Modal tableaux for verifying stream authentication protocols

Guido Governatori[1], Mehmet A. Orgun[2] and Chuchang Liu[3]

[1] School of ITEE, The University of Queensland, Brisbane, QLD 4072, Australia
[2] Department of Computing, Macquarie University, Sydney, NSW 2109, Australia
[3] C3I Division, DSTO, Edinburgh, SA 5111, Australia

Abstract. To develop theories to specify and reason about various aspects of multi-agent systems, many researchers have proposed the use of modal logics such as belief logics, logics of knowledge, and logics of norms. As multi-agent systems operate in dynamic environments, there is also a need to model the evolution of multi-agent systems through time. In order to introduce a temporal dimension to a belief logic, we combine it with a linear-time temporal logic using a powerful technique called fibring for combining logics. We describe a labelled modal tableaux system for the resulting fibred belief logic (FL) which can be used to automatically verify correctness of inter-agent stream authentication protocols. With the resulting fibred belief logic and its associated modal tableaux, one is able to build theories of trust for the description of, and reasoning about, multi-agent systems operating in dynamic environments.

Keywords: belief logic, temporal logic, fibring logics, system-specific trust theories, modal tableaux, security protocols.

1 Introduction

Multi-agent systems (MASs for short) consist of a collection of agents that interact with each other in dynamic and unpredictable environments. Agents communicate with one another by exchanging messages, and they have the ability to cooperate, coordinate and negotiate with each other to achieve their objectives. A fundamental problem regarding security of communication in MAS is that whether a message sent by an agent is reliably received by other agents and whether the message received is regarded as reliable in the view of receivers. The problem generally depends on the trust that agents would put in the security mechanisms of a system [10,27]. The notion of trust is fundamental for understanding the interactions between agents such as human beings, machines, organizations, and other entities [32].

In order to develop theories to specify and reason about various aspects of multi-agent systems including security and trust, many researchers have proposed the use of modal logics such as belief logics [5,9] and logics of knowledge [6,19]. Many of such formalisms have successfully been used for dealing with some particular aspects of agents, but they generally ignore other aspects that are also important. For instance, stream authentication protocols used for communication between agents are often different from the standard class of authentication protocols previously analysed by many

To appear: Journal of Autonomous Agents and Multi-Agent Systems

researchers using belief logics and/or model checking techniques. They involve not only a key management scheme but also its critical use of timing. Therefore it is essential for a logic for an agent-based system to have the ability for modelling and reasoning about the evolution of the system in which they are applied [12].

Several researchers have recently proposed approaches based on temporal epistemic logics for analysing security protocols such as the TESLA (Timed Efficient Stream Loss-tolerant Authentication) protocol [26] and the Needham-Schroeder protocol [8]. In our opinion, these logics are steps in the right direction. It has also been argued that any logical system used for modeling active agents should be a combined system of logics of knowledge, belief, time and norms [14] since these are among the essential concepts to be reasoned about. There is therefore a pressing need for a systematic approach through which salient features of security protocols could be identified, reasoned about, and generalised to other protocols. Accordingly, the aim of this paper is not to provide a complete analysis of stream authentication protocols, but to show how combinations of multi-modal logics can provide effective tools for this task. Specifically our aim is to illustrate how to provide solutions based on logic for analysing communication protocols for autonomous agent-based systems; we are motivated to study suitable logics for specifying and reasoning about agent beliefs, norms and time in dynamic and multi-agent systems; to identify sufficient techniques for combining those logics; to investigate methods for formalizing authentication protocols with theories of trust in combined logics; and to develop techniques for reasoning about security properties that such a protocol may satisfy.

One particular combination technique (called *adding a temporal dimension* or *temporalisation* [11]) can be used to combine logics in a hierarchical way and it could lead to a combined logic suitable for a particular domain. For instance, Liu *et al.* [25] have proposed a *temporalized* belief logic that provides a logical framework for users to specify the dynamics of trust and model evolving theories of trust for multi-agent systems. However, in this logic there are certain restrictions on the use of temporal and belief operators because of the hierarchical combination of belief and temporal logics used. Temporal operators can never be within the scope of a belief operator, hence we cannot express a statement asserting that some agent believes an event to happen at some time, e.g., the logic does not have a formula such as $\mathbf{B}_{john}\,\mathbf{first}\,holds(bob,k)$, which could be used to express an assertion that "*John* believes that at the initial time *Bob* holds the key *k*". Such kind of assertions are often needed, for example, in analysing security protocols, in particular, stream authentication protocols, therefore temporalisation is unlikely to be sufficient as a combination technique in the general case.

We therefore consider a more powerful technique called fibring [14] for combining logics of beliefs, knowledge, norms and time that treats operators of each logic equally in the resulting combined logic. Fibring also allows for the development of proof procedures such as (labelled) tableaux from the constituent logics in a rigorous way. In this paper, we combine, using the fibring technique, the logic TML [25], a variant of the modal logic **KD** of beliefs, with the temporal logic SLTL which is suitable for specifying events, such as those found in reactive systems, that may run on different clocks (time-lines) of varying rates of progress [23,24]. We show that in the resulting fibred belief logic (FL) we can specify and reason about not only dynamics of agent beliefs but

also the timing properties of a system effectively. In order to analyse stream authentication protocols it is necessary to have a logic that can satisfactorily deal with all aspects of those concepts. We also describe a labelled modal tableaux system for FL which can be used to automatically verify correctness of inter-agent stream authentication protocols. Tableaux based proof procedures are well suited to implementing practical proof procedures and they benefit from the underlying Kripke structures effectively [3].

The fibred logic developed in this paper naturally has more expressive power than a temporalised belief logic such as TML$^+$ [25] does. It allows us to express the evolution of beliefs of an agent as well as the beliefs of agents about the evolution of a system. With this logical system one is able to build theories of trust for the description of, and reasoning about, multi-agent systems operating in dynamic environments. For further discussion and motivation on the combination of logics and their relative merits, we refer the reader to the literature [3,7].

The rest of the paper is organised as follows. Section 2 introduces the TESLA stream authentication protocol. Section 3 briefly discusses two logics SLTL and TML. Section 4 presents the fibring technique as specifically applied for combining TML with SLTL, and provides an axiomatisation for the fibred logic called FL. Section 5 adapts KEM [2,15], a labelled modal tableaux system to reason with FL. Section 6 develops a theory of trust in FL for specifying the TESLA protocol and discusses its correctness. Section 7 concludes this paper with a brief summary.

2 The TESLA Protocol

Multi-agent systems, typically real world systems, need to employ application specific protocols for transferring data, such as video, audio and sensory data, among agents. As an example protocol, we consider TESLA (Timed Efficient Stream Loss-tolerant Authentication), a multicast stream authentication protocol proposed by Perrig *et al.* [31]. In TESLA, authentication is based on the timing of the publication of keys and the indirect relation of each new key to an original key commitment. The process for verifying data packets received to be authentic depends on trust of the receiver in the sender, and belief on whether an intruder can have prior knowledge of a key before it is published by the protocol. Note that TESLA does not address denial-of-service attacks.

Perrig *et al.* [31] proposed five schemes for the TESLA Protocol. We consider scheme I (called The Basic Scheme), and simply call it the PCTS (Perrig-Canetti-Tygar-Song) scheme in the rest of the paper. As noted by Perrig, the PCTS scheme cannot tolerate packet loss. In other words, once a packet is lost no further packets can be authenticated. Other schemes of TESLA such as those for tolerating packet loss and achieving faster transfer rates will be analysed in future work.

In the following, the symbols S, R and I are always used to denote the sender, the receiver and the intruder, respectively. We also use the tuple $\langle X,Y\rangle$ to denote the concatenation of X and Y, where X and Y are the parts to be concatenated. A stream α is divided into chunks M_i (called messages), so we may have $\alpha = \langle M_1, M_2, \ldots, M_l\rangle$. A message authentication code (MAC) is derived by applying an authentication scheme, together with a secret key, to a message.

In the PCTS scheme, the basic idea is that each message M_i is sent in a packet P_i, along with additional authentication information [4,31]. The sender issues a signed commitment to a key. The key is only only known to the sender. To send message M_i, the sender uses that key to compute a MAC on a packet P_i, and later discloses the key in packet P_{i+1}, which enables the receiver (or receivers, when multiple receivers are involved) to verify the commitment and the MAC of packet P_i. A successful verification will imply that packet P_i is authenticated and trusted.

The sequence of message packets can be formulated as follows:

$$
\begin{array}{lll}
P_{0r}. & R \rightarrow S: & \langle n_R \rangle \\
P_{0s}. & S \rightarrow R: & \langle \{ f(K_1), n_R \}_{SK(S)} \rangle \\
P_1. & S \rightarrow R: & \langle \langle M_1, f(K_2) \rangle, MAC(K'_1, \langle M_1, f(K_2) \rangle) \rangle \\
 & & \vdots \\
P_i. & S \rightarrow R: & \langle D_i, MAC(K'_i, D_i) \rangle \text{ (for all } i \geq 2) \\
 & & \vdots
\end{array}
$$

where $D_i = \langle M_i, f(K_{i+1}), K_{i-1} \rangle$ for all $i \geq 2$, $K'_j = f'(K_j)$ for all $j \geq 1$, and f and f' are two different pseudo-random functions.

