Security protocols for Secret Santa

Sjouke Mauw, Saša Radomirović, and Peter Ryan

University of Luxembourg
Faculté des Sciences, de la Technologie et de la Communication
6, rue Richard Coudenhove-Kalergi
L-1359, Luxembourg

1 Introduction

The original motivation for the current report lies in a number of questions concerning the current state of the art in security protocol design and analysis. Why is it so hard to develop a complete set of security requirements for a particular problem? After more than twenty years of research on, e.g., e-voting, we have reached a state in which still confusion exists about which properties an e-voting protocol should satisfy. What did we learn from this research experience on e-voting? Can we apply the knowledge accumulated in this domain to similar domains and problems? In short, is there a methodology underlying the process of understanding a security protocol domain?

One could argue that such a methodology would not exist, because security requirements analysis is an instantiation of the general problem of requirements analysis and thus suffers from the same problems, such as ambiguity, incompleteness and implicit assumptions. Nevertheless, security protocol history shows that there are several patterns in the evolution of security requirements that can be identified and from which we can learn.

As a first example we look at the development of the notion of privacy in e-voting. While privacy initially amounted to vote secrecy, it was later on refined into receipt-freeness, stating that a voter cannot prove whom he voted for. Thus, privacy must be guaranteed, even if there is a reason for the voter to cooperate with the adversary. The notion of receipt-freeness, in turn, was refined into coercion resistance, capturing an even stronger form of cooperation with the adversary. As a consequence of the shift from honest participants to participants showing several forms of dishonest behaviour, we see a shift of the security requirements from basic privacy to *enforced privacy*. This drift towards insider attacks was also at the basis of the well-known developments concerning the Needham-Schroeder protocol. Likewise, we may even consider non-repudiation as a form of authentication with a partially malicious agent.

A second class of examples comes from the discrepancy between the real-world problem and the idealized abstraction of that problem. In the real world, there are side-channels, a voter can be physically threatened and random-number generators are predictable, while in the ideal world we consider such things as noise when trying to solve the abstract problem.

Looking at the history of security protocols, we can make a distinction between three approaches towards requirements analysis. The first approach consists of postulating a set of requirements, e.g., based on the similarity of the problem with well-studied existing problems. The second approach is driven by an analysis of possible threats. The set of all threats defines the unwanted behaviour and thus specifies the security requirements. Following Roscoe [4], these approaches can be considered as *extensional*, i.e., they refer to the externally observable effect. Achieving a complete specification in this way is rather hard. The third approach starts by designing an ideal solution that explains how to securely solve the problem under idealized assumptions. For instance, one may assume a Trusted Third Party, resilient, secure and anonymizing communication channels, trusted communication partners, etc. Such an ideal solution may be considered as a specification of what the final solution should achieve. Again using Roscoe's terminology, this is a more *intensional* approach. One of the problems with this approach is that it may suffer from over-specification.

The current state-of-the-art in the field of security protocol design is far from providing clear and explicit answers to the questions raised above. Therefore, we propose to study new, simple security protocol domains as to identify common patterns in the domain analysis and apply them to new situations.

As a first exercise, we propose to study the Secret Santa problem. Secret Santa is known in many western countries under different names, such as Chris Kindle, sinterklaassurprises and julklapp. The informal description is as follows. Members of a group are randomly assigned other members to whom they anonymously give a gift. In order to prevent cheating, simple protocols are used, such as drawing strips of paper (with a name) from a hat. This is a centralized, probabilistic solution. A non-probabilistic service is offered through various web sites, but these require a TTP. An interesting question is whether there are distributed, non-probabilistic solutions without TTP.

2 Related work

The Secret Santa protocol problem was first described by Gerard Tel in his text book [6]. He describes a probabilistic, decentralized solution. One of his students studied the problem in more detail and developed implementations of different solutions [7]. Liberti and Raimondi studied the problem from a different point of view. They are not proposing a protocol but an algorithm to determine if a solution exists under certain constraints, including an anonymity requirement [3].

