To know or not to know –
Epistemic approaches to security protocol verification

Francien Dechesne*
Philosophy section, Department of Technology, Policy and Management,
Delft University of Technology, The Netherlands

Yanjing Wang†
Department of Philosophy, Peking University, China

Abstract. Security properties naturally combine temporal aspects of protocols with aspects of knowledge of the agents. Since BAN-logic, there have been several initiatives and attempts to incorporate epistemics into the analysis of security protocols. In this paper, we give an overview of work in the field and present it in a unified perspective, with comparisons on technical subtleties that have been employed in different approaches.

Also, we study to which degree the use of epistemics is essential for the analysis of security protocols. We look for formal conditions under which knowledge modalities can bring extra expressive power to pure temporal languages. On the other hand, we discuss the cost of the epistemic operators in terms of model checking complexity.

Keywords: security protocols, dynamic epistemic logic, epistemic temporal logic, interpreted systems, verification

1. Knowledge in security protocols

Security protocols are rules (often based on cryptography) that govern communications in hostile environments in order to guarantee certain security goals. Many such goals are *naturally expressed* in terms of knowledge: (only) the right agents should get to *know* the right things. This has to do with the fact that many security properties are about hiding information from the *bad guys* or making sure the *good guys* get their information. For example, here are some intuitive epistemic readings of the security properties mentioned in (Ryan and Schneider, 2001):

- *Sender authentication*: 'the receiver *knows* the sender of a message';

- *Mutual authentication*: 'both parties (commonly) *know* they are talking to each other';

- *Anonymity*: 'the sender is *unknown* (to an eavesdropper)';

- *Secrecy*: 'an intruder does not *know* certain information'.

More specifically, in the area of voting protocols, which recently drew much attention, more involved properties are considered, for example (cf. (Delaune et al., 2009)):

- *Vote-privacy*: nobody other than the voter herself *knows* that a particular voter voted in a particular way;

- *Receipt-freeness*: No voter has any means to let another *know* for sure that she voted in a certain way.

The above list is only indicative, and by no means exhaustive to cover the security properties that have epistemic readings. Although the precise formal meaning of the security properties as above is debatable, the relevance of epistemics in such settings is undeniable (cf. also Kramer (2007) (Slogan 8): "The purpose of a cryptographic protocol is to interactively compute, via message passing, knowledge of the truth of desired and, dually, knowledge of the falsehood of undesired cryptographic states of affairs").

However, security protocols are deceptively simple-looking objects with very subtle behaviours, which require extremely precise formal analysis. Designing a correct protocol can be thought as *programming Satan's computer* as Anderson and Needham (1995) put it. Consider the 3-line Needham-Schroeder authentication protocol (Needham and Schroeder, 1978):[1]

$$
\begin{aligned}
&1.\ \mathtt{A} \rightarrow \mathtt{B}\ :\ \{\mathtt{n_A}, \mathtt{A}\}_{\mathtt{PK_B}} \\
&2.\ \mathtt{B} \rightarrow \mathtt{A}\ :\ \{\mathtt{n_A}, \mathtt{n_B}\}_{\mathtt{PK_A}} \\
&3.\ \mathtt{A} \rightarrow \mathtt{B}\ :\ \{\mathtt{n_B}\}_{\mathtt{PK_B}}
\end{aligned}
$$

which prescribes a set of action patterns with *roles* of agents to authenticate two agents with each other. BAN logic provided a correctness

---

[1] $\mathtt{A}$ generates a random number (a *nonce*), and then sends it to $\mathtt{B}$ in a "locked box" that only $\mathtt{B}$ can open with his private key. $\mathtt{B}$ then sends $\mathtt{A}$'s number back with a random number of his own, in a box that only $\mathtt{A}$ can open. $\mathtt{A}$ then confirms by sending $\mathtt{B}$ his number back. The intended goal is that both $\mathtt{A}$ and $\mathtt{B}$ know that they are talking to each other.

proof of the above protocol, which was later proven flawed due to a man-in-the-middle attack (Lowe, 1996):

$$
\begin{array}{llll}
1 & A \rightarrow I \;:\; \{n_A, A\}_{PK_I} & & \\
1' & & I(A) \rightarrow B \;:\; \{n_A, A\}_{PK_B} \\
2' & & B \rightarrow I(A) \;:\; \{n_A, n_B\}_{PK_A} \\
2 & I \rightarrow A \;:\; \{n_A, n_B\}_{PK_A} & & \\
3 & A \rightarrow I \;:\; \{n_B\}_{PK_I} & & \\
3' & & I(A) \rightarrow B \;:\; \{n_B\}_{PK_B} \\
\end{array}
$$

where A, B, I are concrete agents playing different roles according to the specification of the protocol. After A contacts I, the intruder I can pretend to be A towards B by forwarding A's special number to B. After B's reply, I can use A to obtain B's number and confirm B according to the protocol. Thus B may believe he is talking to A while in fact he is talking to I.

The lack of a proper semantics for its epistemic language and its high level reasoning weaken the value of correctness proofs in BAN-logic. This proves the need for a closer look at the meaning of knowledge and the cryptographic operations used in security protocols.

1.1. Different aspects of knowledge

As an appetiser, consider the property of *Secrecy*:

> "*an intruder does not know certain **information***".

If the information concerned is a bit string s (a *piece of information*), then to *know* it amounts to possessing this piece of bit string, while s itself does not have any truth value. On the other hand, if the information concerned has the form *"it was B who sent the message"* (call it ϕ), then to *know* ϕ means knowing the fact that *it was B who sent the message*, or in other words knowing that the *proposition* ϕ is true. Clearly, the same word *knowledge* can be used for different aspects of what there is to learn. We will, following (Ramanujam and Suresh, 2005b), refer to the first type of knowledge (in the sense of possession of bit strings) as *knowledge of explicit data*,[2] and to the second type of knowledge as *propositional knowledge*.

More subtleties regarding knowledge in a security context, are related to the cryptographic operations used in the security protocols. First of all, based on the bit strings that agents possess and the cryptographic operations available, they can *know* more bit strings by constructing complex message terms from what they possess, or decomposing a composed one into simpler ones. Such knowledge, in terms

---

[2] Kramer (2007) uses the term "individual knowledge" for this.

of possession of bit strings obtained by cryptographic operations, can be classified as *algorithmic knowledge*, as Halpern and Pucella (2003a) put it. More intricately, if an agent A does not possess the symmetric key k, then the encrypted message of m by k ($\{m\}_k$) should mean no more than a random bit string to A, even though she possesses it. Thus we need a notion of knowledge to denote that an agent can *see* the inherent structure of bit strings. We call the last type *"certain knowledge"* following Baskar et al. (2007).

These different ways of using the term "knowledge" (and the verb 'to know') suggest different structures and treatments in the formal models, which we will discuss in Section 2.2.

1.2. Tension between epistemic and temporal structure

Despite the epistemic flavour in expressing security goals, the interchange of messages, which constitutes protocols, occurs over *time*. Thus a rigourous epistemic approach to security protocol verification needs to harmonise the epistemic and temporal aspects. However, the intuition about the expressivity of epistemic logics does not quite coincide with the practice of security protocol verification so far: most of the successful approaches usually model the protocols formally with purely temporal structures, and try to capture the properties in a temporal formalism (cf. e.g., (Ryan and Schneider, 2001; Abadi and Fournet, 2001; Focardi et al., 2004)). The tension between the natural temporal essence of the formal model of protocols, and the natural epistemic formalisation of the security requirements has proven to be a challenge. This raises two natural questions: 1) Does introducing epistemics into the language indeed empower the expressivity in formalising security properties? 2) What is the computational cost of combining epistemic and temporal aspects in security protocol verifications?

Fortunately, recent years have seen a growing interest in epistemic approaches connected to the study of certain security properties that are not easily expressed in terms of events which did or did not happen along a *single* run of the protocol. A list of such properties includes, for example, anonymity (Syverson and Stubblebine, 1999; Halpern and O'Neill, 2005), receipt-freeness (Jonker and de Vink, 2006; Jonker and Pieters, 2006; Baskar et al., 2007), and coercion-resistance (Delaune et al., 2006; Delaune et al., 2009). The verification of such properties depends on whether agents are able to distinguish between different courses of events, which is exactly the idea behind the standard Kripke semantics of knowledge. Formally, this involves the addition of equivalence relations into the temporal model, where it is useful, natural or even necessary as we will argue in Section 4.

Moreover, despite the apparent disguises of the formalisations, the epistemic logical approaches proposed by different research communities do have some important common features, where careful comparisons are needed to pinpoint the differences. This motivates the writing of this paper. Our goal is two-fold: first, we give a brief overview of several epistemic proposals in Section 2 and compare the essential techniques they employ in Section 3. The survey in these sections is intended to be an introduction to this developing field of epistemic verification. Second, in Section 4, we try to give partial answers to the questions we proposed above. Although the survey will be presented mostly in a high level fashion and will only get to some technical details in Sections 3 and 4 when truly necessary, some basic knowledge of epistemic logics and security protocols will definitely help the reader to absorb the compactly presented material more easily.

While we intend this paper to give a brief overview of approaches to modelling knowledge in the analysis of security protocols, we cannot cover all different aspects. The focus in our paper will be on model checking approaches to verification, based on modal logics of *knowledge* rather than belief, that are *possibilistic* rather than probabilistic. For those interested in the other aspects, our introductory text in Subsection 2.1 contains pointers to some work in the areas outside of our focus.

2. Epistemic approaches: A brief survey

2.1. BAN LOGIC

The starting point of formal verification of security protocols is often attributed to the development of BAN-logic (Burrows et al., 1989), named after its inventors Burrows, Abadi and Needham. The syntax of this logic includes predicates of *belief*[3] and *actions* thus it is able to express message passing actions and security goals. In fact, BAN-logic presents a calculus (proof system) by giving a number of inference rules to derive statements. For example, here is a rule for "if A believes he shares key k with B, and A has received a message X encrypted with k, then A believes that it was B who sent the message":

$$\frac{A \ believes \ (A \overset{k}{\leftrightarrow} B), \quad A \ sees \ \{X\}_k}{A \ believes \ (B \ said \ X)}.$$

To handle protocols in this framework, the protocol first needs to be idealised, then the initial assumptions are spelled out in the BAN-

---

[3] However, it is essentially *knowledge*, following the intuition given by the authors.

language, after which each step in the protocol is annotated with a BAN-formula asserting the state of affairs after that step. The statement after the final step describes the outcome of the protocol. The goal of the analysis is to derive a final assertion that implies the protocol is correct.