In packet P_{0r}, the receiver R sends his nonce n_R (generated by himself) to the server S for ensuring freshness. In packet P_{0s}, the signature on $f(K_1)$ acts as an authenticated commitment to K_1. Thus, when R receives K_1 in P_2, he can be sure that it was really sent by S.

Apart from the initial messages P_{0r}, P_{0s} and P_1, any packet P_i has the standard form $\langle D_i, MAC(K'_i, D_i) \rangle$ for all $i \geq 2$. To send message M_i, the sender first picks a fresh random key K_{i+1} to construct packet $P_i = \langle D_i, MAC(K'_i, D_i) \rangle$, then sends the packet to the receiver. When the packet P_i is received, the receiver cannot verify the MAC immediately, since it cannot reconstruct K'_i without knowing K_i, which is contained in packet P_{i+1}. Therefore, only once P_{i+1} is received, the receiver is able to verify the MAC. Packet $P_{i+1} = \langle D_{i+1}, MAC(K'_{i+1}, D_{i+1}) \rangle$ discloses K_i and allows the receiver first to verify that K_i is correct ($f(K_i)$ equals the commitment which was sent in P_{i-1}); and second to compute $K'_i = f'(K_i)$ and check the authenticity of packet P_i by verifying the MAC of P_i.

In analysing the TESLA protocol it is assumed that [31]:

- The sender is honest and works correctly, following all requirements of the protocol strictly.
- The receiver accepts packet P_i as authentic only when it believes the key commitment and the MAC of the packet have been successfully verified. A stream $\alpha = \langle M_1, M_2, \ldots, M_l \rangle$ is considered valid iff the receiver accepts that all packets $P_1, P_2, \ldots, P_l$ are successfully authenticated.
- The intruder has the ability to capture, drop, resend, delay, and alter packets, can access to a fast network with negligible delay, and can perform efficient computations, such as computing a reasonable number of pseudo-random function applications and MACs with negligible delay. Nonetheless, the intruder cannot invert a pseudo-random function with non-negligible probability.

The security property for the TESLA protocol we need to guarantee is that the receiver does not believe any packet P_i to be authenticated unless the message M_i contained in the packet was actually sent by the sender. To prevent any successful attack by an intruder, the receiver only needs to be sure that all packets P_i arrive safely such that the intruder has no time to change the message and commitment in P_i and forge the subsequent traffic.

To analyse the TESLA protocol and show that a particular scheme is secure, we first need a formal framework for formalizing the protocol. As we have mentioned earlier, we consider fibring of a belief logic with a temporal logic in a systematic way.

3 Two Logics: SLTL and TML

We now give a brief introduction to the logics SLTL and TML. Typed Modal Logic (TML) is a variant of the modal logic **KD** of beliefs [25] which is suitable for modelling agent beliefs. In Simple Linear-Time Temporal Logic (SLTL), truth values of formulae vary over clocks (time-lines) of varying rates of progress [23,24].

The choice of these two logics, and consequently of their combination, is motivated by the need to capture some basic notions used in the representation of the TESLA protocols: in the first instance, the notion that an agent believes that a message has been authenticated. The protocols do not impose any condition on the property of beliefs. Thus we have opted for the minimal modal logic for a belief operator satisfying the conditions that the agent is rational (i.e., does not have inconsistent beliefs). Similarly, the TESLA protocols, are based on sequences of messages, and for the temporal properties we need to be able to capture the time when the first packet of a message has been sent (and we can identify this instant with the origin of time, as far as the protocol and the message are concerned), and the packet sent at successive steps. To model this we require two temporal operators: **first** for the origin and **next** for modelling the successive instant in a time-line. The **next** operator is standard in temporal logic; **first** (used in conjunction with successive applications of **next**) allows us to move to any instant along the time-line.

3.1 SLTL: Simple Linear-time Temporal Logic

SLTL offers two operators, **first** and **next**, which refer to the initial moment and the next moment in time respectively. The formulae of SLTL are built with the usual formation rules from standard connectives and quantifiers of classical first order logic, and the temporal operators **first** and **next**.

The collection of moments in time is the set of natural numbers. We define the global clock as the increasing sequence of natural numbers, i.e., $\langle 0,1,2,\ldots\rangle$, and a local clock is an infinite subsequence of the global clock. Thus, we have

Definition 1 (time models) *A time model for the logic SLTL has the form* $\mathbf{c} = \langle C, <, \nu\rangle$, *where* $C = (t_0, t_1, t_2, \ldots)$ *is a clock,* $<$ *is the usual ordering relation over C and* ν *is an assignment function giving a value* $\nu(t,q) \in \{true, false\}$ *for any atomic formula q at time t in C.*

We write $\mathbf{c}, t \models A$ to stand for "A is true at time t in the model $\mathbf{c}$". Then the semantics of the temporal operators with the notion of satisfaction in SLTL is given as follows:

- $\mathbf{c}, t_i \models$ **first** A iff $\mathbf{c}$, $t_0 \models A$.
- $\mathbf{c}, t_i \models$ **next** A iff $\mathbf{c}$, $t_{i+1} \models A$.
- satisfaction in the model $\mathbf{c} = (C, <, \nu)$ is defined as satisfaction at some point on C.

A minimal axiomatic system for the propositional temporal logic consists of the following axioms (axiom schemata). We let $\bigtriangledown$ stand for **first** or **next**.

A0. all axioms of classical first order logic.
A1. $\bigtriangledown($**first** $A) \leftrightarrow$ **first** A.
A2. $\bigtriangledown(\neg A) \leftrightarrow \neg(\bigtriangledown A)$.
A3. $\bigtriangledown(A \wedge B) \leftrightarrow (\bigtriangledown A) \wedge (\bigtriangledown B)$.

Axiom A0 says that initial truths persist. Axiom A2 says that temporal operators are self-dual (over functional frames). Axiom A3 says that the temporal operators commute with $\wedge$.

Apart from the generic substitution rule, SLTL has two rules of inference defined as follows:

MP. From φ and $\varphi \rightarrow \psi$ infer ψ. (Modus Ponens)
TG. From φ infer $\bigtriangledown \varphi$ (Temporal Generalisation)

The soundness and completeness of the axiomatisation system for SLTL with respect the class $\mathscr{C}$ of all local clocks are straightforward [23].

3.2 TML: Typed Modal Logic

TML is a variant of the modal logic **KD** of beliefs [20]. We assume that there are n agents $a_1, \ldots, a_n$ and, correspondingly, n modal operators $\mathbf{B}_1, \ldots, \mathbf{B}_n$ in the logic, where $\mathbf{B}_i$ $(1 \leq i \leq n)$ stands for "agent a_i believes that".

Let Φ_0 be a set of countably many atomic formulae of the (typed[4]) first-order logic.

Definition 2 *syntax We define $\mathscr{L}_{tml}$ as the smallest set of well-formed formulae (wffs) of the logic TML such that:*

- $\Phi_0 \subset \mathscr{L}_{tml}$*;*
- *if φ is in $\mathscr{L}_{tml}$, so are $\neg\varphi$ and $\mathbf{B}_i\varphi$ for all i $(1 \leq i \leq n)$;*
- *if φ and ψ are in $\mathscr{L}_{tml}$, then $\varphi \wedge \psi$ is in $\mathscr{L}_{tml}$; and*
- *if $\varphi(X)$ is in $\mathscr{L}_{tml}$ where X is a free variable, then $\forall X \varphi(X)$ is, too.*

We assume the fixed-domain approach to quantification, that is, the domain of quantification is the same in all possible worlds. This mean than we have the standard first-order logic semantics for the $\forall$ and $\exists$ quantifiers. We also employ rigid denotations for terms, that is, the only dynamic objects are predicates. We also assume that in TML all the wffs built according to the usual formation rules are correctly typed.

[4] By typed we mean that the domain objects (values) are classified into types and each term can only have values of a certain type (e.g., messages).

Definition 3 (Kripke model) *A classical Kripke model [21] for the logic TML is a tuple*

$$\mathbf{m} = \langle S, D, R_1, \ldots, R_n, \pi \rangle,$$

where S is the set of states or possible worlds; D is the domain of the model; each $R_i (1 \leq i \leq n)$ *is a serial relation over S, consisting of state pairs* (s,t) *such that* $(s,t) \in R_i$ *iff, at state s, agent* a_i *considers the state t possible;* π *is the* interpretation function*, which associates every term of the language an element of the domain, and gives the extension of predicates for every possible world.*

- $\pi(x,s) \in D$ *(global interpretation of variables/terms);*
- $\forall s, r' \in W, \pi(x,s) = \pi(x,r)$ *(rigidity of variables/terms);*
- $\pi(\phi(x_1, \ldots, x_n), s) \subseteq D^n$.

Each R_i *called the possibility relation according to agent* a_i*. We write* $\mathbf{m}, s \models \varphi$ *to stand for "*φ *is true at the state s in the model* $\mathbf{m}$*" or "*φ *holds at s in* $\mathbf{m}$*".*

The semantics definition for the belief operators with the notion of satisfaction in TML given an interpretation π is as follows:

- $\mathbf{m}, s \models \varphi(x_1, \ldots, x_n)$ iff $\langle \pi(x_1, s), \ldots, \pi(x_n, s) \rangle \in \pi(\varphi, s)$;
- Standard valuation conditions for negation and boolean connectives;
- $\mathbf{m}, s \models \mathbf{B}_i \varphi$ iff, for all t such that $(s,t) \in R_i$, $\mathbf{m}, t \models \varphi$.
- $\mathbf{m}, s \models \forall x A(x)$ iff for every $d \in \mathscr{T}$ (where $\mathscr{T}$ is the type of x), $\mathbf{m}, s \models A(x)$, based on the interpretation $\pi_{d/x}$ like π except for mapping x to d.
- A formula φ is satisfiable in a model $\mathbf{m}$ if there exists $s \in S$ such that $\mathbf{m}, s \models \varphi$.