3 The Secret Santa problem

Basically, the Secret Santa problem boils down to anonymously establishing a random derangement of the participants. In addition, we require that cheating participants, i.e. those that don't follow this derangement when buying a present, can be identified. Formally,

Definition 1. *Let P be a set of size $N > 2$, containing participants. A* Secret Santa protocol *is a (distributed) protocol that establishes a function $f : P \to P$ such that*

1. *(bijective) f is a bijection;*
2. *(irreflexive) $\forall_{p \in P} f(p) \neq p$;*
3. *(random) f is random, i.e. none of the participants can influence the choice of f in a non-random manner;*
4. *(anonymous) A group of conspiring participants $Q \subseteq P$ will not learn more about function f than can be deduced from $\{f(q) \mid q \in Q\}$.*
5. *(verifiable) If there is a participant $p' \in P$ who has not received a present after the exchange, we can identify the cheater, i.e. the participant $p \in P$ for which $f(p) = p'$.*

These requirements seem to easily follow from the (short and informal) problem description. But there were several choices made that led up to the formulation presented above. For instance, we have taken the stance that in a real-world Secret Santa protocol, there is nothing that can be done to prevent a group of conspirators from exchanging their assignments. Thus the anonymity and verifiability conditions express what we think is the best we can achieve.

Alternatively, we could have argued that unless there is a necessary trade-off present in the requirements, we should always strive for the strongest set of requirements. In this setting, we might state the verifiability requirement as follows.

5'. *(verifiable) After exchanging the presents it is verifiable that every cheating participant $p \in P$, i.e. one that has not given a present to $f(p)$, can be identified.*

Both of these formulations are based on catching cheaters. As a further alternative to the verifiability requirement, we could consider forcing participants to prove compliance.

5". *(verifiable) After successfully executing the protocol, every participant $p \in P$ can prove that the image of p is indeed $f(p)$.*

Aside from the problem of finding the "right" formulation for these requirements, it is not even evident that the list is complete. Indeed, why are notions similar to receipt-freeness, coercion resistance, abuse-freeness and universal verifiability left out? Do we need to require a fairness property stating that whenever a participant gives a present, he will eventually receive a present?

Since correctness of a security protocol is relative to a particular adversary model, a specfication of the adversary model must also be provided. We assume a Dolev-Yao adversary model with conspiring participants. Alternatively, we could have chosen honest but curious participants. This would mean that they strictly follow the protocol, but nevertheless try to learn other participant's secrets, e.g., by performing some extra local calculations. The discrepancy between the real world and the idealized world shows because we ignore covert channels that may appear when people bring or open presents.

4 Solutions

In this section we sketch several solutions for the Secret Santa problem. The traditional solution is to draw strips of paper from a hat. This is a probabilistic protocol, since a participant may draw his own name. Further, the protocol is centralized and it does not require a trusted third party.

4.1 Trusted Third Party

The obvious solution is to use a trusted third party which will generate a random derangement f and securely communicate $f(p)$ to participant p. The trusted third party will then receive from p the present for $f(p)$ and distribute the presents to the participants. The following algorithm will (deterministically) generate a random derangement.

$D := P$;
$R := P$;
while $D \neq \emptyset$
do $p :=$ **any**(D);
  if $|D| = 2 \ \wedge \ |R \cap D| = 1 \ \wedge \ p \notin R \cap D$
    then $f(p) :=$ **random**$(D - \{p\})$;
    else $f(p) :=$ **random**$(R - \{p\})$;
  fi;
  $D := D - \{p\}$;
  $R := R - \{f(p)\}$;
od;

This solution satisfies requirements (1) through (5), but neither (5′) nor (5″). For (5′) the trusted third party must keep the participants in isolation for the duration of the protocol. For (5″) each participant p would need to be given a receipt signed by the trusted third party.

4.2 Decentralized solutions

We give brief descriptions of a couple of decentralized solutions heuristically satisfying the requirements of Definition 1.

