Romantic Cryptography

Frank Stajano[1 2] and William Harris[1]

[1] University of Cambridge Computer Laboratory,
New Museums Site, Cambridge CB2 3QG, UK
http://www.cl.cam.ac.uk/~fms27/, http://will.phase.net/
[2] AT&T Laboratories Cambridge,
24a Trumpington Street, Cambridge CB2 1QA, UK
http://www.uk.research.att.com/~fms/

Abstract. We show how Alice and Bob can establish whether they love each other, but without the embarrassement of revealing that they do if the other party does not share their feelings.
This is a "secure multiparty computation" of the AND function, where the participants cooperate in producing the result of the AND, but without learning the input bit contributed by the other party unless the result implies it.

1 Scenario

Imagine two people A and B[1] coming into social contact. A fancies B, but is too shy to say so. A would like to confess this love to B, but only after being certain that it is reciprocated. On the other hand we hypothesize that B would be equally reluctant to make the first move, for exactly the same reasons.

The resulting deadlock filled our chivalrous hearts with sorrow for A and B, so we set out in search of a solution.

2 Mechanical matchmaking

Many novel cryptographic constructions are too abstract to grasp at first, and are best explained with analogies to tangible artifacts. This time, for once, we proceeded in reverse: to solve the above problem we originally came up with a physical mechanism that we later hoped to turn into mathematics.

Imagine a set of plastic spheres, similar in appearance to golf balls. They all look identical, but some of them have a metallic core that makes them more than twice as heavy as the others (figure 1). All the heavy spheres have the same weight of h, and all the light ones have the same weight of l. As we said, $2l < h$.

[1] In the interest of avoiding discussions on whether it is boys or girls who are most shy, we shall leave it to the reader to decide whether these initials stand for Alice and Bob or for Arthur and Belinda — or even Alice/Belinda or Arthur/Bob if that has greater appeal.

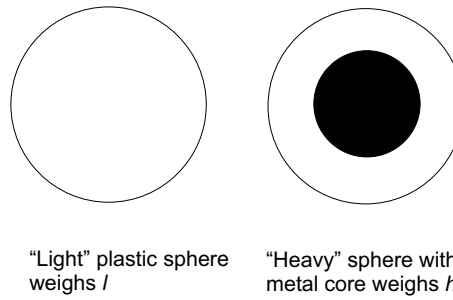


Fig. 1. A light and a heavy sphere, externally identical.

Both A and B are each given a pair of spheres — a light one and a heavy one. We use a simple apparatus consisting of a two-pan balance. In the first (left) pan we place one heavy and two light spheres. This makes it go down.

Now A and B each choose one of their spheres — the heavy one to signify that they fancy the other party, or the light one to mean that they don't — and place it in the second pan, one after the other.

When the first party (no matter who it is) drops the sphere in the second pan, nothing happens: the first pan, containing two light and one heavy spheres, is the heaviest regardless of the type of sphere dropped in the second pan. So the second party learns nothing from this step about the preference of the first party. When the second party also drops his or her sphere in the second pan,