However, the *soundness* of the inference rules in BAN-approach was based on the argument of *stability* of the formulas (once a formula becomes true, it remains true), rather than a formal semantics.[4] The lack of a clear semantics also led BAN-logic to an abstraction level too high to capture the consequences of all the possible intruder behaviours. These drawbacks made the BAN-logic analysis of the Needham-Schroeder authentication protocol overlook the possibility of the man-in-the-middle attack exposed by Lowe (1996), who used a process theoretic analysis in the process algebra CSP (Brookes et al., 1984). At the same time, model checking approaches (Clarke et al., 1999) began to flourish and later became prominent (Halpern and Vardi, 1991). To do model checking on security protocols with epistemic logic, it is necessary to have a suitable formal semantics for knowledge in the security setting.

Despite the efforts made in the literature (Gong et al., 1990; Bieber, 1990; Syverson, 1992), the main hurdle to a reasonable semantics of BAN-like logics was the so-called *logical omniscience* problem, an inherent issue of the standard possible-world semantics of epistemic logics (Von Wright, 1951; Hintikka, 1962): agents know all the valid propositions and all logical consequences of what they know. Under possible-world semantics, the knowledge operator K_i for agent i, and its dual $\widehat{K_i}$ are interpreted through an *equivalence* relation $\sim_i$ capturing the epistemic possibilities over the set of states S of a Kripke model $(S, \{\sim_i\}_{i \in \mathbf{I}}, V)$, where V is a valuation function for the proposition letters in concern. $K_i \phi$ is true at a state s if all the $\sim_i$-reachable worlds satisfy ϕ. Therefore, if a message m is indeed of the form $\{m'\}_k$ (thus $m = \{m'\}_k$ is true everywhere in the model), then an agent knows it, even when she does not possess the key k. This sounds contradictory to our intuition in security analysis.

Many approaches have been suggested to avoid the logical omniscience problem (see Fagin et al. (1995b) [Ch.9] and Halpern and Pucella (2010) for surveys). In the context of security analysis, the most relevant one is the approach of algorithmic knowledge (Halpern et al., 1994), which is prominent in our later introduction of various epistemic approaches. The idea is that an agent *knows* a message term

---

[4] See (Teepe, 2006) for a more elaborate discussion on the soundness of BAN-logic.

only if it is derivable by some algorithm with respect to a deductive system capturing idealised cryptographic operations (Halpern and Pucella, 2003a; Pucella, 2006). For propositional knowledge, a more sophisticated way of avoiding the logical omniscience problem can be obtained by deviating the standard Kripke semantics in the definition of reachable possible worlds, as demonstrated in (Cohen and Dam, 2005b; Cohen and Dam, 2007). Essentially, such approach introduces extra possible worlds which may not be in the model when evaluating epistemic formulas. Awareness can also be used to deal with logical omniscience in the security setting (see Accorsi et al. (2001), for instance), but we will not elaborate on this here.

Before moving on from BAN to the modern model checking epistemic approaches, we should mention that several authors have proposed analyses for security protocols involving belief rather than knowledge, e.g. (Hunter and Delgrande, 2007; van der Meyden and Wilke, 2007; Baltag and Smets, 2008). Also, the epistemic approaches that we survey are *possibilistic* in the sense that an agent knows a fact if he does not consider it possible to be false, while in certain security contexts this may be inappropriate. For example, can we rightfully say that A anonymously sent the message, if the intruder knows A is the sender of a message in 99 out of 100 possible runs? Such example suggests a *probabilistic* approach to knowledge or belief to analyse certain security properties, as in (Reiter and Rubin, 1998; Syverson and Stubblebine, 1999; Halpern and O'Neill, 2002; Halpern and O'Neill, 2005; Shmatikov, 2004; Bhargava and Palamidessi, 2005). These *doxastic* and *probabilistic* approaches are not covered in our survey.

2.2. Basics of Epistemic Approaches

In this section, we will list the commonly used components of most epistemic approaches in the post-BAN era. We first need a logical language $\mathcal{L}_\mathbf{I}$ to specify properties of models, where $\mathbf{I}$ is a (finite) set of agents. Due to the fact that we are talking about message passing in a protocol setting, we need to mention messages in our language. This is often done by introducing the message terms in the logical language in the shape of the following:

$$m ::= \quad c \quad | \quad k \quad | \quad \{m\}_k \quad | \quad (m, m')$$

where c stands for some basic plain terms which may in general be of many types (e.g., names, integers etc.), $\{m\}_k$ is the encryptions of m with key k and (m, m') intuitively represents pairing of m and m'. In

general, arbitrary cryptographic operations f can be introduced in this way.

Associated with the message terms there is a derivation system to capture the cryptographic functions in the message terms (Paulson, 1997; Paulson, 1998; Clarke et al., 1998). For example the following derivation rules capture the symmetric encryption and pairing in of the messages:

$$\texttt{synth}: \frac{m \quad m'}{(m, m')} \quad \frac{m \quad k}{\{m\}_k} \qquad \texttt{analz}: \quad \frac{(m, m')}{m} \quad \frac{(m, m')}{m'} \quad \frac{\{m\}_k \quad k}{m}$$

where $\texttt{synth}$ rules govern the application of cryptographic operations to form new terms from the old, while $\texttt{analz}$ rules intuitively extract information from complex terms. We can alternatively represent $\texttt{analz}$ rules by an equational theory E e.g., $\texttt{dec}(\texttt{enc}(x, y), y) = x$ for the last rule above, if $\texttt{dec}, \texttt{enc}$ are introduced as cryptographic operations with the obvious meaning in the language of message terms. Given a set of messages M, we say $M \vdash m$ if either $m \in M$ or m is *derivable* from M by applying the rules. We write $m =_E m'$, if $m = m'$ is an instantiation of an equation induced by E. Halpern and Pucella (2003a) argue that a derivation system may not be convenient to model certain powerful adversary operations, and propose to use arbitrary algorithms instead of derivation systems. For simplicity, we will not cover such more general cases here.

We build formulas based on message terms which are not formulas themselves. Following the observations in Section 1.1, we need different knowledge operators in the language to cope with various types of knowledge:

1. *Knowledge of explicit data* (possession of bit strings): We build basic propositions in the shape of $has_i m$, where m is a message term, meaning that agent i *possesses m*. For such knowledge we have the *de dicto* reading: $has_i \{m\}_k$ means that the bit string of $\{m\}_k$ is possessed by i. However, i may be unsure about the structure of the message.

2. *Algorithmic knowledge* (possession of derivable bit strings): In the literature, the knowledge of explicit data can be viewed as a special case of algorithmic knowledge. We can use $\overline{has_i} m$ to express that m as a bit string can be derived from the information agent i possesses, by applying corresponding cryptographic operations modelled by $\texttt{synth}$ and $\texttt{analz}$ rules (see Halpern and Pucella (2003a) and Ramanujam and Suresh (2005b) for the detailed rationale).

3. *Propositional knowledge* (what facts are known to the agents): As in the standard epistemic logic, we use $K_i \phi$ to express that "agent

i knows that ϕ is true." Thus the logical language $\mathcal{L}_{\mathbf{I}}$ may look like:

$$\overline{has_i}m \quad | \quad \phi \wedge \psi \quad | \quad \neg\phi \quad | \quad K_i\phi \quad | \quad O\phi$$

where $m \in M$, and O can be any other modal operator, depending on what properties we want to specify. On the other hand, given an existing modal logic language, we can turn it into a language about message passing by adding epistemic operators and taking has_im as the basic propositions[5].

4. *Certain knowledge* (the understanding of the bit strings). This kind of knowledge sits in between algorithmic knowledge and propositional knowledge, since it is not only about message term itself but also about the observational power of agents (Baskar et al., 2007). We may use $K_i has_i m$ to express that agent i knows that m is of certain structure, e.g., $K_i has_i \{c\}_k$ means i knows that he has a bit string which stands for $\{c\}_k$.[6] Thus knowledge operator K_i induces somehow a *de re* reading of $has_i m$.

We interpret the formulas of such epistemic language $\mathcal{L}_{\mathbf{I}}$ on Kripke models $\mathcal{M} = (S, \{\sim_i\}_{i \in \mathbf{I}}, V)$. To evaluate the basic formulae in the shape of $has_i m$, we need to associate a set of message terms for each i at each state. Then $has_i m$ is true at a state s if m is in the set of messages associated with i on s. The semantics of $\overline{has_i}m$ is also straightforward by considering derivable messages at a state.

According to the standard Kripke semantics, $K_i\phi$ is true at a state if ϕ is true anywhere reachable from the current state. The equivalence relations naturally model the epistemic uncertainties of agents. Thus the actual formal meaning of propositional knowledge and certain knowledge depends on the definition of the equivalence relation in the model and the message terms possessed by agents at various states. We will compare different equivalence relations in Section 3.1.

Given an epistemic language in the above style, an epistemic verification framework should give a general way to build up models from a protocol description in order to do model checking. Two approaches are discussed in the next subsections following the traditions of *Epistemic Temporal Logic* (`ETL`) and *Dynamic Epistemic Logic* (`DEL`) approaches.

---

[5] In addition to $has_i m$, it is also common to introduce special propositions to denote the actions happened in the past, e.g., $send_j^i(m)$ (see, for instance (Halpern and Pucella, 2003a)).

[6] Different semantics for K_i operator may cause subtly different readings for such statements. We will see different semantics in Section 3.1.