We denote by $\{m, r\}_k$ an ElGamal encryption of a plaintext m with random number r and key k. Thus $k = (G, q, g, y)$, where $G = \langle g \rangle$ is a cyclic group of order q in which the Decision Diffie-Hellman problem is intractable, and $y = g^s$, s secret. The notation $\{m, r\}_k$ then corresponds to the pair $(g^r, y^r m)$. If we want to be less explicit with respect to the random factor introduced or the public key, we will simply leave them out, writing $\{m\}$ instead of $\{m, r\}_k$.

ElGamal encryption has the following homomorphic properties:

$$\{m_1, r_1\} \cdot \{m_2, r_2\} = \{m_1 m_2, r_1 + r_2\}$$

and

$$\{m_1, r_1\}^n = \{m_1^n, r_1 \cdot n\}.$$

The decentralized secret santa protocol now runs as follows.

1. Setup.
   The participants generate a public key $pk = (G, q, g, y)$ for the ElGamal cryptosystem such that only large enough coalitions of participants may recover the corresponding private key. All encryptions are performed using pk. The participants are labeled 1 through N and assigned the elements g^1 through $g^N \in G$, respectively. The participants publicly form encryptions of each of the elements in the set $\{g^1, \ldots, g^N\}$. Thus the vector $\langle\{g^1\}, \{g^2\}, \ldots, \{g^N\}\rangle$ is obtained.
2. Shuffle.
   Each participant gets to do a re-encryption shuffle of the vector. Thus they end up with

   $$\langle\{g^{\pi(1)}\}, \{g^{\pi(2)}\}, \ldots, \{g^{\pi(N)}\}\rangle$$

   for some permutation π of the set $\{1, \ldots, N\}$.
   The participants can be kept honest during this step using Furukawa and Sako's scheme for proving correctness of a shuffle [1].
3. Derangement test.
   Form the vector

   $$\langle(\{g^{\pi(1)}\}/\{g^1\})^{r_1}, \ldots, (\{g^{\pi(N)}\}/\{g^N\})^{r_N}\rangle,$$

   where r_1,...,r_N are some random values all participants contribute to. One entry will be $\{1\}$ if and only if π is not a derangement. If π is not a derangement, go to step 2. Since the number of derangements of an n-element set is equal to the nearest integer to $n!/e$ [5], only a few repetitions of steps 2 and 3 are expected.
4. Commitment.
   Towards requirement 5 of Definition 1, each participant i chooses a random number ρ_i and submits its encryption $\{\rho_i\}$ as well as the encryption of its hash $\{H(\rho_i)\}$ as a commitment. (H denotes a publicly known cryptographic hash function.) Then the vector $v = \langle\{g^{\pi(1)}\rho_1 H(\rho_1)\}, \ldots, \{g^{\pi(N)}\rho_N H(\rho_N)\}\rangle$ is formed and noted for the verification step. The purpose of the hash of ρ_i is to prevent cheating by modifying ρ_i later.
5. Revealing $\pi(i)$ to participant i only.
   Each participant i blinds $\{g^{\pi(i)}\}$. Then all terms are decrypted and each participant i recovers $\pi(i)$.
   Note that recovering $\pi(i)$ from $g^{\pi(i)}$ will not be a problem, because all $\pi(i)$ will be numbers from 1 to N, where N is the number of participants in the scheme.
6. Presents.
   Each participant i attaches ρ_i imprinted on a label to the present for $\pi(i)$ to prove compliance with his assignment $i \to \pi(i)$.

7. Verification.
   The participants form the vector $\langle\{g^{\pi(1)}\rho_1 H(\rho_1)\},\ldots,\{g^{\pi(N)}\rho_N H(\rho_N)\}\rangle$ and create a reencrypted permutation of its entries. Call this reencrypted, permuted vector u.
   The participants check whether for each of the entries v_i of the vector v there is one entry u_j in u such that $u_j/v_i = \{1\}$.
   That is, they compute for each participant i the vector $\langle u_1/v_i, u_2/v_i, \ldots, u_N/v_i\rangle$ and check whether one of the entries is $\{1\}$. If none of the entries is $\{1\}$, the participant must be a cheater.