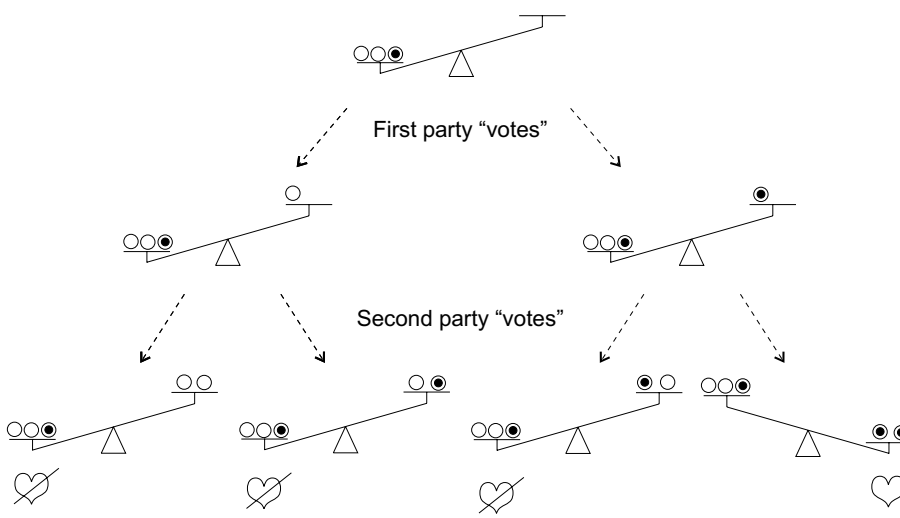


Fig. 2. The tree of choices for the mechanical matchmaker.

the pan will sink if they both fancy each other, and stay up otherwise. However, if it stays up, neither party will know whether the other fancies them or not.

Let us follow this through (figure 2). If they both voted with a heavy sphere, the first and second pan will respectively hold a weight of $2l + h$ and $2h$. Since $2l + h < 2h$, the second pan goes down and both participants get to know about their common love. This is the happy ending where no more secrecy is needed. Any other possible combination in the second pan, i.e. either $l + h$ or $2l$, is lighter than $2l + h$, so the second pan stays up all the way through, no matter what the two spheres are. Since the balance does not move at all for the whole duration of the protocol, neither party can infer anything about the other's choice.

We hasten to point out that we are using an ideal balance whose arm does not oscillate as people slap weights on the pans. The obvious engineering fixes (e.g. a mechanical lock to the arm, to be released only on completion of the protocol) can be applied to idealize a real-world balance.

3 Moving to bits

We then tried to translate this construction into bits. It is of course trivial to substitute the balance with a computer, to which A and B send their yes or no. But whoever controls the computer gets to know the choice of both parties. This is equivalent to A and B giving their spheres to a passer-by who is then supposed to tell them whether or not both spheres were heavy. With A and B being shy to the point of paranoia, this disclosure is not acceptable unless the passer-by is immediately murdered after the operation. However such a practice could never succeed on a large scale, because passers-by would soon become wary of weighing spheres for strangers.

So we seek a protocol that only involves A and B; or, alternatively, a three-principal protocol in which the third party does not learn anything about the individual preferences of A and B beyond what it publicly reveals to both of them on completion.

We originally thought that this second case could be implemented by having A and B send an obfuscated version of their choice to some server S. The obvious implementation does not work, but illustrates the principle. Each of the two parties generates two bits: the love bit l, indicating whether they fancy the other party, and the pad bit p, chosen at random. A sends $l_A \oplus p_A$ to S, and p_A to B. B acts symmetrically. After these four messages have been exchanged, S sends out to both A and B an identical message obtained by combining the two bits it received from A and B. From this message, each of the two parties can derive the logical AND of the love bits, $l_A \wedge l_B$, without learning the l of the other party if their own was 0. Unfortunately it can be shown by exhaustive examination of all possible truth tables that this last part will not work, no matter what the boolean function used by S to combine the bits it receives.

4 From millionaires to lovers

The solution we propose builds upon a result by Yao [6]:

> Two millionaires wish to know who is richer; however, they do not want to find out inadvertently any additional information about each other's wealth. How can they carry out such a conversation?

Yao gives a clever solution based on public key cryptography and modular arithmetic which is sufficiently complicated that Schneier [4] moves it from the front to the back of his book[2]. Fortunately we do not need to go into details here, because our solution uses Yao's as a black box.

He provides a protocol that runs between A and B with no third party. A thinks of a number x_A, B thinks of a number x_B, and at the end of the protocol A and B get to know whether $x_A < x_B$ or not. Here we present a simple construction for the secure computation of the AND function using Yao's protocol as a primitive.

