this assignment is to get a first hands-on experience with them.

Exercise 12.1 (VOTEPF and MIXNET). (6+6 points)

Consider VOTEPF and MIXNET. What is the purpose of these protocols? Answer with a complete English sentence without mathematical symbols. Also, state the information that is verified in each case. Discuss further important properties. 6+6

Exercise 12.2. (7+7 points)

Answer the following questions for the named protocol. 7+7

REENCPF What happens if a lazy prover chooses the random value $t_i = s_i$ in step 3.

VOTEPF What are similarities and differences to KNOWDLOG?

PET What are similarities and differences to EQDLOG?

MIXNET What is the purpose of the q_i?

REGISTER Why the use of nonces instead of simple random choices?

VOTE Why is this called protocol, not algorithm?

TABULATE Concerning the chronological sequence of steps 4-11, which can be run in parallel, for which can the order be reversed?

Exercise 12.3. (0+4 points)

Assume a scenario where n Tabulation tellers and m voters are involved. How often is every protocol executed (on average/at least/at most)? +4

A. Appendix

Protocol A.1. Reencryption proof (REENCPF).

Public input: A list $C = [(T_i, Y_i)]_i$ of (reencrypted) ciphertexts, a particular ciphertext $\widehat{C} = (T, Y)$, and the recipients' public key X.

Private input to the prover: An index j into the list C and the reencryption randomness t' such that $\widehat{C} = C_j + \mathrm{enc}_X(\mathcal{O}; t')$.

Output to the prover: $\mathrm{REENCPF}(j, t') = (\check{s}, \check{t})$

1. The prover performs 2–8.
2. For all indices i of C do 3–5
3. She picks random values $s_i, t_i \longleftarrow \mathbb{Z}_q$.
4. $\widetilde{T}_i = s_i(T_i - T) + t_i P$ and
5. $\widetilde{Y}_i = s_i(Y_i - Y) + t_i X$.
6. The prover computes $c \leftarrow \mathbb{Z}_q(\mathrm{hash}(\widehat{C}, C, [(\widetilde{T}_i, \widetilde{Y}_i)]_i))$.
7. The prover computes
   $\check{s}_j \leftarrow c - \sum_{i \neq j} s_i$, and for $i \neq j$ let $\check{s}_i \leftarrow s_i$,
   $\check{t}_j \leftarrow t_j - t' (\check{s}_j - s_j)$, and for $i \neq j$ let $\check{t}_i \leftarrow t_i$.
8. He sends $(\check{s}, \check{t})$. $\xrightarrow{(\check{s}, \check{t})}$
9. The verifier performs 10–15.
10. He reconstructs $\widetilde{T}$ and $\widetilde{Y}$:
11. For all indices i of C do 12–13
12. $\widetilde{T}'_i = \check{s}_i(T_i - T) + \check{t}_i P$ and
13. $\widetilde{Y}'_i = \check{s}_i(Y_i - Y) + \check{t}_i X$.
14. He computes $c' \leftarrow \mathbb{Z}_q(\mathrm{hash}(\widehat{C}, C, [(\widetilde{T}'_i, \widetilde{Y}'_i)]_i))$, and $d' \leftarrow \sum_i \check{s}_i$.
15. He verifies $c' \stackrel{?}{=} d'$.

Protocol A.2. Vote Proof (VOTEPF).

Public input: Encrypted credential $(T_1, Y_1, c, r) = \mathrm{CredEnc}(s, t, K_{TT}, rid, vid)$, encrypted choice (T_2, Y_2), the prover's public key X.

Private input to the prover: Temporary keys $t_1, t_2 \in \mathbb{Z}_q$ such that $T_i = t_i P$.

1. The prover picks $s_1, s_2 \longleftarrow \mathbb{Z}_q$.
2. The prover computes $c \leftarrow \mathbb{Z}_q(\mathrm{hash}(P, X, T_1, Y_1, T_2, Y_2, s_1 P, s_2 P))$.
3. The prover computes $r_i \leftarrow s_i - c t_i$ in $\mathbb{Z}_q$.
4. He sends (c, r_1, r_2). $\xrightarrow{(c, r_1, r_2)}$
5. The verifier checks $c \stackrel{?}{=} \mathbb{Z}_q(\mathrm{hash}(P, X, T_1, Y_1, T_2, Y_2, r_1 P + c T_1, r_2 P + c T_2))$.