In preparation for fibring TML with SLTL, we now consider monadic models for TML defined as follows:

Definition 4 (monadic models) *A monadic model for TML is a structure*

$$\mathbf{m} = \langle S, D, R_1, \ldots, R_n, \pi, u \rangle$$

where $\langle S, D, R_1, \ldots, R_n, \pi \rangle$ *is a classical model for TML and* $u \in S$ *is called the actual world.* φ *is satisfiable in the monadic model* $\mathbf{m}$ *if and only if* $\mathbf{m}, u \models \varphi$.

We define $\mathscr{K}_{tml}$ as a class of monadic models of the form $\langle S, R_1, \ldots, R_n, \pi, u \rangle$, where the set of states is assumed to be

(1) $S = \{x \mid \exists R_{i_1} \ldots R_{i_k}\ u R_{i_1} \circ \ldots \circ R_{i_k} x, R_{i_1}, \ldots, R_{i_k} \in \{R_1, \ldots, R_n\}\}$,

where $R_i \circ R_j$ represents the relative product (or composition) of R_i and R_j. This means that a monadic models in u is connected and rooted, where the root is u. Alternatively, a monadic model is a generated model.

Note that this assumption does not affect satisfaction in models as points not accessible from u by any composition of some of these relations, $R_1, \ldots$, and R_n, can not affect truth values of a formula at u. Furthermore, using the notation $\mathbf{m}$ for a model in $\mathscr{K}_{tml}$, we write $\mathbf{m} = \langle S^{(\mathbf{m})}, R_1^{(\mathbf{m})}, \ldots, R_n^{(\mathbf{m})}, \pi^{(\mathbf{m})}, u^{(\mathbf{m})} \rangle$, and we also use the notation $\mathbf{m}^u$ to denote the model rooted in u. In addition we assume:

(2) if $\mathbf{m}_1 \neq \mathbf{m}_2$, then $S^{(\mathbf{m}_1)} \cap S^{(\mathbf{m}_2)} = \emptyset$.
(3) $\mathbf{m}_1 = \mathbf{m}_2$ iff $u^{(\mathbf{m}_1)} = u^{(\mathbf{m}_2)}$.

Assumption (2) indicates that all sets of possible worlds in $\mathscr{K}_{tml}$ are all pairwise disjoint, and that there are infinitely many isomorphic (but disjoint) copies of each model; assumption (3) means that a model in $\mathscr{K}_{tml}$ can in fact be identified by the actual world in it.

TML is axiomatised by the following axiom schemata and inference rules:

B0. all axioms of classical first-order logic.
B1. $\mathbf{B}_i(\varphi \rightarrow \psi) \wedge \mathbf{B}_i\varphi \rightarrow \mathbf{B}_i\psi$ for all i $(1 \leq i \leq n)$.
B2. $\mathbf{B}_i(\neg\varphi) \rightarrow \neg(\mathbf{B}_i\varphi)$ for all i $(1 \leq i \leq n)$.
B3. $\forall X\mathbf{B}_i\varphi(X) \rightarrow \mathbf{B}_i\forall X\varphi(X)$ for all i $(1 \leq i \leq n)$.
I1. From φ and $\varphi \rightarrow \psi$ infer ψ. (Modus Ponens)
I2. From $\forall X\varphi(X)$ infer $\varphi(Y)$. (Instantiation)
I3. From $\varphi(X)$ infer $\forall X\varphi(X)$. (Generalisation)
I4. From φ infer $\mathbf{B}_i\varphi$ for all i $(1 \leq i \leq n)$. (Necessitation)

The soundness and completeness of the axiomatisation system for TML can be proved in a standard pattern [20].

4 FL: Fibred Logic

In this section, we discuss how the logic FL is obtained through the use of fibring technique for combining the logics TML and SLTL. The formulae of FL may contain any number of applications of the temporal operators and/or the belief operators without any restrictions. To be able to interpret a formula of TML whose main operator is a temporal operator, we need to use the meaning of the temporal operators with a time reference. To be able to interpret a formula whose main operator is a belief operator, similarly, we need to use the meaning of the belief operators with a state reference. This will require us to move between time references and state references freely. The fibring method [14] is used to interweave the semantics of the constituent logics using fibring functions (that move context between time references and state references) in such a way that any formula of FL is interpreted in its proper context. What the fibred logic (FL for short) looks like depends on the conditions on the fibring and assignment functions $\mathbf{F}$ and π, which will be discussed later.

Let $\mathscr{O} = \{\mathbf{B}_1, \ldots, \mathbf{B}_n, \mathbf{first}, \mathbf{next}\}$ be the set of modal connectives of FL. Then the formulae of FL are obtained from the usual formation rules. As before we assume that in FL all the wffs are correctly typed. Then we have

Definition 5 (syntax) *Let Φ_0 be a set of countably many atomic formulae of the (typed) first-order logic, then $\mathscr{L}_{fl}$ as the smallest set of well-formed formulae (wffs) of the logic FL is defined by the following formation rules:*

1. *$\Phi_0 \subset \mathscr{L}_{fl}$;*
2. *if φ is in $\mathscr{L}_{fl}$, so are $\neg\varphi$ and $\bigtriangledown\ \varphi$ for all $\bigtriangledown \in \mathscr{O}$;*
3. *if φ and ψ are in $\mathscr{L}_{fl}$, then $\varphi \wedge \psi$ is in $\mathscr{L}_{fl}$; and*

4. *if $\varphi(X)$ is in $\mathscr{L}_{fl}$ where X is a free variable, then $\forall X\varphi(X)$ is, too.*

The discussion of the fibred semantics in the case of the Kripke monadic models for TML with time models for SLTL can be laid out in three levels: using a single time model, or considering a set of time models with the same clock, or based on different clock models. In this paper we restrict ourselves to the first level. Following Gabbay [14], we define the fibred semantics arising from the Kripke models for TML with a single time model based on *simplified fibred models* (simply, *sfm models*) defined as follows:

Definition 6 (sfm models) *A* simplified fibred model *or* sfm model *is a tuple*

$$(W, D, S, R, \pi, \mathbf{F})$$

where

1. *W is a set of possible worlds;*
2. *D is a (typed) domain of individuals;*
3. *S is a function giving for each w a set of possible worlds, $S^w \subseteq W$;*
4. *R is a function giving for each w, a relation $R^w \subseteq S^w \times S^w$;*
5. *π is an interpretation function as in Definition 3;*
6. *$\mathbf{F}$, the fibring function, is a function $\mathbf{F} : \mathscr{O} \times W \mapsto W$. A fibring function $\mathbf{F}$ is a function giving for each $\bigtriangledown$ and each $w \in W$ another point (actual world) in W as follows:*

$$\mathbf{F}(\bigtriangledown, w) = \begin{cases} w & \text{if } w \in S^u \text{ and } u^{(\mathbf{m})} \text{ is a semantics for } \bigtriangledown \\ w' & \text{otherwise, where } w' \in S^v, \text{ and } v^{(\mathbf{m})} \text{ is a semantics for } \bigtriangledown. \end{cases}$$

The above definition is based on the above intuition: (i) for every world w we associate to it a model $\mathbf{m}^{(w)} = (S^w, R_w, \pi, w)$ and we have to ensure that it is a model for one of the logic to be combined; and (ii) whenever we evaluate a modal operator, first we check the type of model we are in. If the model is a semantics for the modal operator, we use the standard evaluation conditions for it; otherwise we use the fibring function to move to a new model (world) in the appropriate semantics.

An sfm model for FL can be generated from fibring the semantics for SLTL and TML. The detailed construction runs as follows

Definition 7 (Model for FL) *Let $\mathscr{B}$ and $\mathscr{C}$ be respectively set of monadic modals and time model. Let W_b be the set of worlds in $\mathscr{B}$ and W_t the set of natural number in $\mathscr{C}$. Then an sfm-model for FL is*

$$\langle W, D, S, R_0, R_1, \ldots, R_n, \pi, \mathbf{F}, w_0 \rangle$$

where

1. *$W = W_t \cup W_b$;*
2. *D is the set of the (typed) individual common to all models in $\mathscr{B}$ and $\mathscr{C}$;*
3. *For $s \in W_b$, $S^s = \{x | sR_{i_1} \circ \cdots \circ R_{i_k} x$, for some $R_{i_1}, \ldots, R_{i_k} \in \{R_1, \ldots, R_n\}\}$ such that (1) $S^{(s)} \cap W_t = \emptyset$, (2) for $s, r \in W_b$, if $s \neq r$, then $S^{(s)} \cap S^{(r)} = \emptyset$;*

4. $R_0 = \{(x,y) | x,y \in W_t \& x < y\}$;
5. $R_i = \bigcup_{R_i \in \mathscr{B}}$;
6. $\mathbf{F}$ *is such that,*
   - *if* $\bigtriangledown$ *is* **first** *or* **next**, *then* $\mathbf{F}(\bigtriangledown, w) = w$ *if* $w \in W_t$, *otherwise* $\mathbf{F}(\bigtriangledown, w) \in W_b$;
   - *if* $\bigtriangledown$ *is* $\mathbf{B}_i$, *then* $\mathbf{F}(\bigtriangledown, w) = w$ *if* $w \in W_b$, *otherwise* $\mathbf{F}(\bigtriangledown, w) \in W_t$.

