Receipt-freeness: formal definition and fault attacks[⋆] (Extended Abstract)

Stéphanie Delaune[1,2], Steve Kremer[2], and Mark Ryan[3]

[1] France Télécom R& D
[2] Laboratoire Spécification et Vérification
ENS Cachan, France
{delaune,kremer}@lsv.ens-cachan.fr
[3] School of Computer Science
University of Birmingham, UK
M.D.Ryan@cs.bham.ac.uk

1 Introduction

Intuitively, an election protocol is *receipt-free* if a voter A cannot prove to a potential coercer C that she voted in a particular way. We assume that A wishes to cooperate with C; receipt-freeness guarantees that such cooperation will not be worthwhile, because it will be impossible for C to obtain proof about how A voted. Receipt-freeness is a similar property to *privacy*, which asserts that an intruder cannot gain information about how A voted. Receipt-freeness is like privacy with the additional assumption that A cooperates with the intruder C, for example by sharing her secret key and other secret information generated during the protocol. Thus, receipt-freeness implies privacy.

Receipt-freeness is also related to the property of *individual verifiability*, which guarantees that a voter A can obtain proof that her vote was counted in the final tally of the election. In order to satisfy both receipt-freeness and individual verifiability, it is necessary that the verification constitute proof for A, but that it is insufficient proof for the coercer C. Therefore, some bounds on the communication between A and C are necessary. If C has verifiable access to all of the secrets held by A, then intuitively the verification that A receives will constitute proof for C as well.

Such communication bounds between A and C may in practice be provided by the *voting booth* which A enters in order to cast her vote. In this private booth, C cannot see what A actually does. A wishes to cooperate with C, and may have shared secrets with C before entering the booth. We may even assume that A and C can communicate while A is inside the booth (for example, by mobile phone), so that A can inform C about the messages she receives from the election administrator and take instructions from C about what messages to send back. But C may not be able to verify the veracity of all the messages that A claims to receive from the administrator (for example, the values of nonces). Similarly, C may not be able to verify that A sends the agreed messages back to the administrator.

C's ability to verify these things will depend on the details of the protocol: what information is revealed at what time, which channels are secret and which public, etc. We will assume that C has the capabilities (such as control of public channels) of a Dolev-Yao attacker.

In this paper we propose a formalisation of receipt-freeness in terms of observational equivalence in the applied pi calculus. The formalisation of receipt-freeness is non trivial and gives interesting insights. Among others, our formalisation highlights a new kind of *fault attacks*. The idea of a fault attack is to let the coercer test the *loyalty* of a coerced voter. The coercer gives the voter messages that she should use during the protocol. The coercer can send a *garbage* message to the voter. If the voter is unable to decide whether the message is garbage or not (for instance when a ciphertext is sent), the attacker may distinguish a voter who follows the coercer's instructions from a voter who is trying to cheat the coercer, as the protocol would block on the incorrect message. In practice, a coercer can use this technique to check whether a coerced voter is behaving as expected. We illustrate the attack on a simple protocol inspired by a voting protocol proposed by Lee et al. [5], which we refer to as the Lee et al. 03 protocol. Independently, Juels et al. [3] proposed formal definitions for a strong version of receipt-freeness, they call coercion resistance. Their and our definition seem to have similarities and consider stronger attack models than the usual informal definitions. However, their definitions are given in a computational setting, rather than the symbolic model considered here. We do not yet know how these definitions compare.

[⋆] S. Delaune and S. Kremer have been partly supported by the RNTL project PROUVÉ 03V360 and the ACI-SI Rossignol.

