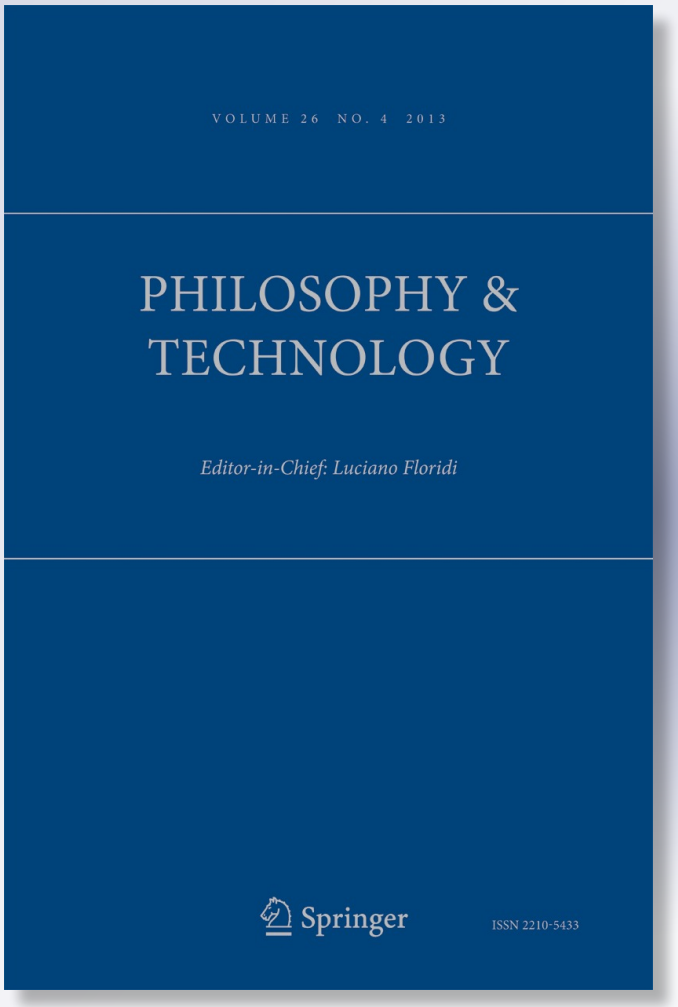
Living in an Impossible World: Real-izing the Consequences of Intransitive Trust

Bruce Christianson

Philosophy & Technology

ISSN 2210-5433
Volume 26
Number 4

Philos. Technol. (2013) 26:411-429
DOI 10.1007/s13347-013-0128-5



Springer

Your article is published under the Creative Commons Attribution license which allows users to read, copy, distribute and make derivative works, as long as the author of the original work is cited. You may self-archive this article on your own website, an institutional repository or funder's repository and make it publicly available immediately.

Philos. Technol. (2013) 26:411–429
DOI 10.1007/s13347-013-0128-5

SPECIAL ISSUE

Living in an Impossible World: Real-izing the Consequences of Intransitive Trust

Bruce Christianson

Received: 25 January 2013 / Accepted: 1 August 2013 / Published online: 21 August 2013
© The Author(s) 2013. This article is published with open access at Springerlink.com

Abstract Many accounts of online trust are based upon mechanisms for building reputation. Trust is portrayed as desirable, and handing off trust is easier if trust is modelled to be transitive. But in the analysis of cyber-security protocols, trust is usually used as a substitute for certain knowledge: it follows that if there is no residual risk, then there is no need for trust. On this grimmer understanding, the less that users are required to trust, the better. Involuntary transitivity of trust becomes corrosive, because it prevents participants from having control—or even knowledge—of the risks to which their trust assumptions expose them. In this paper, we take the stance that controlling the transitivity of trust requires us to recognise trust as a non-referentially transparent modality, similar to but significantly weaker than the epistemic modalities, and to accept the corollary that imaginary (indeed—even impossible) threats can have real consequences that adversely affect online security. An apparently paradoxical outcome is that the desire of principals to keep their trust assumptions private can actually assist the design of systems to satisfy multiple security agendas. However, this approach requires agents to have the capability to predicate accurately about states of affairs that are logically inconsistent with their beliefs, and consequently, designing systems in this way becomes more akin to diplomacy than engineering.

Keywords Counterfactual reasoning · Doxastic logic · Intensional properties · Privacy · Security · Trust

1 Introduction

This paper examines some issues concerning the use of trust assumptions to underpin the design of secure online systems.

B. Christianson (✉)
Science and Technology Research Institute, University of Hertfordshire, College Lane, Hatfield AL10 9AB England, UK
e-mail: b.christianson@herts.ac.uk

Trust is an essential ingredient in any distributed system: Alice cannot directly inspect the state of remote parts of the system, so she must trust Bob to tell her. So far so good—but what if Bob is trusting Carol? Does this mean that Alice has to trust Carol too?

This didn't matter in the old days, when systems were monolithic. The system designers specified a uniform security policy, which set out whom everybody trusted, and for what purposes. Those days are gone. Online systems are increasingly built using repurposed components, by co-operating participants who have different agendas, different beliefs about what is acceptable—or even possible—and very different views about whom they trust.[1] There is no "lowest common boss" to whom conflicts can be appealed, and while governments may be participants, they are no longer in charge and in most cases don't really want to be. And the stakes are high: online systems broker a vast amount of sensitive personal information. This sensitive information includes Bob's trust assumptions, which he may be very unwilling to reveal to Alice.

But trust involves risk. If Alice has to trust Carol whenever Bob does, but has no way of knowing whom Bob is trusting, then Alice has no way to control the risks to which she is exposed. This is the problem that we call the *promiscuous transitivity* of trust.

The keystone of this paper is an argument that trust is not promiscuously transitive. According to this argument, trust turns out to be an intrinsically intensional notion, one which involves a doxastic agent and the world as that agent understands it to be: trust cannot be unpacked into a purely extensional relationship between an agent and the world as it actually is. However, inferences derived from trust have an intrinsically weaker epistemic status than knowledge, and agents adhering to different trust policies may have mutually inconsistent understandings of the world.

The second part of the paper unpicks some of the consequences of this for the design of online systems that are required to comply with a range of mutually incompatible, and only partially disclosed, security policies. Our perhaps slightly surprising conclusion is that the desire of agents to keep their trust assumptions private can potentially assist the seemingly impossible task of building such systems. However, the resulting approach to system design is more akin to diplomacy than to engineering.

Compared to the vast amount that has been written on, say, knowledge or truth, the philosophical literature has historically had surprisingly little to say about trust.[2] Fortunately, that situation is now changing, and recent years have seen an explosion of interest in the philosophy of trust, particularly of trust on the internet. I'm not going to review this literature systematically here, but Simpson (2012) gives a superb genealogical map of the different ways in which the concept of trust has been evoked.

