Formal Definitions and Complexity Results for Trust Relations and Trust Domains*

Simon Kramer**, Rajeev Goré, and Eiji Okamoto

[1] University of Tsukuba, Japan
simon.kramer@a3.epfl.ch
[2] Australian National University
rajeev.gore@anu.edu.au
[3] University of Tsukuba, Japan
okamoto@risk.tsukuba.ac.jp

Abstract. We propose computational, declarative definitions of the concepts of weak and strong *trust relations* between interacting agents, and *trust domains* of trust-related agents in distributed systems. Our definitions yield computational complexity results for deciding potential and actual trust relationships and membership in trust domains, as well as a positive (negative) compositionality result for strong (weak) trust domains. Our defining principle for weak and strong trust is (common) belief in and knowledge of agent correctness, respectively.

Keywords computability, compositionality, and scalability of trust and trustworthiness; computer-aided decision making (CADM); dependable distributed or multi-agent systems; modal logics of belief and knowledge.

1 Introduction

1.1 Motivation

Dependable distributed systems guarantee their functionality both in spite of and thanks to the technologies that implement these systems, and the (man and/or machine) agents[4] that partake in them. The functionality guarantee is conditioned on *naming* and *number*, i.e., on agent identity for the information security aspect [2, Chapter 6] (anonymity, pseudonymity), and on a minimal number of correct (or dually, a maximal number of faulty/corrupt) agents for the aspects of fault tolerance [26] (classical distributed computation) and corruption tolerance [37] (secure multiparty computation). The notion of *agent correctness* (e.g., as induced by a *policy*) in turn depends on the system itself. For example, agent correctness can include: absence of crash (liveness); absence of cryptographic-key compromise; algorithmic, legal, and policy compliance; due

* A technical-report version of this paper including the appendix appears in [22].
** This (corresponding) author's contribution was funded with Grant P 08742 from the Japan Society for the Promotion of Science.
[4] network nodes, parties, processes, processors, real or virtual machines, users, etc.

debt repayment in credit banking; fairness in scheduling; etc. In any case, agent correctness captures the *predictability* of each correct partaking agent that guarantees the functionality of the system to all, correct or faulty, partaking agents. We stress that this predictability is one of agent behaviour rather than mental attitude. All that matters is the behavioural *effect* of an attitude, not the attitude itself. (An attitude may have no or no relevant effect.) In sum, system functionality depends on agent correctness, and agents depend on each other via each other's correctness. Whence the social need, called *trust*, to know whether or not someone behaves correctly. At first sight, such knowledge may seem difficult to attain. Yet in our standard understanding of knowledge defined in terms of indistinguishability of system states, an agent a simply attains knowledge about a fact ϕ in a given state s as soon as ϕ also holds in all the states that are indistinguishable from s to a (cf. Section 2). In particular, a need not be in control of the system.

The concept of trust has at least three different aspects, namely trust *relations* and *domains,* and trust *management.* Our intuitions of them are as follows.

Trust relations An agent a trusts an agent b when a believes or even knows that b behaves correctly, independently of b's mental attitude. Hence, defining trust based on the correct behaviour of agents is more general than defining trust based on their mental attitude. The reader interested in the latter is referred to [11], which is a substantial study of various kinds of trust relations based on belief in and knowledge of mental attitudes.

Trust domains A trust domain is a community of mutually trusting agents with the common belief in or even knowledge of the trust relationships in the community. Informally, a statement ϕ is common belief or knowledge in a community $\mathcal{C}$ when all agents in $\mathcal{C}$ believe or know that ϕ is true (call this new statement ϕ'), all believe or know that ϕ' is true (call this new statement ϕ''), all believe or know that ϕ'' is true (call this new statement ϕ'''), etc. Notice the *mutual awareness* among agents w.r.t. the commonality of their knowledge or belief. This awareness is computationally costly (cf. Section 3). More intuitions on common knowledge in distributed systems can be found in [15].

Trust management Trust management is (cf. [30] for a recent survey):

1. the organisation of trust relations into trust domains (compartments), i.e., the sociology
2. the coordination of trust-building actions, i.e., the flow of trust (partial dehierarchisation and decompartmentation, e.g., by building reputation [31]).

The organisation of trust relations into trust domains requires the ability to decide whether or not a given relation is a trust relation, and a given domain is a trust domain. Ideally, this ability appeals to formal definitions and decidability results for trust relations and trust domains, in order to support the human brain with *computer-aided decision making (CADM)*, as motivated by the following example.

Example 1 (Group size and the human brain). According to [32], "150 is the cognitive limit to the number of people a human brain can maintain a coherent social relationship with". "More generally, there are several layers of natural human group size that increase with a ratio of approximately three: 5, 15, 50, 150, 500, and 1,500". And "[a]s group sizes grow across these boundaries, they have more externally imposed infrastructure—and more formalized security systems."

The motivation for formal definitions now follows from the assumption that trust is a fundamental element of any coherent social relationship; and the motivation for CADM (requiring decidability results) additionally from the desire to extend the cognitive limit of the human brain.

It turns out that deciding trust relationships can be tractable with the aid of modern computers. However, deciding membership in trust domains, though computationally possible in theory, is computationally intractable in practice, even with the aid of clusters or clouds of (super-)computers. What is worse: not only can we not make use of the promised power of cloud computing[5] [1] for deciding membership in trust domains in general, but also in particular when the candidate domains are the computing clouds themselves!

Example 2 (Trust and Cloud Computing). According to [21], in cloud computing, "users are universally required to accept the underlying premise of trust. In fact, some have conjectured that trust is the biggest concern facing cloud computing. Nowhere is the element of trust more apparent than in security, and many believe trust and security to be synonymous." Also according to [28]: "The growing importance of cloud computing makes it increasingly imperative that we grapple with the meaning of trust in the cloud and how the customer, provider, and society in general establish that trust."

Indeed, the automatic validation of the underlying premise of cloud computing, i.e., trust, is an intractably big concern for the clouds themselves, as indicated above. However, the validation of trust can of course still be a tractably big concern for the relations between the cloud members. Anyway, what remains formally elusive is the *declarative meaning of trust*. As a matter of fact, the vast majority of the research literature on trust focuses on *how* to (operationally) establish and maintain trust of some form (e.g., with protocols, recommendation/reputation systems, reference monitors, trusted computing bases [14], etc.), but without actually defining *what* trust of that form (declaratively) means. And the very few works that do attempt declarative definitions of forms of trust do not provide insights in the decidability or complexity of trust domains, or applications to security such as trust in cryptographic-key management (cf. Section 4)

The bottom line is that declaratively defining the meaning of trust and obtaining estimates of the decidability or complexity of trust can be difficult. Yet formally defining trust in terms of (declarative) belief or knowledge of behavioural correctness turns out to be natural, since humans often naturally refer

[5] Cloud computing is automated outsourcing of IT (data storage, calculation routines) into evolving opaque clouds of anonymous computing units.

to these or similar notions when informally explaining what they mean by trust. In distributed or multi-agent systems, attaining an *individual* consciousness of trust in terms of belief (weak trust) or knowledge (strong trust) from agent to agent can be computationally *tractable*. Whereas attaining a *collective* consciousness (mutual awareness) of trust in terms of *common* belief or knowledge in a greater domain (e.g., a cluster, cloud, or other collectives) of agents is computationally *in*tractable. Computationally speaking, *collective trust does not scale*. Trust domains should be family-sized, so to speak.

1.2 Goal

Our goal is four-fold, namely:

1. to provide *computational, declarative definitions for trust relations* between interacting agents, *and trust domains* of trust-related agents in distributed systems
2. to obtain *computational complexity results* for deciding trust relationships and membership in trust domains
3. to instantiate our concepts of trust relations and trust domains in four major applications of trust, namely: Trusted Third Parties (TTPs), the Web of Trust, Public-Key Infrastructures (PKIs), and Identity-Based Cryptography (ID-Based Cryptography).
4. to point out limited computational means for *building trust*, and by that, building up trust relations and trust domains in computer-communication networks.

Contribution To the best of our knowledge, general formal definitions for trust domains, general complexity results for trust relations as well as trust domains, a positive (negative) compositionality result for strong (weak) trust domains, and a generic formalisation of trust in TTPs, the Web of Trust, PKIs, and ID-Based Cryptography are all novel. The resulting (in)tractability insights are of great practical importance for cryptographic-key management, and could may well be similarly important for computing clouds, which we believe should be conceived as trust domains, and for which trust is the underlying premise [21].

1.3 Methodology

Our methodology is to develop our formal definitions for trust relations and trust domains in a framework that is a semantically defined, standard modal logic of belief and knowledge (cf. Section 2). In that, we are interested in the *descriptive* (as opposed to deductive) use of an off-the-shelf, general-purpose (as opposed to special-purpose, e.g., the famous BAN-logic, which uses but does not define trust) logic that is as simple as possible and as complex as necessary—both syntactically and semantically as well as computationally. Our *defining principle* for weak and strong trust is belief in and knowledge of agent correctness, respectively. We then derive our complexity results for deciding trust relationships and

membership in trust domains by reduction to known results for the complexity of belief and knowledge (cf. Section 3). In spite of the practical significance of our results, their derivation is quite simple (which increases their value), thanks to our modal logical framework. The difficulty was to find what we believe to be an interesting formal point of view on trust, which happens to be modal. Other points of view have, to the best of our knowledge, not resulted in general (in)tractability insights into trust, nor a positive (negative) compositionality result for strong (weak) trust domains, nor a generic formalisation of TTPs as well as trust in the Web of Trust, PKIs, and ID-Based Cryptography (cf. Section 4).

2 Formal definitions

We develop our formal definitions of trust relations and trust domains in a framework that is a semantically defined, standard modal logic of common belief and knowledge. The logic is *parametric* in the notion of agent correctness, to be instantiated for each considered distributed system (cf. Appendices C.2–C.4 for our three examples).

Let S designate the considered distributed system (e.g., of sub-systems).