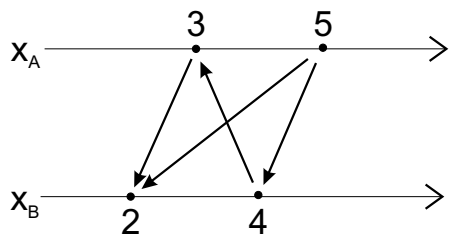


Fig. 3. Only one pair has $x_A < x_B$ (arrow pointing towards x_A).

We give A two numbers to be used as his/her "Yao wealth", one if s/he wishes to say "I love you", or boolean 1, and one for "I don't", or boolean 0. We do the same with B, with two other numbers. We choose the numbers so that the boolean value of the predicate $x_A < x_B$ corresponds to that of the predicate $\tilde{x}_A \wedge \tilde{x}_B$ where x is the chosen number and $\tilde{x}$ its boolean semantics.

The four numbers are easily chosen with the help of the diagram of figure 3. A gets any two numbers, say 3 and 5, and B gets two other numbers of which the first is smaller than both of A's numbers, while the second is between them — say 2 and 4 respectively. If we draw arrows between A's numbers and B's, with the tip of the arrow always pointing towards the lower number (like the $<$ symbol does), we notice that all the arrows go from A to B except the one from 4 to 3, which is the only case where $x_A < x_B$. This three-against-one situation matches that of the AND truth table. We therefore tell A and B that their "1"

[2] The problem is described in section 6.2, but the mathematical details of Yao's solution are relegated to section 23.14.

bits are represented by 3 and 4 respectively. This mapping establishes the desired homomorphism between $<$ and AND, thereby solving the problem.

Our romantic cryptography protocol is therefore as follows. A thinks of a number: 3 if s/he fancies B, 5 otherwise. B thinks of a number: 4 if s/he fancies A, 2 otherwise. Then A and B run Yao's protocol, with their chosen number as their "wealth". If Yao says that A's number is smaller than B's, then they love each other, otherwise not.

5 Other approaches

Before coming up with this solution, we discussed the idea of romantic cryptography at the weekly meeting of the Cambridge security group[3]. This elicited stimulating discussions and a number of interesting suggestions, some of which we feel are worth mentioning here for comparison.

5.1 A probabilistic scheme

The following scheme is due to Markus Kuhn.

A and B take turns in sending messages to each other. At every turn, a principal who fancies the other transmits a 1, while a principal who does not transmits a random bit. As soon as one of the participants sends a 0, s/he reveals that s/he does not fancy the other party. However, for as long as a participant keeps sending "1" bits, nothing can be said for sure about his/her choice. The longer the unbroken chain of "1" bits, though, the more likely that s/he fancies the other party: for an unbroken chain of length n, the probability is

$$P(n) = 1 - \left(\frac{1}{2}\right)^n .$$

As soon as one of the parties issues a 0, the protocol obviously terminates with the outcome of a boolean 0, or "you two do not mutually fancy each other". The party who issued the 0, say A, still doesn't know whether B fancies him/her — all that is known is that, if B so far issued a sequence of n "1"s, that s/he fancies A with probability $P(n)$. But then, until A issued that 0, the same could have been said of him/her with probability $P(n')$, with n' differing from n by at most 1 (depending on who sent the first message). Most importantly, the unloved party B can still plausibly claim that s/he did not love A either, even if s/he in fact did.

If both parties fancy each other, neither of them will ever issue a 0, so the protocol has a termination problem. To overcome this, either party is free to drop out at any time. This means that this protocol never gives a definite "yes" answer (boolean 1) — only a "yes with a certain probability". The participants can of course make this probability arbitrarily high by continuing the protocol for as long as they consider necessary.

[3] On Friday 1999-03-26.