In this paper, I use the word trust in a number of different but interlocking senses,[3] some of them quite technical. Although I maintain that each of these uses has enough in common with the folk-concept of trust to justify using the word "trust" for it (rather than, say, "squibble"), I make no attempt to give any complete unpacking of trust. Nor do I seek

[1] For example, when I purchase an e-book, the vendor may not be the owner of the digital rights, nor the host of the website I buy through; I pay using a service such as PayPal, and the content is delivered through an infrastructure operated by yet another party. On what basis do I believe that I shall receive the correct e-book, on the correct device, in exchange for the correct charge to my credit card? Who is trusting whom, to do what? And who has learned what personal information about me in the process?

[2] Baier (1986) comments on this "strange silence" in the specific context of moral philosophy.

[3] I had some idea of distinguishing these by writing them trust_1, trust_2 and so on, but gave up shortly after reaching double figures. I rely instead upon the reader to remain alert to context.

to establish a semantic monopoly—there is plenty of room for others to use the word differently. However, I shall mean what I stipulate at each point.

The connotations of "Alice trusts Bob" that I exploit systematically are, firstly that Alice believes (or at least, commits to behave as if it were the case) that Bob's actions will respect Alice's interests, in some highly context-dependent sense, and secondly that Alice is nevertheless open to the possibility that Bob might (might actually, rather than counterfactually) act otherwise, in such a way that Alice suffers some damage to her integrity.

Similar remarks about stipulation apply to the term "believes". In this paper, we deliberately draw no clear distinction between human principals and automated agents acting on their behalf. Those concerned about attributing opinions to automata can read "believes" as "is authorised to act as if", following in the tradition of Burrows et al. (1990), Abadi et al. (1993), etc.

Here is a roadmap for the paper. We start with a quick review of the essential role played by trust in distributed systems and the reasons why transitivity of trust is sometimes enabling and sometimes corrosive. Next, we disentangle a particular conception of trust, one we claim will turn out to be useful to the system designer, from a number of other concepts (reliance, trustworthiness, jurisdiction) with which trust is often naturally aligned. We then demonstrate, via a flawed proof giving rise to a counterexample, that this conception of trust is not promiscuously transitive. It turns out that this is the case precisely because the notion of trust we are using is intrinsically intensional.

Next, we examine in more detail the implications for system design of users being reticent about their trust assumptions. A consequence of reticence, in the light of the intensional nature of trust, is that trust-derived beliefs have an intrinsically weak epistemic status that generally cannot be traded up to knowledge even in principle. We argue that the resulting ambiguity, about which purpose is served by the various measures introduced to counter perceived threats, can be exploited so as to provide system designers with more options and make it easier for users with different agendas to collaborate. Our conclusion is that it doesn't matter if there is no consensus about *why* the system is safe to use, so long as each of the participants has appropriate reasons to believe that it is.

2 The Need for Transitivity

Systems have *requirements*, which we can think of as desired behaviours for the system. In terms of the design of the system, requirements generally correspond to predicates that the global state of the system must satisfy. In a distributed system Alice cannot directly inspect the state of remote elements of the system. Thus, she is dependent upon representations, in the form of messages informing her, for example, that the state of Bob's part of the system satisfies a particular predicate at a particular time. These messages are generally sent as part of a *protocol* for changing the global system state.[4]

[4] A *protocol* is a set of rules which, when correctly implemented, has the effect of changing the global state of the system in a way that satisfies a particular (protocol dependent) predicate. The rules specify the syntax of the messages to be exchanged and the semantics of the operations to be carried out when any given message is received: such operations may involve the making of changes to the local state and the sending of further messages. Cryptographic techniques can be used to provide Alice with an assurance that a message is from Bob and is the correct message. However, the protocols used to distribute the cryptographic keys typically themselves presume correct operation of parts of the system remote from Alice, a point to which we shall return below.

Alice's *security policy* must specify the predicates that each part of the system is required to satisfy in order for her transaction to be carried out securely. But her security policy must also allow Alice to determine whether or not she is justified in proceeding to the next step in the protocol on the basis of the messages she has received: in particular, should Alice believe Bob's representation or not? Alice will need to make some trust assumptions in order to perform any non-trivial distributed operations at all.[5]

Many approaches to online trust are based upon some notion of reputation. The idea is that Bob decides whether or not to trust Carol based upon the reputation that Carol has. Reputation is initially a feature of the mechanism by which Bob is introduced to Carol, for example via a trusted third party, such as a close relative or an online trading site. This reputation is modified in the light of Bob's subsequent beliefs about Carol's dealings with himself and with other principals, and this in turn affects the initial reputation that Carol acquires when Bob introduces her to Alice, who trusts Bob.

On this view of trust, it is natural to seek ways of enabling the hand-off of trust by making reputation more public. This enables others to trust someone who is already highly trusted. Finding the right mechanisms for doing this might be problematic (in a purely technical sense), but it seems clear that it would be desirable to facilitate the *transitivity* of trust, so that:

If A trusts B and B trusts C, then it is possible for A to trust C.

Here is a premonition that transitivity of trust is not without its problems:

Bob trusts Carol to service his car. Alice trusts Bob to recommend a suitable garage. Bob recommends Carol to Alice, based on the excellent service he has received. Is Alice justified in trusting Carol to service her car?

In fact, Carol really fancies Bob and would like to go out with him, which is why he has had such a good experience. Bob's trust in Carol is justified, and Bob has been honest with Alice, but nevertheless, Alice ends up with an inferior deal.

The reputation-based approach to trust focuses upon what we might describe as the supply side—how can we provide more trust? I want to start my argument by looking instead at the demand side of trust—why do we need to have trust online at all? Looking at the electronic protocols in which trust features, the answer is almost always that trust is being used as a substitute for knowledge.

A consequence of using trust as a substitute for knowledge is that there is no trust without risk: to the extent that we can be certain, we have no need for trust. Helen Nissenbaum (2001) puts this very well: suppose that Bob is a psychopath who has been shackled to the wall. Although Alice may feel secure while she is in the room with Bob, this is not because Alice trusts Bob.[6]

Trust thus comes with a price tag. The cost of using a trust assumption is exposure to risk. The situation online is similar to that described by Piero Ferrucci (2006):

[5] Generally, Alice will need to combine the *dynamic* representations made by trusted *counterparties* in the system, such as Bob, with *static* representations about the properties of the system made to Alice by the system designer. Usually, in today's systems, Alice ends up being effectively forced to trust the system designer or, at least, to commit to behaving as if she did. As we shall see, this is not actually inevitable. Das Chowdhury and Christianson (2013) refers to the need for users to trust large parts of the system infrastructure before being able to accomplish anything at all as "compulsive trust", in analogy with compulsive gambling.