A couple of variant solutions. If we permit ourselves the use of a *fully homomorphic* crypto algorithm, such as that presented in [2], we can make the above protocol more efficient by replacing the N plaintext equivalence tests (PET) above by a simple PET. As before, we compute the $\{\pi(i) - i\}$ using the additive homomorphism. Now we exploit the multiplicative homomorphism to compute the encryption of the product of the $(\pi(i) - i)$ terms:

$$\lambda := \prod_1^N \pi(i) - i$$

If λ is PET equivalent to $\{0\}$ then we know that there is at least one fixed point and so the permutation is not a derangement. If λ is not an encryption of 0, then we know that the permutation is a derangement but nothing further.

If we want to avoid the N PET tests but prefer not to use a fully homomorphic encryption scheme, we can use another approach that exploits verifiable parallel permutations. Suppose that our encryption algorithm satisfies only a multiplicative homomorphism, ElGamal for example.

We start by constructing publicly a vector w that is simply the encryptions of the first N prime numbers in order, denoted p_i:

$$\langle\{p_1\},\{p_2\},\ldots,\{p_N\}\rangle$$

As before, we generate a secret permutation of w in a distributed fashion with each participant contributing a shuffle. Call the vector representing this shuffle w'. Now a vector z is created that is the encryption of the values 1 through N in order:

We now parallel shuffle these two vectors to get the two shuffled vectors w' and z'. We now form the term:

$$\lambda := \prod_1^N (w'_i/w_i)^i$$

If, when we decrypt λ we find that its factorisation includes at least one occurrence of each of the first N primes then we have a derangement. Given that we applied identical shuffles to w and z we know that if the shuffle applied to w is a derangement then so is that applied to z.

Where w_i denotes the ith component of the vector w.

4.3 Conclusions

The purpose of this paper was to stress the problem of requirements analysis for security protocols. In order to make a step towards a methodology for the understanding of security protocol problems, we propose to conduct a series of case studies on new problems. We used the Secret Santa problem as an exercise and discussed requirements and some possible solutions. Although we think that we have identified the most interesting requirements they are not arguably complete. As indicated, there are several ways in which the set of requirements can be extended and there are different variations of the given requirements. It is also interesting to formalize the requirements and the proposed solutions as a first step towards formal verification. An interesting question is also to design efficient non-probabilistic decentralized solutions. By studying the relation between the TTP-based ideal solution and the decentralised solutions one could achieve a better understanding of the different requirements analysis approaches that we identified.

An interesting approach for identifying new case studies could be found in the distributed algorithms literature, which is rich of problems that in addition to the functional requirements often have a security dimension as well.

References

1. Jun Furukawa and Kazue Sako. An efficient scheme for proving a shuffle. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 368–387. Springer, 2001.
2. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC '09: Proceedings of the 41st annual ACM symposium on Theory of computing*, pages 169–178, New York, NY, USA, 2009. ACM.
3. Leo Liberti and Franco Raimondi. The Secret Santa problem. In R. Fleischer and J. Xu, editors, *AAIM08 Proceedings*, LNCS 5034, pages 271–279. Springer-Verlag, 2008.
4. A.W. Roscoe. Intensional Specifications of Security Protocols. In *Proc. CSFW '96*, pages 28–38. IEEE, 1996.
5. R. P. Stanley. *Enumerative combinatorics, Volume 1.* Wadsworth Publ. Co., Belmont, CA, USA, 1986.
6. Gerard Tel. *Cryptografie – Beveiliging van de digitale maatschappij.* Addison Wesley, 2002.
7. J. Verelst. Secure computing and distributed solutions to the Secret Santa problem. Master's thesis, Computer Science Dept., University of Utrecht, 2003.