2.3. Epistemic Temporal Approaches

Developed independently by Parikh and Ramanujam (1985) and Halpern and Moses (1990), and made popular later by the seminal book (Fagin et al., 1995b), the *Interpreted Systems* (IS) nicely combine the temporal developments of a system (in runs) with epistemic ones in a distributed setting. Despite their different appearances, most of the epistemic temporal verification languages and models can be reformulated into this very general framework, e.g., epistemic linear-time logic LTL+K (Dixon et al., 2003), epistemic branching-time logic CTL+K (Lomuscio et al., 2009), and epistemic alternating time temporal logic ATL+K (van der Hoek and Wooldridge, 2002). Moreover, some process models like Strand Spaces (Javier et al., 1999) can be related to ISs (Halpern and Pucella, 2003b).[7]

Based on the expositions in (Fagin et al., 1995b) and (Parikh and Ramanujam, 2003), we now define the interpreted systems with explicit events. Given a set of agents $\mathbf{I} = \{i_1, \ldots, i_n, \epsilon\}$, with ϵ modelling the environment, and $n+1$ corresponding non-empty sets $L_1, \ldots, L_n, L_\epsilon$ of local states, the set of *global states* for an interpreted system is a set $S \subseteq L_\epsilon \times L_1 \times \ldots \times L_n$. Given a set of events E, we associate with each $e \in E$ a transition relation $\stackrel{e}{\to} \subseteq S \times S$. An infinite *run* r on S is a function $r : \mathbb{N} \mapsto S \times E$. Let $r_S(u)$ and $r_E(u)$ be the corresponding global state and event (to happen) at the uth point of the run r respectively. We say a run r is *admissible* if $\forall u \geq 0 : r_S(u) \stackrel{r_E(u)}{\to} r_S(u+1)$. An interpreted system $\mathcal{I}$ is then defined as a pair (R, V) where R is a set of admissible runs, and $V : S \mapsto 2^{\mathbf{P}}$ is a valuation function that at each global state in S, assigns a truth value to each proposition atom in $\mathbf{P}$. We denote by $(\mathcal{I}, r, u)$ the point $r(u)$ in interpreted system $\mathcal{I}$.

To verify a protocol in the presence of an adversary, we need to formalise the protocols and the adversary model, describing the possible actions of an adversary. Here we show an example of a formalisation of the Needham-Schroeder authentication protocol mentioned in Section 1, with the Dolev-Yao adversary model (Dolev and Yao, 1983) where all the messages are delivered via the intruder role (E) acting as a *buffer* (see, e.g., Cremers (2006) for rationale):[8]

---

[7] Halpern and Pucella (2003b) show that strand spaces correspond to a subset of a class of ISs, which they call Strand Systems.

[8] Due to space limitation, we cannot go into the details of the various specification languages and adversary models proposed in the literature. For example, Halpern and Pucella (2003a) provide the possibility of modelling different adversaries in the IS-framework.

$$\text{for A}: \begin{array}{llll} 1. & \text{A} & \text{send} & \text{E}: \{n_A, A\}_{PK_B} \\ 2. & \text{A} & \text{rec} & \text{E}: \{n_A, n_B\}_{PK_A} \\ 3. & \text{A} & \text{send} & \text{E}: \{n_B\}_{PK_B} \end{array} \qquad \text{for B}: \begin{array}{llll} 1. & \text{B} & \text{rec} & \text{E}: \{n_A, A\}_{PK_B} \\ 2. & \text{B} & \text{send} & \text{E}: \{n_A, n_B\}_{PK_A} \\ 3. & \text{B} & \text{rec} & \text{E}: \{n_B\}_{PK_B} \end{array}$$

Here the action patterns in a protocol are broken down and grouped into *local protocols* by roles. Note that, in the above formalisation, the intruder implicitly *eavesdrops* all the messages and the agents will accept any message that the intruder may possess, as long as it is in the forms specified (thus modelling the intruder's ability to *manipulate* messages).

Despite differences in details in each specific framework, e.g. (Halpern and Pucella, 2003a; van der Meyden and Su, 2004; Ramanujam and Suresh, 2005b; Boureanu et al., 2009), we can summarise the merit of the general ETL approach for modelling protocols under an adversary model, in the following steps.

Step 1. Suppose the set of agents is $\mathbf{I} = \{1, 2, \ldots, n, \epsilon\}$, where ϵ indicates the intruder. We start from a set S_0 (usually a singleton) of initial states which are tuples of local states $\langle l_1, \ldots l_n, l_\epsilon \rangle$. An initial local state for agent i should, among other things, encode a set of message terms representing the messages that agent i initially possesses (i.e. the *information states* of agents (Ramanujam and Suresh, 2005b)). In such a setting, we can retrieve the information state of i at global state s by $info_i(s)$. We can then define the semantics of $has_i m$ and $\overline{has_i} m$ at $(\mathcal{I}, r, u)$ by $info_i(r_S(u))$ in a straightforward way.

Step 2. We can generate a temporal structure, based on the initial states, by collecting all the admissible sequences of global states according to the protocol under the adversary model. The protocol specification and the adversary model define a set of events (instantiated action patterns). To give the transition relation $\xrightarrow{e}$ for the events on the global states, we can give each event e a precondition and a postcondition. The first specifies when the event can happen and the latter one changes the local states of agents to model information updates by the events. In the above example, an instantiated action: $(j \; \texttt{send} \; \epsilon : \{n_j, j\}_{PK_i})$ has the precondition that $\{n_j, j\}_{PK_i}$ is in the current information set of j and the postcondition that $\{n_j, j\}_{PK_i}$ is added to the information state of the intruder. In general, agents can send a message only if they possess it, and the effect of a $\texttt{send}$ action is that the message is delivered to the intruder (under the Dolev-Yao model). The order of the actions according to the protocol can be encoded also by preconditions requiring that a certain action happened in the earlier stage of the run. We call the resulting set of runs the *generated temporal structure* $T(S_0)$.

We choose to let each e be *observable* to an agent i iff i herself is involved, e.g. $(j \text{ send } \epsilon : m)$ is only observable by ϵ and j. Similarly, the i-observable subsequence of $(j \text{ send } \epsilon : m)(i \text{ rec } \epsilon : m')$ is $(i \text{ rec } \epsilon : m')$. In the Dolev-Yao setting we presented above, the intruder can observe all the events. In a more sophisticated analysis, the events are composed by synchronising local events with respect to each agent according to their local protocols, cf. e.g. (Boureanu et al., 2009).

Step 3. From $T(S_0)$, we build up the epistemic temporal model $E(T(S_0))$ by defining epistemic relations $\sim_i$ between points $(T(S_0), r, k)$. The standard way of defining $\sim_i$ in IS is by matching local states of i, or local views of i of the histories of events. However, the information sets and local histories in the protocol setting, do not capture how the messages are understood by the agents (recall what we called *certain knowledge*, Section 2.2). It is possible that two message terms are different, but still *regarded* as the same by an agent e.g., events $\text{rec} : \{m\}_k$ and $\text{rec} : \{m'\}_k$ are not distinguishable to an agent who does not have the key k. Moreover, if an agent later obtains the key k, then she can tell $\{m\}_k$ and $\{m'\}_k$ apart by *"looking back with a fresh eye"*. Thus we need to build $\sim_i$ on some sophisticated equivalence relation on messages $(\approx)$. In Section 3.1, we will discuss different existing definitions for $\approx$ on *lists* of message terms, since we usually assume that the agents can remember the order of the messages passing actions that she can observe.

It is not hard to see that we can lift $\approx$ to equivalence between points in an IS. Suppose each information set is represented by a list of messages. Let $M_i(e_0, \ldots, e_u)$ be the list of messages occurring in i's observable subsequence of events in $e_0, \ldots, e_u$. Two obvious possibilities are:

- *Asynchronous*: $(s_0 \overset{e_0}{\to} s_1 \ldots s_{u-1} \overset{e_{u-1}}{\to} s_u) \sim_i (s_0' \overset{e_0'}{\to} s_1' \ldots s_{u'-1}' \overset{e_{u'-1}'}{\to} s_{u'}')$ iff $info_i(s_u) \approx info_i(s_{u'}')$.[9]

- *Synchronous*: $(s_0 \overset{e_0}{\to} s_1 \ldots s_{u-1} \overset{e_{u-1}}{\to} s_u) \sim_i (s_0' \overset{e_0'}{\to} s_1' \ldots s_{u-1}' \overset{e_{u-1}'}{\to} s_u')$ iff $\langle info_i(s_0), M_i(e_0, \ldots, e_{u-1}) \rangle \approx \langle info_i(s_0'), M_i(e_0', \ldots, e_{u-1}') \rangle$.

The above procedure can be summarised with the slogan:

First temporal then epistemic.

Notably, Boureanu et al. (2009) presents an fully automated method to generate interpreted systems from formal specification of protocols

---

[9] This is an example of asynchronous and *forgetful* agents (Shilov and Garanina, 2002), other memory conditions can be applied here.

taking many details into consideration. Other methods to generate IS-like models include process algebra with epistemic annotations, e.g. Dechesne et al. (2007), who make use of an operational semantics to generate the model from the protocol specified in process algebra terms.