[6] Although, if Alice lacks the requisite technical expertise herself, she may feel secure because she trusts the manufacturer of the shackles.

> "To trust is to bet. Each time we trust, we put ourselves on the line. If we confide in a friend, we can be betrayed. If we put faith in a partner we can be abandoned. If we trust in the world, we can be crushed. Far too often it ends that way. But the alternative is worse still, because if we do not put ourselves on the line, nothing will happen."

Usually, when Alice chooses to trust Bob in the online world, it is not because she particularly desires to accept the risk, but because there is no better available alternative: if Alice does not make some trust assumptions, she incurs the opportunity cost of being unable to complete the transaction she wants to make.

On this understanding, the less our system *requires* Alice to trust, the better, because the less involuntary risk she is exposed to that something will go wrong. Limiting the scope of the trust assumptions required by the design of a system is called *trust confinement*. We cannot do without trust online, any more than we can in society generally. But it is desirable that those who use a system be able to keep track of the risks to which they are being exposed and to choose wisely which risks they invest.

3 The Problem with Promiscuity

Different users of the system generally have different requirements, corresponding to their different agendas and security policies.[7] For example, Alice wants the system to give her money when she puts her card into a cash machine, even though her account is at another branch. Alice's primary concern is being able to withdraw money when she does have it. On the other hand, the bank is concerned to make sure that Alice cannot withdraw money that she does not have. It is generally very hard to design systems so that they will demonstrably meet the requirements of disparate groups of users who subscribe to different security policies and trust assumptions.[8]

Some users of the system have desires that correspond to behaviour that the designers do not wish the system to exhibit. For example, Rob would quite like to be able to take Alice's money out of her account without her permission or prior knowledge. Such users[9] are called attackers. Note that Rob may also be a legitimate user of the system, with an account at the same branch as Alice.

In order to satisfy Alice's requirements, the system must behave in a way consistent with Alice's security policy. Alice needs to be able to prove that the system will comply with her security policy—for example that only she, and merchants that she has authorised, are able to remove money from her account. To this end, Alice will need to make some trust assumptions in order to be able to prove that the system (as specified) will exhibit the behaviours that she wants it to have.[10]

[7] In this paper, we draw no particular distinction between security requirements and other types of requirement, preferring to treat all requirements as having potential implications for security. The point is that (a) Alice may have requirements that translate into restrictions upon how Bob may interact with the system, and (b) Bob may not share Alice's view about whether these restrictions are appropriate.

[8] We shall see shortly that the designer's task is apparently made even harder by the fact that users are often unwilling to reveal their trust assumptions or are even willing to be deceitful about them.

[9] Or, at least, users behaving in such a way.

[10] We include non-performance as a type of behaviour, so this statement can include assertions that the system does not exhibit behaviours that Alice wants it to not have. In technical terms, behaviour specification includes both *safety* and *liveness* properties.

Alice thus has a trust confinement problem: in order to control the risks to which she is exposed, she needs to be able to ensure that assertions such as

< Alice trusts Rob >

cannot somehow find their way into the set of statements that can be proved about the system.

However, if trust is *promiscuously* transitive, in the sense that we have:

If < < A trusts B > and < B trusts C > > then < A trusts C >

as a rule of inference, then Alice is in trouble: Bob could decide to trust Rob without Alice's consent—or even knowledge—and Alice would then be trusting Rob even though she believes that Rob is an attacker—and worse, if Bob does not share his trust assumptions with Alice, then Alice may not even be aware that she is trusting Rob until it is too late.

Hidden trust assumptions lie at the root of many security problems. As an example, consider key distribution: how can Edwina be sure that she is using the correct public key for George, whom she has never met in real life?

A classical hierarchy of public key certification authorities (A, B and C) is shown in Fig. 1, along with some of their clients (D, E, F and G). The solid arrows show direct knowledge of public keys, and the dotted arrows show trust about key provision.

E wants to learn the public key of G. E trusts B to provide E with the correct public key for G; however, B does not know the public key of G. B trusts A to provide B with the correct public key for G, but A does not know the public key of G either. C knows the public key of G, so in order for E to prove that he has the correct public key for G, E must assume that A trusts C.

Thus, as well as assuming that trust is transitive, clients must also assume that trust between certification authorities is symmetric. An immediate corollary of this is that all certification authorities in the tree trust one another, whence clients must trust the entire tree.[11] At this point, the hierarchical approach to key certification becomes hopelessly risky.

This type of trust explosion is not confined to hierarchical approaches: analogous problems arise in key handoffs based on web-of-trust models such as Phil Zimmerman's Pretty Good Privacy.[12]

A major advantage of public key over symmetric key cryptography is the capability it provides for localisation of trust, but this localization is lost if Alice's trust can be handed off promiscuously, thereby exposing Alice to risks of which she is not aware.

How can Alice limit the risks to which she is exposed or even maintain awareness of what these risks are? If trust is promiscuously transitive, then trust confinement appears to be an intractable problem.

4 Disentangling Trust

How can we limit the transitivity of trust? Some approaches to unpacking the semantics of trust have the effect of forcing trust to be promiscuously transitive, and I shall argue that such approaches are mistaken. Before we pass on to this, it will

[11] Neither F nor G played any active part in the protocol, but the only way E can distinguish F's key from G's is for E to trust C.

[12] See Herald et al. (2010) for details.

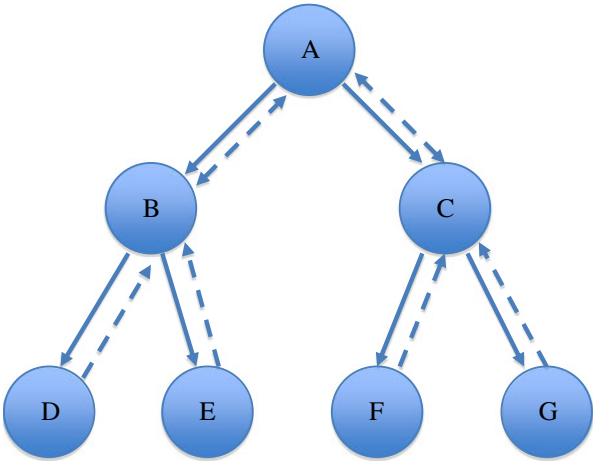


Fig. 1 A public key certification hierarchy

be helpful to distinguish trust from some other notions with which trust is often naturally aligned.