Notice that condition 3 means that for each $s \in W_b$ we associate a model of TML which is an isomorphic copy of a submodel generated by s.

Definition 8 (semantics) *The semantics of formulae for the logic FL is defined inductively with respect to an sfm-model* $\langle W, R_0, R_1, \ldots, R_n, \pi, \mathbf{F}, w_0 \rangle$. *For any* $w \in W$,

1. *given an intepretation* π, $s \models \varphi(x_1, \ldots, x_n)$ *iff* $\langle \pi(x_1, s), \ldots, \pi(x_n, s) \rangle \in \pi(\varphi, s)$;
2. $w \models \neg\varphi$ *iff it is not the case that* $w \models \varphi$.
3. $w \models (\varphi \wedge \psi)$ *iff* $w \models \varphi$ *and* $w \models \psi$.
4. $w \models \forall X \varphi(X)$ *iff, for all* $d \in \mathscr{T}$, *where* $\mathscr{T}$ *is the type of* X, $w \models \varphi(d)$, *based on the interpretation* $\pi_{d/x}$ *like* π *except for mapping* x *to* d.
5. $w \models \bigtriangledown\ \varphi$ *iff* $\mathbf{F}(\bigtriangledown, w) \models \bigtriangledown\ \varphi$.
6. $w \models$ **first** φ *when* $w \in W_t$ *iff* $\min\{t \mid t \in W_t\} \models \varphi$.
7. $w \models$ **next** φ *when* $w \in W_t$ *iff* $\min\{t \mid w R_0 t\} \models \varphi$.
8. $w \models \mathbf{B}_i \varphi$ *when* $w \notin W_t$ *and* $1 \leq i \leq n$ *iff, for all* s *such that* $w R_i s$, $s \models \varphi$, *assuming* $s \in S^{(\mathbf{m})}$ *and* $\mathbf{m} \in \mathscr{K}_{bl}$.

With the sfm model $\langle W, R_0, R_1, \ldots, R_n, \pi, \mathbf{F}, w_0 \rangle$ we say that it satisfies the formula φ iff $w_0 \models \varphi$. Furthermore, $\mathbf{m} = \langle W, R_0, R_1, \ldots, R_n, \pi, \mathbf{F} \rangle$ is called a *regular fibred semantics model* for the logic FL. We say φ is valid in the model $\mathbf{m}$, and written as $\mathbf{m} \models \varphi$, if, for all $w_0 \in W$, the model $\langle W, R_0, R_1, \ldots, R_n, \pi, \mathbf{F}, w_0 \rangle$ satisfies φ; we say that φ is satisfied in the model $\mathbf{m}$ if, for some $w_0 \in W$, the model $\langle W, R_0, R_1, \ldots, R_n, \pi, \mathbf{F}, w_0 \rangle$ satisfies φ. Let $\mathscr{K}_{fl}$ be the set of regular fibred semantics models which defines the fibred logic FL, then we say φ is valid in the logic FL if, for all $\mathbf{m} \in \mathscr{K}_{fl}$, $\mathbf{m} \models \varphi$.

The axiom set of FL consists of the combination of the axioms for SLTL and TML and their inference rules; we omit the details.

The soundness for the logic FL depends on the soundness theorems for logics TML and SLTL, and is not difficult to prove; the completeness can be proved by the techniques used in Gabbay [14].

5 Labelled Tableaux for FL

In this section we show how to adapt KEM, a labelled modal tableaux system, to reason with FL. The system can be used to automatically check for formal properties of security protocols, in particular for TESLA, in FL.

A tableaux system is a semantic based refutation method that systematically tries to build a (counter-)model for a set of formulae. A failed attempt to refute (invalidate) a set of formulae generates a model where the set of formulae is true. To show that a property A follows from a theory (a protocol) $B_1, \ldots, B_n$ we verify whether a model for $\{B_1, \ldots, B_n, \neg A\}$ exists. If it does not then A is a consequence of the protocol.

In labelled tableaux systems, the object language is supplemented by labels meant to represent semantic structures (possible worlds in the case of modal and temporal logics). Thus the formulae of a labelled tableaux system are expressions of the form $A : i$, where A is a formula of the logic and i is a label. The intuitive interpretation of $A : i$ is that A is true at (the possible world(s) denoted by) i.

KEM [2,16,17] is a labelled tableaux for logics admitting possible world semantics whose inferential engine is based on a combination of standard tableaux linear expansion rules and natural deduction rules supplemented by an analytic version of the cut rule. In addition it utilises a sophisticated but powerful label formalism that enables the logic to deal with a large class of (quantified) modal and non-classical logics. Furthermore the label mechanism corresponds to fibring thus it is possible to define tableaux systems for multi-modal logic by a seamless combination of the (sub)tableaux systems for the component logics of the combination.

5.1 Label Formalism

KEM uses *Labelled Formulas* (*L*-formulae for short), where an *L*-formula is an expression of the form $A : i$, where A is a wff of the logic, and i is a label. For FL we have a type of labels to various modalities for each agent (belief) plus a type of labels for the temporal modalities. The set of atomic labels is

$$\Phi = \Phi_T \cup \bigcup_{i \in Agt} \Phi^i,$$

where $\Phi_T = \{t_0, t_1, \ldots\}$ and every Φ^i is partitioned into (non-empty) sets of variables and constants: $\Phi^i = \Phi^i_V \cup \Phi^i_C$ where $\Phi^i_V = \{W^i_1, W^i_2, \ldots\}$ and $\Phi^i_C = \{w^i_1, w^i_2, \ldots\}$. Finally we add a sets of auxiliary unindexed atomic labels $\Phi^A = \Phi^A_V \cup \Phi^A_C$ where $\Phi^A_V = \{W_1, W_2, \ldots\}$ and $\Phi^A_C = \{w_1, w_2, \ldots\}$. Φ^A will be used in unifications and proofs. Φ_C and Φ_V denote the set of constants and the set of variables.

Definition 9 (labels) *The set of labels $\Im$ is then defined inductively as follows:*

$$\begin{aligned}
\Im = \bigcup \Im_k \textit{ where } \Im_k,\ k \in \mathbb{N}:& \\
&\Im_1 = \Phi_C \cup \Phi_V; \\
&\Im_2 = \Im_1 \times \Phi_C; \\
&\Im_{n+1} = \Im_1 \times \Im_n, (n > 1).
\end{aligned}$$

According to the above definition a label is either a (i) an element of the set Φ_C, or (ii) an element of the set Φ_V, or (iii) a path term (u', u) where (iiia) $u' \in \Phi_C \cup \Phi_V$ and (iiib) $u \in \Phi_C$ or $u = (v', v)$ where (v', v) is a label. As an intuitive explanation, we may think of a label $u \in \Phi_C$ as denoting a world (a *given* one), and a label $u \in \Phi_V$ as denoting a set of worlds (*any* world) in some Kripke model. A label $u = (v', v)$ may be viewed as representing a path from v to a (set of) world(s) v' accessible from v (the world(s) denoted by v).

For any label $u = (v', v)$ we shall call v' the *head* of u, v the *body* of u, and denote them by $h(u)$ and $b(u)$ respectively. If $b(u)$ denotes the body of u, then $b(b(u))$ will denote the body of $b(u)$, and so on. We call each of $b(u)$, $b(b(u))$, etc., a *segment* of u.

The length of a label u, $\ell(u)$, is the number of world-symbols in it. $s^n(u)$ will denote the segment of u of length n and we shall use $h^n(u)$ as an abbreviations for $h(s^n(u))$. Notice that $h(u) = h^{\ell(u)}(u)$. Let u be a label and u' an atomic label. We use $(u';u)$ as a notation for the label (u',u) if $u' \neq h(u)$, or for u otherwise.

For any label $u, \ell(u) > n$, we define the *counter-segment-n* of u, as follows (for $n < k < \ell(u)$):

$$c^n(u) = h(u) \times (\cdots \times (h^k(u) \times (\cdots \times (h^{n+1}(u), w_0))))$$

where w_0 is a dummy label, i.e., a label not appearing in u (the context in which such a notion occurs will tell us what w_0 stands for). The counter-segment-n defines what remains of a given label after having identified the segment of length n with a 'dummy' label w_0. The appropriate dummy label will be specified in the applications where such a notion is used. However, it can be viewed also as an independent atomic label in the set of auxiliary labels.

So far we have provided definitions about the structure of the labels without regard of the elements they are made of. The following definitions will be concerned with the type of world symbols occurring in a label.

We say that a label u is τ-*preferred* iff $\tau = i$ and $h(u) \in \Phi^i$, or $\tau = t$ and $h(u) \in \Phi_T$; a label u is τ-*pure* iff each segment of u of length $n > 1$ is τ-preferred. With $\Im^i$ we denote the set of i-preferred labels where $i \in Agt$.