2.4. Dynamic Epistemic Logic Approaches

A different perspective towards the dynamics of multi-agent system is provided by the development of the so-called Dynamic Epistemic Logic (DEL) (Plaza, 1989; Gerbrandy and Groeneveld, 1997; Baltag et al., 1999). The focus of DEL is not on the temporal structure of the system but rather on the epistemic impact of the events as the agents perceive them. A typical DEL-language is defined as follows:

$$\phi ::= \top \mid p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid [\mathcal{A}, e]\phi \mid K_i\phi$$

where p is in a set of basic propositions $Prop$, $G \subseteq I$ and $\mathcal{A}$ is an *action model* with e as its designated action. $[\mathcal{A}, e]\phi$ intuitively says after execution of the event $(\mathcal{A}, e)$, ϕ is true. Action models are tuples of the form $(E, \{\asymp_i\}_{i \in \mathbf{I}}, Pre, Pos)$ where $\asymp_i$ models agents i's observational power on events in E (e.g. $e_1 \asymp_i e_2$ means i is not sure which one of e_1 and e_2 happened); the precondition function $Pre : E \to Form(\texttt{DEL})$ assigns each event a DEL-formula such that the event can only happen when the formula is satisfiable; the postcondition $Pos : E \to (Prop \to Form(\texttt{DEL}))$ models the factual changes caused by the event by changing the truth values of basic propositions p to the truth value of $Pos(e)(p)$ (see van Benthem et al. (2006) for details). The semantics for epistemic formulas is as usual and for $[\mathcal{A}, e]\phi$ formulas:

$$\mathcal{M}, s \vDash [\mathcal{A}, e]\phi \iff (\mathcal{M}, s \vDash Pre(e) \Rightarrow \mathcal{M} \otimes \mathcal{A}, (s, e) \vDash \phi)$$

Where, given a static Kripke model $\mathcal{M} = (S, \{\sim_i\}_{i \in \mathbf{I}}, V)$ and an action model $\mathcal{A} = (E, \{\asymp_i\}_{i \in \mathbf{I}}, Pre, Pos)$, the updated model is $\mathcal{M} \otimes \mathcal{A} = (S', \{\sim'_i\}_{i \in \mathbf{I}}, V')$ with:

$$\begin{aligned} S' &= \{\langle s, e \rangle \mid \mathcal{M}, w \Vdash Pre(e)\} \\ \sim'_i &= \{(\langle s, e \rangle, \langle t, e' \rangle) \mid s \sim_i v \text{ and } e \asymp_i e'\} \\ V'(\langle s, t \rangle)(p) &= \top \iff \mathcal{M}, w \Vdash Pos(e)(p) \end{aligned}$$

DEL has been successful in modelling what agents learn through different communication acts according to epistemic reasoning, for example in the scenario where an agent shows his cards to the players on his team without the opponent noticing (van Ditmarsch, 2003). Thus it looks promising to analyse security protocols by modelling protocols in terms of action models. In (Hommersom et al., 2005), (van Eijck and

13

Orzan, 2007), and (Dechesne and Wang, 2007) the first attempts were made towards the security protocol verification by `DEL`. We summarize the modeling steps as follows:

Step 1. We start with a finite initial static model $\mathcal{M}$ with given epistemic relations $\sim_i$. Similar as in the interpreted system approach, a state is associated with a tuple of information sets modelling the messages that agents possess. The epistemic relations can be given similarly according to the equivalence $\approx$ on lists of messages.

Step 2. We need to build an event model $\mathcal{A}$ which captures all the protocol actions with suitable pre- and postconditions similar to what we described at step 2 for `ETL` approaches. For example, to model the Needham-Schroeder authentication protocol mentioned above, we can build action model $\mathcal{A} = (E, \{\asymp_i\}_{i \in \mathbf{I}}, Pre, Pos)$ such that E includes all instantiated actions of the protocol, for example: event $e = (j \ \texttt{send} \ \epsilon : \{n_j, j\}_{PK_i})$ with precondition $Pre(e) = \overline{has}_j(\{n_j, j\}_{PK_i})$ and postcondition $Pos(e)(has_\epsilon(\{n_j, j\}_{PK_i})) = \top$. The epistemic relations $\asymp_i$ between events can be generated by lifting $\approx$ on lists of messages to events, under the constraint that agent can always distinguish the events that she is involved from other actions.

Step 3. The update execution computes the result of performing $\mathcal{A}$ on $\mathcal{M}$ iteratively, thereby it essentially builds up all the possible runs of the protocol.

The above procedure can be summarised as the slogan:

First epistemic then temporal.

Although it seems that `DEL` modelling is very similar to `ETL`modelling, we will pinpoint the tricky differences between the two approaches in details in Section 3.2.

2.5. Tools

In the last decade, many tools have been developed to handle formal verification in the setting of `ETL` or `DEL`, with potential application in security analysis. For `ETL` model checking, we have MCK: Model Checking Knowledge (Gammie and van der Meyden, 2004; van der Meyden and Su, 2004) and MCMAS: Model Checker for Multi-Agents Systems (Lomuscio and Raimondi, 2006b; Lomuscio et al., 2009). Boureanu et al. (2009) recently presented a fully automatic translation from protocol descriptions given in CAPSL (Common Authentication Protocol Specification Language) into the input language for MCMAS, enabling the automated checking of the security protocols from the Clark-Jacobs security protocol library by means of temporal epistemic logic. For `DEL` model checking, we have DEMO: Dynamic Epistemic MOdelling (van

Eijck, 2005) and LYS: a knowledge anaLYSis toolset (Orzan, 2005). Other relevant tool sets include the ETL-model checker MCTK (Su, 2004), the ATL-model checker (Alur et al., 1998), and the real-time system model checker (Kacprzak et al., 2008).[10]

In the literature, various tools are presented with some case studies demonstrating how the framework can be applied. For these demonstrations, often well-known situations or protocols are chosen which require relatively small models. The *classic* examples in the epistemic verification demonstrations are the *Dining Cryptographers* protocol for anonymous broadcast (Chaum, 1988), the *Muddy Children puzzle* (see, e.g., (Fagin et al., 1995b)) for demonstrating the effect of (repetition of) public announcements, and *Russian Cards problem* (see van Ditmarsch (2003)) for secure public announcements. Such common examples facilitate comparisons of the modelling and efficiency among different tools based on different frameworks, see for example (van Ditmarsch et al., 2006), which takes the Russian Cards problem as a test case for MCK, MCMAS and DEMO.

3. Comparisons

In this section we will compare more technical aspects of the approaches mentioned in the previous section. In the first part, we discuss the different versions of equivalence. In the second part, we compare the epistemic temporal approach with the dynamic epistemic one in the security setting.

3.1. ON EQUIVALENCES

Some well known formal methods have been adapted or designed to include (trace) equivalences to deal with multi-trace security properties (e.g., *applied pi-calculus* (Abadi and Fournet, 2001)). In this part, we focus on how the equivalence relations of agents are defined, based on the lists $\langle m_1, \cdots m_n \rangle$ that record the messages that this agent received in order. The rest of this subsection will be devoted to the comparison of the following equivalence relations:

- simple deduction equivalence $\approx_d$

- *pattern matching equivalence* $\approx_{pat}$ (in (Abadi and Tuttle, 1991; Abadi and Rogaway, 2002) and (Baskar et al., 2007));

---

[10] This is definitely not a complete list, see (Lomuscio and Penczek, 2007) for a survey of symbolic model checking for ETL.

15

- *static equivalence* $\approx_s$ (defined in (Abadi and Fournet, 2001; Abadi and Cortier, 2004), and later used in (Ciobâcǎ et al., 2009; Chadha et al., 2009));

- *permutation equivalence* $\approx_{per}$ (in (Cohen and Dam, 2005a; Garcia et al., 2005), and later used in (Cohen and Dam, 2007; Jonker and Pieters, 2006)).

We assume there is a fixed equational theory E corresponding to the derivation system on terms of messages. Let $M = \langle m_1, \ldots, m_n \rangle$ and $M' = \langle m'_1, \ldots, m'_n \rangle$[11] then:

- $M \approx_d M'$ iff for all message terms m: $M \vdash m \iff M' \vdash m$.

- $M \approx_{pat} M'$ iff M and M' induce the same recognisable message patterns, i.e. for all j: $pat(m_j, M) = pat(m'_j, M')$, where $pat(m_j, M)$ is roughly the message term in which the unconstructable parts are replaced by an uninterpreted symbol $\square$. For example:

$$pat(\{m\}_k, M) = \begin{cases} \{pat(m, M)\}_k & \text{if } M \vdash k \\ \square & \text{otherwise} \end{cases}$$

For formal details on various cryptographic operations we refer to (Abadi and Rogaway, 2002; Baskar et al., 2007).

- $M \approx_s M'$ iff M and M' satisfy the same equality tests. Formally, defining $\sigma_M, \sigma_{M'}$ to be the substitutions replacing x_j with m_j and m'_j respectively, then $M \approx_s M'$ iff for any message terms with variables $t(x_1, \ldots, x_n)$ and $t'(x_1, \ldots, x_n)$:

$$\sigma_M(t) =_E \sigma_M(t') \iff \sigma_{M'}(t) =_E \sigma_{M'}(t').\text{[12]}$$

- $M \approx_{per} M'$ iff there is a permutation $\pi : M \to M'$ such that for all j: $\pi(m_j) = m'_j$ and $\pi(t(\overline{m})) = t(\overline{\pi(m)})$ for any message term with variables t and any suitable list $\overline{m}$ from $\{m \mid M \vdash m\}$. Cohen and Dam (2005a) show that $\approx_{per}$ is indeed an equivalence relation.

The relation $\approx_d$ is very fine (despite the fact it does not require a one-one correspondence of messages) and thereby assigns strong observational power to the agents: e.g. $M_1 = \langle \{c\}_k \rangle \not\approx_d M_2 = \langle \{c'\}_k \rangle$. It may only make sense to employ such equivalence relation for the intruder if we need to guarantee extreme security. On the other hand

---

[11] Note that the equivalences we consider here all respect the number of messages.

[12] Here we leave out the details about protected names in the original *frame* (our σ) in applied-pi calculus.

$\approx_{pat}$ is rather coarse as it treats all the unreadable parts as the same: e.g. $M_3 = \langle \{c\}_k, \{c'\}_k \rangle \approx_{pat} M_4 = \langle \{c\}_k, \{c\}_k \rangle$ since $pat(\{c\}_k, M_3) = pat(\{c'\}_k, M_3) = pat(\{c\}_k, M_4) = \square$.

Static equivalence is somewhere in between, e.g. $M_1 \approx_s M_2$ but $M_3 \not\approx_s M_4$ since $\{c\}_k \neq_E \{c'\}_k$ but $\{c\}_k =_E \{c\}_k$. To relate $\approx_s$ and $\approx_{per}$, Cohen and Dam show that:

THEOREM 1. (Cohen and Dam, 2007). *For any lists of messages M and M' satisfying $|\{m \mid M \not\vdash m\}| = |\{m \mid M' \not\vdash m\}| = \omega$:*

$$M \approx_s M' \iff M \approx_{per} M'$$

where the cardinality condition allows us to permute all the non-derivable messages in M to the non-derivable messages in M'.