The first of these is *reliance*. If Alice relies upon Bob to carry out some operation, and Bob in turn relies upon Carol, then Alice is relying upon Carol whether she knows it or not. However, it does not follow that any trust is involved. Reliance need not involve openness to risk in the same way as trust: Alice may have a way of checking whether or not Bob has done what he is supposed to do. Alice is not authorised to issue an ISO/IEC 27001 security compliance certificate for her software, whereas Bob is. Bob is relying upon Carol to carry out the appropriate tests before Bob issues the certificate. However, the tests that Carol is required to carry out are in the public domain, and if Alice has the required skill,[13] she can carry out the same tests and determine whether or not Bob has acted correctly, regardless of whether Alice is aware of Carol's involvement. The case of online transaction validation is similar: for example, Alice may be able to verify, but not generate, Bob's asymmetric key signature on the transaction. Conversely, Alice may trust Bob to do something for her that she could alternatively do perfectly well for herself: in this case, she is not reliant upon Bob.

Another notion from which trust should be distinguished is *belief in trustworthiness*. Saying that Alice trusts Bob is conceptually a very different statement to saying that Alice believes that Bob is trustworthy.[14] There are many scenarios in which these two concepts have a natural alignment.[15] However, I claim that for some of the scenarios

[13] And, if Alice does not have the skill herself, then Alice can ask Dolby, whom she does trust, to do the checking for her.

[14] I part company with a number of philosophers at this point, notably with Russell Hardin (2004). I should say at once that there are also perfectly respectable computer scientists who, like Hardin, unpack trust as belief in trustworthiness, although, in some cases, these are regarded as metric, rather than binary, quantities, with the *amount* of trust being a measure of the *strength* of the belief in trustworthiness. Bishop (2003), for example, defines trust as "a measure of trustworthiness, relying on the evidence provided". Others, such as Anderson (2010), follow a similar line to ours: Alice trusts Bob if Bob has the power to violate Alice's security policy.

[15] Simpson's "cognitive trust" is an example of such an alignment, but he is careful to qualify the contexts in which it applies.

discussed in this paper, forcing trust to align with trustworthiness in this way distorts the semantics of one or the other to the point where a useful word is wasted.

That Alice believes that Bob is trustworthy does not entail that Alice actually trusts him: Alice may have no need to do so or may prefer to trust somebody else. Conversely, the fact that Alice trusts Bob online does not entail that Alice believes that Bob is trustworthy: Alice may trust Bob because she has no choice, because she prefers accepting the risk to foregoing the transaction or because Alice's security policy says that her insurance underwriter will accept liability for the consequences if Bob does something wrong.[16] Of course, Alice may be more willing to trust Bob if she believes that the risk is lower, but this is not quite the same relationship between the concepts.

Finally, trust is not the same as *belief in jurisdiction*. Many of the messages which are exchanged in our protocols are effectively performative speech acts[17]: if Bob has control over some aspect of the system state, such as key allocation, then the fact that Bob asserts "Alice, your key is K" is precisely what makes that statement true. We need to distinguish carefully between:

Alice believes < X because < Bob says X > >

and the more general

< Alice believes X > because < Bob says X >

The first of these statements expresses Alice's belief in Bob's jurisdiction over X; the second is more like what we intend by trust. In the first case, Alice might be factually wrong (if it turns out that B saying X is not what makes X true after all); in the second case, Alice cannot be factually wrong (since Alice does, as a matter of fact, believe X because Bob said X) although she may be wrong (in a different sense) to have trusted Bob about X if it turns out that her trust was misplaced.

In the next section, we construct an invalid argument for the promiscuous transitivity of trust. We shall see that this "proof" of transitivity fails in an interesting way, which sheds light on the intensional nature of trust.

In the following sections, we shall exploit this insight to explore a novel understanding of secure system design—specifically to draw some lessons about how to model systems that are going to be shared by groups of users who do not share their security policies.

5 Trust and Reticence

For the purpose of this section, we shall formally model trust as an inference rule, along the line identified towards the end of the previous section:

A trusts < B about X > =

If < A believes < B says X > > then < A believes X >

where each schema applies to all X satisfying some syntactic condition that depends on A and B.

[16] An extension of this argument, by allowing principals to act as underwriters for one another, leads to a genuinely transitive notion called Trust*, described by Clarke et al. (2012).
[17] In the sense of Austin (1962)

Schneier (2012) gives some everyday examples for X:

> "I trust Alice to return a $10 loan but not a $10,000 loan, Bob to return a $10,000 loan but not to babysit an infant, Carol to babysit but not with my house key, Dave with my house key but not my intimate secrets, and Ellen with my intimate secrets but not to return a $10 loan. I trust Frank if a friend vouches for him, a taxi driver as long as he's displaying his license, and Gail as long as she hasn't been drinking."

More formally, X may stand for an assertion about the present (or future) state of part of the system, a predicate or a relation (which may be a second-order relation) involving A, B and possibly other parties, or even another schema.[18] We remark in passing that this approach encompasses delegation.[19]

A more systematic formal development could proceed along the lines of Primiero and Taddeo (2012), adding epistemic contexts and justifications to the environments of the schemata, but I intend not to say too much about further details of the formalism as I want the argument to be quite general. What we have specified here will suffice for the purpose at hand.

At first sight, our trust inference rule appears to make trust a consequence of belief in trustworthiness after all: for if we unpack

B is trustworthy about X =

If < B says X > then X

then it follows that

If < A believes < B is trustworthy about X > > then

If < A believes < B says X > > then < A believes X >

But we need to be careful here: the inference rule given above for trust addresses only one of trust's crucial properties. What is being modelled here is what we might call *latent* trust: Alice *would* believe X *if* Bob said X, and this is all that can be derived from belief in trustworthiness. The case where Alice actually *does* believe X *because* she believes that Bob *did* say X, we might call *actuated* trust.[20]

It is the possibility that trust might become actuated that makes trust risky, and it is arguable that without this possibility, Alice isn't *actually* trusting Bob at all, even though (counterfactually) she *might* have trusted him.[21]

[18] We shall see later that X may stand for an intensional relation, for example
X = < C believes < D trusts < A about < good_wine (T) > > > > >.

[19] In the schema < Alice trusts < Bob about X > >, X can take the form "Bob has used Alice's credit card to purchase G" or "Bob has committed Alice to do Y". See Crispo and Christianson (1999) for more on the semantics of delegation.

[20] The distinction is similar to that between "I would trust Bob with my life (if I ever needed to)", and "I trust Bob with my life (every weekend when we go rock climbing)."

[21] For the distinction between actual and counterfactual possibilities, I am formally following a standard two-dimensional modal semantics, similar to that debated by Stalnaker and Baldwin (2001), although I do not commit here to any particular metaphysical position on the status of other worlds.

For example, it may be that Alice already knows X, won't ever make a decision that is based on X, prefers to trust somebody other than Bob about X or knows that Bob will never say X: in all these cases, the inference rule has no effect.