5.2 Label Unifications

One of the key features of KEM is its logic dependent label unification mechanism. In the same way as each modal logic is characterised by a combination of modal axioms (or semantic conditions on models), KEM defines a unification for each modality and axiom/ semantic condition and then combines them in a recursive and modular way. In this case for SLTL we have to provide a characterisation of the two modalities **first** and **next** in terms of relationships over labels; similarly for TML we have to give conditions on labels for the $\mathbf{B}_i$ modal operators. In particular we use what we call unification to determine whether the denotation of two labels have a non empty intersection, or in other terms whether two labels can be mapped to the same possible world in the possible worlds semantics.

The second key issue is the ability to split labels and to work with parts of labels. The mechanism permits the encapsulation of operations on sub-labels. This is an important feature that, in the present context, allows us to correlate unifications and fibring functions. Given the modularity of the approach the first step of the construction is to define unifications (pattern matching for labels) corresponding to the single modality in the logic we want to study.

Every unification is built from a basic unification defined in terms of a substitution $\rho : \Phi \mapsto \Im$ such that:

$$\begin{aligned} \rho : & \mathbf{1}_{\Phi_C} \\ & \Phi_V^i \mapsto \Im^i \text{ for every } i \in Agt \\ & \Phi_V^A \mapsto \Im \end{aligned}$$

This means that a substitution ρ replaces a constant with the same constant; a variable of type i can be replaced by any i-preferred label, while an auxiliary variable can be freely replaced by any label. This is in agreement with the intuitive meaning of labels that a constant stands for a possible world, and a variable stands for a set of possible worlds (of the appropriate type). Accordingly, we have that two atomic ("world") labels u and v σ-unify iff there is a substitution ρ such that $\rho(u) = \rho(v)$. We shall use $[u;v]\sigma$ both to indicate that there is a substitution ρ for u and v, and the result of the substitution. The σ-unification is extended to the case of composite labels (path labels) as follows:

$$[u;v]\sigma = z \text{ iff } \exists\rho : h(z) = \rho(h(u)) = \rho(h(v)) \text{ and } b(z) = [b(u);b(v)]\sigma$$

For example given the labels $u = (W_1^i, (w_2^j, w_0))$ and $v = (W_2^i, (W_2^j, w_0))$, the σ-unification of the two labels is $w = (W_1^i(w_2^j, w_0))$, where each $h^n(w)$ has been obtained as the unification of $h^n(u)$ and $h^n(v)$. Clearly σ is symmetric, i.e., $[u;v]\sigma$ iff $[v;u]\sigma$. Moreover this definition offers a flexible and powerful mechanism: it allows for an independent computation of the elements of the result of the unification, and variables can be freely renamed without affecting the result of a unification.

We are now ready to introduce the unifications corresponding to the modal operators at hand. For these unification we assume that the labels involved are τ-pure. The first unification is that for **first**.

$$[u;v]\sigma^{\mathbf{first}} = (t_0;[h^1(u);h^1(v)]\sigma) \text{ iff } h(u) = h(v) = t_0 \text{ and } [h^1(u);h^1(v)]\sigma$$

The unification for **first** ($\sigma^{\mathbf{first}}$-unification) corresponds to a constant function (the initial time is unique for the model). Accordingly if two labels end with the same atomic label (t_0) then the two labels denote the same time instant, namely the start of the clock.

For the unification for **next** we will use the fact that the time line is a discrete total order, thus two labels denote the same time instant if they have the same length.

$$[u;v]\sigma = u \text{ iff } \ell(u) = \ell(v),\ [h^1(u);h^1(v)]\sigma \text{ and } c^1(u), c^1(v) \text{ do not contain } t_0.$$

The unification for the logic SLTL is defined by the combination of the unifications for **first** and **next**. Formally

$$[u;v]\sigma_{SLTL} = \begin{cases} [u;v]\sigma^{\mathbf{first}} \\ [c^n(u);c^m(v)]\sigma, \quad w_0 = [s^n(u);s^m(v)]\sigma_{SLTL} \end{cases}$$

For example, according to the above definition of the unification for the SLTL, we have that the labels $u = (t_4, (t_3, (t_0, (t_1, t_0))))$ and $v = (t_6, (t_5, t_0))$ σ_{SLTL}-unify, since we can split the labels as follows: $c^3(u) = (t_4, (t_3, w_0))$, $s^3(u) = (t_0, (t_1, t_0))$ and $c^1(v) = (t_6, (t_5, w_0))$, and $s^1(v) = t_0$, it is immediate to see that $[c^3(u);c^1(v)]\sigma$ and $[s^3(u);s^1(v)]\sigma^{\mathbf{first}}$.

The belief logic can be understood a the combination of multiple **KD** modal logics, one for each agent $i \in Agt$. Thus we first give the unification for each of such logics and then we combine in a single unification to be used with the unification for SLTL for FL.

$$[u;v]\sigma^{TML_i} = [u;v]\sigma$$

where u and v are i-pure. Notice that using the mechanism of counter-segment it is always possible to split labels into pure sub-labels. Accordingly the definition of the unification for TML is

$$[u;v]\sigma_{TML} = \begin{cases} [u;v]\sigma^{TML_i} & u,v \text{ are } i\text{-pure, or} \\ [c^n(u);c^m(v)]\sigma^{TML_i} & c^n(u), c^m(v) \text{ are } i\text{-pure, and} \\ & w_0 = [s^n(u);s^m(v)]\sigma_{TML}. \end{cases}$$

The logic FL is is the fibred combination of TML and SLTL, thus according to the results in Gabbay and Governatori [15] we can obtain the unification for it based on the unifications for the component logics. With σ^{TBL} we understood either σ_{SLTL} or σ_{TML}. The unification for FL is:

$$[u;v]\sigma_{FL} = \begin{cases} [u;v]\sigma^{TBL} & \\ [c^n(u);c^m(v)]\sigma^{TBL} & c^n(u), c^m(v) \text{ are } i\text{-pure, and} \\ & w_0 = [s^n(u);s^m(v)]\sigma_{FL} \end{cases}$$

Theorem 1 *The σ_{FL}-unification of two labels u and v can be computed in linear time.*

Proof. The complexity of $\sigma^{\mathbf{first}}$ is $O(1)$: all we have to do is to compare the first and last element of the two labels. For σ the complexity is $O(n)$, we count the number of elements in the two labels and we check that the right-most elements unify. For σ_{SLTL} we have $O(n)$ again. This unification can be computed by counting the elements of labels starting from the right-most element and reset the count to 0 every time we encounter t_0. Accordingly, two labels σ_{SLTL}-unify if they have the same count.

For σ^{TML_i} the complexity is $O(n)$. The definition of σ implies that the two labels have the same length and thus we perform n unification of pairs of atomic labels, where the unifications of atomic labels can be computed in constant time. For σ_{TML} again the complexity is $O(n)$. We split each labels into maximal i-pure counter-segments. This operation can be computed in linear time, and then we perform at most n ($n \leq \max\{\ell(u),\ell(v)\}$) σ^{TML_i}-unifications of linear complexity.

Finally for σ_{FL} we split the labels into maximal τ-pure counter-segments and we have n linear unifications. ⊣

5.3 Inference Rules

For the presentation of the inference rules we assume familiarity with Smullyan-Fitting unifying notation [13].

$$\begin{array}{cccc} \alpha : u & A \wedge B : u & \neg(A \vee B) : u & \neg(A \rightarrow B) : u \\ \hline \alpha_1 : u & \hline A : u & \hline \neg A : u & \hline A : u \\ \alpha_2 : u & B : u & \neg B : u & \neg B : u \end{array} \qquad (\alpha)$$

The α-rules are just the familiar linear branch-expansion rules of the tableau method. For the β-rules (formulae behaving disjunctively) we exemplify only the rules for im-

plication ($\rightarrow$).[5]

$$\frac{\begin{array}{c}\beta : u \\ \beta_i^C : v\end{array}(i=1,2)}{\beta_{3-i} : [u;v]\sigma_{FL}} \quad \frac{A \rightarrow B : u \quad A : v}{B : [u;v]\sigma_{FL}} \quad \frac{A \rightarrow B : u \quad \neg B : v}{\neg A : [u;v]\sigma_{FL}} \qquad (\beta)$$

The β-rules are nothing but natural inference patterns such as Modus Ponens, Modus Tollens and Disjunctive syllogism generalised to the modal case. In order to apply such rules it is required that the labels of the premises unify and the label of the conclusion is the result of their unification.

$$\frac{\gamma : u}{\gamma_0(x_n) : u} \quad \frac{\forall x P(x) : u}{P(x_n) : u} \quad \frac{\neg\exists x P(x) : u}{\neg P(x_n) : u} \qquad (\gamma)$$

The γ rules are the usual "universal" rules of tableaux method with the usual proviso that x_n is a variable not previously occurring in the tree [13,2].

$$\frac{\delta : u}{\delta_0(c_n) : u} \quad \frac{\exists x P(x) : u}{P(c_n) : u} \quad \frac{\neg\forall x P(x) : u}{\neg P(c_n) : u} \qquad (\delta)$$

The δ rules are the usual "existential" rules of the tableau method, where c_n is a constant that does not occur previously in the tree.

$$\nu \ \frac{\mathbf{B}_i A : u}{A : (W_n^i, u)} \quad \pi \ \frac{\neg\mathbf{B}_i A : u}{\neg A : (w^i, u)} \qquad (\mu_B)$$

where W_n^i and w_n^i are new labels.