Definition 1 (Framework). *Let*

- $\mathcal{A}$ *designate an arbitrary finite set of unique* agent names[6] *a, b, c, etc.*
- $\mathcal{C} \subseteq \mathcal{A}$ *denote (finite and not necessarily disjoint) communities of agents (referred to by their name)*
- $\mathcal{P} := \{\ \mathsf{correct}(a) \mid a \in \mathcal{A}\ \}$ *designate our (finite) set of* atomic propositions *P for referring to agent correctness*
- $\mathcal{L} \ni \phi\ ::=\ P \mid \neg\phi \mid \phi \wedge \phi \mid \mathsf{CB}_{\mathcal{C}}(\phi) \mid \mathsf{CK}_{\mathcal{C}}(\phi)$ *designate our modal language of formulae ϕ, with $\mathsf{CB}_{\mathcal{C}}(\phi)$ for "it is common belief in the community $\mathcal{C}$ that ϕ", and $\mathsf{CK}_{\mathcal{C}}(\phi)$ for "it is common knowledge in the community $\mathcal{C}$ that ϕ".*

Then given the set $\mathcal{S}$ (the state space) of system states *s induced by S (e.g., via a reachability or, in modal jargon, temporal accessibility relation)*[7]*, we define the* satisfaction relation $\models$ *of our framework in Table 1. There,*

- *"iff" abbreviates "by definition, if and only if"*
- $(\mathfrak{S}, \mathcal{V})$ *designates the (modal)* model *of our framework*
- $\mathfrak{S} := (\mathcal{S}, \{\mathrm{D}_a\}_{a\in\mathcal{A}}, \{\mathrm{E}_a\}_{a\in\mathcal{A}})$ *designates the (modal)* frame *with appropriate (for the system S)*
  - *serial*[8]*, transitive, and Euclidean*[9] *relations* $\mathrm{D}_a \subseteq \mathcal{S} \times \mathcal{S}$ *of* doxastic accessibility *(used for defining* belief*)*

[6] i.e., agent names injectively map to agents (here, names are identifiers)

[7] For example, suppose that there is a set $\mathcal{S}_i$ of initial states for every system S, T designates the system's reachability or, synonymously, temporal accessibility relation, and T^* designates the reflexive transitive closure of T. Then, $\mathcal{S}$ is induced by S in that sense that $\mathcal{S} := \{\ s \mid \text{there is } s_i \in \mathcal{S}_i \text{ such that } s_i\ \mathrm{T}^*\ s\ \}$.

[8] for all $s \in \mathcal{S}$, there is $s' \in \mathcal{S}$ s.t. $s\ \mathrm{D}_a\ s'$

[9] for all $s, s', s'' \in \mathcal{S}$, if $s\ \mathrm{D}_a\ s'$ and $s\ \mathrm{D}_a\ s''$ then $s'\ \mathrm{D}_a\ s''$

Table 1. Satisfaction relation

$(\mathfrak{S},\mathcal{V}),s \models P$:iff $s \in \mathcal{V}(P)$
$(\mathfrak{S},\mathcal{V}),s \models \neg\phi$:iff not $(\mathfrak{S},\mathcal{V}),s \models \phi$
$(\mathfrak{S},\mathcal{V}),s \models \phi \wedge \phi'$:iff $(\mathfrak{S},\mathcal{V}),s \models \phi$ and $(\mathfrak{S},\mathcal{V}),s \models \phi'$
$(\mathfrak{S},\mathcal{V}),s \models \mathsf{CB}_{\mathcal{C}}(\phi)$:iff for all $s' \in \mathcal{S}$, if $s\ \mathrm{D}^{+}_{\mathcal{C}}\ s'$ then $(\mathfrak{S},\mathcal{V}),s' \models \phi$
$(\mathfrak{S},\mathcal{V}),s \models \mathsf{CK}_{\mathcal{C}}(\phi)$:iff for all $s' \in \mathcal{S}$, if $s\ \mathrm{E}^{*}_{\mathcal{C}}\ s'$ then $(\mathfrak{S},\mathcal{V}),s' \models \phi$

  - *equivalence relations* $\mathrm{E}_a \subseteq \mathcal{S} \times \mathcal{S}$ *of* epistemic accessibility *(e.g., state* indistinguishability*, used for defining* knowledge*)*

  such that $\mathrm{D}_a \subseteq \mathrm{E}_a$ *for any* $a \in \mathcal{A}$
- $\mathcal{V} : \mathcal{P} \to 2^{\mathcal{S}}$ *designates the* valuation function *(returning for every* $P \in \mathcal{P}$ *the set of states where* P *is true) to be defined according to the appropriate notion of agent correctness for the system* S *(e.g., see Appendix C)*
- $\mathrm{D}^{+}_{\mathcal{C}}$ *designates the transitive closure of* $\bigcup_{a\in\mathcal{C}} \mathrm{D}_a$
- $\mathrm{E}^{*}_{\mathcal{C}}$ *designates the reflexive transitive closure of* $\bigcup_{a\in\mathcal{C}} \mathrm{E}_a$.

Note that defining (common) belief and knowledge abstractly with a serial, transitive, and Euclidean relation, and an equivalence relation, respectively, has emerged as a common practice that gives greater generality over more concrete approaches [27, Section 7.1]: the concrete definitions of the accessibility relations can be freely determined for a given distributed system provided they comply with the prescribed, characteristic properties. Typically, these definitions involve the projection of global states onto agents' local views [13]. For example, let $a \in \mathcal{A}$, and let π_a designate such a projection (function) for a. Then, epistemic accessibility in the sense of state indistinguishability can be defined such that for all $s, s' \in \mathcal{S}$,

$$s\ \mathrm{E}_a\ s' \text{ :iff } \pi_a(s) = \pi_a(s'),$$

which guarantees that E_a is an equivalence relation. Doxastic accessibility $\mathrm{D}_a \subseteq \mathrm{E}_a$ can be defined from E_a by weakening the reflexivity of E_a to seriality as appropriate for the considered application.

Further note the following macro-definitions: $\phi \vee \phi' := \neg(\neg\phi \wedge \neg\phi')$, $\top := \mathsf{correct}(a) \vee \neg\mathsf{correct}(a)$, $\bot := \neg\top$, $\phi \to \phi' := \neg\phi \vee \phi'$, $\phi \leftrightarrow \phi' := (\phi \to \phi') \wedge (\phi' \to \phi)$, $\mathsf{B}_a(\phi) := \mathsf{CB}_{\{a\}}(\phi)$ (for "a believes that ϕ"), and $\mathsf{K}_a(\phi) := \mathsf{CK}_{\{a\}}(\phi)$ (for "a knows that ϕ"). Likewise we now obtain our declarative definitions of weak and strong trust relations and domains as mere macro-definitions, i.e., as simple syntactic constructions of semantically defined building blocks (cf. Table 2, conjunction over $\mathcal{C} = \emptyset$ being $\top$). Note that we could dually define weak and strong *dis*trust relations, i.e., as belief in and knowledge of agent *in*correctness, respectively, which use *verb-phrase* negation. The reader is invited not to confuse distrust with *absence of trust*, e.g., with $\neg\mathsf{B}_a(\mathsf{correct}(b))$ or $\neg\mathsf{K}_a(\mathsf{correct}(b))$, which use *sentence* negation. Further note that we could define *weak-strong* (dis)trust domains and *strong-weak* (dis)trust domains, i.e., as the common knowledge of weak (dis)trust relations, and the common belief of strong (dis)trust relations,

Table 2. Weak and strong trust relations and domains

$a\ \mathsf{wTrusts}\ b := \mathsf{B}_a(\mathsf{correct}(b))$	a weakly trusts b
$\mathsf{wTD}(\mathcal{C}) := \mathsf{CB}_{\mathcal{C}}(\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{wTrusts}\ b)$	$\mathcal{C}$ is a weak trust domain.
$a\ \mathsf{sTrusts}\ b := \mathsf{K}_a(\mathsf{correct}(b))$	a strongly trusts b
$\mathsf{sTD}(\mathcal{C}) := \mathsf{CK}_{\mathcal{C}}(\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b)$	$\mathcal{C}$ is a strong trust domain.

respectively. The difference between weak and strong trust is induced by the difference between belief and knowledge, respectively: weak trust possibly is wrong (i.e., mistaken belief), whereas strong trust necessarily is right (i.e., truthful belief).

Social networking systems furnish evidence for the adequacy of defining trust domains in terms of common knowledge or at least belief. As a matter of fact, the enumeration of "friends" on a member page in such systems constitutes a public announcement to the readers of that page who are logged in, who see all logged-in readers, etc. And it is common knowledge in the community of (dynamic) epistemic logicians that public announcements of (verifiable) elementary facts induce common knowledge within the addressed public (cf. [34] for a public announcement). So, suppose that you are a member of Facebook, and $\mathcal{C}$ designates the set consisting of you and those of your "friends" that are enumerated on your member page. Further, fix the current moment in time, and call it s. (We may talk about time here; see Footnote 7.) Then the formula $\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b \leftrightarrow \mathsf{sTD}(\mathcal{C})$ is true at s (with no outermost $\mathsf{CK}_{\mathcal{C}}$ operator on the left since the aforementioned public announcement implies it). The formula is a (bi-)conditional because we have not fixed the notion of agent correctness for Facebook (*they* should). Note that the trust relations between you and your "friends" are symmetric, because each "friend" had to give their consent for having the privilege of being enumerated as such on your member page in Facebook.

Definition 2 (Truth & Validity). *The formula ϕ is* true *(or* satisfied*) in the model $(\mathfrak{S},\mathcal{V})$ at the state $s \in \mathcal{S}$:iff $(\mathfrak{S},\mathcal{V}), s \models \phi$. The formula ϕ is* satisfiable *in the model $(\mathfrak{S},\mathcal{V})$:iff there is $s \in \mathcal{S}$ such that $(\mathfrak{S},\mathcal{V}), s \models \phi$. The formula ϕ is* globally true *(or* globally satisfied*) in the model $(\mathfrak{S},\mathcal{V})$, written $(\mathfrak{S},\mathcal{V}) \models \phi$, :iff for all $s \in \mathcal{S}$, $(\mathfrak{S},\mathcal{V}), s \models \phi$. The formula ϕ is* satisfiable *:iff there is a model $(\mathfrak{S},\mathcal{V})$ and a state $s \in \mathcal{S}$ such that $(\mathfrak{S},\mathcal{V}), s \models \phi$. The formula ϕ is* valid*, written $\models \phi$, :iff for all models $(\mathfrak{S},\mathcal{V})$, $(\mathfrak{S},\mathcal{V}) \models \phi$. (cf. [5])*

Fact 1 (Common belief) *Being defined in terms of a serial, transitive, and Euclidean relation, $\mathsf{CB}_{\{a\}}$ is KD45 for any $a \in \mathcal{C} \subseteq \mathcal{A}$, i.e.:*