Paradoxically, this is arguably also the case if Alice can be *certain* that Bob is trustworthy—for example, because she has personally certified the code that she herself loaded into his tamper-proof hardware—the argument is that Alice doesn't *actually* trust Bob, because there is no risk: the effect of a cast-iron proof of trustworthiness is to shackle Bob to the wall.

Next, we define two subsidiary notions:

A trusts < B *competent* about X > =

If < A believes < B believes X > > then < A believes X >

A trusts < B *honest* about X > =

If < A believes < B says X > > then < A believes < B believes X > >

Certainly, trust in competence and trust in honesty jointly entail trust.[22]

Now let us attempt to establish promiscuous transitivity of trust as a derived inference rule and see what goes wrong.

Assume:

A believes < A trusts < B about X > >

A believes < B trusts < C about X > >

A believes < C says X >

Required:

A believes X

Remember, we are going to establish that the conclusion does *not* follow, so it is all the better if we do this under strengthened premises. We therefore choose to substitute for the assumption that A trusts B, the jointly stronger assumptions[23]:

A believes < A trusts < B competent about X > >

A believes < A trusts < B honest about X > >

We suppose that A has an excellent inference engine, which reasons as follows:

Suppose that B were aware of the facts, i.e. that B believes what A believes about what C said. Then B *would* believe < C says X >. Therefore, since

< B trusts < C about X > >

[22] A similar decomposition of trust is made by Hardwig (1991), but in a starkly different setting: his context is the role played by trust in sharing *knowledge* (not mere belief) in Science and Mathematics—extreme instances of monolithic fully shared security policy—for which honesty is crucial. I shall be using this decomposition only for the purpose of the present argument, the formal point being that the conjunction of the two is a strictly stronger premise. In a subsequent section, I shall argue against the utility of honesty as a component of trust in the context of distributed systems security.

[23] Note that we do this only for A trusts B, not for B trusts C.

B *would* believe X under circumstances that are, in fact, true. Therefore, since

A believes < A trusts < B competent about X > >

A *would* believe X under circumstances that are, in fact, true.

Thus, from < A believes < C says X > >, A can deduce < A believes X >. Therefore, A believes < A trusts < C about X > >. QED.

Clearly, something is wrong: nowhere have we used the assumption

< A trusts < B honest about X > >.

Our "proof" would work even if B were not honest.

Here is a counterexample[24] to demonstrate that the conclusion really is not entailed by the premises:

Carol holds the back-up key for Bob's presence in the Cloud. Alice trusts Bob; Bob trusts Carol. Carol is threatened with a fatal power-down while Bob is offline.

Should Carol broadcast the key or destroy the key?

This reflects a common situation, where two security requirements conflict. The first requirement is an *availability* requirement: Bob must have access to the key. The second is a *confidentiality* requirement: only Bob may have access to the key. Broadcasting the key will ensure that the first requirement continues to be met, but will break the second. Destroying the key will ensure that the second requirement continues to be met, but will break the first. Which requirement is primary? This is specified by Bob's security policy, which Alice does not know, but which Bob trusts Carol to follow.

Carol says to Alice:

"Here is a message for Bob. Please tell him his key is 281478a6f9032e49."

Bob knows that this means that Carol has successfully destroyed the key.

Carol's message said something different to Bob to what it said to Alice. Bob and Carol have deceived Alice. Alice trusts Bob, but Bob did not betray Alice's trust: Bob did not say anything to Alice or to anyone else.[25] Even if Alice believes that Carol will be honest to Bob, this does not require Carol to be honest to Alice.

What can we conclude from this? The moral of our failed proof is that reticence[26] can block transitivity. What B would believe about what C says isn't just a question of fact.

A cannot deduce

< B believes < C says X to B > >

from

< A believes < C says X to A > >.

Our crude model of trust is therefore in need of refinement. A number of possibilities present themselves. One possibility is:

A trusts < B about X > =

If < A believes < B says X to A > > then < A believes X >

[24] Adapted from Christianson and Harbison (1997)

[25] Bob could actuate Alice's trust by saying to Alice something different to what Carol said: but even in this case, the fact that Carol could break Bob's security policy does not entail that Carol could break Alice's—although Bob can. The point is that Alice's *inference* about what Bob believes is invalid.

[26] We use the term *reticence* to include deception as well as concealment of beliefs. The term is not intended to be pejorative.

This reformulation leaves it open to A to make additional assumptions, if she chooses, of the form[27]:

If < A believes < B says X to C > > then < A believes < B says X to A > >

It is tempting to go on and consider how to refine our logic, but this would take us too far afield. We already have enough for what we need, and we turn now to considering the consequences of intransitivity for our agenda of secure system design.

The messages to take away from this formal interlude are that trust is not promiscuously transitive and that the intransitivity comes about as a result of the intensional nature of trust, combined with the reticence of principals about their trust assumptions.

6 Intensional Trust

Our conclusion from the previous section is that trust is a modality similar to an epistemic modality, indeed that trust is *doxastic*: statements about beliefs held by particular principals must be unpacked in terms of their reasons for holding those beliefs, not in terms of the facts which would make those beliefs true in the actual world.

The analysis of trust must thus consider intensional conditions and not just extensional properties of the system.

In particular trust, like knowledge, is not referentially transparent. For example, Alice may trust Heosphoros but not trust Hesperos, because she does not realise that these two entities are the same in real life. One of the very problematic aspects of cyberspace, as distinct to the physical world, is the great difficulty involved in knowing when two entities are in fact the same.[28] This is why it is not safe for Alice to adopt a strategy of "trust by default" for remote individuals, trusting Rob until he proves himself untrustworthy, since Rob can simply re-present himself as Bob, and Alice has no way of knowing whether Bob and Rob are the same or different principals.[29]

Distributed systems are ultimately built in order to enable individuals to extend the reach of their societal interactions. However, the various individuals and organisations interacting online increasingly have different requirements, different security policies, including different trust assumptions, and consequently may hold radically different beliefs about the properties of the system. The state of the system that Alice believes actual may be one that Bob believes impossible, and vice versa.[30]

[27] This form includes the case where a public announcement by B suffices.

[28] In cyberspace, we generally have no way to distinguish digital entities by inspecting the physical matrices upon which they are instantiated. All that we can access remotely are abstract bit patterns. Bit patterns may be linked to other bit patterns by cryptography, but establishing remotely verifiable bindings between bit patterns in cyberspace and the real-world physical entities to which they refer turns out to be a startlingly difficult problem. In part, this is because, considered as abstractions, bit patterns have no provenance: it's not an epistemic problem to decide which is the original and which is the copy, for example, there really is no fact of the matter. See Christianson and Malcolm (1998) for more on this issue.