2 Formalisation of receipt-freeness

We will use the applied pi calculus [1] as our formalism. The applied pi calculus is an "impure", but more convenient, version of the pi calculus. A convenient feature of the applied pi calculus is the fact that it is parameterized by a signature and an arbitrary equational theory. The equational theory allows one to model cryptographic primitives in a Dolev-Yao style, but is flexible enough to model some less standard cryptographic primitives, such as re-encryption and non-interactive designated verifier proofs, as shown in the section 4. Moreover, observational equivalence, which is a powerful tool for specifying security properties, is defined for an arbitrary equational theory. The applied pi calculus has been useful to analyze many cryptographic protocols, in particular an electronic voting protocol [4].

Similarly to privacy, receipt-freeness may be formalised as an observational equivalence. In [4], we modelled an election scheme as an applied pi-calculus process P, and formalised privacy as the observational equivalence between P and another version of P in which two voters A and B have swapped their votes. Let $A(v)$ represent the voting process with identity A (i.e. A's voting process) and vote v. Privacy is expressed as:

$$A(v_A) \mid B(v_B) \mid S \approx A(v_B) \mid B(v_A) \mid S$$

Here, S is the rest of the system (the other voters and the administrators, collectors, etc.) The intuition is that if an intruder cannot detect if arbitrary honest voters A and B swap their votes, then in general he cannot know anything about how A (or B) voted. Note that this definition is robust even in situations where the result of the election is such that the votes of A and B are necessarily revealed: for example, if the vote is unanimous, or if all other voters reveal how they voted and thus allow the votes of A and B to be deduced.

Formalising receipt-freeness is also done by observational equivalence, but we need to model the fact that A is willing to cooperate with the coercer C. We assume that the coercer is in fact the intruder who, as usual in the Dolev-Yao model, has access to the public channels. To model A's communication with C, we consider that A executes a voting process which has been modified.

Definition 1. *Let P be a process and c_1, c_2 be channels. The process P^{c_1,c_2} is a process like P but which copies all messages it receives on c_1 to c_2, and accepts inputs on c_2 for all messages it sends on c_1. Specifically,*

- *Every* $\mathsf{in}(c_1, y)$ *in P is replaced by* $\mathsf{in}(c_1, y); \mathsf{out}(c_2, y)$.
- *Every* $\mathsf{out}(c_1, m)$ *in P is replaced by* $\mathsf{in}(c_2, x); \mathsf{out}(c_1, x)$, *where x is a variable not occurring in P;*

Moreover, we prefix P by copying every restricted name used in the process, except channel names (we suppose that restricted channel names are never used otherwise than as a channel name), on c_2 (note that using scope extrusion all the restricted names can be generated at the beginning of the process).

Intuitively, the protocol is receipt-free if, for all voters A, the process in which A votes according to the intruder's wishes is indistinguishable from the one in which she votes something else, even if all secrets are (apparently) shared with the intruder. As in the case of privacy, we express this as an observational equivalence to a process in which A swaps her vote with B, in order to avoid the case in which the intruder can distinguish the situations merely by counting the votes at the end. Suppose that the voter communicates with the election administrator along channel cha, and suppose chc is the channel on which the coercer C communicates with corrupt voters.

Definition 2 (Receipt freeness). *A voting protocol in which voters $A, B, \ldots$ communicate with the voting administrator along a channel cha is* receipt free *if: there exists a process A' such that if A' votes then it votes v_A, and for all coercers C, there exists a vote v, such that*

$$\nu\, chc\ (C \mid A^{cha,chc} \mid B(v_A) \mid S) \quad \approx \quad \nu\, chc\ (C \mid A' \mid B(v) \mid S)$$

where chc is the channel used for the communication between A and C, $B(v)$ is the process of an honest voter voting v, and S is the remainder of the system.

A' is a process in which voter A votes v_A but communicates with the coercer C in order to feign cooperation with him. Thus, the equivalence says that the coercer cannot tell the difference between a situation in which A genuinely cooperates with him in order to cast his chosen vote, and one in which she pretends to cooperate but actually casts the vote v_A, provided there is some counterbalancing voter B that votes the other way around.