Petride and Pucella (2007) plead for a principled approach to model indistinguishability relations that is worth elaborating upon. They define two states to be indistinguishable for an agent if the agent can compute the same observations from both states. There observations can be considered as tests in the spirit of static equivalence. They generate relations on the basis of the computational power of the agents: taking Θ to be a set of observations θ (tests), and A an algorithm returning for each $\theta \in \Theta$ and M the answer "yes", "no" or "unknown" to the question whether θ holds at M, they let $M \approx_{\Theta,A} M'$ iff for all $\theta \in \Theta$ $A(\theta, M) = A(\theta, M')$. For example $\approx_{pat}$ can be reformulated as $\approx_{\Theta,A}$ where θ is built as follows:

$$t ::= x \mid c \mid k \mid \{t\}_k \mid (t, t)$$
$$\theta ::= has(t) \mid \exists x.\theta \text{ where the only free variable in } \theta \text{ is } x.$$

It is easy to see that θ expresses the pattern of a message. The corresponding algorithm A then takes a pattern and then try to match it in M. On the other hand, to have Θ define $\approx_s$, we at least need to introduce equality into the language of Θ. If fact, if we take Θ as a logical language then this proposal is actually asking for logical characterisations to different equivalence relations with corresponding "model checking" algorithms for Θ on M. As another example, a logical characterisation of $\approx_{per}$ is given in (Cohen and Dam, 2007, Theorem 3).

Regarding the complexity of checking such equivalence, we should first note that the decidability of $M \vdash m$ can be encoded by the decidability of $\approx_s$ or $\approx_{pat}$. However, checking $\vdash$ can be undecidable (Abadi and Cortier, 2004) depending on the underlying derivation system. Baskar et al. (2007) show that when M is finite, a derivation system containing encryption and blind signature can be decided in PTIME . This implies the decidability of $\approx_{pat}$ according to the definition of $\approx_{pat}$

17

in (Baskar et al., 2007). More general results in (Abadi and Cortier, 2004) show that when the E is a *convergent subterm theory* that can cover many important cryptographic operations, both $\approx_s$ and $\vdash$ are decidable in PTIME .

3.2. ETL VS. DEL IN MODELLING

We now compare the epistemic temporal approach with the dynamic epistemic approach in modelling.
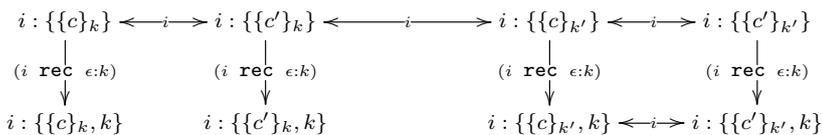
3.2.1. *Limitations of* DEL

As epistemic temporal logic and dynamic epistemic logic are two important methods of describing epistemic interaction over time, technical comparisons have been done to pinpoint the differences between the two. From an abstract point of view, ignoring the structure of the local states, an ETL-model is a tree-like Kripke structure with relations labelled by events and agent names. We get a similar structure, if we start from a static initial Kripke model, performing sequences of DEL-updates, and link each state and its update by the corresponding event $(s \xrightarrow{e} \langle s, e \rangle)$.

Van Benthem et al. (2007) characterise the class of ETL tree-like structures that are DEL constructable in the above sense, by the notions of *Synchronicity* (agents are always aware if something has happened), *Perfect Recall* (the local history is remembered), *No Miracles*, and *Epistemic Bisimulation Invariance* (see below). This means that standard DEL can only deal with idealised agents who satisfy those properties. If we model intruders with enough observation power for better security, then Synchronicity and Perfect Recall can be intuitively assumed. However, No Miracles and Epistemic Bisimulation Invariance may lead to some drawbacks of DEL approaches in security verification:

No Miracles: An ETL-model $\mathcal{M}$, considered as a Kripke model with temporal action transitions $\xrightarrow{e}$ and epistemic relations $\sim_i$, has the property *No Miracles* if the following holds: *for all states s, s' and events e, e' such that $s \xrightarrow{e} t$ and $s' \xrightarrow{e'} t'$, for some t, t': if $s \sim_i s'$ and there are s'', s''' with $s'' \xrightarrow{e} t''$, $s''' \xrightarrow{e'} t'''$ for some $t'' \sim_i t'''$, then $t \sim_i t'$*. (If two actions lead to indistinguishable states somewhere in the model, then it can't be the case that performing these actions on indistinguishable states will lead to distinguishable states.)
However consider the following (partial) model where $\sim_i$ denotes an equivalence relation based on $\approx_{pat}$:

$$i : \{\{c\}_k\} \xleftarrow{\ \ i\ \ } i : \{\{c'\}_k\} \xleftarrow{\qquad i \qquad} i : \{\{c\}_{k'}\} \xleftarrow{\ \ i\ \ } i : \{\{c'\}_{k'}\}$$

$$\downarrow (i\ \texttt{rec}\ \epsilon{:}k) \qquad \downarrow (i\ \texttt{rec}\ \epsilon{:}k) \qquad \downarrow (i\ \texttt{rec}\ \epsilon{:}k) \qquad \downarrow (i\ \texttt{rec}\ \epsilon{:}k)$$

$$i : \{\{c\}_k, k\} \qquad i : \{\{c'\}_k, k\} \qquad\qquad i : \{\{c\}_{k'}, k\} \xleftarrow{\ i\ } i : \{\{c'\}_{k'}, k\}$$

where $k' \neq k$ and $c' \neq c$. Clearly, this model violates *No Miracle*, so it is impossible to be generated by the standard `DEL` approach. The problem roots in the definition of epistemic relations in the action models. Recall that the epistemic relations in the updated model are defined by the synchronisation of the epistemic relations in the static model and those in the action model. However, in the security protocol setting, the same receive action on indistinguishable states may cause the resulting states to be distinguishable, as the example shows. One way to go around this is to "*split*" each action into multiple *copies* with different preconditions such that an action can be looked differently under different conditions. For example, in the action model, the action $(i\ \texttt{rec}\ \epsilon : k)$ with the precondition $has_i\{c\}_k \wedge \neg\overline{has_i}k$ should be i-distinguishable from the same action with the precondition $has_i\{c'\}_k \wedge \neg\overline{has_i}k$. However, this ad-hoc method may introduce infinitely many copies of actions in the action model, which is not allowed by the standard `DEL`.

Epistemic Bisimulation Invariance requires the same event to happen at the states that are epistemically bisimilar (i.e. bisimulation disregarding the temporal relations). This is because the pre-conditions in the action model are formalised in the dynamic epistemic language. This may cause problems if we want to model protocol actions with temporal preconditions in terms of the past (for example, if i sends k only if j sent k'). The usual solution is to encode the history of actions by new basic propositions.

3.2.2. *Limitations of* `ETL`

According to the modelling procedure we described in Section 2.3, the epistemic relations are built after the temporal structures. This may prevent us from handling *knowledge-based protocols*, which have preconditions in terms of knowledge, e.g., i sends m only if i knows that j has k. As shown in (Halpern and Fagin, 1989; Fagin et al., 1995a), it is possible to construct the unique temporal structure and epistemic relations simultaneously according to a knowledge-based protocol, if the system is synchronous and the epistemic preconditions are not about the *future*.[13] On the other hand, `DEL` by definition can handle conditions about what *may* happen in the future, since in action models we can

_____

[13] As argued in (Halpern and Fagin, 1989), if a protocol has *"forward-looking"* conditions (like $K_i F\phi$: "i knows that ϕ will hold eventually"), it is circular to define

have preconditions like $K_i\langle\mathcal{A}, e\rangle\phi$ (i knows that e may happen and in that case ϕ will be true).

In the ETL modelling of security protocols, the initial (global) states represent the initial distribution of names, keys, and other messages. If we focus on a particular distribution, then we can start with a unique initial state. By doing so, we implicitly assume that the distribution of the information, e.g, who has what key, is commonly known (Baskar et al., 2007). However, what if one agent is uncertain about whether another agent knows that she has a public key? Such higher order uncertainties are not well-handled if we generate epistemic relations between initial states based on matching local states. For example, suppose the only message term is a public key k and agent i has it while agent j does not. To make the formula $\phi = K_i(has_ik \wedge \neg has_jk) \wedge \widehat{K_i}K_jhas_ik \wedge \widehat{K_i}\neg K_jhas_ik$ true in an initial model, we need at least two states which represent the same initial distribution of messages, as the following model shows:

$$i : \{k\}; j : \{\} \leftarrow\!\!-i-\!\!\rightarrow i : \{k\}; j : \{\}$$
$$\begin{array}{c}\uparrow\\j\\\downarrow\end{array}$$
$$i : \{\}; j : \{\}$$

It is clear that ϕ holds at the upper two states. However, if the epistemic relations are generated by matching local states or other local information, then there should be a j relation linking all the states. But then formula $K_i\neg K_jhas_ik$ will be true at the upper worlds, contradictory to our initial intention. In fact, if we want to handle higher order uncertainties by the generated epistemic relations, we need to introduce some extra *tokens* in the local states of j to distinguish the two upper states. Intuitively, a local state of one agent, though called *local*, should also contain information about one's opinion on others, in order to handle higher order uncertainties. However, it is rather ad-hoc to introduce those auxiliary tokens. On the other hand, DEL is more flexible in modelling how agents reason about each other, because the equivalences can be defined by choice. More flexibility is also offered by the possibility of modelling higher order uncertainties in the action models.

To summarize, the distinct features in either DEL or ETL approaches are usually double-edged swords:

---

the admissible runs uniquely. Therefore there may be none or several solutions to the fix-point-like definition of the admissible runs.

Features	ETL	DEL
Equivalence relations	generated by matching local information	generated by product update or by hand (for initial models)
Pros	flexible and automatic; generated in a distributed fashion	easy to handle higher order uncertainties; update mechanism is formally defined
Cons	inconvenient for higher order uncertainties at initial states	inconvenient in a cryptographic setting
Events	represented by transitions on global states	modelled in action models
Pros	flexible	pre- and postconditions are encoded in the DEL language thus easy to handle epistemic conditions in protocols
Cons	detailed modelling (e.g. pre- and postconditions) is outside the framework	equivalence relations between events are designed by hand

Based on the above observation, we may want to combine the two frameworks, as already attempted in (Hoshi and Yap, 2009; van Benthem et al., 2009; Wang et al., 2010).