[29] This form of attack is a partial converse to *identity theft*: "Bob" *is* Rob, but *doesn't* wish to be identified as Rob. If the cost of cloning identity is low, reputation systems are vulnerable to a *Sybil* attack, the online equivalent of ballot stuffing.

[30] Indeed, Alice and Bob may not agree even about which states of the system are alethically admissible, let alone acceptable. We return to this point below. A regrettably classic example is the case of phantom withdrawals: the bank is sure that the customer authorised the withdrawal (because the bank trusts the security module), and the customer is sure that they did not.

Even when Alice and Bob do share a belief, they may have radically different grounds for holding it—and they may be unwilling to share these grounds. There are many motives for not sharing information, including commercial confidentiality and considerations of personal privacy.

Among the information which they may not be willing to share is the content of their security policies: our security policy and our trust assumptions reveal a great deal about us, and we may not wish to violate our privacy by allowing other parties unfettered access to them.

As well as trust assumptions, security policies may also include *knowledge* that the principals are unwilling to share.[31] But in contrast with knowledge, which could in principle be shared,[32] beliefs based upon trust may be innately subject dependent: if Alice trusts reticent Bob, then she shares some of his beliefs but not all, and their full belief sets may be mutually inconsistent. Thus, there is in general no possibility of "trading up" trust-derived beliefs to knowledge.[33]

What are the implications of this for secure systems design? We shall argue that the weak epistemic status of trust-based beliefs presents us with an opportunity as well as a problem. But in order to lay the ground for this argument, we must first examine the methodology of the design process in more detail.

7 Imaginary Threats with Real Effects

The conventional narrative of secure systems development goes like this.[34] Legitimate users of a system have *requirements*, which we can think of as predicates that the system state must be constrained to satisfy. Attackers may potentially perform actions that would cause these predicates to be violated. Such potential actions are *threats*. Threats are addressed by incorporating *countermeasures* into the design of the system. These countermeasures provide a security *service* that blocks the threat.

Clients of a telecommunications service need to know the correct account numbers to use for various online payment transactions. The service posts the up-to-date numbers on a publicly visible notice board. The requirement is that these numbers be correct.[35] An attacker wishes to substitute her own bank accounts for these numbers. The countermeasure is to provide a digital signature for the content of the notice board and a key certification service which ensures that only authorised signatures verify correctly. These mechanisms provide the security service of *integrity*.

[31] Baier (1986) makes this very point in the context of security: "some ignorance in the trusting is proper, and awareness that such persons [as security officers] may be relying upon one's not knowing what they know will not destabilize any trust one has in them to do what they are entrusted to do."

[32] I shan't commit to a particular semantics of knowledge, but on most accounts, we have without further assumption that (a) If Bob knows X, then X, which guarantees inter-subjective consistency, and (b) If Alice knows that Bob knows X, then Alice knows X, which is precisely Hardwig's testamentary principle T′.

[33] Our notion of trust-based belief is thus epistemically rather weaker than the strong form of belief discussed by Stalnaker (2006). If his Alice believes X, then she must also believe that she knows X (by his property SB), whereas our Alice may believe X (because Bob says X), but also believe that she does not know X (because she is taking a risk trusting Bob.)

[34] See Stallings (2011) for an even-handed account of the received view.

[35] Or at least should be the same as the numbers that the telecommunications company attempted to post.

Once a countermeasure is deployed, it becomes a part of the system. Consequently, threat analysis is a recursive process. In a mature system, most threats arise as a result of countermeasures placed in the system to counteract a different threat.[36]

A warehouse has a secure area. The threat is that Rob, an unauthorised person, may enter the secure area. The countermeasure is that Alice, the authorised employee, must enter her unique pass number into a keypad to open the door. A new threat is that Rob may attempt to determine Alice's pass number by examining the keypad, for example using an infrared detector or fingerprint dust. A second countermeasure is that motion-sensitive video cameras are set to record anything in the vicinity of the door. An interactional threat is that Rob gains access to a discarded video recording and sees Alice's pass number being typed in. This imposes a new security requirement upon old video recordings: they must not be viewed by unauthorised persons. The designers of the countermeasures may not have adequately considered the interaction between them.[37]

Thus, an imaginary[38] threat can summon a real one into being: a particular countermeasure may be essential even though the corresponding threat is imaginary, because a user who falsely believes the threat to be real will not use the system unless the countermeasure is present. Which threats are actually real and which are in fact imaginary is not relevant here: what matters are the beliefs of the individual principals, derived in accordance with their respective security policies and trust assumptions.

Alice believes that she is at risk from Carol revealing the secret key they share. Bob knows that Alice is not at risk, because Bob has secretly installed spyware in Carol's computer, which ensures that she cannot transmit the key. However, Bob cannot disclose this fact to Alice. In order to persuade Alice that it is safe to use the system, Bob must agree to use a key escrow service, which Alice believes will block the imaginary threat. However, unbeknown to Bob, the escrow service has been penetrated by the CIA. This exposes everyone, including Bob, to a real threat.

Principles may be reluctant to reveal which threats they regard as real and which threats they regard as imaginary. It is unhelpful to attempt to settle the status of a threat by an appeal to "facts": after all, as already remarked, if principals could agree

[36] In the same way as, in most mature systems, the majority of bugs have their origin in an unsuccessful or partially successful attempt to fix a previous bug. There are interesting interactions between the analysis of trust used in game theory (see for example Eckel and Wilson (2003)) and the approach to exploiting trust in the design process described here. For a wonderful account of the threat–countermeasure arms races being played out in casinos, see Forte (2004), and for an application of this to protocol design, see Clark et al. (2013).

[37] This kind of destructive interference between two perfectly rational countermeasures to separate threats is a common source of the security errors exploited by attackers in practice.

[38] Here, by *imaginary*, I mean something that has certain specified properties but does not exist; see, for example, Lambert (1983) or Priest (2005). It is tempting to regard the existence or non-existence of a threat as an ontological issue, but the situation is more complicated. Recall that a threat comprises a sequence of admissible state transitions that has the cumulative effect of allowing the global state to violate a predicate corresponding to a system requirement. Alice and Bob may have different beliefs about which states and transitions are alethically possible. They may also have conflicting beliefs about which current and recent states of the system are epistemically open, particularly when they are relying on different trust assumptions in respect of parts remote from themselves; and the predicates that correspond to their requirements are in any case intensional. Stalnaker (2006) sets out a very sensible conceptual framework for reasoning about such systems, which does consider the possibility of beliefs being false in fact: but (since his beliefs are epistemically strong) he follows the traditional position of regarding false beliefs as corresponding to system faults, rather than seeing their (hopefully unrealized) possibility as a fair price paid for privacy.

on the facts, there would be no need for trust at all, and we have already seen that beliefs mediated by trust cannot be cashed out in terms of extensional predicates on the system.