Protocol A.3. Plaintext equivalence test (PET).

Public input: Two ciphertexts $C_j = (T_j, Y_j)$, encrpyted with the tabulation tellers' common public key $X_{\mathrm{TT}} = \sum_i X_i$.

Private input to tabulation teller i: The private key share x_i.
Output: PET(C_1, C_2)

1. Tabulation teller i performs 2–6.
2. Pick a randomizer $z_i \in \mathbb{Z}_q$ and compute $\widetilde{T}_i \leftarrow z_i(T_1 - T_2)$, $\widetilde{Y}_i \leftarrow z_i(Y_1 - Y_2)$.
3. Publish a commitment to $(\widetilde{T}_i, \widetilde{Y}_i)$. $\xrightarrow{\text{commit}(\widetilde{T}_i, \widetilde{Y}_i)}$
4. Wait until commitments of all tabulation tellers are available. $\longleftarrow$
5. Publish $(\widetilde{T}_i, \widetilde{Y}_i)$ and a proof of equality of discrete logarithms for $(T, Y, \widetilde{T}_i, \widetilde{Y}_i)$. $\xrightarrow{(\widetilde{T}_i, \widetilde{Y}_i, \text{EqDlogs}(\ldots))}$
6. Wait and verify all commitments and proofs. $\longleftarrow$
7. Let $\widetilde{T} \leftarrow \sum_i \widetilde{T}_i$, $\widetilde{Y} \leftarrow \sum_i \widetilde{Y}_i$.
8. All tabulation tellers jointly decrypt $(\widetilde{T}, \widetilde{Y})$: $\longrightarrow$ $\longleftarrow$

$$m' \leftarrow \text{DistDec}(\widetilde{T}, \widetilde{Y}).$$

9. If $m' = \mathcal{O}$ then Return Equal Else Return Unequal .

Algorithm A.4. Atomic mix operation (MIX).
Input: A list $C = [C_i]_i$ of ciphertexts, and a direction $d \in \{\mathsf{In}, \mathsf{Out}\}$.
Output: An anonymized reencryption $M = \text{Mix}(C)$ of C, and a list of commitments.
Private output: r, w, p.

1. Pick a permutation π of the indices of C. (Instead of picking it, you can also compute it such that the reencrypted list M is sorted.)
2. If $d = \mathsf{In}$ then $p \leftarrow \pi^{-1}$ Else $p \leftarrow \pi$.
3. Pick reencryption randomnesses $r_i \xleftarrow{} \mathbb{Z}_q^\times$ and commitment randomizers $w_i \xleftarrow{} \mathcal{R}$.
4. Let $M \leftarrow [\text{Reenc}(C_{\pi(i)}; r_i)]_i$.
5. Let $S \leftarrow [\text{Commit}(w_i, p(i))]$.
6. Return M, S .

Protocol A.5. The anonymizing mix net (MIXNET).
Public input: A list $C = [C_i]_i$ of ciphertexts.
Output: Anonymization MIXNET(C) of C.

1. Let $M_{0,2} \leftarrow C$.
2. For $i = 1 \ldots n$ do 3–6
3. Wait for $M_{i-1,2}$. $\longleftarrow$
4. Mix i computes $(M_{i,1}, S_{i,1}) \leftarrow \text{Mix}(M_{i-1,2}, \mathsf{Out})$ and publishes that. $\xrightarrow{M_{i,1}, S_{i,1}}$