K: $\models \mathsf{CB}_{\mathcal{C}}(\phi \to \phi') \to (\mathsf{CB}_{\mathcal{C}}(\phi) \to \mathsf{CB}_{\mathcal{C}}(\phi'))$ *(Kripke's law)*
D: $\models \mathsf{CB}_{\{a\}}(\phi) \to \neg\mathsf{CB}_{\{a\}}(\neg\phi)$ *(consistency of beliefs, seriality)*
4: $\models \mathsf{CB}_{\mathcal{C}}(\phi) \to \mathsf{CB}_{\mathcal{C}}(\mathsf{CB}_{\mathcal{C}}(\phi))$ *(positive introspection, transitivity)*
5: $\models \neg\mathsf{CB}_{\mathcal{C}}(\phi) \to \mathsf{CB}_{\mathcal{C}}(\neg\mathsf{CB}_{\mathcal{C}}(\phi))$ *(negative introspection, Euclideanness)*
N: *if* $\models \phi$ *then* $\models \mathsf{CB}_{\mathcal{C}}(\phi)$ *(necessitation).*

Further, let $\mathsf{EB}_{\mathcal{C}}(\phi) := \bigwedge_{a \in \mathcal{C}} \mathsf{B}_a(\phi)$ *("everybody in* $\mathcal{C}$ *believes that* ϕ*"). Then:*

- $\models \mathsf{CB}_{\mathcal{C}}(\phi) \rightarrow \mathsf{EB}_{\mathcal{C}}(\phi)$
- $\models \mathsf{CB}_{\mathcal{C}}(\phi) \rightarrow \mathsf{EB}_{\mathcal{C}}(\mathsf{CB}_{\mathcal{C}}(\phi))$
- $\models \mathsf{CB}_{\mathcal{C}}(\phi \rightarrow \mathsf{EB}_{\mathcal{C}}(\phi)) \rightarrow (\mathsf{EB}_{\mathcal{C}}(\phi) \rightarrow \mathsf{CB}_{\mathcal{C}}(\phi))$.

For details see [27, Section 7.1].

The difference between belief and knowledge is that belief possibly is wrong (cf. the **D** law), whereas knowledge necessarily is right (cf. the following **T** law).

Fact 2 (Common knowledge) *Being defined in terms of an equivalence relation,* $\mathsf{CK}_{\mathcal{C}}$ *is S5 for any* $\mathcal{C} \subseteq \mathcal{A}$*, i.e.:*

K: $\models \mathsf{CK}_{\mathcal{C}}(\phi \rightarrow \phi') \rightarrow (\mathsf{CK}_{\mathcal{C}}(\phi) \rightarrow \mathsf{CK}_{\mathcal{C}}(\phi'))$ *(Kripke's law)*
T: $\models \mathsf{CK}_{\mathcal{C}}(\phi) \rightarrow \phi$ *(truth law, reflexivity)*
4: $\models \mathsf{CK}_{\mathcal{C}}(\phi) \rightarrow \mathsf{CK}_{\mathcal{C}}(\mathsf{CK}_{\mathcal{C}}(\phi))$ *(positive introspection)*
5: $\models \neg\mathsf{CK}_{\mathcal{C}}(\phi) \rightarrow \mathsf{CK}_{\mathcal{C}}(\neg\mathsf{CK}_{\mathcal{C}}(\phi))$ *(negative introspection)*
N: *if* $\models \phi$ *then* $\models \mathsf{CK}_{\mathcal{C}}(\phi)$ *(necessitation).*

Further, let $\mathsf{EK}_{\mathcal{C}}(\phi) := \bigwedge_{a \in \mathcal{C}} \mathsf{K}_a(\phi)$ *("everybody in* $\mathcal{C}$ *knows that* ϕ*"). Then:*

- $\models \mathsf{CK}_{\mathcal{C}}(\phi) \rightarrow \mathsf{EK}_{\mathcal{C}}(\mathsf{CK}_{\mathcal{C}}(\phi))$
- $\models \mathsf{CK}_{\mathcal{C}}(\phi \rightarrow \mathsf{EK}_{\mathcal{C}}(\phi)) \rightarrow (\phi \rightarrow \mathsf{CK}_{\mathcal{C}}(\phi))$.

For details see [27, Section 7.1].

Note that depending on the properties of the employed communication lines, common knowledge may have to be pre-established, i.e., off those lines [15].

Fact 3 (Knowledge versus belief) *For all* $\mathcal{C} \subseteq \mathcal{A}$*,* $\models \mathsf{CK}_{\mathcal{C}}(\phi) \rightarrow \mathsf{CB}_{\mathcal{C}}(\phi)$*. In particular when* $\mathcal{C} = \{a\}$*,* $\models \mathsf{K}_a(\phi) \rightarrow \mathsf{B}_a(\phi)$*.*

Proof. By the fact that for all $a \in \mathcal{A}$, $\mathrm{D}_a \subseteq \mathrm{E}_a$ (cf. Definition 1).

The following corollary is immediate.

Corollary 1 (Strong versus weak trust).

1. *For all* $a, b \in \mathcal{A}$*,* $\models a \ \mathsf{sTrusts}\ b \rightarrow a \ \mathsf{wTrusts}\ b$*.*
2. *For all* $\mathcal{C} \subseteq \mathcal{A}$*,* $\models \mathsf{sTD}(\mathcal{C}) \rightarrow \mathsf{wTD}(\mathcal{C})$*.*

Trust relations and trust domains can be related as follows.

Proposition 1 (Trust relations and domains). *In trust domains, trust relations are* universal *(i.e., correspond to the Cartesian product on those domains). That is, for all* $a, b \in \mathcal{C}$*,* $\models \mathsf{wTD}(\mathcal{C}) \rightarrow a \ \mathsf{wTrusts}\ b$ *and* $\models \mathsf{sTD}(\mathcal{C}) \rightarrow a \ \mathsf{sTrusts}\ b$*.*

Proof. Almost by definition of weak and strong trust relations and domains.

Hence in trust domains, trust relations are equivalence relations. (The universal relation contains all other relations.)

Corollary 2 (Trust relations and domains). *In trust domains, trust relations are* $(a, b, c \in \mathcal{C} \subseteq \mathcal{A})$*:* reflexive *(i.e.,* $\models \mathsf{wTD}(\mathcal{C}) \rightarrow a\ \mathsf{wTrusts}\ a$ *and* $\models \mathsf{sTD}(\mathcal{C}) \rightarrow a\ \mathsf{sTrusts}\ a$*),* symmetric *(i.e.,* $\models \mathsf{wTD}(\mathcal{C}) \rightarrow (a\ \mathsf{wTrusts}\ b \rightarrow b\ \mathsf{wTrusts}\ a)$ *and* $\models \mathsf{sTD}(\mathcal{C}) \rightarrow (a\ \mathsf{sTrusts}\ b \rightarrow b\ \mathsf{sTrusts}\ a)$*), and* transitive *(i.e.,* $\models \mathsf{wTD}(\mathcal{C}) \rightarrow ((a\ \mathsf{wTrusts}\ b \wedge b\ \mathsf{wTrusts}\ c) \rightarrow a\ \mathsf{wTrusts}\ c)$ *and* $\models \mathsf{sTD}(\mathcal{C}) \rightarrow ((a\ \mathsf{sTrusts}\ b \wedge b\ \mathsf{sTrusts}\ c) \rightarrow a\ \mathsf{sTrusts}\ c)$*).*

A more interesting condition for the transitivity of trust relations than their universality is knowledge (which is implied in strong trust domains by common knowledge) in the following sense.

Lemma 1 (Transitivity Lemma).

1. $\models \mathsf{B}_a(b\ \mathsf{sTrusts}\ c) \rightarrow a\ \mathsf{wTrusts}\ c$
2. $\models \mathsf{K}_a(b\ \mathsf{sTrusts}\ c) \rightarrow a\ \mathsf{sTrusts}\ c.$

Proof. The first validity follows from $\mathbf{T}(\mathsf{K}_b)$ and $\models (\mathsf{B}_a(\phi) \wedge (\phi \rightarrow \phi')) \rightarrow \mathsf{B}_a(\phi')$, and the second from $\mathbf{T}(\mathsf{K}_b)$ and $\models (\mathsf{K}_a(\phi) \wedge (\phi \rightarrow \phi')) \rightarrow \mathsf{K}_a(\phi')$.

Note that b acts as a *reference* of c's trustworthiness to a. This is an important concept for applications. An example is the construction of transitive *trust paths,* e.g., of length 3: $\models \mathsf{B}_a(b\ \mathsf{sTrusts}\ c) \rightarrow ((a\ \mathsf{wTrusts}\ b \wedge b\ \mathsf{wTrusts}\ c) \rightarrow a\ \mathsf{wTrusts}\ c)$ and $\models \mathsf{K}_a(b\ \mathsf{sTrusts}\ c) \rightarrow ((a\ \mathsf{sTrusts}\ b \wedge b\ \mathsf{sTrusts}\ c) \rightarrow a\ \mathsf{sTrusts}\ c)$, respectively.

Fact 4 1. $\not\models \mathsf{B}_a(b\ \mathsf{wTrusts}\ c) \rightarrow a\ \mathsf{wTrusts}\ c$
2. $\not\models \mathsf{K}_a(b\ \mathsf{wTrusts}\ c) \rightarrow a\ \mathsf{wTrusts}\ c.$

Proof. By the absence of the **T**-law for B_b.

Here are some simple facts about trust domains.

Proposition 2 (Trust domains).

0. $\models \mathsf{wTD}(\emptyset)$ *and* $\models \mathsf{sTD}(\emptyset)$
1. *Separating a trust domain:*
   $\models \mathsf{wTD}(\mathcal{C} \cup \mathcal{C}') \rightarrow (\mathsf{wTD}(\mathcal{C}) \wedge \mathsf{wTD}(\mathcal{C}'))$
   $\models \mathsf{sTD}(\mathcal{C} \cup \mathcal{C}') \rightarrow (\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}'))$
2. $\models (\mathsf{wTD}(\mathcal{C}) \wedge \mathsf{wTD}(\mathcal{C}')) \rightarrow \mathsf{wTD}(\mathcal{C} \cap \mathcal{C}')$
   $\models (\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}')) \rightarrow \mathsf{sTD}(\mathcal{C} \cap \mathcal{C}')$
3. *if* $\mathcal{C} \subseteq \mathcal{C}'$ *then* $\models \mathsf{wTD}(\mathcal{C}') \rightarrow \mathsf{wTD}(\mathcal{C})$ *and* $\models \mathsf{sTD}(\mathcal{C}') \rightarrow \mathsf{sTD}(\mathcal{C})$.

Proof. Straightforward from definitions.

Now consider a more complex fact about (actually an insight into) trust domains.