If A does not fancy B, the probability of this non-love being revealed is doubled at each message that A sends. This increase is quite steep and puts at an unfair disadvantage the party who starts the game. To compensate for this, Kuhn actually substitutes the $\frac{1}{2}$ above with a tunable parameter α, with $0 \ll \alpha < 1$, to which both parties commit before the start of the protocol. If A fancies B, s/he always transmits a 1 as before. If s/he does not, she sends a 1 with probability α or a 0 with probability $1 - \alpha$. By choosing an α arbitrarily close to 1, the likelihood of long unbroken chains of "1"s even for non-loving parties can be increased at will, therefore evening out the inbalance coming from the fact that the lengths of the two chains may differ by 1 when the first 0 bit comes out.

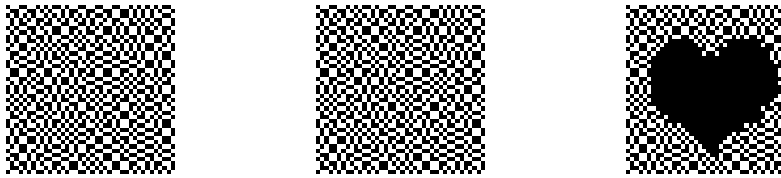
We observe two minor problems. Firstly, as we said, the "yes" answer is only probabilistic. Secondly, the party who first issues a 0, say A, *reveals* that s/he does not fancy the other. If B fancies A then this is not a problem, because A's non-love would be implied by the outcome of the protocol anyway. But if neither B nor A fancies the other, then there is no need to disclose to B that A does not fancy him/her. In such a situation, we would prefer the protocol to leave both parties in the dark about each other's preference.

Mike Roe extended Kuhn's construction to a multiuser environment where everybody runs the protocol against everybody else — a scenario of perhaps even greater practical significance, as we shall see in section 6. Each pair of players establishes a secret key using Diffie-Hellman [1]. The key is used to seed a pseudorandom keystream generator. To say "I don't fancy you", people send a bit chosen at random; while to say "I fancy you" they send the next bit from the common keystream. (There is little difference between this arrangement and one in which each pair establishes an encrypted channel, using Diffie-Hellman to bootstrap a stream cipher, and then runs Kuhn's protocol over it.)

5.2 A visual scheme (or two)

An entirely different approach, based on Naor and Shamir's "visual cryptography" [3], was proposed by Bruno Crispo. Visual cryptography is a secret sharing scheme in which the shares can be reassembled without using a computer, by simple superposition of transparencies inscribed with random-looking dot patterns. In its simplest incarnation, visual cryptography is a visual one-time pad in which a cryptogram and a key are combined to yield a plaintext: by superposing two transparencies with random-looking patterns, a picture emerges — and no information about that picture can be obtained, in the Shannon sense, from either of the transparencies on its own.

Crispo suggests generating four transparencies: two really random ones, and two shares that combine to form a picture of a heart (see figure 4). At a casual inspection, all four transparencies look random and indistinguishable from each other. A and B each receive one random transparency and one with a share. To say "I fancy you", they exhibit the transparency with the share; to say they don't, they exhibit the random one. Then the two chosen transparencies are superposed: if the heart appears, it's mutual love (boolean 1); all other combinations yield a random pattern and signify boolean 0.



Share 1 Share 2 Combination

Fig. 4. Visual cryptography: superposing the two shares reveals a plaintext picture with reduced contrast.

This scheme is interesting for several reasons. It performs a logical AND using visual cryptography, which instead implements an XOR, which in turn is implemented on top of a physical OR (the superposition of "paper" or "ink" pixels, with "ink" being 1). In fact, the construction to implement XOR out of OR at the price of a reduction in contrast is precisely the clever idea at the core of visual cryptography. It is quite difficult to obtain XOR from OR, and so is obtaining AND from XOR. Yet, obtaining AND from OR is trivial — just exchange the meanings of 0 and 1. So it is natural to wonder if the same result could not be achieved more simply by avoiding the XOR step. A moment's reasoning shows that yes, the AND can be trivially computed by superposition of transparencies (just say that "transparent" means 1), but then the contributions of the two participants become public.