5. Mix i computes $(M_{i,2}, S_{i,2}) \leftarrow \mathsf{Mix}(M_{i,1}, \mathsf{In})$ and publishes that. $\xrightarrow{M_{i,2},\, S_{i,2}}$
6. Pick a further random value $q_i \xleftarrow{\text{🎲}} \mathcal{R}$ and publish a commitment to it. $\xrightarrow{\text{Commit}(q_i)}$
7. Wait for all mixes to finish. $\longleftarrow$
8. Then each mix publishes q_i. $\xrightarrow{q_i}$
9. Wait and verify all other mixes' commitments. $\longleftarrow$
10. Let $q \leftarrow \text{hash}(q_1, \dots, q_n)$.
11. Compute the challenge $c_i \leftarrow \text{hash}(q, i)$.
12. For $i \in \{1, \dots, n\}$ in parallel do 13–20
13. Mix i publishes r_j or $r_{p(j)}$ depending on $\text{bit}_j(c_i)$, w_j and $p(j)$ from the mixing resulting in $M_{i,1+\text{bit}_j(c_i)}$ for all indices j of C. $\xrightarrow{\left[\left(\begin{cases} r_j & \text{if bit}_j(c_i) = 0 \\ r_{p(j)} & \text{if bit}_j(c_i) = 1 \end{cases}, w_j, p(j)\right)\right]_j}$
14. Now all the mixing information can be erased.
15. Wait for the other mixes' responses. $\longleftarrow$
16. Verify $\text{Commit}(w_j, p(j)) = S_{i,1+\text{bit}_j(c_i)}$.
17. If $\text{bit}_j(c_i) = 0$ then
18. Verify $\text{Reenc}_X(M_{i-1,2,p(j)}; r_j) = M_{i,1,j}$.
19. Else
20. Verify $\text{Reenc}_X(M_{i,1,j}; r_{p(j)}) = M_{i,2,p(j)}$.
21. Return $M_{n,2}$

Protocol A.6. Registration (REGISTER).

Public input: The distributed public key X_{TT} of the tabulation tellers, a public RSA key K_{RT_i} of the registration teller i. The voter's public designation key X_{vid}. The voter's public registration RSA key K_{vid}. Identifiers of election (eid), voter (vid), registration tellers (rid), and block (bid). Public credentials $S_j = \text{CredEnc}(s_j; r; X_{TT}; \text{rid}, \text{vid})$ for each registration teller $j \in \text{rid}$.

Private input to registration teller RT_i: Private credential $s_i \in \mathcal{M}$ and encryption randomness $r \in \mathbb{Z}_q^\times$.

Private input to the voter: Private registration RSA key $k_{\text{vid}}, \dots$

Output to the voter: private credentials $\text{Register}(vid, rid, sid)$

1. The voter picks a nonce N_{vid} and sends the election id eid, his id vid, and the nonce encrypted to the registration teller i. $\xrightarrow{\text{RSAenc}_{K_{\text{RT}_i}}(\text{eid}, \text{vid}, N_{\text{vid}})}$
2. The registration teller RT_i verifies that vid is a voter in block (precinct) bid in election eid, and that for each registration tellers j in

rid the public credential S_j is available and $\mathrm{CredVer}(S_j; j, \mathrm{vid})$ succeeds.

3. The registration teller picks a nonce N_R and an AES key k (of security level ℓ).
4. Send the registration teller ids rid, the nonces N_R and N_V and the chosen AES key k to the voter. $\xleftarrow{\mathrm{RSAenc}_{K_{\mathrm{vid}}}(\mathrm{rid}, N_R, N_V, k)}$
5. The voter decrypts and verifies rid and N_V, and sends the nonce N_R back to the registration teller RT_i. $\xrightarrow{N_R}$
6. The registration teller RT_i verifies N_R.
7. The registration teller picks $r' \longleftarrow \mathbb{Z}_q^\times$ and computes $w \leftarrow r' - r$ and another encryption $S'_i \leftarrow \mathrm{Enc}(s_i; r', X_{TT})$ of the private credential.
8. The registration teller sends AES encrypted the private credential share and the new randomness r' together with a designated verifier proof that S_i and S'_i encrypt the same message. $\xleftarrow{\mathrm{AESenc}_k(s_i, r', \mathrm{DVRP}(\dots), \mathrm{bid})}$
9. The voter decrypts and verifies the designated verifier proof against S_i from the bulletin board.

Algorithm A.7. Fake credentials (FACECREDENTIAL).

Input obtained from registration: Private credential shares s_i, public credential shares S_i, reencryption factors r_i, and designated verifier proofs D_i from each registration teller RT_i.

Input: Index set L of registration teller for which to fake shares. The voter's designation key pair $(X_{\mathrm{vid}}, x_{\mathrm{vid}})$.