To better understand the definition, consider the following cases:

– The process $C \mid A^{cha,chc}$ votes the coercer's vote v_C. Then we pick $v = v_C$ and A' should vote v_A in order to preserve the equivalence.
– The process $C \mid A^{cha,chc}$ votes v_A. Then we pick $v = v_A$ and A' should again vote v_A in order to preserve the equivalence.
– The process $C \mid A^{cha,chc}$ does not vote, perhaps because C sends some incoherent messages to A and they do not constitute a valid vote. Then we pick $v = v_A$ and A' should not vote, in order to preserve the equivalence.

3 Fault attack

Looking closely at the definition of receipt freeness, one may notice that the intruder could have an unexpected strategy to distinguish the left-hand process from the right-hand process. In protocols in which the voter is expected to submit certain messages to the election administrators, the arrangement between the voter and the coercer may involve the coercer preparing those messages for the voter. In this case, the coercer may submit messages to A with the pure intention to block the process on the left-hand side of the observational equivalence. If A' is unable to detect that the message is incoherent, the process will not block on the right-hand side, thus yielding an observable difference.

In a real-life scenario the intruder may in that way test the loyalty of A in a probabilistic way. For example, in one out of every one hundred voters, the coercer might submit such a garbage message to the voter. If the voter is not genuinely cooperating with him by submitting the garbage message, and instead votes his own vote successfully, the attacker can perceive a difference.

We argue that this attack, if successful, is an attack against receipt-freeness, since it means that the voter knows that the coercer has the ability to detect whether the voter is cooperating with him or not. Launching the attack costs the coercer that particular vote, but it is a means of applying pressure on the voter to cooperate.

Our definition of receipt-freeness is correct with respect to the attack. That is, a protocol which is vulnerable to the attack is not receipt-free according to our definition. A protocol is receipt-free if A' can be chosen to mimick $A^{cha,chc}$. Thus, if A' can detect that the message from the coercer is incoherent, she can act in order to block the protocol, preserving the equivalence.

As an aside, we note that in protocols such as [2] the voter is not required to submit complicated messages to the administrators. They may still be vulnerable to the attack, however. The coercer may be able to provide a malformed ballot paper to the voter.

4 The Lee et al. 03 protocol

In this section we introduce a simple protocol inspired by the protocol proposed by Lee et al. [5]. We simplified the protocol in order to concentrate on the aspects that are important with respect to receipt-freeness. In particular we do not consider distributed authorities. The protocol relies on two less usual cryptographic primitives: re-encryption and designated verifier proofs (DVP) of re-encryption.

A re-encryption of a ciphertext (obtained using a randomized encryption scheme) changes the random coins, without changing or revealing the plaintext. In the ElGamal scheme for instance, if (x, y) is the ciphertext, this is simply done by computing (xg^r, yh^r), where r is a random number, and g and h are the subgroup generator and the public key respectively. Note that neither the creator of the original ciphertext nor the person re-encrypting knows the random coins used in the re-encrypted ciphertext, for they are a function of the coins chosen by both parties. In particular, a voter cannot reveal the coins to a potential coercer who could use this information to verify the value of the vote, by ciphering his expected vote with these coins.

A DVP of the re-encryption proves that the two ciphertexts contain indeed the same plaintext. However, a designated verifier proof only convinces one intended person, e.g. the voter, that the re-encrypted ciphertext contains the original plaintext. In particular this proof cannot be used to convince the coercer. Technically, this is achieved by giving the designated verifier the ability to simulate the transcripts of the proof.

Our simplified protocol can be described in three phases.

1. The voter encrypts his vote with the collector's public key, signs the encrypted vote and sends it to an administrator. The administrator checks whether the voter is a legitimate voter and has not voted yet. Then the administrator

re-encrypts the given ciphertext, signs it and sends it back to the voter. The administrator also provides a (non-interactive) DVP that the two ciphertexts contain indeed the same plaintext. In practice, this first stage of the protocol can be done using a voting booth where eligibility of the voter is tested at the entrance of the booth. The booth contains a tamper-proof device which performs re-encryptions, signatures and dvp proofs.