Theorem 1 (Merging strong trust domains). *Merging two strong trust domains is compositional in the sense that a necessary and sufficient condition for merging two strong trust domains is that it be common knowledge in the union of both domains that each domain is a strong trust domain. Formally, for all* $\mathcal{C}, \mathcal{C}' \subseteq \mathcal{A}$*,*

$$\models \mathsf{CK}_{\mathcal{C} \cup \mathcal{C}'}(\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}')) \leftrightarrow \mathsf{sTD}(\mathcal{C} \cup \mathcal{C}').$$

Proof. See Appendix A.

Note that there is no analogous result for weak trust domains due to Fact 4. That is, merging *weak* trust domains is *non*-compositional in the sense that the result of merging two weak trust domains is not necessarily again a weak trust domain — not even when there is common knowledge (instead of only common belief) in the union of both domains that each domain is a weak trust domain (cf. Fact 4.2). The theorem yields a simple, though computationally intrinsically costly *design pattern* for recursively building up strong trust domains (cf. Appendix B). Again, due to the absence of an analogous theorem for weak trust domains, there is no analogous design pattern either. Hence, there really is a strong practical interest in strong trust domains. As a matter of fact, this practical interest is even stronger because checking membership in strong trust domains is computationally no more complex (up to a constant) than checking membership in weak trust domains (cf. Section 3).

We continue to define *potential* trust between two agents a, called *potential truster*, and b, called *potential trustee*, and within communities $\mathcal{C}$. The idea is to define *potentiality* as satis*fiability*.

Definition 3 (Potential trust).

- *There is a potential weak (strong) trust relationship between a and b in the system S :iff a* $\mathsf{wTrusts}$ *b (a* $\mathsf{sTrusts}$ *b) is satisfiable in the model $(\mathfrak{S}, \mathcal{V})$ induced by S.*
- *The community $\mathcal{C}$ is a potential weak (strong) trust domain in the system S :iff* $\mathsf{wTD}(\mathcal{C})$ *(*$\mathsf{sTD}(\mathcal{C})$*) is satisfiable in the model $(\mathfrak{S}, \mathcal{V})$ induced by S.*

Similarly, we define *actual* trust between two agents a, called *truster*, and b, called *trustee*, and within communities $\mathcal{C}$. The idea is to define (two degrees of) *actuality* as (two degrees of) satis*faction*.

Definition 4 (Actual trust).

- *There is a weak (strong) trust relationship between a and b in the model $(\mathfrak{S}, \mathcal{V})$ at the state $s \in \mathcal{S}$:iff a* $\mathsf{wTrusts}$ *b (a* $\mathsf{sTrusts}$ *b) is satisfied in $(\mathfrak{S}, \mathcal{V})$ at s.*
- *There is a weak (strong) trust relationship between a and b in the model $(\mathfrak{S}, \mathcal{V})$:iff a* $\mathsf{wTrusts}$ *b (a* $\mathsf{sTrusts}$ *b) is globally satisfied in $(\mathfrak{S}, \mathcal{V})$.*
- *The community $\mathcal{C}$ is a weak (strong) trust domain in the model $(\mathfrak{S}, \mathcal{V})$ at the state $s \in \mathcal{S}$:iff* $\mathsf{wTD}(\mathcal{C})$ *(*$\mathsf{sTD}(\mathcal{C})$*) is satisfied in $(\mathfrak{S}, \mathcal{V})$ at s.*
- *The community $\mathcal{C}$ is a weak (strong) trust domain in the model $(\mathfrak{S}, \mathcal{V})$:iff* $\mathsf{wTD}(\mathcal{C})$ *(*$\mathsf{sTD}(\mathcal{C})$*) is globally satisfied in $(\mathfrak{S}, \mathcal{V})$.*

Since satisfaction implies satisfiability, but not vice versa, actual trust implies potential trust, but not vice versa. For example, if two agents do not even know each other then they can not be in an actual trust relationship. However, they may be in a potential trust relationship: maybe in another system state, their trust potential can become actualised. On the other hand, in a given system two agents may well know each other but not be in a potential trust relationship: the system may be designed so that trust between them is impossible — in any system state.

Table 3. Computational time complexities

		Trust relations		**Trust domains**	
	degree	*weak* a wTrusts b	*strong* a sTrusts b	*weak* wTD($\mathcal{C}$)	*strong* sTD($\mathcal{C}$)
actual	local satisfaction in models $(\mathfrak{S}, \mathcal{V})$ and states s	$\mathcal{O}(f_S(\lvert s\rvert))$		$\mathcal{O}(2^{\lvert\mathcal{C}\rvert \cdot f_S(\lvert s\rvert)})$	
	global satisfaction in models $(\mathfrak{S}, \mathcal{V})$				
potential	satisfiability in models $(\mathfrak{S}, \mathcal{V})$				

Recall that there is a universal quantification over states s in the definition of global satisfaction, and an existential quantification over states s in the definitions of satisfiability in models.

3 Complexity results

We obtain our complexity results for deciding trust relationships and membership in trust domains by reduction to known results for the complexity of common belief and knowledge (cf. [17] and [16]). As usual for logics, the valuation function (here $\mathcal{V}$) acts as an oracle, which is assumed to decide in a single step whether or not an atomic proposition is true at the current state in the model induced by the considered distributed system S. However, deciding agent correctness is not trivial and depends on S. For example, in our applications (cf. Appendix C), deciding agent correctness can be at least polynomial in the size of the current state, which depending on the system modelling, may contain the history of system events. Another example is the case study in [24], where deterministically deciding agent correctness is quadratic in the size of the current state (containing the history of system events). The notion of state size really is system-specific. For example, when states contain the history of system events, state size can be defined as history length. Hence, we may have to account for the complexity of deciding agent correctness in the complexity of deciding trust relationships and membership in trust domains.

So, suppose that the truth of each atomic proposition can be deterministically decided in a polynomial number $f_S(|s|)$ of steps in the size $|s|$ of the state s of the model $(\mathfrak{S}, \mathcal{V})$ induced by S. We recall that since potential trust is defined in terms of satisfiability, and actual trust in terms of satisfaction, and since decidability of satisfiability implies decidability of satisfaction, decidability of potential trust implies decidability of actual trust, and complexities of potential trust are upper bounds for complexities of actual trust. Furthermore, satisfiability and validity are inter-solvable (ϕ is valid iff $\neg\phi$ is not satisfiable), and satisfiability complexities yield satisfaction (model-checking) complexities.

Theorem 2. *The computational time complexities of deterministically deciding trust relations is $\mathcal{O}(f_S(|s|))$ for potential and actual, weak and strong trust (cf. Table 3).*

Proof. Notice that our definitions of trust relations refer to a finite number (i.e., $|\mathcal{P}|$) of atomic propositions P, and that each definition uses exactly one atomic proposition (e.g., $\mathsf{correct}(b)$) and exactly one modal operator (e.g., B_a or K_a). Now according to [17], the complexity of the satisfiability of formulae $\mathsf{B}_a(\phi)$ and $\mathsf{K}_a(\phi)$ in a language with a finite number of atomic propositions and a

bounded nesting depth of modal operators B_a and K_a is in (oracle) linear time in the length (here constantly 1) of the formula. Hence, the complexity of the satisfiability of formulae expressing weak and strong trust relations is even in (oracle) constant time, and thus $\mathcal{O}(f_S(|s|))$ without oracle. Yet $\mathcal{O}(f_S(|s|))$ is an absolute lower bound and thus the complexity of *all* trust relationships.

We can learn at least two lessons from these results. The first lesson is that we do have to account for the complexity of deciding agent correctness in the complexity of deciding trust relationships. The second lesson is that, surprisingly, deciding agent correctness is, up to a constant, equionerous to deciding potential and actual as well as weak and strong trust relationships.

Theorem 3. *The computational time complexities of deterministically deciding trust domains is* $\mathcal{O}(2^{|\mathcal{C}|\cdot f_S(|s|)})$ *for potential and actual, weak and strong trust (cf. Table 3).*

Proof. According to [16], the complexity of the satisfiability of formulae $\mathsf{CB}_{\mathcal{C}}(\phi)$ and $\mathsf{CK}_{\mathcal{C}}(\phi)$ is in (oracle) deterministic single exponential time in the length of the sub-formula ϕ. The intuition is that formulae $\mathsf{CB}_{\mathcal{C}}(\phi)$ and $\mathsf{CK}_{\mathcal{C}}(\phi)$ correspond to formulae of infinitely deeply nested operators B_a and K_a with $a \in \mathcal{C}$, respectively, and that in that case, a finite number of atomic propositions does not help. In our case, the length of the conjunctive sub-formula in $\mathsf{wTD}(\mathcal{C})$ and $\mathsf{sTD}(\mathcal{C})$ is polynomial in the size $|\mathcal{C}|$ of the community $\mathcal{C}$. Further, the complexity of each conjunct is $\mathcal{O}(f_S(|s|))$, which we assumed to be polynomial. Yet a single exponential of a polynomial cost is still "only" a single exponential cost. Hence, the complexity of the satisfiability of formulae $\mathsf{wTD}(\mathcal{C})$ and $\mathsf{sTD}(\mathcal{C})$ is deterministic single exponential time in the size of $\mathcal{C}$ times $f_S(|s|)$. That is, the complexity of membership in potential weak and strong trust domains is $\mathcal{O}(2^{|\mathcal{C}|\cdot f_S(|s|)})$. Finally according to [16], the operators $\mathsf{CB}_{\mathcal{C}}$ and $\mathsf{CK}_{\mathcal{C}}$ force satisfying models of a size that is exponential in $|\mathcal{C}|$. Hence, $\mathcal{O}(2^{|\mathcal{C}|\cdot f_S(|s|)})$ is the complexity of *all*, potential or actual, memberships problems.

4 Related work

There is a huge literature on notions of trust that are not formal in the sense of formal languages and semantics, and also on trust management, which however is not the subject matter of this paper. We are aware of the following formal work related to ours.

4.1 Trust relations

As explained in the introduction, [11] presents various kinds of trust relations based on belief in and knowledge of mental attitudes, which is less general than belief in or knowledge of agent behaviour.

In [39], trust relationships are defined as four-tuples of a set R of trusters, a set E of trustees, a set C of conditions, and a set P of properties (the actions or

attributes of the trustees). Thereby, conditions and properties are fully abstract, i.e., without pre-determined form. According to the authors, "Trust relationship T means that under the condition set C, truster set R trust that trustee set E have the properties in set P.", where the meaning of "trust that" is formally primitive and thus left to interpretation. Given that trust can possibly be wrong, a plausible interpretation of "trust that" could be "believe that" (rather than "know that"). The authors then define operations on trust relationships in terms of set-theoretic operations on the projections of their tuples. We attempt to relate the authors' notion of trust relationship to our notion of weak trust relation (based on belief rather than knowledge) by coercing their definition of T in the following macro-definition of our logic, assuming their P is included in our $\mathcal{P}$:

$$T(C,R,E,P) := (\bigwedge_{c\in C} c) \rightarrow \bigwedge_{r\in R} \mathsf{B}_r(\bigwedge_{e\in E}\bigwedge_{p\in P} p(e)).$$

If the authors agree with this coercion, we have the following definability result:

$$\models T(\{\top\},\{a\},\{b\},\{\mathsf{correct}\}) \leftrightarrow a \ \mathsf{wTrusts}\ b,$$

where $\mathsf{correct}$ is an attribute in the authors' sense.