The rules for $\mathbf{B}_i$ are the normal expansion rule for modal operators of labelled tableaux with free variables. The intuition for the ν rule is that if $\mathbf{B}_i A$ is true at u, then A is true at all worlds accessible via R_i from u, and this is the interpretation of the label (W_n^i, u); similarly if $\mathbf{B}_i A$ is false at u (i.e., $\neg\mathbf{B}_i A$ is true), then there must be a world, let us say w_n^i accessible from u, where $\neg A$ is true.

$$\frac{\mathbf{first} A : u}{A : (t_0, u)} \quad \frac{\neg\mathbf{first} A : u}{\neg A : (t_0, u)} \quad \frac{\mathbf{next} A : u}{A : (t_n, u)} \quad \frac{\neg\mathbf{next} A : u}{\neg A : (t_n, u)} \qquad (\mu_T)$$

where t_n is new.

Given the functional interpretation of the temporal accessibility relation and that the initial instant is fixed, we have the same expansion of the labels and there is no need to introduce variables.

$$\frac{}{A : u \quad | \quad \neg A : u} \qquad (PB)$$

The "Principle of Bivalence" represents the semantic counterpart of the cut rule of the sequent calculus (intuitive meaning: a formula A is either true or false in any given world). PB is a zero-premise inference rule, so in its unrestricted version can be applied whenever we like. However, we impose a restriction on its application. PB can be only

[5] Where β_i^C denotes the complement of β_i.

applied w.r.t. immediate sub-formulae of unanalysed β-formulae, that is β formulae for which we have no immediate sub-formulae with the appropriate labels in the tree.

$$\frac{\begin{array}{c} A(x) : u \\ \neg A(y) : v \end{array}}{\times} [\text{ if } [u;v]\sigma_{FL} \text{ and } x \text{ and } y \text{ unify}] \qquad (PNC)$$

The rule PNC (*Principle of Non-Contradiction*) states that two labelled formulae are σ_{FL}-complementary when the two formulae are complementary (i.e., the terms in the formula unify according to the standard unification for terms) and their labels σ_{FL}-unify.

5.4 Proof Search

Let $\Gamma = \{X_1, \ldots, X_m\}$ be a set of formulae. Then $\mathscr{T}$ is a *KEM-tree for* Γ if there exists a finite sequence $(\mathscr{T}_1, \mathscr{T}_2, \ldots, \mathscr{T}_n)$ such that (i) $\mathscr{T}_1$ is a 1-branch tree consisting of $\{X_1 : t_1, \ldots, X_m : t_m\}$; (ii) $\mathscr{T}_n = \mathscr{T}$, and (iii) for each $i < n$, $\mathscr{T}_{i+1}$ results from $\mathscr{T}_i$ by an application of a rule of KEM. A branch θ of a KEM-tree $\mathscr{T}$ of L-formulae is said to be σ_{FL}*-closed* if it ends with an application of *PNC*, open otherwise. As usual with tableau methods, a set Γ of formulae is checked for consistency by constructing a KEM-tree for Γ. Moreover we say that a formula A is a *KEM-consequence of a set of formulae* $\Gamma = \{X_1, \ldots, X_n\}$ ($\Gamma \vdash_{KEM(L)} A$) if a KEM-tree for $\{X_1 : u_1, \ldots, X_n : u_n, \neg A : v\}$ is closed using the unification for the logic L, where $v \in \Phi_C^A$, and $u_i \in \Phi_V^A$. The intuition behind this definition is that A is a consequence of Γ when we take Γ as a set of global assumptions [13], i.e., true in every world in a Kripke model.

We now describe a systematic procedure for KEM. First we define the following notions. Given a branch θ of a KEM-tree, we shall call an L-formula $X : u$ *E-analysed in* θ if either (i) X is of type α and both $\alpha_1 : t$ and $\alpha_2 : u$ occur in θ; or (ii) X is of type β and one of the following conditions is satisfied: (a) if $\beta_1^C : v$ occurs in θ and $[u;v]\sigma_{FL}$, then also $\beta_2 : [u;v]\sigma_{FL}$ occurs in θ, (b) if $\beta_2^C : v$ occurs in θ and $[u;v]\sigma_{FL}$, then also $\beta_1 : [u;v]\sigma_{FL}$ occurs in θ; or (iii) X is of type μ and $\mu_0 : (u', u)$ occurs in θ for some appropriate u' of the right type, not previously occurring in θ, or (iv) X is of type γ and $\gamma_0(x_n) : u$ occurs in θ for some variable x_n not previously occurring in θ or (v) X is of type δ and $\delta_0(c_n) : u$ occurs in θ for some variable c_n not previously occurring in θ.

We shall call a branch θ of a KEM-tree *E-completed* if every L-formula in it is E-analysed and it contains no complementary formulae which are not σ_{FL}-complementary. We shall say a branch θ of a KEM-tree *completed* if it is E-completed and all the L-formulae of type β in it either are analysed or cannot be analysed. We shall call a KEM-tree *completed* if every branch is completed.

The following procedure starts from the 1-branch, 1-node tree consisting of $\{X_1 : u, \ldots, X_m : v\}$ and applies the inference rules until the resulting KEM-tree is either closed or completed.

At each stage of proof search (i) we choose an open non completed branch θ. If θ is not E-completed, then (ii) we apply the 1-premise rules until θ becomes E-completed. If the resulting branch θ' is neither closed nor completed, then (iii) we apply the 2-premise rules until θ becomes E-completed. If the resulting branch θ' is neither closed nor completed, then (iv) we choose an L-formula of type β which is not yet analysed

in the branch and apply *PB* so that the resulting *LS*-formulae are $\beta_1 : u'$ and $\beta_1^C : u'$ (or, equivalently $\beta_2 : u'$ and $\beta_2^C : u'$), where $u = u'$ if u is restricted (and already occurring when $h(u) \in \Phi_C$), otherwise u' is obtained from u by instantiating $h(u)$ to a constant not occurring in u; (v) ("Modal *PB*") if the branch is not E-completed nor closed, because of complementary formulae which are not σ_{FL}-complementary, then we have to see whether a restricted label unifying with both the labels of the complementary formulae occurs previously in the branch; if such a label exists, or can be built using already existing labels and the unification rules, then the branch is closed, (vi) we repeat the procedure in each branch generated by *PB*.

5.5 Soundness and Completeness

The resulting tableaux system is sound and complete for the logics presented in this paper. As usual with tableaux systems a proof of A is a closed tableaux for $\neg A$, thus a tableaux systems is sound and complete for a particular logic if it is able to generate closed tableaux for all negation of valid formula, and open tableaux (models) for all satisfiable formulae. In proving the results for the logics at hand we will make use of the main result (Theorem 22) of Gabbay and Governatori [15] that allows one to obtain a sound and complete labelled tableaux system for a fibred logic based on sound and complete labelled tableaux systems (of the same type of the tableaux system for the fibred logic) for the logics to be combined. The key idea of the Theorem is to conceive the join point of a unification where the labels are split in segments and counter-segments as the counterpart of the fibring function in fibred models. The soundness and completeness of *KEM* for FL follows from its being sound and complete for the component logics SLTL and TML.

Lemma 1 *$KEM(SLTL)$ is sound and complete for SLTL.*

Proof. The logic SLTL can be seen as the fibring of two modal logics whose accessibility relation is a total function for **next** and a constant function for **first**. Governatori [16] proved that KEM with the $\sigma^{\mathbf{first}}$-unification is sound and complete for a modal logic with a constant accessibility relation and later [17] showed the same result for functional accessibility relation and σ. Theorem 22 of Gabbay and Governatori [15] allows us to combine the unification for **first** and **next** as in the unification for σ_{SLTL} to obtain a sound and complete tableau systems for SLTL. ⊣

Lemma 2 *$KEM(TML)$ is sound and complete for TML.*

Proof. [2] shows that KEM with the σ-unification is sound and complete for the modal logic **KD**. The multi-modal case is a consequence of Theorem 22 of [15] ⊣

Armed of these results we can now prove the result for FL.

Theorem 2 *$KEM(FL)$ is sound and complete for FL.*

Proof. From Lemma 1 and 2 and Theorem 22 of [15]. ⊣

6 Analysing Authentication Protocols

In this section, we first build a theory of trust to specify the TESLA protocol, then discuss its correctness. With the purpose of making the logic FL appropriate for specifying the protocol, we restrict the time model of FL to guarantee that the time interval between any moment and its next moment in time has the same length, 1 unit time. This restriction matches the special timing property that the TESLA scheme satisfies: *the sender sends packets at regular time intervals*. The assumption simplifies our discussion without harming its correctness.

6.1 The Formalization of TESLA

We now establish a theory that describes the behaviour or functions of the protocol with the scheme PCTS. The basic types of the theory include: $Agents = \{A, B, S, R, I\}$, $Messages = \{X, Y, D, D'\}$ and $Keys = \{K, K_1, K_2\}$ where S, R, I stand for the sender, the receiver, and the intruder, respectively. In case there are multiple receivers, we may have $R_1, R_2, \ldots$ in the type $Agents$.