Consequently, although the first instinct of a systems designer dealing with a problematic inconsistency is to attempt to settle the matter by looking at how things actually are (the extensional properties of the state the system is actually in), such an approach cannot settle inconsistencies arising from incompatible trust assumptions.

The situation for secure distributed systems design now appears desperate: it is not possible for the principals to agree on the facts, and the principals themselves are unwilling to reveal their beliefs.[39] How can the system designer possibly satisfy Alice that the system is safe for her to use without inadvertently introducing countermeasures that Bob believes make the system unsafe for him[40]?

We shall argue that the intensional need for reticence, which helps to control the transitivity of trust and limits risk, is a cause for hope as well as for despair. Far from needing to arrive at consensus about which threats are real, it suffices in practice to agree to useful public fictions. The issue is not whether a threat "really is" imaginary, but whether it justifies an "appropriate" countermeasure.

However, this argument requires us first to analyse more closely the relationship between countermeasures and threats.

8 Threat/Service Duality

There is a duality between security threats and security services.[41] To explain this duality, we shall use the example of *authentication* versus *plausible deniability.*

Alice explains to Bob that from time to time she will provide him with inside information texted from a series of pay-as-you-go phones. Alice explains the threat of plausible deniability: an impostor such as Alice's mutant twin sister *m*Alice might text Bob information claiming that it came from Alice. When Bob acts upon the information, Alice denies that the information came from her. Bob could not distinguish deception by *m*Alice from the case in which Alice had given him the information and then changed her mind about the wisdom of doing so.[42]

Alice therefore provides Bob with a mechanism that provides an *authentication* service. This mechanism[43] is based upon a secret key that Alice shares with Bob. This allows Bob to verify that Alice is the source of a particular text message.

Alice has deceived Bob about her security policy: in fact, Alice needs to ensure that the leaked information cannot be attributed to her. For Alice, plausible deniability is the requirement, and unforgeable authentication is the threat. This is the reason that

[39] Whether these are interpreted as opinions or as commitments.

[40] Bearing in mind that Bob may not even be willing to say on the record why he considers the system to be unsafe.

[41] This duality was first pointed out by Michael Roe (1997), but the implications appear still not to be widely understood.

[42] In technical terms, the first of these attacks is a *masquerade*; the second is *sender repudiation*.

[43] For the sake of argument, assume that the mechanism is a Message Authentication Code appended to the text.

Alice has chosen an authentication protocol based upon symmetric key, rather than asymmetric key, cryptography.

Alice has provided Bob with a mechanism that allows him to verify that she is the source of the information that he receives. However, the secret key is shared, and so Bob can produce the authenticator for any message which Alice can. Thus, Alice's authentication cannot be passed to policeman Carol or reporter Darren unless they trust Bob implicitly.[44] Alice can deny giving Bob the message and claim that Bob sent it to himself. Alice has deceived Bob about which was the security requirement and which the threat.[45]

A corollary of Threat/Service duality is that there is not a unique mapping between threats and countermeasures: just as there is more than one possible countermeasure to block a given threat, so for any given countermeasure, there can be more than one threat which it could plausibly be intended to block. Indeed, in many cases, a single countermeasure does block more than one threat. For example, a digital signature based upon an asymmetric key can be used to block both masquerade and sender repudiation.

The following hypothetical examples illustrate the difficulty of inferring a threat from a countermeasure. They should not be regarded as indicative of any actual bank's practice.

It requires two bank employees with different keys to open the safe. Is this because the bank does not trust its managers? Is the threat that the manager might take money from the safe for their own use?

An alternative explanation of the two-key rule is that it keeps the families of bank managers safe and reduces the risk that they will be taken hostage. The threat is that a bank robber will tell the manager to open the safe if she ever wants to see her children again. The need to coordinate the kidnapping of two families increases the number of shares into which the stolen money must be divided and the chances that at least one member of the gang is a police informer. The bank publishes the countermeasure widely (so that bank robbers realise that a single manager cannot open the safe.) However, the threat remains secret: the bank does not need to reveal whether or not it trusts its managers.

A video camera records the teller's hands as she carries out banking transactions. Is this because tellers might short-change customers, by deliberately miscounting cash and pocketing the difference later? This is the threat perceived by the customer and may be the reason that the customer regards the countermeasure as a good idea and is willing for it to be paid for out of the overheads on her account.

However, an alternative threat is that a customer waits until the cash is in the till before falsely accusing the teller of giving a receipt for an amount that is slightly too low, in the hope that the bank will simply pay up the difference rather than close out the till and take another teller off duty to reconcile the cash tray with the recorded transactions. With the countermeasure in place, a simple replay of the tape in the bank manager's office in front of witnesses allows the fraudulent claim to be dismissed. The bank need not reveal to the teller, or to the customer, which of them they do not trust.

A security screen separates the tellers from the customers. Is the threat that the robber might shoot a teller who does not comply with a request to empty her till? Or is the threat that a member of the bank staff might otherwise be tempted to tackle the robber when his back is turned[46]? The bank need not reveal whom the screen is intended to protect.

---

[44] In which case they might as well just take Bob's word for the source of the message. Bob giving Carol the key he shares with Alice is no more convincing for Carol, but has the effect that Bob then has to trust Carol not to masquerade as Alice.

[45] Well, that's Bob's story and it's plausible enough. At the meta-level, Alice and Bob might trust each other absolutely and be colluding against the District Attorney. As Roe points out, precise requirements for the plausible deniability service tend not to be specified in system design documents.

[46] Statistically, this type of event is when customers are most likely to get injured, an outcome that neither banks nor their insurers enjoy.

The consequence of this is that principals do not need to reveal which threats are the ones they regard as real: it suffices if they subscribe publicly to a threat that justifies the countermeasure they seek.[47]

It may be that all the users of the system believe that a particular threat is imaginary, but subscribe publicly to it because it justifies a countermeasure that they all want (perhaps for different private reasons.) Bob does not need to say "I don't trust Carol", but can say "Potential future users might be deterred from joining the system if they were required to trust the key server."

This provides us with a way out of the design paradox.

9 Designing for the Impossible

Increasingly, online systems are required to support diverse groups of clients who have very different beliefs about what threats need to be countered and very different assumptions about whom they trust.

Our task is to design online systems that people believe are safe to use and yet which support privacy by not requiring people to disclose too much about their security policy and trust assumptions. Our argument is that, to enable this, we should shift the focus of our modelling: from extensional properties of the real world in which the system will be deployed, to intensional properties of the partially overlapping but mutually inconsistent virtual worlds in which the various rival principals, stakeholders and protagonists of the system believe themselves to be living.