In [18], so-called *trust in belief* and *trust in performance* are formalised in the situation calculus, such that "axioms and theorems can be used to infer whether a thing can be believed". Whereas we define (weak) trust *in agents*, and moreover *in terms of belief* taken off-the-shelf as a standard primitive. In [19], the ideas of trust in belief and trust in performance are taken up again similarly.

In [36], "trust is a state at which the host believes, expects, or accepts that the effects from the cleint are the positive", although belief is not formal in the sense of modal logic. The authors' idea is that "the host's trust on a client is obtained based on the trust evaluation of the client by the host. When the host trusts a client, the host will believe, expect or accept that the client will do no harm to the host in the given context". Hence, this notion of trust is agent-centric in the sense of being defined in terms of (local) effects at an agent's location. This is a less general notion of trust than ours, which is *systemic* in the sense of being defined in terms of correct agent behaviour within a certain system. Also, we recall again that agent correctness is a standard primitive in the distributed-systems community [26].

In [9], a domain-theoretic model of trust relations between agents is presented. In that model, a given directed trust relation from a truster to a trustee is abstracted as a value reflecting the truster's degree of trust in the trustee. Thus, [9]'s notion of trust is, as opposed to ours, quantitative, but, as opposed to ours, lacks a behavioural (e.g., in terms of agent correctness) and doxastic/epistemic explication (in terms of belief/knowledge). The purpose of the model is the definition of a unique global trust state arising from the trust relations between the agents via trust policies. Complexities for computing portions of that global state are given in [25].

4.2 Trust domains

To the best of our knowledge, the only formal piece of work on trust domains is [38], based on description logic. However, the authors' definition is a conceptual modelling of trust domains limited to PKIs.

5 Conclusion

Assessment We have provided simple, smooth definitions and complexity results for multi-agent trust in a single, standard framework. More precisely, we have delivered definitions for weak and strong trust relations and trust domains, as well as potential and actual trust relationships and membership in trust domains. All our definitions have the advantage of being declarative *and* computational, as well as being parametric in the notion of agent correctness. Thanks to being declarative, our definitions are independent of the manifold manifestations of trust establishment (e.g., via recommendations and/or reputation). They are meaningful in any concrete distributed system with a notion of agent correctness and state space. We recall that agent correctness is a primitive in the distributed-systems community [26], and that state space is forced in a world of digital computers. A surprising insight gained from our computational analysis of trust is that given weak trust, strong trust is for free (up to a constant) from the point of view of complexity theory. Finally, we have shown that our trust domains are fit as such for TTPs and the Web of Trust, and that with a minor modification in the form of a constraint, they can be made to fit PKIs, ID-Based Cryptography, and others.

Future work We could unify our notions of weak and strong trust relation (trust domain) in a notion of *graded trust relation* (*graded trust domain*) defined in terms of graded (common) belief instead of plain (common) belief and plain (common) knowledge, respectively [23]. Informally, knowledge is belief with 100% certitude. With the additional introduction of *temporal* modalities, it then becomes possible to study the evolution of the quality and quantity of trust in a given distributed system, by observing the evolution of the grade of each trust relation and trust domain in the system. Finally, we could build actual trust-management systems for trust relations and trust domains in our present sense of building trust *from absence of trust* and in a future sense of *re*building trust from *dis*trust.

Acknowledgements The first author thanks Jean-Luc Beuchat, Akira Kanaoka, Kanta Matsuura and the members of his laboratory, and Andrey Rybalchenko for stimulating discussions.

References

1. Cloud computing. `http://csrc.nist.gov/groups/SNS/cloud-computing/index.html`.

2. R. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*. Wiley, second edition, 2008.
3. R. Anderson, B. Crispo, J.-H. Lee, Ch. Manifavas, V. Matyas, and F. Petitcolas. *The Global Internet Trust Register*. The MIT Press, 1999.
4. C. Areces and B. ten Cate. *Handbook of Modal Logic*, chapter Hybrid Logics. Volume 3 of Blackburn et al. [6], 2007.
5. P. Blackburn and J. van Benthem. *Handbook of Modal Logic*, chapter Modal Logic: A Semantic Perspective. Volume 3 of Blackburn et al. [6], 2007.
6. P. Blackburn, J. van Benthem, and F. Wolter, editors. *Handbook of Modal Logic*, volume 3 of *Studies in Logic and Practical Reasoning*. Elsevier, 2007.
7. C. Boyd and A. Mathuria. *Protocols for Authentication and Key Establishment*. Springer, 2003.
8. J. Bradfield and C. Stirling. *Handbook of Modal Logic*, chapter Modal Mu-Calculi. Volume 3 of Blackburn et al. [6], 2007.
9. M. Carbone, M. Nielsen, and V. Sassone. A formal model for trust in dynamic networks. In *Proceedings of the IEEE Conference on Software Engineering and Formal Methods*, 2003.
10. B.A. Davey and H.A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 1990 (2002).
11. R. Demolombe. Reasoning about trust: A formal logical framework. In *Proceedings of the Conference on Trust Management*, volume 2995 of *LNCS*. Springer, 2004.
12. D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(12), 1983.
13. R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning about Knowledge*. MIT Press, 1995.
14. E. Gallery and Ch.J. Mitchell. Trusted computing: Security and applications. *Cryptologia*, 33(3), 2009.
15. J.Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, 37(3), 1990.
16. J.Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54(3), 1992.
17. J.Y. Halpern and Y. Moses. The effect of bounding the number of primitive propositions and the depth of nesting on the complexity of modal logic. *Artificial Intelligence*, 75(2), 1995.
18. J. Huang and M.S. Fox. An ontology of trust — formal semantics and transitivity. In *Proceedings of the ACM Conference on Electronic Commerce*, 2006.
19. J. Huang and D. Nicol. A calculus of trust and its application to PKI and identity management. In *Proceedings of the ACM Symposium on Identity and Trust on the Internet*, 2009.
20. M. Joye and G. Neven. *Identity-Based Cryptography*. IOS Press, 2009.
21. L.M. Kaufman. Data security in the world of cloud computing. *IEEE Security & Privacy*, 7(4), 2009.
22. S. Kramer, R. Goré, and E. Okamoto. Formal definitions and complexity results for trust relations and trust domains fit for TTPs, the Web of Trust, PKIs, and ID-Based Cryptography. Logic Column of ACM SIGACT News, March 2010.
23. S. Kramer, C. Palamidessi, R. Segala, A. Turrini, and Ch. Braun. A quantitative doxastic logic for probabilistic processes and applications to information-hiding. *Journal of Applied Non-Classical Logic*, 19(4), 2009.
24. S. Kramer and A. Rybalchenko. A multi-modal framework for achieving accountability in multi-agent systems. In *Proceedings of the ESSLLI-affiliated Workshop on Logics in Security*, 2010.

25. K. Krukowa and A. Twigg. The complexity of fixed point models of trust in distributed networks. *Theoretical Computer Science*, 389(3), 2007.
26. N.A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers, 1996.
27. J.-J. Meyer and F. Veltnam. *Handbook of Modal Logic*, chapter Intelligent Agents and Common Sense Reasoning. Volume 3 of Blackburn et al. [6], 2007.
28. B. Michael. In clouds shall we trust? *IEEE Security & Privacy*, 7(5), 2009.
29. L.C. Paulson. The inductive approach to verifying cryptographic protocols. *Journal of Computer Security*, 6(1), 1998.
30. S. Ruohomaa and L. Kutvonen. Trust management survey. In *Proceedings of the Conference on Trust Management*, volume 3477 of *LNCS*. Springer, 2005.
31. A. Jøsang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2), 2007.
32. B. Schneier. Security, group size, and the human brain. *IEEE Security & Privacy*, 7(4), 2009.
33. A. Tiu and R. Goré. A proof theoretic analysis of intruder theories. volume 5595 of *Lecture Notes in Computer Science*. Springer, 2009.
34. H. van Ditmarsch, W. van der Hoek, and B. Kooi. *Dynamic Epistemic Logic*, volume 337 of *Synthese Library*. Springer, 2007.
35. H.C.A. van Tilborg, editor. *Encyclopedia of Cryptography and Security*, pages 398–400. Springer, 2005.
36. D. Xiu and Z. Liu. A formal definition for trust in distributed systems. In *Proceedings of the Information Security Conference*, volume 3650 of *LNCS*. Springer, 2005.
37. A. Yao. Protocols for secure computations. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, 1982.
38. H. Yu, Ch. Jin, and H. Che. A description logic for PKI trust domain modeling. In *Proceedings of the IEEE Conference on Information Technology and Applications*, 2005.
39. W. Zhao, V. Varadharajan, and G. Bryan. Analysis and modelling of trust in distributed information systems. In *Proceedings of the Conference on Information Systems Security*, volume 3803 of *LNCS*. Springer, 2005.
40. Ph. Zimmermann. `http://www.philzimmermann.com/`.