Through an analysis of the TESLA protocol, we set a theory to specify it consisting of four modules, M_{sr} (*send-receive mode specification*), M_{mk} (*message receiving and knowledge gained*), M_{ms} (*message sending*), and M_{ar} (*authentication rules*). Each module consists of several axiom schemata). Several predicates are used to express these axioms. Given the intuitive reading of the predicates we omit their explanations.

Send-receive mode specification depends on what kind of mode is adopted. We first consider a simple mode called the *zero-delay mode*, which is based on two assumptions: (1) zero time is spent between sending a message and receiving this message, i.e., the sending time of a packet P_i is equal to the receiving time of the packet on the synchronized receiver's clock, for any P_i; and (2) the packet rate is assumed to be 1 (i.e., 1 packet per unit time). With this mode, module M_{sr} consists of the following axiom schemata:

Z1. $sends(S, R, X) \leftrightarrow receives(R, X)$.
Z2. $sends(S, R, \langle D, MAC(f'(K), D)\rangle) \leftrightarrow \mathbf{next}\ sends(S, R, X) \wedge K \in X$.

The first rule says that, if the sender sends the receiver a message, then the receiver will receive the message at the same time; and the second one says that the sender sends the receiver a message packet with a signed commitment to a key if it will send the receiver a packet containing that key at the next moment in time.

Zero-delay mode is an idealized mode. However, generally the time spent between sending and receiving messages cannot be zero. Considering this point, we give the definition of send-receive modes by introducing a generic form.

Definition 10 (time intervals) *For a send-receive mode, all packets P_i must arrive within a certain time interval $[u, v]$ relative to the current time defined as follows:*

$$sends(S, R, P_i) \rightarrow \mathbf{next}^{(m)}\ receives(R, P_i), u \leq m \leq v.$$

Let the current time be t_c (time of sending a packet). Definition 10 indicates that any packet sent by the sender must arrive at a moment between t_{c+u} and t_{c+v}.

Definition 11 (time distance of sending) *Let $d = 1/r$, where r is the packet rate (i.e., number of packets sent per unit time). We call d the* time distance of sending *between two packets.*

Noting that a send-receive mode is in fact determined based on the time interval of packet arrival and the time distance of sending, we have the formal definition of a mode as follows:

Definition 12 (send-receive modes) *We use the notation $m([u,v],d)$ to represent a* send-receive mode of the PCTS scheme of TESLA *or, simply, a* mode *if $u,v,d \in \mathcal{N}$, the set of all natural numbers, and $u \leq v$, where $[u,v]$ is the time interval of this mode, and d the time distance of sending with it. We say that $m([u,v],d)$ is a* safe mode *if $v < d$.*

The following generic rules specify a given mode $m([u,v],d)$:[6]

G1. $sends(S,R,X) \leftrightarrow \mathbf{next}^{(u)}\ receives(R,X) \vee \ldots \vee \mathbf{next}^{(v)} receives(R,X)$.
G2. $sends(S,R,\langle D, MAC(f'(K),D)\rangle) \leftrightarrow \mathbf{next}^{(d)}\ sends(S,R,X) \wedge K \in X$.

Mode-specific rules are determined when u,v and d are given. For example, within the mode $m([2,3],4)$, we have

S1. $sends(S,R,X) \leftrightarrow \mathbf{next}^{(2)}\ receives(R,X) \vee \mathbf{next}^{(3)} receives(R,X)$.
S2. $sends(S,R,\langle D, MAC(f'(K),D)\rangle) \leftrightarrow \mathbf{next}^{(4)}\ sends(S,R,X) \wedge K \in X$.

Modules M_{mk}, M_{ms}, and M_{ar} are fixed for any mode. Due to space limitations, they are listed below without explanations.

M_{mk} **(message receiving and knowledge gained)**

G3. $receives(A,\langle X,Y\rangle) \rightarrow receives(A,X) \wedge receives(A,Y)$.
G4. $receives(A,X) \rightarrow knows(A,X)$.
G5. $knows(A,K) \rightarrow knows(A,f(K)) \wedge knows(A,f'(K))$.
G6. $knows(A,\{X\}_{SK(B)}) \rightarrow knows(A,X)$.
G7. $knows(A,K) \wedge knows(A,X) \rightarrow knows(A,MAC(K,X))$.
G8. $knows(A,X) \rightarrow \mathbf{next}\ knows(A,X)$.

where $SK(B)$ is the private key of agent B and its corresponding public key can be known by anybody, so we have G8. Here knowledge only refers to knowing parts of messages rather than propositions hence it is modeled using a predicate.

M_{ms}**(Message sending)**
G9. $sends(A,B,\langle X,Y\rangle) \rightarrow sends(A,B,X) \wedge sends(A,B,Y)$.
G10. $sends(A,B,X) \rightarrow has\_sent(A,B,X)$.
G11. $has\_sent(A,B,X) \rightarrow \mathbf{next}\ has\_sent(A,B,X)$.

M_{ar} **(Authentication rules)**
G12. $is\_auth(\langle X, MAC(f'(k),D)\rangle) \leftrightarrow verify\_success(f(K)) \wedge verify\_success(MAC(f'(K),D))$.
G13. $is\_auth(X) \rightarrow has\_been\_auth(X)$.

[6] In what follows we will use $\mathbf{next}^{(m)}$ to indicate m consecutive occurrences of **next**.

G14. $\mathbf{B}_R\ has\_been\_auth(X) \rightarrow \mathbf{next}\ \mathbf{B}_R\ has\_been\_auth(X)$.
G15. $receives(R, \langle X, MAC(f'(K), D)\rangle) \wedge$
$\mathbf{B}_R\ \neg has\_sent(S,R,K) \rightarrow \mathbf{B}_R\ arrive\_safe(\langle X, MAC(f'(K), D)\rangle)$.
G16. $arrive\_safe(X) \rightarrow has\_arrive\_safe(X)$.
G17. $\mathbf{B}_R\ has\_arrive\_safe(X) \rightarrow \mathbf{next}\ \mathbf{B}_R\ has\_arrive\_safe(X)$.
G18. $\mathbf{B}_R\ verify\_success(f(K)) \leftrightarrow \mathbf{B}_R\ has\_arrive\_safe(\langle X, MAC(f'(K), D)\rangle) \wedge knows(R,K) \wedge$
$\mathbf{B}_R\ has\_been\_auth(\langle D', MAC(f'(K), D')\rangle) \wedge f(K) \in D'$.
G19. $\mathbf{B}_R\ verify\_success(MAC(f'(K), D)) \leftrightarrow \mathbf{B}_R\ has\_arrive\_safe(\langle X, MAC(f'(K), D)\rangle) \wedge$
$knows(R,K) \wedge MAC(f'(K), X) = MAC(f'(K), D)$.

Thus, we have obtained a theory $\mathbf{T} = \mathrm{M}_{sr} \cup \mathrm{M}_{mk} \cup \mathrm{M}_{ms} \cup \mathrm{M}_{ar}$ specifying the PCTS scheme of TESLA given in Section 2, where each module contains the relevant axioms given above: $\mathrm{M}_{sr} = \{\mathrm{G1}, \mathrm{G2}\}$, $\mathrm{M}_{mk} = \{\mathrm{G3}, \ldots, \mathrm{G8}\}$, $\mathrm{M}_{ms} = \{\mathrm{G9}, \mathrm{G10}, \mathrm{G11}\}$, and $\mathrm{M}_{ar} = \{\mathrm{G12}, \ldots, \mathrm{G19}\}$.

6.2 Correctness Analysis

The correctness condition for a given TESLA scheme should guarantee that if the receiver (receivers) can verify that a packet is authentic, then the packet was indeed sent by the sender.

Definition 13 (correctness condition) *The local correctness for a TESLA scheme to the receiver R who receives messages from the sender S means that, if R has verified that a packet is authentic, then the packet was indeed sent by S. That is,*

$$\forall X\ (\mathbf{B}_R\ has\_been\_auth(X) \wedge has\_sent(A,R,X) \rightarrow A = S).$$

Furthermore, the (global) correctness for the TESLA scheme means that the local correctness for the scheme to all receivers holds.

The theory discussed above is based on a time model where the clock is regarded as the synchronized receiver's clock (correspondingly to the global clock). It provides a basis for the receiver to verify stream messages received through the PCTS scheme of TESLA if the scheme with its send-receive mode satisfies the correctness condition.

Based on the theory developed above, we can show that the correctness condition of the TESLA protocol holds within the scheme.

Proposition 1 *The PCTS scheme with the mode $m([u,v],d)$ mode is secure (i.e., it satisfies the correctness condition)if $m([u,v],d)$ is a safe mode.*

Proof. We sketch the proof: Within the PCTS scheme, packet P_1 is authenticated with the regular digital signature scheme and can therefore be checked using a standard verification method. Therefore, the correctness for the PCTS scheme may be inductively shown based on the assumption that the receiver has the authenticated packet P_1, and it was indeed sent by the sender. That is, we have that

$$\mathbf{B}_R\ has\_been\_auth(P_1) \wedge has\_sent(A,R,P_1) \rightarrow A = S.$$

Then, assuming for all i ($1 \leq i \leq n-1$),

$$\mathbf{B}_R\ has\_been\_auth(P_i) \wedge has\_sent(A,R,P_i) \rightarrow A = S$$

holds, we only need to show that

$$\mathbf{B}_R\ has\_been\_auth(P_n) \wedge has\_sent(A,R,P_n) \rightarrow A = S.$$

It can be shown that the assertion holds true based on axioms G8 and G11 if $v < d$, i.e., $m([u,v],d)$ is a safe model. $\dashv$

We can also use the theory to show that the PCTS scheme with an unsafe mode, e.g, the mode $m([1,4],2)$, provides chances for the intruder to attack the system. Consider the case: assume that packets P_i and P_{i+1} are sent out by the sender at time t (the current moment in time) and at $t+2$ (the next next moment), respectively. The intruder, I, first intercepts P_i at $t+2$ and then, at $t+3$, again intercepts P_{i+1} when it arrives. By creating a packet P_i', instead of P_i, using key K_i in packet P_{i+1}, I masquerades as the sender send packet P_i' to the receiver. The attack will be successful if P_i' reaches the receiver at $t+4$.