2. The voter sends (via an anonymous channel) the re-encrypted vote, which has been signed by the administrator to the public board.
3. The collector checks the administrator's signature on each of the votes and, if valid, decrypts the votes and publishes the final results.

One of the main advantages of this protocol is that it is *vote and go*.

Our contention is that this protocol (and hence the Lee et al. 03 protocol) is vulnerable to the fault attack described in the previous section. Let us suppose that the coercer C's arrangement with the voter A is that he gives her the vote encrypted with the collector's public key, to be used in the first phase. A has no means of verifying whether this message is a correct encryption of a valid vote, or not. If she substitutes her own vote, and C's message turns out to be garbage, then this can be detected.

The protocol can be fixed by coding things so that a simple syntactic check will determine whether a particular text is a valid encrypted vote or not. For example, one can use an encryption algorithm with the property that every number is a valid encryption, and every decryption is acceptable as a valid vote; for instance an even number is interpreted as 'yes', while an odd number means 'no'.

Verifying that the fixed protocol satisfies our definition of receipt freeness is currently work in progress. The cryptographic primitives used by the protocol can be modeled by the following equational theory.

```
decrypt(pencrypt(m,pk(sk),r),sk) = m
rencrypt(pencrypt(m,pk(sk),r1),r2) = pencrypt(m,pk(sk),f(r1,r2))
checksign(sign(m,sk),pk(sk)) = m
checkdvp(dvp(x,rencrypt(x,r),pkv),x,rencrypt(x,r),pkv) = ok
checkdvp(dvp(x, y, skv), x, y, pk(skv)) = ok
```

The last equation models that the designated verifier can use his secret key to produce an indistinguishable DVP (the function pk returns the public key corresponding to the given secret key). The other equations are rather straightforward. The voters and the regulator process are modelled as applied pi calculus processes in a similar way as in [4]. The voting booth is modelled by a private channel with the regulator.

5 Conclusions

We have formalised a rather strong notion of receipt-freeness: it expresses that a voter A can appear to satisfy the demands of a coercer not only in respect of how she votes, but also in respect of whether she votes. We have shown informally that the Lee et al. 03 protocol [5] fails to satisfy our definition. We have proposed a simple way to fix it. Verifying that the corrected protocol is indeed correct is currently work in progress.

References

1. M. Abadi and C. Fournet. Mobile values, new names, and secure communication. In H. R. Nielson, editor, *Proceedings of the 28th ACM Symposium on Principles of Programming Languages*, pages 104–115, London, UK, Jan. 2001. ACM.
2. D. Chaum, P. Y. A. Ryan, and S. Schneider. A practical, voter-verifiable election scheme. In *Proceedings of the 10th European Symposium On Research In Computer Security – ESORICS 2005*, Lecture Notes in Computer Science, Milan, Italy, Sept. 2005. Springer. To appear.
3. A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant electronic elections. In *Workshop on Privacy in the Electronic Society – WPES 2005*, Alexandria, VA, USA, Nov. 2005. ACM. To appear.
4. S. Kremer and M. D. Ryan. Analysis of an electronic voting protocol in the applied pi-calculus. In M. Sagiv, editor, *Programming Languages and Systems — Proceedings of the 14th European Symposium on Programming (ESOP'05)*, volume 3444 of *Lecture Notes in Computer Science*, pages 186–200, Edinburgh, U.K., Apr. 2005. Springer.
5. B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang, and S. Yoo. Providing receipt-freeness in mixnet-based voting protocols. In J. I. Lim and D. H. Lee, editors, *Information Security and Cryptology – ICISC 2003*, volume 2971 of *Lecture Notes in Computer Science*, pages 245–258, Seoul, Korea, Nov. 2004. Springer.