A A formal proof

We recall the laws $\models (\mathsf{CK}_{\mathcal{C}}(\phi) \wedge (\phi \to \phi')) \to \mathsf{CK}_{\mathcal{C}}(\phi')$, $\models (\mathsf{CK}_{\mathcal{C}}(\phi) \wedge \mathsf{CK}_{\mathcal{C}}(\phi')) \leftrightarrow \mathsf{CK}_{\mathcal{C}}(\phi \wedge \phi')$, and $\models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\phi) \to (\mathsf{CK}_{\mathcal{C}}(\phi) \wedge \mathsf{CK}_{\mathcal{C}'}(\phi))$; and call them $\mathbf{L1}(\mathsf{CK}_{\mathcal{C}})$, $\mathbf{L2}(\mathsf{CK}_{\mathcal{C}})$, and $\mathbf{L3}(\mathsf{CK}_{\mathcal{C}})$, respectively. Now, let $\mathfrak{S}$ designate an arbitrary frame for our logic, and $\mathcal{V}$ an arbitrary valuation function for agent correctness. Then:

1	$s \in \mathcal{S}$	hyp.
2	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}'))$	hyp.
3	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C})) \wedge \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C}'))$	2, $\mathbf{L2}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
4	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}}(\mathsf{sTD}(\mathcal{C}))$ and $(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C}'))$	3, def.
5	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{sTD}(\mathcal{C}) \to \bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b$	$\mathbf{T}(\mathsf{CK}_{\mathcal{C}})$
6	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{sTD}(\mathcal{C}') \to \bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b$	$\mathbf{T}(\mathsf{CK}_{\mathcal{C}'})$
7	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b)$	4, 5 $\mathbf{L1}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
8	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b)$	4, 6 $\mathbf{L1}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
9	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b)$ and $(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b)$	7, 8
10	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b) \wedge \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b)$	9, def.
11	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}\left(\begin{smallmatrix}\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b \wedge \\ \bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b\end{smallmatrix}\right)$	10, $\mathbf{L2}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
12	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}\cup\mathcal{C}'} a\ \mathsf{sTrusts}\ b)$	11, Lemma 1.2
13	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{sTD}(\mathcal{C} \cup \mathcal{C}')$	12, def.
14	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{sTD}(\mathcal{C} \cup \mathcal{C}')$	hyp.
15	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}\cup\mathcal{C}'} a\ \mathsf{sTrusts}\ b)$	14, def.
16	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}\left(\begin{smallmatrix}\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b \wedge \\ \bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b\end{smallmatrix}\right)$	15
17	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}\left(\begin{smallmatrix}\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b \wedge \\ \bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b\end{smallmatrix}\right))$	16, $\mathbf{4}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
18	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b) \wedge \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b)$	16, $\mathbf{L2}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
19	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}}(\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b) \wedge \mathsf{CK}_{\mathcal{C}'}(\bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b)$	18, $\mathbf{L3}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
20	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}')$	19, def.
21	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}\left(\begin{smallmatrix}\bigwedge_{a,b\in\mathcal{C}} a\ \mathsf{sTrusts}\ b \wedge \\ \bigwedge_{a,b\in\mathcal{C}'} a\ \mathsf{sTrusts}\ b\end{smallmatrix}\right) \to (\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}'))$	20
22	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}'))$	17, 21, $\mathbf{L1}(\mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'})$
23	$(\mathfrak{S}, \mathcal{V}), s \models \mathsf{sTD}(\mathcal{C} \cup \mathcal{C}')$ if and only if $(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}'))$	2–13, 14–22

24 for all $s \in \mathcal{S}$, $(\mathfrak{S}, \mathcal{V}), s \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}')) \leftrightarrow \mathsf{sTD}(\mathcal{C} \cup \mathcal{C}')$ 1–23
25 $(\mathfrak{S}, \mathcal{V}) \models \mathsf{CK}_{\mathcal{C}\cup\mathcal{C}'}(\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}')) \leftrightarrow \mathsf{sTD}(\mathcal{C} \cup \mathcal{C}')$ 24

B Building trust

Building trust in the sense of building *from absence of trust* (as opposed to *re*building from *dis*trust, cf. 2nd paragraph after Definition 1) is possible if and only if there is at least *potential* trust in the sense of Definition 3. That is, given $a, b \in \mathcal{A}$ and $\mathcal{C} \subseteq \mathcal{A}$, the formulae a $\mathsf{wTrusts}$ b or a $\mathsf{sTrusts}$ b, and $\mathsf{wTD}(\mathcal{C})$ or $\mathsf{sTD}(\mathcal{C})$ must at least be *satisfiable in the model induced by the considered system*, in order for a weak or strong trust relationship from a to b to possibly exist, and in order for $\mathcal{C}$ to possibly be a weak or strong trust domain, respectively. Yet thanks to the computability of our notions of trust, a computer can aid us in our decision of whether or not building trust from a current absence of trust in a given system, and between a given pair of agents, or within a given (small) group of agents is actually possible.

When it is indeed possible to actualise a certain potential trust, the next question is how to actually *build up* the trust. A potential trustee has at least two non-mutally-exclusive possibilities of earning the trust of a potential truster:

1. by *behaving correctly* according to the notion of agent correctness (cf. $\mathsf{correct}$) of the system where the computer has found the considered kind of trust to be actually possible. (Behaving correctly can include doing nothing, when the considered notion of agent correctness does not exclude such inactivity.)
2. by *producing evidence* (e.g., a recommendation) for *or proof* (e.g., a signed log file) of behavioural correctness, whenever requested to do so by the potential truster (here in the role of an auditor).

The potential truster may then decide to trust the potential trustee based on the observation of the potential trustee's behaviour, and/or based on the knowledge of certain data, i.e., evidence or even proof produced by the potential trustee. Depending on the value of the information obtained, a relationship of weak or strong trust is established from the potential truster to the potential trustee, who have now become *de facto* an actual truster and trustee, respectively. This process of building up oriented trust relations can be extended to building up symmetric trust relationships, and trust domains of agents in such relationships. As a matter of fact, Theorem 1 yields the following *design pattern*, called RD, for building up via recursive descent strong trust domains from a possible absence of trust in a given system. RD relies on the communication abstraction of *public announcement* (cf. 3rd paragraph after Definition 1), which we use as a black-box plug-in. We recall that the effect of a public announcement of a fact is to induce the common knowledge of that fact with the addressed public by minimally changing the epistemic state of each agent in that public (cf. [34] for details). Depending on the communication context, the mechanism ranges from trivial (e.g., in a public assembly) to difficult (e.g., with communication asynchrony), possibly requiring implementation as a full-fledged communication protocol. So,

although 'announcement' may suggest triviality rather than difficulty, public announcements may well be non-trivial to implement in a given context, which is why we call RD *design pattern* rather than *algorithm*:

1. **Input:** a model $(\mathfrak{S}, \mathcal{V})$, a state s, and $\mathcal{C}_1, \mathcal{C}_2 \in \mathcal{A}$;
2. **Divide:** for $i \in \{1, 2\}$ do {
   when $(\mathfrak{S}, \mathcal{V}), s \models \neg\mathsf{sTD}(\mathcal{C}_i)$:
   (a) divide $\mathcal{C}_i$ freely into $\mathcal{C}_{i.1}$ and $\mathcal{C}_{i.2}$;
   (b) $s := \mathrm{RD}((\mathfrak{S}, \mathcal{V}), s, \mathcal{C}_{i.1}, \mathcal{C}_{i.2})$; };
3. **Conquer:** announce to the community $\mathcal{C}_1 \cup \mathcal{C}_2$ that $\mathsf{sTD}(\mathcal{C}_1) \wedge \mathsf{sTD}(\mathcal{C}_2)$ is true (choose appropriate communication channels and an appropriate protocol), which takes the system from s to some $s' \in \mathcal{S}$ such that s' is reachable (cf. Footnote 7) from s and

$$(\mathfrak{S}, \mathcal{V}), s' \models \mathsf{CK}_{\mathcal{C} \cup \mathcal{C}'}(\mathsf{sTD}(\mathcal{C}) \wedge \mathsf{sTD}(\mathcal{C}'));$$

4. **Output:** $s' \in \mathcal{S}$;
5. **Effect:** $(\mathfrak{S}, \mathcal{V}), s' \models \mathsf{sTD}(\mathcal{C}_1 \cup \mathcal{C}_2)$.

Building up trust domains from possible absence of trust generally is computationally costly, in particular when made by means of the recursive-descent design pattern RD.

Theorem 4. *The complexity of building up trust domains is exponential in the number of potential members.*

Proof. By the fact that membership in trust domains has to be checked for each potential new member anew, which is exponential in the size of the trust domain being checked in a given model and at a given state (cf. Table 3).

In contrast, it is common knowledge (among humans) that for *destroying* actual trust relationships, and thus also trust domains, a single (side) step is sufficient — metaphorically speaking. And *re*building trust from *dis*trust is difficult.

C Application to cryptographic-key management

We instantiate our concepts of trust relations and trust domains in four major applications of trust, namely Trusted Third Parties (TTPs), the Web of Trust, Public-Key Infrastructures (PKIs), and ID-Based Cryptography. For the latter three, we will have to define the valuation function $\mathcal{V}$ on the atomic propositions $\mathsf{correct}(a)$ about agent correctness (cf. Definition 1 and Table 2). (We will also have to refine the definition of trust domains in Table 2 for the latter two.) That is, ***each notion of agent correctness is specific***[10] ***to each system rather than general to all systems.*** (Thus, ***trust is system-specific to some extent.***) However, we can define agent correctness *generically* for the Web of Trust and PKIs, by means of the following, common auxiliary logic,

[10] like policies, which induce notions of agent correctness, as mentioned in Section 1.1

called AuxLog. The logic is a modal fixpoint logic [8] operating on points that are agents $a \in \mathcal{A}$ rather than states $s \in \mathcal{S}$. AuxLog is *parametric* in a binary relation $R \subseteq \mathcal{A} \times \mathcal{A}$ to be fixed separately for the Web of Trust and PKIs, but with the commonality of depending on a fixed state s ($R \in \{\mathrm{DTI}_s, \mathrm{CERT}_s\}$).

Definition 5 (Auxiliary Logic). *Let $\mathcal{X}$ designate a countable set of propositional variables C, and let*

$$\mathcal{L}' \ni \alpha ::= \mathsf{OK} \mid C \mid \neg\alpha \mid \alpha \wedge \alpha \mid \overline{\Box}\alpha \mid \boldsymbol{\nu} C(\alpha)$$

designate the language $\mathcal{L}'$ of AuxLog where all free occurrences of C in α of $\boldsymbol{\nu} C(\alpha)$ are assumed to occur within an even number of occurrences of $\neg$ to guarantee the existence of (greatest) fixpoints (expressed by $\boldsymbol{\nu} C(\alpha)$) [8]. Then, given a relation $R \subseteq \mathcal{A} \times \mathcal{A}$ decidable in deterministic constant time but structurally arbitrary, and an auxiliary interpretation $[\![\cdot]\!] : \mathcal{X} \cup \{\mathsf{OK}\} \to 2^{\mathcal{A}}$ partially pre-defined as

$$[\![\mathsf{OK}]\!] := \{\ a \in \mathcal{A} \mid \textit{at most } a \textit{ can access } a\textit{'s private key}\ \}^{11},$$

the interpretation $\|\cdot\|_{[\![\cdot]\!]} : \mathcal{L}' \to 2^{\mathcal{A}}$ of AuxLog-propositions is as follows:

$$\begin{aligned}
\|C\|_{[\![\cdot]\!]} &:= [\![C]\!] \\
\|\mathsf{OK}\|_{[\![\cdot]\!]} &:= [\![\mathsf{OK}]\!] \\
\|\neg\alpha\|_{[\![\cdot]\!]} &:= \mathcal{A} \setminus \|\alpha\|_{[\![\cdot]\!]} \\
\|\alpha \wedge \alpha'\|_{[\![\cdot]\!]} &:= \|\alpha\|_{[\![\cdot]\!]} \cap \|\alpha'\|_{[\![\cdot]\!]} \\
\|\overline{\Box}\alpha\|_{[\![\cdot]\!]} &:= \{\ a \in \mathcal{A} \mid \textit{for all } b \in \mathcal{A}, \textit{ if } b\ R\ a \textit{ then } b \in \|\alpha\|_{[\![\cdot]\!]}\ \} \\
\|\boldsymbol{\nu} C(\alpha)\|_{[\![\cdot]\!]} &:= \bigcup\{\ A \subseteq \mathcal{A} \mid A \subseteq \|\alpha\|_{[\![\cdot]\!]_{[C \mapsto A]}}\ \},
\end{aligned}$$

where $[\![\cdot]\!]_{[C \mapsto A]}$ maps C to A and otherwise agrees with $[\![\cdot]\!]$.