Comparing to the ETL approach, the standard DEL-approach has limitations in generating suitable equivalence relations for the security setting. On the other hand, DEL seems convenient for the knowledge-based protocols (or simply *epistemic protocols*) where:

— preconditions are in terms of knowledge of the agents;

— higher order uncertainties are crucial (e.g., higher order uncertainty about initial distribution of information or observation of actions);

— protocol goals are in terms of *nested* knowledge form.

Epistemic protocols use epistemic reasoning rather than cryptography to obtain security. Examples of such protocols include e.g., Dining Cryptographers (Chaum et al., 1988) and Card Cryptography (Fischer and Wright, 1996; van Ditmarsch, 2003; van Ditmarsch, 2008). Notably, to verify such protocols, meta-knowledge of the protocols themselves matters and creates some complications (Wang et al., 2009).

4. To know or not, towards a technical answer

As emphasised in the previous sections, many epistemic approaches are motivated by a common conviction that epistemic logic can express security properties "more naturally". However, in practice, in most of the formal frameworks, security properties are formalised as temporal formulae rather than in terms of knowledge. To really justify the use of epistemics, it is crucial to understand better whether adding epistemics into the temporal logic can indeed help to express more security properties, and if so, what the cost is for the improved expressivity.

4.1. ON EXPRESSIVITY OF ETL

In this section, we try to formally address the expressivity of epistemic temporal logic (ETL) versus pure temporal logic (TL), aiming at a technical basis to answer the above questions. Here we regard ETL and TL as *classes* of logics: we do not fix the exact logic unless necessary. The comparison will always be between a temporal logic L and an epistemic temporal logic that extends L with epistemic operators.

Roughly speaking, a logic L_1 is strictly *more expressive* than L_2, if (1) for every formula in L_2 there is a formula in L_1 defining the same class of models (i.e. they have exactly the same models.); but (2) there is a formula in L_1 which does not have a corresponding formula in L_2. Note that the comparison of the expressivity of different logics is usually studied given the condition that the logics concerned are defined on the same type of models. However, in the case of ETL and TL, this condition does not hold: the models of ETL involve epistemic relations, while these are absent in the TL-models. This complicates formal comparisons of the two logics in terms of expressivity. To make the comparison of ETL and TL possible, we need to provide the common playground for these two logics.

A rather straightforward observation is that if we consider the epistemic relations of agent i to be just another kind of transitions, labelled 'i', then ETL can be "reduced" to TL. Let $\text{CTL}_\mathbf{I}^*$ and $\text{Mu}_\mathbf{I}$ be CTL^* and modal $\mu-$calculus with extra actions labelled by the names of agents in $\mathbf{I}$ respectively. Let C^{ETL} and C^{TL} be the classes of all ETL- and TL-models respectively. Then:

THEOREM 2. *There exists a language translation $t_\text{L} : L_{\text{ETL}} \to L_{\text{TL}}$ and a model transformation $t_\mathcal{M} : C^{\text{ETL}} \to C^{\text{TL}}$ where $\text{TL} \in \{\text{CTL}_\mathbf{I}^*, \text{PDL}, \text{Mu}_\mathbf{I}\}$ such that:*

$$\forall \varphi \in \text{ETL} \forall \mathcal{M} \in C^{\text{ETL}} (\mathcal{M} \models_{\text{ETL}} \varphi \Leftrightarrow t_\mathcal{M}(\mathcal{M}) \models_{\text{TL}} t_\text{L}(\psi)).$$

PROOF We only discuss the CTL* case. Let t_L be the translation that, for each formula, 1) replaces each occurrence of K_i by AX_i, 2) recursively replaces each common knowledge operator $C_{\mathbf{I'}}$ (with $\mathbf{I'} \subseteq \mathbf{I}$) by $A(\neg((\bigvee_{i \in I'} X_i \top) U t_\mathsf{L}(\neg\phi)))$. Let $t_\mathcal{M}$ be the transformation which unravels the epistemic relations into labelled temporal relations. □

This observation suggests a way to reduce ETL model checking into TL model checking, with the help of some small model property. However, the unravelling of epistemic relations may introduce an exponential blow-up of the models, see for instance (Alur et al., 2007).

On the other hand, the above result is somehow misleading in understanding the expressivity of ETL and TL, since we reinterpret epistemic relations as temporal operators by introducing *new* operators in the temporal language. To address the comparison of expressivity without manipulating the language we can consider the following case:

Suppose that the epistemic relations of the ETL*-models are generated by the temporal structures as explained in Section 2.3. We can turn the* ETL*-models into* TL*-models by ignoring the generated epistemic relations. A straightforward question is to ask whether explicit epistemics helps to define more classes of such temporal models, or if the epistemic information can be retrieved from the temporal structure. Formally we need to prove or disprove the following:*

$$\exists\phi \in L_{\mathsf{ETL}}, \forall\psi \in L_{\mathsf{TL}} : t_\mathcal{M}^-(C_\phi) \neq C_\psi.$$

where C_ϕ (C_ψ) is the class of ETL *(*TL*) models which satisfy ϕ (ψ); $t_\mathcal{M}^-$ transforms the* ETL *models in C_ϕ into corresponding* TL *models by ignoring the generated epistemic relations.*

In case that the epistemic relations are generated respecting synchronicity (i.e. epistemic relations only appear in the same *level* of the tree unravelling of the temporal model), then we have a clear answer to the above question. We can reformulate Theorem 1 of (Alur et al., 2006) in spirit as follows:

THEOREM 3. *If we only consider the* ETL *models satisfying synchronicity, then the secrecy flavoured* ETL *formula $AXAG(\neg K\neg p \wedge \neg Kp)$ (never be sure about p in the future) is not $t_\mathcal{M}^-$-translatable into the language of modal μ-calculus.*

The proof is essentially hidden in Emerson (1987), which shows that the class of the trees that have a level where p is true everywhere, is not recognisable by non-deterministic Muller tree automata. We can employ the pumping-lemma-like argument of Emerson (1987) to obtain this result.

The merit of the above untranslatability result may actually root in the property of synchronicity, since it is known that Monadic Sec-

ond Order logic cannot express "x and y are at the same level" on trees (Läuchli and Savioz, 1987). Hence, although synchronicity is a commonly accepted idealisation of the agents, we still want to know whether we can ignore it or replace it by other properties but get a similar untranslatability result. This is still open.

4.2. Relevant results of Model Checking ETL

The previous sections gave both the intuitive and technical arguments on the usability and expressivity of the epistemic approaches in protocol verification. However, do we pay any cost in the complexity of model checking? In this section we summarize the important model checking results of the literature. For complexity results regarding the satisfiability problems of the corresponding logics we refer to (Halpern and Vardi, 1986; Shilov and Garanina, 2002).

Shilov and Garanina (2002) show that on explicit Kripke models the model checking problem of CTL with common knowledge operators (CTL + C) can be done in PTIME and van der Hoek and Wooldridge (2002) proved that for Alternating Time Logic (ATL) with knowledge, it is PTIME -complete. This looks similar to the logics without knowledge operators. However, due to the construction of epistemic models in the protocol verification setting, we are more interested in the model checking problem on finitely generated infinite epistemic temporal models. Results in (Shilov and Garanina, 2002) indicate that on asynchronous generated models with forgetful agents, the complexity of model checking complies to the general case on Kripke structures. However, we are more interested in the finitely generated synchronous system with perfect recall agents as intruders. Here are some results for this situation:

Reference	Logic	Fragment	Complexity
(van der Meyden and Shilov, 1999)	LTL + K	full	non-elementary
(van der Meyden and Shilov, 1999)	LTL + C	full	undecidable
(van der Meyden and Shilov, 1999)	LTL + C	UNTIL-free	PSPACE-complete
(Engelhardt et al., 2007)	LTL + C	single agent	PSPACE-complete
(Shilov and Garanina, 2002)	CTL + K	full	non-elementary
(Shilov and Garanina, 2002)	CTL + C	full	undecidable
(Alur et al., 2007)	CTL + K	nesting-free	PSPACE-complete
(Shilov and Garanina, 2002)	PDL + C	full	PSPACE-complete
(Shilov and Garanina, 2002)	MU + K	full	undecidable
(Alur et al., 2007)	MU + K	nesting-free	EXPTIME-complete

Putting together the decidability of $\approx$ on messages terms (cf. Section 3.1) and the model checking results above, we can obtain decidability results for security verification (e.g. Baskar et al. (2007)).[14]

---

[14] Important security properties are generally undecidable if there are no restrictions on the number of messages and nonces, for example, cf. (Durgin et al., 1999)

The above results suggest that we may focus on single agent cases or nesting-free `ETL` formulas due to the complexity issues. This somehow coincides with the disadvantages of `ETL` modelling we mentioned in Section 3: `ETL` modelling is not very suitable for multi-agent cases with higher order uncertainty. On the other hand, although multi-agent cases are often undecidable in general, we can still have some hope by restricting ourselves to certain class where equivalence relations of agents have certain pattern (e.g. Engelhardt et al. (2002)). Moreover, some model checking techniques such as abstraction and symmetry reduction that are specific to `ETL` or `DEL` can be found in (Dechesne et al., 2008; Cohen et al., 2009a; Cohen et al., 2009b). Due to the space limitations we will not go into the details.

As a final note, in practice, the performance of an `ETL` model checking tool kit relies on the particular class of models in concern and their representations, for example, model checking `CTL+K` against *"compact models"* is in PSPACE (Lomuscio and Raimondi, 2006a).