A simpler alternative to Crispo's scheme is therefore as follows: we prepare a mask that is completely black except for one transparent pixel at specified coordinates somewhere in the middle of the sheet. Each user generates[4] a transparency made of random pixels everywhere, except for a well-chosen value in that specific location (with transparent = love = 1, black = non-love = 0). Then the transparencies from the two users, plus the mask, are superposed. If the result is all black, the answer is negative; but if a white pixel shines through, it's mutual love. The fact that the patterns are random ensures that neither A nor B can "see" each other's contribution at a glance while manipulating the transparencies.

This method is more steganographic than cryptographic: it just hides the only relevant pixel in an ocean of random covertext. It can be seen as a Cardano grille [2] with only one hole.

The security of this scheme, just like that of Crispo's original one, relies on the assumption that a human won't be able to see anything other than random

[4] Or picks from a pre-made stack of random transparencies, in the appropriate trade-off between security and convenience.

noise in those transparencies. Things will fail if the parties have an eye for detail, if they videotape the session or if they are allowed to examine each other's transparencies too closely. This construction, unlike visual cryptography, is not cryptographically strong. Like our first scheme with the balance and spheres, it cannot be translated from the world of physics into that of mathematics.

6 The social context

A fundamental objection to all the schemes presented above, first presented to us by Alan Jones, is that they are vulnerable to liars. If A wishes to know whether B loves him/her, A just has to say that s/he loves B, and the outcome of the AND will immediately tell him/her whether B loves him/her or not. At that point A can say "I didn't fancy you really, I said I did just because I was curious about whether you fancied me".

We recognize this situation as isomorphic to going for a job interview in a place where one doesn't really want to work, just to find out whether a job offer would actually be forthcoming, and at what salary. This scenario, by the way, highlights how the secure multiparty computation of AND is something for which a need is sometimes felt in the commercial world as well as between potential lovers. It has been observed that employers would much rather make an offer not to all the candidates they like, but just to the ones who are also going to accept it. Obviously, being rejected hurts your pride regardless of whether you are a spotty teenager or a mature managing director.

We do not believe that the problem of liars can be dealt with at the protocol level — if the liar keeps quiet about having lied, the external trace of his/her protocol inputs and outputs will be indistinguishable from that of an honest player. Perhaps the most appropriate safeguard is "contractual": the principals who agree to play the protocol also agree to be to some extent bound by its outcome. So, if A and B run the protocol in sight of all their schoolmates, then if the observed outcome is "mutual love" then the hypothetical liar cannot renege on his/her word without losing face.

Another basic objection to the very idea of romantic cryptography is that whoever starts the protocol probably fancies the other — why would they bother otherwise? One solution to this is to make it socially acceptable to run the protocol between randomly chosen pairs of participants. For example, the school could have a "love day" in which, by decree, everyone runs the protocol against everyone else. Before our patient readers dismiss all this as silly fantasies unworthy of their honourable attention, let us just point out that something similar *has already been organized* at Cambridge in at least two independent efforts[5]. Richard Neill and John Surcombe created a matchmaking server for Cambridge students at `http://romance.ucam.org/`. After a few months of activity, the site

[5] Which perhaps says more about Cambridge social life than about the soundness of the idea — but our view on the matter might be biased, so we shall leave such judgements to the reader.

now claims to get 10,000 page views per day, which makes it "the most popular student-run website in Cambridge by some margin". In their own words:

> The RUO[6] Love Web is a variant on the KCSU[7] Love-in organised for Valentine's Day 2000, which KCSU describe as "the geekiest, most cowardly way ever of finding yourself a partner". We have shamelessly stolen this idea and added our own special flavour to it. The basic idea is the same: everyone tells RUO who they fancy, and if there are any matches, we put you in touch. The big differences are that it runs all year round, and when we find a match we let you exchange messages on RUO so you can organise a proper meeting...