Further, $\alpha \vee \alpha' := \neg(\neg\alpha \wedge \neg\alpha')$, $\top := \alpha \vee \neg\alpha$, $\bot := \neg\top$, $\alpha \to \alpha' := \neg\alpha \vee \alpha'$, $\alpha \leftrightarrow \alpha' := (\alpha \to \alpha') \wedge (\alpha' \to \alpha)$, $\overline{\Diamond}\alpha := \neg\overline{\Box}(\neg\alpha)$, and, notably, $\boldsymbol{\mu} C(\alpha(C)) := \neg\boldsymbol{\nu} C(\neg\alpha(\neg C))$.

Finally, for all $a \in \mathcal{A}$ and $\alpha \in \mathcal{L}'$,

$$\langle (\mathcal{A}, R), [\![\cdot]\!] \rangle, a \vDash \alpha \quad \textit{:iff} \quad a \in \|\alpha\|_{[\![\cdot]\!]},$$

and for all $\alpha \in \mathcal{L}'$,

$$\vDash \alpha \quad \textit{:iff} \quad \textit{for all } [\![\cdot]\!] \textit{ and } a \in \mathcal{A},\ \langle (\mathcal{A}, R), [\![\cdot]\!] \rangle, a \vDash \alpha.$$

[11] This phrasing can be made formal provided that a notion of data space $\mathcal{D}$ (including keys) and data derivation for agents is fixed, e.g., à la Dolev-Yao [12]. Then data derivation can be formalised as a relation $\vdash \subseteq 2^{\mathcal{D}} \times \mathcal{D}$ (the first formalisation was in terms of closure operators [29]), and the phrase "at most a can access a's private key" as "for all $b \in \mathcal{A}$, if k is the private key of a and $\mathrm{m}_b(s) \vdash k$ then $b = a$", where $\mathrm{m}_b(s)$ returns the set of data that b generated or received as such in s. Note however that depending on the structure of $\mathcal{D}$, the computability of $\vdash$ may range from polynomial time to undecidability [33].

Table 4. Trustworthy Trusted Third Parties

$\mathsf{wtTTP}(c,a,b) := \mathsf{CB}_{\{a,b,c\}}(\mathsf{wTD}(\{c,a\}) \wedge \mathsf{wTD}(\{c,b\}))$	c is a weakly trustworthy TTP (wtTTP) of a and b
$\mathsf{stTTP}(c,a,b) := \mathsf{CK}_{\{a,b,c\}}(\mathsf{sTD}(\{c,a\}) \wedge \mathsf{sTD}(\{c,b\}))$	c is a strongly trustworthy TTP (stTTP) of a and b

The reader is invited not to confuse the auxiliary satisfaction relation $\vDash$ of AuxLog with $\models$, the main one from Definition 1. Further note that AuxLog is a member of the family of μ-calculi over the modal system **K**, which is characterised by the laws of propositional logic and the modal laws $\vDash \overline{\Box}(\alpha \rightarrow \alpha') \rightarrow (\overline{\Box}\alpha \rightarrow \overline{\Box}\alpha')$ and "if $\vDash \alpha$ then $\vDash \overline{\Box}\alpha$". The reason is that, as mentioned, $R \subseteq \mathcal{A} \times \mathcal{A}$ is structurally arbitrary. Hence, no more structural properties than those of the modal system **K**, i.e., none, can generally be assumed to hold for $\overline{\Box}$. As a corollary, the model-checking problem, i.e., "Given $a \in \mathcal{A}$ and $\alpha \in \mathcal{L}'$, is it the case that $\langle(\mathcal{A}, R), [\![\cdot]\!]\rangle, a \vDash \alpha$?" is decidable in deterministic polynomial time in the size of α. See [8] for details.

C.1 Trusted Third Parties

The concept of a Trusted Third Party (TTP) is a folklore concept for much of information security, e.g., for many protocols for authentication and key establishment [7], as well as for secure multiparty computation [37]. Recall that "a secure multiparty computation for function f can be viewed as an implementation of a trusted third party T, which, upon receipt of the input values $x_1, \ldots, x_n$ from parties $P_1, \ldots, P_n$, respectively, produces the output value $y = f(x_1, \ldots, x_n)$. Party T is trusted for (i) providing the correct value for y and (ii) [n]ot revealing any further information to parties $P_1, \ldots, P_n$" [35]. In our terminology of agent correctness, the conjunction of Condition (i) and (ii) informally stipulates what it means for T to be correct. Notice that the above definition of secure multiparty computation merely defines the *object* of trust (i.e., agent correctness), but not trust itself (which, in our definition, is belief or even knowledge *of* agent correctness). To the best of our knowledge, the concept of a TTP has never been formally defined, i.e., mathematically analysed into its conceptual constituents. Here, we are able to define the TTP-concept in terms of our concept of trust domains (cf. Table 4), and instantiate TTPs in the Web of Trust (cf. Section C.2) and Public-Key Infrastructures (PKIs) (cf. Section C.3). More precisely, we define the concepts of a weakly and a strongly *trustworthy* TTP, i.e., TTPs that may or even must *deserve* the trust of their trusters—and vice versa. Note that the trustworthiness of TTPs (e.g., the certification authorities in a PKI) is absolutely crucial, without which whole security architectures (e.g., a PKI) can break down. Observe that thanks to Theorem 1, the two sides (e.g., $\{c,a\}$ and $\{c,b\}$) in a strongly (but not in a weakly) trustworthy TTP constitute a (strong) trust domain as a whole (i.e., as $\{c,a\} \cup \{c,b\}$).

C.2 The Web of Trust

In the (decentralised) Web of Trust, as defined by Philip Zimmermann [40] in 1992, any agent can independently establish its own domain of trusted correspondents by publicly designating (e.g., on their homepage) so-called *trusted introducers*, who by this very act become commonly known as such. In PGP, the designation of a trusted introducer is implemented as the (publicly exportable) signing of the designated trusted introducer's public key with the designator's private key. Additional assurance can be provided by The Global Internet Trust Register [3]. The role of an agent a's trusted introducer b is to act as a *guarantor* for the trustworthiness of a, and by that, to catalyse the building up of trust relationships between a and those agents c who are only potential (not yet actual) trustees of a but who are (already) actual trustees of b. Notice the importance of distinguishing between potential and actual trust (cf. Definition 3 and 4). Thus, the more guarantors (actual trustees) an agent (as an actual truster) has, the more potential trustees the agent (as a potential truster) has. In the Web of Trust, agents are (socially speaking) trustworthy, or (technically speaking) ***correct if and only if all their designated trusted introducers are, and at most they (the correct agents) can access their (own) private key.*** (Agents with untrustworthy introducers or a corrupt private key are untrustworthy.) Notice the possible *mutuality* in this social notion of agent correctness.

We model the designated-trusted-introducer relationships between agents in system states $s \in \mathcal{S}$ with a family of relations (a kind of data base) $\mathrm{DTI}_s \subseteq \mathcal{A} \times \mathcal{A}$ such that

$$b \,\mathrm{DTI}_s\, a \text{ :iff } b \text{ is a designated trusted introducer of } a \text{ in } s.$$

The valuation function $\mathcal{V}$ on the propositions $\mathsf{correct}(a)$ can then be formally defined with the aid of AuxLog as follows:

$$\boxed{\mathcal{V}(\mathsf{correct}(a)) := \{\ s \mid \langle(\mathcal{A}, \mathrm{DTI}_s), \emptyset\rangle, a \vDash \boldsymbol{\nu} C(\mathsf{OK} \wedge \overline{\Box} C)\ \},}$$

where $\emptyset$ designates the empty auxiliary interpretation (C is bound!). The greatest-fixpoint assertion $\langle(\mathcal{A}, \mathrm{DTI}_s), \emptyset\rangle, a \vDash \boldsymbol{\nu} C(\mathsf{OK} \wedge \overline{\Box} C)$ says that a is in the greatest fixpoint of the interpretation of *the property C such that:*

> if a satisfies C (i.e., a is in the interpretation of C) then a satisfies $\mathsf{OK} \wedge \overline{\Box} C$, which in turn says that at most a can access a's private key and for all $b \in \mathcal{A}$, if b is a designated trusted introducer of a in the state s then b satisfies C.

Observe that all (=1) free occurrences of C in $\mathsf{OK} \wedge \overline{\Box} C$ of $\boldsymbol{\nu} C(\mathsf{OK} \wedge \overline{\Box} C)$ occur within an even (=0) number of occurrences of $\neg$. Hence, our definition is formally well-defined. Further observe that the possible mutuality in our notion of agent correctness corresponds to the co-inductiveness of the greatest fixpoint, which allows direct (*self-designation*) and indirect (*mutual designation*) loops in the designated-trusted-introducer relationships. The interpretation of the corresponding (inductive) least-fixpoint formula would wrongly not allow such loops.

Of course, the language of AuxLog allows for other, more complex definitions of agent correctness, e.g., ones disallowing self-designation[12], and/or ones with a more complex notion of being OK. Our present definition is just an inceptive example proposal. The *co*-inductive definition has the following iterative paraphrase from *above* (iterated *de*construction).

> Everybody is correct (the Web of Trust is born in the plenum, so to say);
> except for the following agents (*ex*clude those which are clearly not OK):
> 0. agents with a corrupt private key (Type 0 agents)
> 1. agents with a designated trusted introducer of Type 0 (Type 1 agents)
> 2. agents with a designated trusted introducer of Type 1 (Type 2 agents)
> 3. etc.