5. Conclusions

In this paper, we surveyed the epistemic approaches to security protocol verification with the questions: are security protocols essentially about knowledge (what is there *to know*?), what kinds of knowledge can we distinguish (how *to know*?), and does an epistemic approach bring benefits (why *to know*?). We first made the distinctions between different types of knowledge relevant in the security setting and then gave an overview of commonly used techniques in the epistemic approaches. In particular, we compared various equivalence relations defined in the literature that correspond to the semantics of propositional knowledge. We also compared two major epistemic logical approaches proposed to model interaction in multi-agent systems. It turns out that, in a security protocol verification setting, `ETL` approaches are more suitable to model message passing over time, based on which appropriate equivalence relations can be generated. On the other hand, the `DEL` approach offers all freedom to model higher order information and uncertainties in terms of agents' knowledge about each other. The model checking results of `ETL` also confirm that it is better to focus on a single agent case: in the security setting, this would be the intruder. Finally, we collected clues for the comparison of the expressivity of `ETL` and `TL`, in order to see when an epistemic approach is inevitable. We showed under

---

for the undecidability for secrecy. A solution is to focus on decidable subclasses as in e.g. (Ramanujam and Suresh, 2005a).

the assumption of synchronicity, that `ETL` can define more (security) properties of the temporal structures.

References

Abadi, M. and V. Cortier: 2004, 'Deciding Knowledge in Security Protocols Under Equational Theories'. In: *ICALP04*, Vol. 3142 of *LNCS*. pp. 46–58.

Abadi, M. and C. Fournet: 2001, 'Mobile values, new names, and secure communication'. In: *POPL '01*. pp. 104–115.

Abadi, M. and P. Rogaway: 2002, 'Reconciling Two Views of Cryptography (The Computational Soundness of Formal Encryption)'. *Journal of Cryptology* **15**(2), 103–127.

Abadi, M. and M. R. Tuttle: 1991, 'A semantics for a logic of authentication (extended abstract)'. In: *PODC '91*. New York, NY, USA, pp. 201–216.

Accorsi, R., D. Basin, and L. Vigano: 2001, 'Towards an Awareness-Based Semantics for Security Protocol Analysis'. In: *Logical Aspects of Cryptographic Protocol Verification*, Vol. 55 of *ENTCS*. pp. 5–24.

Alur, R., T. A. Henzinger, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Tasiran: 1998, 'Mocha: Modularity in Model Checking'. In: *CAV'98*, Vol. 1427 of *LNCS*. pp. 521–525.

Alur, R., P. Černý, and S. Chaudhuri: 2007, 'Model Checking on Trees with Path Equivalences'. In: *TACAS'07*, Vol. 4424 of *LNCS*. pp. 664–678.

Alur, R., P. Černý, and S. Zdancewic: 2006, 'Preserving Secrecy Under Refinement'. In: *ICALP'06*, Vol. 4052 of *LNCS*. pp. 107–118.

Anderson, R. and R. Needham: 1995, 'Programming Satan's computer'. In: J. Leeuwen (ed.): *Computer Science Today*, Vol. 1000 of *LNCS*. Berlin/Heidelberg: Springer, Chapt. 26, pp. 426–440.

Baltag, A., L. S. Moss, and S. Solecki: 1999, 'The logic of public announcements, common knowledge, and private suspicions'. Technical Report SEN-R9922, CWI, Amsterdam.

Baltag, A. and S. Smets: 2008, 'Probabilistic dynamic belief revision'. *Synthese* **165**(2), 179–202.

Baskar, A., R. Ramanujam, and S. P. Suresh: 2007, 'Knowledge-based modelling of voting protocols'. In: *TARK '07*. pp. 62–71.

Bhargava, M. and C. Palamidessi: 2005, 'Probabilistic Anonymity'. In: *CONCUR'05*, Vol. 3653 of *LNCS*. pp. 171–185.

Bieber, P.: 1990, 'A logic of communication in hostile environment'. In: *Computer Security Foundations Workshop III*. pp. 14–22.

Boureanu, I., M. Cohen, and A. Lomuscio: 2009, 'Automatic verification of temporal-epistemic properties of cryptographic protocols'. *Journal of Applied Non-Classical Logics* **19**(4), 463–487.

Brookes, D., C. A. R. Hoare, and A. W. Roscoe: 1984, 'A theory of communicating sequential processes'. *Journal of the ACM* **31**(3), 560–599.

Burrows, M., M. Abadi, and R. Needham: 1989, 'A Logic of Authentication'. *Proceedings of the Royal Society of London, Series A, Mathematical and Physical Sciences* **426**(1871), 233–271.

Chadha, R., S. Delaune, and S. Kremer: 2009, 'Epistemic Logic for the Applied Pi Calculus'. In: *FMOODS '09/FORTE '09*, Vol. 5522 of *LNCS*. pp. 182–197.

Chaum, D.: 1988, 'The dining cryptographers problem: unconditional sender and receiver untraceability'. *Journal of Cryptology* **1**, 65–75.

Chaum, D., C. Crépeau, and I. Damgard: 1988, 'Multiparty unconditionally secure protocols'. In: *STOC '88*. pp. 11–19.

Ciobâcă, S., S. Delaune, and S. Kremer: 2009, 'Computing Knowledge in Security Protocols under Convergent Equational Theories'. In: *CADE-22*, Vol. 5663 of *LNCS*. pp. 355–370.

Clarke, E. M., O. Grumberg, and D. A. Peled: 1999, *Model Checking*. The MIT Press.

Clarke, E. M., S. Jha, and W. R. Marrero: 1998, 'Using state space exploration and a natural deduction style message derivation engine to verify security protocols'. In: *PROCOMET '98*. pp. 87–106.

Cohen, M. and M. Dam: 2005a, 'A completeness result for BAN Logic'. In: *Methods for Modalities*. pp. 202–219.

Cohen, M. and M. Dam: 2005b, 'Logical Omniscience in the Semantics of BAN Logic'. In: *FCS'05*. pp. 121–132.

Cohen, M. and M. Dam: 2007, 'A Complete Axiomatization of Knowledge and Cryptography'. In: *LICS*. pp. 77–88.

Cohen, M., M. Dam, A. Lomuscio, and F. Russo:: 2009a, 'Abstraction in Model Checking Multi-agent Systems'. In: *AAMAS '09*. pp. 945–952.

Cohen, M., A. Lomuscio, M. Dam, and H. Qu: 2009b, 'A symmetry reduction technique for model checking temporal epistemic logic'. In: *IJCAI'09*. pp. 721–726.

Cremers, C. J. F.: 2006, 'Scyther - Semantics and Verification of Security Protocols'. Ph.D. dissertation, Eindhoven University of Technology.

Dechesne, F., M. R. Mousavi, and S. Orzan: 2007, 'Operational and Epistemic Approaches to Protocol Analysis: Bridging the Gap'. In: *LPAR*. pp. 226–241.

Dechesne, F., S. Orzan, and Y. Wang: 2008, 'Refinement of Kripke Models for Dynamics'. In: *ICTAC'08*, Vol. 5160 of *LNCS*. pp. 111–125.

Dechesne, F. and Y. Wang: 2007, 'Dynamic epistemic verification of security protocols: framework and case study'. In: *A Meeting of the minds (LORI 2008)*, Vol. 8 of *Texts in Computer Science*. pp. 129–144.

Delaune, S., S. Kremer, and M. Ryan: 2006, 'Coercion-Resistance and Receipt-Freeness in Electronic Voting'. In: *CSFW'06*. pp. 28–42.

Delaune, S., S. Kremer, and M. Ryan: 2009, 'Verifying Privacy-type properties of Electronic Voting Protocols'. *Journal of Computer Security* **17**(4), 435–487.

Dixon, C., Carmen, M. Fisher, and W. van der Hoek: 2003, 'Using Temporal Logics of Knowledge in the Formal Verification of Security Protocols.'. Technical Report ULCS-03-022, University of Liverpool, Department of Computer Science.

Dolev, D. and A. Yao: 1983, 'On the security of public key protocols'. *IEEE Transactions on Information Theory* **29**(2), 198–208.

Durgin, N. A., P. D. Lincoln, J. C. Mitchell, and A. Scedrov: 1999, 'Undecidability of Bounded Security Protocols'. In: *Proceedings of the workshop on Formal Methods and Security Protocols*.

Emerson, E. A.: 1987, 'Uniform inevitability is tree automaton ineffable'. *Inf. Process. Lett.* **24**(2), 77–79.

Engelhardt, K., P. Gammie, and R. van der Meyden: 2007, 'Model Checking Knowledge and Linear Time: PSPACE Cases'. In: *LFCS*. pp. 195–211.

Engelhardt, K., R. Van Der Meyden, and K. Su: 2002, 'Modal Logics with a Linear Hierarchy of Local Propositional Quantifiers'. In: *Advances in Modal Logic*, Vol. 9. pp. 9–30.

Fagin, R., J. Y. Halpern, Y. Moses, and M. Y. Vardi: 1995a, 'Knowledge-Based Programs'. In: *Symposium on Principles of Distributed Computing*. pp. 153–163.

Fagin, R., J. Y. Halpern, M. Y. Vardi, and Y. Moses: 1995b, *Reasoning about knowledge*. Cambridge, MA, USA: MIT Press.

Fischer, M. J. and R. N. Wright: 1996, 'Bounds on secret key exchange using a random deal of cards'. *Journal of Cryptology* **9**, 71–99.

Focardi, R., R. Gorrieri, and F. Martinelli: 2004, *Classification of Security Properties (Part II: Network Security)*, Vol. 2946 of *LNCS*, pp. 139–185. Springer.

Gammie, P. and R. van der Meyden: 2004, 'MCK: Model Checking the Logic of Knowledge'. In: *CAV'04*, Vol. 3114 of *LNCS*. Springer, pp. 256–259.

Garcia, F. D., I. Hasuo, P. van Rossum, and W. Pieters: 2005, 'Provable Anonymity'. In: *Formal Methods in Security Engineering '05*. pp. 63–72.

Gerbrandy, J. and W. Groeneveld: 1997, 'Reasoning about Information Change'. *Journal of Logic, Language and Information* **6**(2), 147–169.

Gong, L., R. Needham, and R. Yahalom: 1990, 'Reasoning about Belief in Cryptographic Protocols'. In: *Research in Security and Privacy*. pp. 234–248.