Of course the services offered by KCSU and RUO rely on a *Trusted* Third Party, namely KCSU or RUO, and therefore do not count as romantic cryptography — these servers run the role of the passer-by weighing the spheres, only on an industrial scale.

In the past we examined a similar problem in the context of anonymous auctions [5]. It is trivial to provide anonymity if one accepts the presence of a trusted third party through which all transactions are routed; but it is much more interesting to devise schemes that work without the trusted third party, either because the participants communicate directly, or because the third party they use does not have to be trusted with secret information. As we pointed out at the time, this has advantages for the servers as well, in terms of reduced liability.

Here, too, it would be possible to run a more discreet and robust matchmaking service if the server were only used for introductions ("user `abc12`, now please run the romantic cryptography protocol with user `lmn567`", repeated for all the possible pairs of users who have expressed an interest in the service) instead of being trusted with sensitive information about whom each user fancies.

7 Conclusions

The romantic cryptography problem is a secure multiparty computation of the AND function, and we have shown how to solve it by reducing it to Yao's secure multiparty computation of the $<$ function.

In a world that becomes ever more web-centric, with thin clients supported by fat servers, secure multiparty computation may be a useful tool to avoid having to trust the fat servers unconditionally.

The constructions presented in this paper are little more than recreations, but they help us observe how the real-world semantics of the problem may add subtle security constraints to those implied by the underlying mathematics. Saying that "romantic cryptography is secure multiparty AND" seems at first to capture all the technical aspects of the problem, but in raw "secure multiparty

[6] RUO = `romance.ucam.org`.
[7] KCSU = King's College Student Union.

AND" there is nothing to suggest that the initiator of the protocol is likely to contribute a "1" bit. There is a lesson in there.

We shall of course be pleased to hear from any real-world users who manage to find a partner using any of the techniques presented in this paper. We suspect, however, that success in this endeavour might involve rather more than the correct application of our protocols.

8 Acknowledgements

We are grateful to the members of the security group at Cambridge for being receptive, creative and critical about these ideas. Those who offered the most significant contributions have been credited in the text, but many others took part in interesting related discussions.

References

1. Whitfield Diffie and Martin E. Hellman. "New Directions in Cryptography". *IEEE Transactions on Information Theory*, **IT-22**(6):644–654, Nov 1976.
2. David Kahn. *The Codebreakers: The Story of Secret Writing.* MacMillan Publishing Company, New York, NY, USA, 1967.
3. Moni Naor and Adi Shamir. "Visual Cryptography". In Alfredo De Santis (ed.), "Advances in cryptology — EUROCRYPT '94: Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9–12, 1994: proceedings", vol. 950 of *Lecture Notes in Computer Science*, pp. 1–12. Springer-Verlag, 1995. ISBN 3-540-60176-7. ISSN 0302-9743. http://www.wisdom.weizmann.ac.il/~naor/PAPERS/visual_pap.ps.gz.
4. Bruce Schneier. *Applied Cryptography,* 2^{nd} *ed., Protocols, Algorithms, and Source Code in C.* Wiley, 1996. ISBN 0-471-11709-9.
5. Frank Stajano and Ross Anderson. "The Cocaine Auction Protocol: On The Power Of Anonymous Broadcast". In Andreas Pfitzmann (ed.), "Proceedings of 3^{rd} International Information Hiding Workshop", Lecture Notes in Computer Science. Springer-Verlag, Dresden, Germany, Oct 1999. http://www.cl.cam.ac.uk/~fms27/cocaine/. Also available as AT&T Laboratories Cambridge Technical Report 1999.4.
6. Andrew C. Yao. "Protocols for Secure Computations". In "23^{rd} Annual Symposium on Foundations of Computer Science", pp. 160–164. IEEE Computer Society Press, Los Alamitos, Ca., USA, Nov 1982. ISSN 0272-5428.