Clearly, weak or strong trust relations in the Web of Trust must be universal within an agent's domain of correspondents in the sense of Proposition 1: designated trusted introducers are trusted; and they would not act as such, if they did not trust their designator and their designator's other designated trusted introducers, etc. Hence, our trust relations and trust domains from Table 2 as well as our weakly and strongly trustworthy TTPs from Table 4 are fit for the Web of Trust without further adaptation.

C.3 Public-Key Infrastructures

In Public-Key Infrastructures (PKIs), centralised *certificate authorities* (CAs) act as guarantors for the trustworthiness of the public key of their clients by issuing certificates that bind the public key of each client (the legitimate key owner) to the client's (unique) name. In PKIs, agents are (socially speaking) trustworthy, or (technically speaking) ***correct if and only if all their certified agents are, and at most they (the correct agents) can access their (own) private key.*** (Agents who certify incorrect agents or agents with a corrupt private key are incorrect.) Notice the absence of mutuality in this notion of agent correctness; it is intrinsically unilateral. However, the notion of PKI trust to be built from this notion of agent correctness will be again bilateral (i.e., mutual, and thus symmetric): the certifying correct agent trusts the certified correct agent, and vice versa.

We model the relationships from certifying agents to certified agents (which may themselves be certifying agents to agents certified by them, etc.) in system states $s \in \mathcal{S}$ with a family of relations (a kind of data base) $\mathrm{CRT}_s \subseteq \mathcal{A} \times \mathcal{A}$ such that

$$b \; \mathrm{CRT}_s \; a \text{ :iff } b \text{ is certified by } a \text{ in } s,$$

[12] Yet whether or not you declare yourself as trustworthy may well be legally critical (cf. for example such hand-written self-declarations in certain immigration procedures).

Anyway, the disallowance of self-designation could be implemented by introducing a binding operator $\downarrow$ à la hybrid logic [4] into the language of AuxLog, such that $\langle (\mathcal{A}, R), [\![\cdot]\!] \rangle, a \vDash \downarrow C(\alpha)$:iff $\langle (\mathcal{A}, R), [\![\cdot]\!]_{[C \mapsto \{a\}]} \rangle, a \vDash \alpha$, and stipulating that $\mathcal{V}(\mathsf{correct}(a)) := \{ \; s \mid \langle (\mathcal{A}, \mathrm{DTI}_s), \emptyset \rangle, a \vDash \boldsymbol{\mu} C(\mathsf{OK} \wedge \neg \downarrow C'(\overline{\Diamond} C') \wedge \overline{\Box} C) \; \}$.

where "b is certified by a in s" means "a has issued a valid certificate for b in s", i.e., a certificate that is non-revoked in s and signed by a with the private key of a. The valuation function $\mathcal{V}$ on the propositions $\mathsf{correct}(a)$ can then be formally defined with the aid of AuxLog as follows:

$$\boxed{\mathcal{V}(\mathsf{correct}(a)) := \{\ s \mid \langle(\mathcal{A}, \mathrm{CRT}_s), \emptyset\rangle, a \vDash \boldsymbol{\mu} C(\mathsf{OK} \wedge \overline{\Box} C)\ \}.}$$

The least-fixpoint assertion $\langle(\mathcal{A}, \mathrm{CRT}_s), \emptyset\rangle, a \vDash \boldsymbol{\mu} C(\mathsf{OK} \wedge \overline{\Box} C)$ says that a is in the least fixpoint of the interpretation of *the property C such that:*

> if a satisfies $\mathsf{OK} \wedge \overline{\Box} C$ (i.e., a is in the interpretation of $\mathsf{OK} \wedge \overline{\Box} C$)—which in turn says that at most a can access a's private key and for all $b \in \mathcal{A}$, if b is certified by a in the state s then b satisfies C—then a satisfies C.

Observe that certification is unilateral (i.e., non-mutual, and thus not symmetric) in the sense that certification relationships must not be directly (*self-certification*) nor indirectly (*mutual certification*) looping, which forces a least-fixpoint formulation. The PKI-approach to trustworthiness is thus diametrically opposed to the approach of the Web of Trust. This opposition is reflected first, in the least/greatest fixpoint "duality" of the two paradigms; and second, in the fact that PKIs are based on (ultimately national) *authority* (hierarchical CAs), whereas the Web of Trust is based on (borderless) *peership* (peer-guarantors). (At the international level, it makes sense to organise the totality of national CAs as a Web of Trust.) Yet again of course, as with the Web of Trust, the language of AuxLog allows for other, more complex definitions of agent correctness, e.g., ones with self-certification for the root-CA[13] (the *trust anchor*), and/or ones with a more complex notion of being OK (e.g., including the possibility of *key escrow*). Again, our present definition is just an inceptive example proposal. The *in*ductive definition has the following iterative paraphrase from *below* (iterated *con*struction).

> Nobody is correct (PKIs are born *ex nihilo*, so to say); except for the following agents (*in*clude those which are clearly OK): agents without a corrupt private key (Type 0 agents), whose certified agents are also of Type 0 (Type 1 agents), whose certified agents are again also of Type 0 (Type 2 agents), etc. (In other words, being of Type 0 is an invariant in the transitive closure of certification relationships.)

Notice the structural difference between this paraphrase for agent correctness in PKIs and the previous one for agent correctness in the Web of Trust. The "duality" is not pure.

As suggested, CAs are commonly organised in a hierarchy, which induces *structured trust domains* in the form of finite trees. Recall that a finite tree is a partially-ordered set (here say $\langle \mathcal{C}, \leq \rangle$) with a bottom (top) element such that

[13] Self-certification for the root-CA could be implemented by introducing an atomic proposition root true at and only at the root-CA agent, and stipulating that $\mathcal{V}(\mathsf{correct}(a)) := \{\ s \mid \langle(\mathcal{A}, \mathrm{CRT}_s), \emptyset\rangle, a \vDash (\mathsf{root} \rightarrow \overline{\Diamond}\mathsf{root}) \wedge \boldsymbol{\mu} C(\mathsf{OK} \wedge \overline{\Box} C)\ \}$.

Table 5. Public-Key Infrastructure trust domains

$\mathsf{wTD}_{\mathsf{PKI}}(\mathcal{C}) := \mathsf{CB}_{\mathcal{C}}(\bigwedge_{a,\,b\,\in\,\mathcal{C}\ \text{and}\ (a \leq b\ \text{or}\ b \leq a)} a\ \mathsf{wTrusts}\ b)$	$\mathcal{C}$ is a weak PKI trust domain
$\mathsf{sTD}_{\mathsf{PKI}}(\mathcal{C}) := \mathsf{CK}_{\mathcal{C}}(\bigwedge_{a,\,b\,\in\,\mathcal{C}\ \text{and}\ (a \leq b\ \text{or}\ b \leq a)} a\ \mathsf{sTrusts}\ b)$	$\mathcal{C}$ is a strong PKI trust domain.

for each element in the set, the down-set (up-set) of the element is a finite chain [10]. In PKI trust domains, trust relations are symmetric (up- and downwards the tree branches) and transitive (along the tree branches) but not universal (after all, a tree is a tree and not felt fabric), and the root CA corresponds to the tree root, the intermediate CA's correspond to the intermediate tree nodes, and the clients to the tree leafs. Hence we can fit our weak and strong trust domains to PKI trust domains $\langle \mathcal{C}, \leq \rangle$ by simply stipulating that the conjunction in the respective definition respect the finite-tree structure $\leq$ of $\mathcal{C}$, and reflect the symmetry and transitivity of the trust relations. And that is all: see Table 5. We can now instantiate our weakly and strongly trustworthy TTPs from Table 4 for PKIs as $\mathsf{wtTTP}_{\mathsf{PKI}}(c,a,b) := \mathsf{CB}_{\{a,b,c\}}(\mathsf{wTD}_{\mathsf{PKI}}(\{c,a\}) \wedge \mathsf{wTD}_{\mathsf{PKI}}(\{c,b\}))$ and $\mathsf{stTTP}_{\mathsf{PKI}}(c,a,b) := \mathsf{CK}_{\{a,b,c\}}(\mathsf{sTD}_{\mathsf{PKI}}(\{c,a\}) \wedge \mathsf{sTD}_{\mathsf{PKI}}(\{c,b\}))$, respectively, with $\leq := \{(c,a),(c,b)\}$ as (tree) domain structure. Fits for trust domains with other structures (e.g., buses, chains, rings, stars, etc.) can be made by similarly simple stipulations (e.g., for computing clouds, which have not a fixed but a dynamic, an evolving structure).

In sum, a weak or strong PKI trust domain $\mathcal{C}$ is built from a *certification hierarchy* $\leq$ of certifying (CAs) and certifiable agents $a \in \mathcal{C}$ such that for all states $s \in \mathcal{S}$ there is a (possibly empty) *certification record* $\{\ b \in \mathcal{C} \mid b\ \mathrm{CRT}_s\ a\ \}$. Thereby, the certification hierarchy acts as a constraining skeleton (there is no such skeleton in the Web of Trust; it is unconstrained) for the potential and the actual trust relationships in $\mathcal{C}$, and, by that, the respective memberships in the trust domain $\mathcal{C}$ itself. And the certification records act as evidential support for the actuality of the trust relationships and the memberships in the trust domain.

C.4 Identity-Based Cryptography

Identity-Based Cryptography is a variation of Public-Key Cryptography in which the intending sender of a message derives the (public) encryption key from the public identity (e.g., a telephone number, an email address, etc., or a combination thereof) of the intended recipient [20]. In our setting, we abstractly model an agent a's public identity with the symbol 'a'. For the sake of the security of ID-based encryption, an ID-based private key must not be derivable from its corresponding public counterpart without an additional trap-door information. This trap-door information is owned by a central CA (cCA $\in \mathcal{A}$), which therefore can derive the private keys of all its certified agents. (Thus ID-based domains have a star structure: $\leq$ is an n-ary tree of depth 1 with as root cCA and as leaves the n agents certified by cCA.) Hence for ID-Based Cryptography, the

definition of an agent being OK (cf. Section C) must be weakened, e.g.,

$$[\![\mathsf{OK}]\!] := \{\ a \in \mathcal{A} \mid \text{at most } a \text{ and cCA can access } a\text{'s private key }\}.$$

Note that as a consequence, the notion of trust (and thus the value of the trustworthiness of cCA) is weakened! Of course, a restrengthening is possible, e.g., by stipulating that cCA has not *used* the private keys of its certified agents (except possibly for *key escrow*). In sum, the flexibility of ID-Based Cryptography for its users (the certified agents) is paid with a devaluation of the trustworthiness of its provider (cCA).