Bob believes that Alice is living in an impossible world or rather that the world in which Alice falsely believes herself to live is impossible.[48] Bob has no privileged position. Bob must assume that Alice might have the same view of him and his beliefs (if she knew what they were): in fact, Bob must consider even the possibility that Alice's view could be correct, because she may know something that Bob does not. Nevertheless, Alice and Bob need to be able to trust one another sufficiently to enable them to cooperate, and for this, they need to be able to interpret each other's stated beliefs appropriately in the context of their own.[49]

We need the ability to predicate accurately[50] about imaginary threats in impossible worlds if we are to build systems that allow effective interworking between groups adhering to conflicting security policies.

We cannot do without trust online: we can never prove that a distributed system meets our requirements without making some assumptions based on trust; but trust entails risk, so we need to be able to contain transitivity; but users are unwilling to reveal to us whom they trust and for what. Fortunately trust-derived beliefs about remote parts of the system

[47] Selection of the preferred countermeasure from the ones that would block the real threat may be conditioned by which of these countermeasures has a suitable publicly acceptable alternative threat to act as cover story. And as usual, a good strategy for preserving privacy is to behave *slightly* irrationally.

[48] By impossible here we mean something similar to the "open" worlds of Priest (2005) it is not simply that Alice believes the system to be in a state inconsistent with Bob's beliefs about the actual state; Alice believes the system to be in a state which Bob believes it could never conceivably be in under any circumstances, even counterfactual ones.

[49] Alice should ask not "What would make that true?" but "What would make Bob assert that to me?"

[50] And this in turn requires us to be able to reason counterfactually, for example in the style of Stalnaker (2006), about intensions.

are not extensional propositions, so it doesn't fatally matter if different groups of users hold mutually inconsistent beliefs about the state of the system.

All the stakeholders must agree that the system is secure, in the sense that they believe the properties of the system to be consistent with their own security policy. Otherwise, they would be unwilling to use the system. However, there does not need to be a consensus about *why* the system is secure or about what facts and assumptions (including trust assumptions) underpin "the proof" that the system is secure. There doesn't need to be an agreed proof: it suffices that each group has their own reasons to believe their own proof. If we unpack trust intensionally, then this can become an achievable goal.

Under this approach, designing secure systems to be used across multiple security policy domains is more like diplomacy than engineering. The real threats to security need not be agreed or even revealed. Although the agreed countermeasures may be real and necessary, the threats they have been agreed to guard against (and the services they are intended to provide) may be imaginary or impossible.

Open Access This article is distributed under the terms of the Creative Commons Attribution License which permits any use, distribution, and reproduction in any medium, provided the original author(s) and the source are credited.

References

Abadi, M., Burrows, M., Lampson, B., & Plotkin, G. (1993). A calculus for access control in distributed systems. *ACM Transactions on Programming Languages and Systems, 15*(4), 706–734.

Anderson, R. (2010). *Security engineering: a guide to building dependable distributed systems*. New York: Wiley.

Austin, J. L. (1962). *How to do things with words: the William James lectures delivered at Harvard University in 1955*. Oxford: Clarendon. Ed. J.O. Urmson.

Baier, A. (1986). Trust and antitrust. *Ethics, 96*(2), 231–260.

Bishop, M. (2003). *Computer security: art and science*. Reading: Addison Wesley.

Burrows, M., Abadi, M., & Needham, R. (1990). A logic of authentication. *ACM Transactions on Computer Systems (TOCS), 8*(1), 18–36.

Christianson, B., Harbison, W. (1997). Why isn't trust transitive? Security Protocols 4, 171–176, LNCS 1189, Springer

Christianson, B., Malcolm, J. (1998). Binding bit patterns to real world entities. Security Protocols 5, 105–113, LNCS 1361, Springer

Clark, S., Blaze, M., Smith, J. (2012). The casino and the OODA loop: why our protocols always eventually fail. Security Protocols 20, 60–75, LNCS 7622, Springer

Clarke, S., Christianson, B., Xiao, H. (2013). Trust*: using local guarantees to extend the reach of trust. Security Protocols 17, 171–188, LNCS 7028, Springer.

Crispo, B., Christianson, B. (1999). A note about the semantics of delegation. In Autonomous agents: deception, fraud and trust in agent societies. ACM: Seattle, pp 55–64

Das Chowdhury, P., Christianson, B. (2013). More security or less insecurity? Security Protocols 18, LNCS 7061, Springer, (in press).

Eckel, C., & Wilson, R. (2003). The human face of game theory: trust and reciprocity in sequential games. In E. Ostrom & J. Walker (Eds.), *Trust and reciprocity: interdisciplinary lessons from experimental research* (pp. 245–274). New York: Russell Sage.

Ferrucci, P. (2006). *The power of kindness: the unexpected benefits of leading a compassionate life*. Baltimore: Penguin.

Forte, S. (2004). *Casino game protection: a comprehensive guide*. Las Vegas: SLF Publishing.

Hardin, R. (2004). *Trust and trustworthiness*. New York: Russell Sage.

Hardwig, J. (1991). The role of trust in knowledge. *Journal of Philosophy, 88*(12), 693–708.

Herald, S., Clarke, S., & Christianson, B. (2010). A non-transitive trust model for key distribution. *Journal of Information Assurance and Security, 5*(6), 618–625.

Lambert, K. (1983). *Meinong and the principle of independence*. Cambridge: Cambridge University Press.
Nissenbaum, H. (2001). Securing trust online: wisdom or oxymoron? *Boston University Law Review, 81*(3), 635–664.
Priest, G. (2005). *Towards non-being: the logic and metaphysics of intentionality*. Oxford: Clarendon.
Primiero, G., & Taddeo, M. (2012). A modal type theory for formalizing trusted communications. *Journal of Applied Logic, 10*(1), 92–114.
Roe, M. (1997). Cryptography and evidence. PhD thesis, University of Cambridge Computer Laboratory; available online as Technical Report 780 http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-780.pdf. Accessed 1 Aug 2013.
Schneier, B. (2012). *Liars and outliers: enabling the trust that society needs to thrive*. New York: Wiley.
Simpson, T. (2012). What is trust? *Pacific Philosophical Quarterly, 93*(4), 550–569.
Stallings, W. (2011). *Cryptography and network security* (5th ed.). Englewood Cliffs: Prentice Hall.
Stalnaker, R. (2006). On logics of knowledge and belief. *Philosophical Studies, 128*, 169–199.
Stalnaker, R., & Baldwin, T. (2001). On considering a possible world as actual. *Proceedings of the Aristotelian Society, Supplementary Volumes, 75*(1), 141–174.