Halpern, J. and K. O'Neill: 2002, 'Secrecy in multiagent systems'. In: *Proc. 15th IEEE Computer Security Foundations Workshop*. pp. 32–46.

Halpern, J. and K. O'Neill: 2005, 'Anonymity and information hiding in multiagent systems'. *Journal of Computer Security* **13**(3), 483–514.

Halpern, J. Y. and R. Fagin: 1989, 'Modelling knowledge and action in distributed systems'. *Distributed Computing* **3**(4), 159–177.

Halpern, J. Y. and Y. Moses: 1990, 'Knowledge and common knowledge in a distributed environment'. *J. ACM* **37**(3), 549–587.

Halpern, J. Y., Y. Moses, and M. Y. Vardi: 1994, 'Algorithmic knowledge'. In: *TARK '94*. pp. 255–266.

Halpern, J. Y. and R. Pucella: 2003a, 'Modeling Adversaries in a Logic for Security Protocol Analysis'. In: *Formal Aspects of Security*, Vol. 2629 of *LNCS*. Springer, pp. 87–100.

Halpern, J. Y. and R. Pucella: 2003b, 'On the relationship between strand spaces and multi-agent systems'. *ACM Trans. Inf. Syst. Secur.* **6**(1), 43–70.

Halpern, J. Y. and R. Pucella: 2010, 'Dealing With Logical Omniscience: Expressiveness and Pragmatics'. *Artificial Intelligence*.

Halpern, J. Y. and M. Y. Vardi: 1986, 'The complexity of reasoning about knowledge and time'. In: *STOC '86*. pp. 304–315.

Halpern, J. Y. and M. Y. Vardi: 1991, 'Model checking vs. theorem proving: a manifesto'. In: *Artificial intelligence and mathematical theory of computation: papers in honor of John McCarthy*. Academic Press Professional, Inc., pp. 151–176.

Hintikka, J.: 1962, *Knowledge and Belief: An Introduction to the Logic of the Two Notions*. Ithaca N.Y.: Cornell University Press.

Hommersom, A., J.-J. Meyer, and E. Vink: 2005, 'Update Semantics of Security Protocols'. In: *Information, Interaction and Agency*. Springer, pp. 289–327.

Hoshi, T. and A. Yap: 2009, 'Dynamic epistemic logic with branching temporal structures'. *Synthese* **169**(2), 259–281.

Hunter, A. and J. P. Delgrande: 2007, 'Belief Change and Cryptographic Protocol Verification'. In: *AAAI*. pp. 427–433.

Javier, J. C. Herzog, and J. D. Guttman: 1999, 'Strand spaces: proving security protocols correct'. *Journal of Computer Security* **7**(2-3), 191–230.

Jonker, H. and W. Pieters: 2006, 'Receipt-Freeness as a special case of Anonymity in Epistemic Logic'. In: *Workshop On Trustworthy Elections 2006*.

Jonker, H. L. and E. P. de Vink: 2006, 'Formalising Receipt-Freeness'. In: *Information Security*, Vol. 4176 of *LNCS*. pp. 476–488.

Kacprzak, M., W. Nabiałek, A. Niewiadomski, W. Penczek, A. Półrola, M. Szreter, B. Woźna, and A. Zbrzezny: 2008, 'VerICS 2007 - a Model Checker for Knowledge and Real-Time'. *Fundamenta Informaticae* **85**(1), 313–328.

Kramer, S.: 2007, 'Logical concepts in cryptography'. Ph.D. thesis, EPFL.

Läuchli, H. and C. Savioz: 1987, 'Monadic Second Order Definable Relations on the Binary Tree'. *The Journal of Symbolic Logic* **52**(1), 219–226.

Lomuscio, A. and W. Penczek: 2007, 'Symbolic model checking for temporal-epistemic logics'. *SIGACT News* **38**(3), 77–99.

Lomuscio, A., H. Qu, and F. Raimondi: 2009, 'MCMAS: A Model Checker for the Verification of Multi-Agent Systems'. In: *CAV'09*, Vol. 5643 of *LNCS*. pp. 682–688.

Lomuscio, A. and F. Raimondi: 2006a, 'The complexity of model checking concurrent programs against CTLK specifications'. In: *AAMAS '06*. pp. 548–550.

Lomuscio, A. and F. Raimondi: 2006b, 'MCMAS: A Model Checker for Multi-agent Systems'. In: *TACAS '06*, Vol. 3920 of *LNCS*. pp. 450–454.

Lowe, G.: 1996, 'Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR'. In: *TACAS '96*, Vol. 1055 of *LNCS*. pp. 147–166.

Needham, R. M. and M. D. Schroeder: 1978, 'Using encryption for authentication in large networks of computers'. *Commun. ACM* **21**(12), 993–999.

Orzan, S.: 2005, 'LYS'. Available from `http://www.mobanet.nl/simona/lys/`.

Parikh, R. and R. Ramanujam: 1985, 'Distributed Processes and the Logic of Knowledge'. In: *Logic of Programs*, Vol. 193 of *LNCS*. pp. 256–268.

Parikh, R. and R. Ramanujam: 2003, 'A knowledge based semantics of messages'. *Journal of Logic, Language and Information* **12**(4), 453–467.

Paulson, L. C.: 1997, 'Proving Properties of Security Protocols by Induction'. In: *10th Computer Security Foundations Workshop*. pp. 70–83.

Paulson, L. C.: 1998, 'The inductive approach to verifying cryptographic protocols'. *Journal of Computer Security* **6**, 85–128.

Petride, S. and R. Pucella: 2007, 'Perfect cryptography, S5 knowledge, and algorithmic knowledge'. In: *TARK '07*. pp. 239–247.

Plaza, J. A.: 1989, 'Logics of public communications'. In: *Proceedings of the 4th International Symposium on Methodologies for Intelligent Systems*. pp. 201–216.

Pucella, R.: 2006, 'Deductive Algorithmic Knowledge'. *Journal of Logic and Computation* **16**(2), 287–309.

Ramanujam, R. and S. P. Suresh: 2005a, 'Decidability of context-explicit security protocols'. *J. Comput. Secur.* **13**(1), 135–165.

Ramanujam, R. and S. P. Suresh: 2005b, 'Deciding knowledge properties of security protocols'. In: *TARK '05*. pp. 219–235.

Reiter, M. K. and A. D. Rubin: 1998, 'Crowds: Anonymity for Web Transactions.'. *ACM Transactions on Information and System Security* **1**, 66–92.

Ryan, P. and S. Schneider: 2001, *Modelling and analysis of security protocols*. Addison Wesley.

Shilov, N. V. and N. O. Garanina: 2002, 'Model checking knowledge and fixpoints'. In: *FICS*, Vol. NS-02-2 of *BRICS Notes Series*. pp. 25–39.

Shmatikov, V.: 2004, 'Probabilistic Model Checking of an Anonymity System'. *Journal of Computer Security* **12**(3/4), 355–377.

Su, K.: 2004, 'Model Checking Temporal Logics of Knowledge in Distributed Systems'. In: *AAAI*. pp. 98–103.

Syverson, P. F.: 1992, 'Knowledge, Belief, and Semantics in the Analysis of Cryptographic Protocols'. *Journal of Computer Security* **1**(3-4), 317–334.

Syverson, P. F. and S. G. Stubblebine: 1999, 'Group Principals and the Formalization of Anonymity'. In: *World Congress on Formal Methods*, Vol. 1708 of *LNCS*. pp. 814–833.

Teepe, W.: 2006, 'BAN Logic is Not 'Sound', Constructing Epistemic Logics for Security is Difficult'. In: *Proceedings of FAMAS'06*. pp. 79–91.

van Benthem, J., J. Gerbrandy, T. Hoshi, and E. Pacuit: 2009, 'Merging Frameworks for Interaction'. *Journal of Philosophical Logic* **38**(5), 491–526.

van Benthem, J., J. Gerbrandy, and E. Pacuit: 2007, 'Merging frameworks for interaction: DEL and ETL'. In: *TARK '07*. pp. 72–81.

van Benthem, J., J. van Eijck, and B. Kooi: 2006, 'Logics of communication and change'. *Information and Computation* **204**(11), 1620–1662.

van der Hoek, W. and M. Wooldridge: 2002, 'Tractable multiagent planning for epistemic goals'. In: *AAMAS '02*. pp. 1167–1174.

van der Meyden, R. and N. Shilov: 1999, 'Model Checking Knowledge and Time in Systems with Perfect Recall'. In: *FSTTCS '99*, Vol. 1738 of *LNCS*. pp. 432–445.

van der Meyden, R. and K. Su: 2004, 'Symbolic Model Checking the Knowledge of the Dining Cryptographers'. In: *CSFW '04*. pp. 280–291.

van der Meyden, R. and T. Wilke: 2007, 'Preservation of epistemic properties in security protocol implementations'. In: *TARK '07*. pp. 212–221.

van Ditmarsch, H.: 2003, 'The Russian Cards Problem'. *Studia Logica* **75**(1), 31–62.

van Ditmarsch, H.: 2008, 'Unconditionally secure protocols with card deals'. Presentation, available at `http://www.cs.otago.ac.nz/staffpriv/hans/lorentz/niaslorentz.pdf`.

van Ditmarsch, H., W. van der Hoek, R. van der Meyden, and J. Ruan: 2006, 'Model Checking Russian Cards'. In: *MoChArt 05*, Vol. 149(2) of *ENTCS*. pp. 105–123.

van Eijck, J.: 2005, 'DEMO program and documentation'. Available from `http://www.cwi.nl/~jve/demo/`.

van Eijck, J. and S. Orzan: 2007, 'Epistemic Verification of Anonymity'. *Electron. Notes Theor. Comput. Sci.* **168**, 159–174.

Von Wright, G. H.: 1951, *An Essay in Modal Logic*. Amsterdam: North Holland.

Wang, Y., L. Kuppusamy, and J. van Eijck: 2009, 'Verifying epistemic protocols under common knowledge'. In: *TARK '09*. pp. 257–266.

Wang, Y., F. Sietsma, and J. van Eijck: 2010, 'Logic of Information Flow on Communication Channels (extended abstract)'. In: *AAMAS '10*. to appear.