Output: Fake private credential shares …

1. For i do 2–9
2. If $i \in L$ then
3. Pick $\widetilde{r}_i \longleftarrow \mathbb{Z}_q^\times$.
4. Pick $\widetilde{s}_i$ randomly.
5. Else
6. Let $\widetilde{r}_i \leftarrow r_i$.
7. Let $\widetilde{s}_i \leftarrow s_i$.
8. $\widetilde{S}_i \leftarrow \mathrm{enc}(\widetilde{s}_i; \widetilde{r}_i; X_{\mathrm{TT}})$.
9. Compute a non-interactive fake designated verifier proof $\widetilde{D}_i$ by Protocol 6.4
10. Return $[(\widetilde{s}_i, \widetilde{r}_i, \widetilde{D}_i)]_i$

Protocol A.8. Vote (VOTE).

Public input: The distributed public key X_{TT} of the tabulation tellers. Well-known choice ciphertext list C.

Private input: The voter's choice t and his credentials s.

Output to the ballot box: Vote(t, s)

1. The voter picks a randomness r_s and encrypts his credentials $S \leftarrow \mathrm{enc}(s; r_s; X_{TT})$ for the tabulation tellers.
2. He picks a randomness r_v and reencrypts his choise C_t: $V \leftarrow \mathrm{reenc}(C_t; r_v)$.
3. He prepares a vote proof P_w of correct voting by Protocol A.2 with inputs S, V, r_s, r_v, and further context.
4. He prepares a REENCPF P_k that V is a reencryption of one of the cipher texts C by Protocol A.1.
5. Let vote $\leftarrow (S, V, P_w, P_k)$ and send this to the ballot box. $\xrightarrow{\text{vote}}$

Protocol A.9. Tabulate (TABULATE).

Principals: Tabulation tellers $\mathrm{TT}_1, \ldots, \mathrm{TT}_n$, broadcast bulletin board ABB, ballot boxes $\mathrm{VBB}_1, \ldots, \mathrm{VBB}_m$, supervisor Sup.

Public input: X_{TT}, contents of bulletin board ABB.

Private input to TT_i: Private key share x_i of X_{TT}.

Output: Election tally for one block.

1. Each ballot box VBB_i posts commitments on the list of all votes on the tabulation board ABB. $\xrightarrow{\text{Commit(received votes)}}$
2. The supervisor signs the list of all received VBB commitments. $\xrightarrow{\mathrm{sign}_{\mathrm{Sup}}\text{(ABB so far)}}$
3. The tabulation tellers TT_i jointly execute 4–11.
4. **Retrieve votes**. Retrieve all votes from all endorsed ballot boxes VBB_i. Verify the commitments. Let $A \leftarrow$ list of votes. $\xleftarrow{\text{votes}}$ $\xrightarrow{A}$
5. **Check proofs**. Verify all VotePfs and ReencPfs in retrieved votes. Eliminate any votes with an invalid proof. Let B be the list of remaining votes. $\xrightarrow{B}$
6. **Duplicate elimination**. Run the plaintext equivalence test $\mathrm{PET}(S'_i, S'_j)$ for all pairs (i, j), where S'_x is the encrypted credential in vote B_x. Eliminate equivalent votes according to a revoting policy. Let C be the list of remaining votes. $\xrightarrow{C}$
7. **Mix votes**. $D \leftarrow \mathrm{MixNet}(C)$. $\xrightarrow{D}$
8. **Mix credentials**. Let E be the list of all initially created encrypted credentials. Anonymize it: $F \leftarrow \mathrm{MixNet}(E)$. $\xleftarrow{E}$ $\xrightarrow{F}$

9.	**Invalid elimination**. Run the plaintext equivalence test $\mathrm{PET}(S_i, S'_j)$ for all pairs (i, j) where $S_i = F_i$, $S_j = D_j$. Eliminate votes from D for which there is no equivalent credential found in F. Let G be the list of remaining votes.	$\xrightarrow{\quad G \quad}$
10.	**Decrypt**. Let $H_i \leftarrow \mathrm{DistDec}(G_i)$ for all i.	$\xrightarrow{\quad H \quad}$
11.	**Tally**. Compute the tally of H according to an election method specified by the supervisor.	$\xrightarrow{\quad \text{tally} \quad}$
12.	Finally, the supervisor endorses the tally (if …).	$\xrightarrow{\text{Sign ABB so far.}}$