The correctness of a TESLA scheme depends on the send-receive mode that the scheme adopts. With an appropriate mode (a safe mode), the TESLA scheme can guarantee that the intruder does not have enough time to make a fake message and then masquerade as the sender to send it to the receiver, even when the intruder is able to intercept a message that is sent to the receiver. Focusing on the discussion of the correctness condition for the TESLA scheme, we do not specifically consider how the intruder may work within a given TESLA scheme, but there is no difficulty for the reader themself to construct some successful attack by the intruder within a scheme that adopts an unsafe mode. The interested reader can refer to the literature [28] for the specification of an intruder module in the proposed fibred logic. Similarly, other variants of the TESLA protocol have been proposed, and it is not our aim to give a complete analysis of how to encode them.

6.3 Mechanising Correctness Proofs

In order to automatically analyse the correctness of a scheme of the protocol, we need to mechanize the theory describing the behaviour of the protocol in an appropriate proof system. In our approach, such system-specific trust theories developed for specifying communications protocols do not depend on a specific implementation. The user is therefore allowed to freely choose the tools for mechanizing these theories. Below we show how modal tableaux can be used to verify the properties of the TESLA protocol. Modular structure offers convenience to the user for translating a theory to an executable code (program) in a mechanised proof system, such as Isabelle [29].

With the labelled modal tableaux system KEM, to show a safe mode satisfies the correctness condition, we only need to show that in this mode $A = S$ is a *KEM-consequence of a set of formulae* $\Gamma = \{\mathbf{B}_R\ has\_been\_auth(X), has\_sent(A,R,X)\}$. Due to space limitations, we only give a simple case to show how the labelled tableaux system works on checking the properties of TESLA. With the send-receive mode $m([2,3],4)$, we assume that the message has arrived safely and it has been authenticated based on the time the message was received and the contents of the message:

H1. $\mathbf{first}\ next^{(3)}\ receives(R,\langle X,Y\rangle)$
H2. $\mathbf{first\ next}^{(7)}\ receives(R,X1) \wedge K \in X1$
H3. $MAC(f'(K),X) = Y$
H4. $\mathbf{first\ next}^{(8)}\ \mathbf{B}_R\ is\_auth(\langle X,Y\rangle)$

Then, we can prove the following property:

(A). $\mathbf{first\ next}^{(8)}\ \mathbf{B}_R\ (is\_auth(\langle X,Y\rangle) \rightarrow (\mathbf{first}\ sends(S,R,\langle X,Y\rangle) \vee \mathbf{first\ next}\ sends(S,R,\langle X,Y\rangle)))$

It basically says that if at time t_8, agent R believes that if the message is authenticated, then it must have been sent at either time t_0 or time t_1 (agent R does not really know the exact time when the message was sent, however, it knows about the time interval).

In the following we show the tableaux proof of the property. All the rules of the PCTS scheme of TESLA are at our disposal as well as the assumptions made above; each is labelled with a generic universal label that would unify with any given label. Tableaux rules have been applied exhaustively until all the branches have been completed (details of proof steps are omitted). We also assume a that biconditional (such as rule S1 used in the proof) is the conjunction of two implications.

1. $sends(S,R,\langle X,Y\rangle) \leftrightarrow \mathbf{next}^{(2)}\ receives(R,\langle X,Y\rangle) \vee \mathbf{next}^{(3)} receives(R,\langle X,Y\rangle) : W1$
2. $\mathbf{first}\ next^{(3)}\ receives(R,\langle X,Y\rangle) : W2$
3. $\mathbf{first\ next}^{(8)}\ \mathbf{B}_R\ is\_auth(\langle X,Y\rangle) : W3$
4. $\neg\mathbf{first\ next}^{(8)}\ \mathbf{B}_R\ (is\_auth(\langle X,Y\rangle) \rightarrow (\mathbf{first}\ sends(S,R,\langle X,Y\rangle) \vee \mathbf{first\ next}\ sends(S,R,\langle X,Y\rangle))) : W4$
5. $\neg\mathbf{next}^{(8)}\ \mathbf{B}_R\ (is\_auth(\langle X,Y\rangle) \rightarrow (\mathbf{first}\ sends(S,R,\langle X,Y\rangle) \vee \mathbf{first\ next}\ sends(S,R,\langle X,Y\rangle))) : (t_0, W4)$
6. $\neg\mathbf{next}^{(7)}\ \mathbf{B}_R\ (is\_auth(\langle X,Y\rangle) \rightarrow (\mathbf{first}\ sends(S,R,\langle X,Y\rangle) \vee \mathbf{first\ next}\ sends(S,R,\langle X,Y\rangle))) : (t_1,(t_0, W4))$
7. ... (expansion rule for **next** is applied 7 times, μ_T)
8. $\neg\mathbf{B}_R\ (is\_auth(\langle X,Y\rangle) \rightarrow (\mathbf{first}\ sends(S,R,\langle X,Y\rangle) \vee \mathbf{first\ next} sends(S,R,\langle X,Y\rangle))) : (t_8,(\ldots(t_1,(t_0,W4))\ldots)$
9. $\neg(is\_auth(\langle X,Y\rangle) \rightarrow (\mathbf{first}\ sends(S,R,\langle X,Y\rangle) \vee \mathbf{first\ next}\ sends(S,R,\langle X,Y\rangle))) : (w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))$
10. $is\_auth(\langle X,Y\rangle) : (w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))$
11. $\neg(\mathbf{first}\ sends(S,R,\langle X,Y\rangle) \vee \mathbf{first\ next}\ sends(S,R,\langle X,Y\rangle)) : (w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))$
12. $\neg\mathbf{first}\ sends(S,R,\langle X,Y\rangle) : (w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))$
13. $\neg\mathbf{first\ next}\ sends(S,R,\langle X,Y\rangle)) : (w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))$
14. $\neg sends(S,R,\langle X,Y\rangle) : (t_0,(w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots)))$
15. $\neg\mathbf{next}\ sends(S,R,\langle X,Y\rangle)) : (t_0,((w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots)))$
16. $\neg sends(S,R,\langle X,Y\rangle)) : (t_1,(t_0,((w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))))$
17. $sends(S,R,\langle X,Y\rangle) \rightarrow \mathbf{next}^{(2)}\ receives(R,\langle X,Y\rangle) \vee \mathbf{next}^{(3)} receives(R,\langle X,Y\rangle) : W1$
18. $\mathbf{next}^{(2)}\ receives(R,\langle X,Y\rangle) \vee \mathbf{next}^{(3)} receives(R,\langle X,Y\rangle) \rightarrow sends(S,R,\langle X,Y\rangle) : W1$
19. $\neg(\mathbf{next}^{(2)}\ receives(R,\langle X,Y\rangle) \vee \mathbf{next}^{(3)} receives(R,\langle X,Y\rangle)) : (t_1,(t_0,((w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))))$
20. $\neg\mathbf{next}^{(2)}\ receives(R,\langle X,Y\rangle) : (t_1,(t_0,((w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))))$
21. $\neg\mathbf{next}^{(3)} receives(R,\langle X,Y\rangle)) : (t_1,(t_0,((w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))))$
22. $\mathbf{next}^{(3)}\ receives(R,\langle X,Y\rangle) : (t_0,W2)$
23. $\mathbf{next}^{(2)}\ receives(R,\langle X,Y\rangle) : (t_1,(t_0,W2))$

24. $\times[(t_1,(t_0,((w^i,(t_8,(\ldots(t_1,(t_0,W4))\ldots))))$ and $(t_1,(t_0,W2))$ unify]
(obtained from steps 20 and 23)

This proof has only one branch which is closed. This shows that agent R's belief has been justified based on the assumptions. Notice that, according to the proof search procedure given in Section 5.4, the labels $W1$, $W2$, $W3$ and $W4$ are auxiliary labels, and as such they can be uniformly replaced by any suitable atomic label.

7 Concluding Remarks

With the logic FL, we use a simple case of the fibred semantics arising from Kripke models with a single time model. However, it is not difficult to extend it with other different time models. Such extensions would be needed when one wants to deal with different local clocks for multi-agent systems. The logic resulting from fibring SLTL and TML is far more expressive than the temporalized belief logic [25].

We have also discussed an application of the logic FL in analysing the TESLA protocol. In other approaches to the analysis of TESLA, Archer [1] uses the theorem prover TAME, and Broadfoot *et al* [4] use model checking techniques, to analyse TESLA. The advantage of these methods is that some properties of the protocol can easily be captured through proof systems, but a drawback is that the formal representations involved in such proofs are often not easily validated by the user. Our approach separates the theory from its implementation and helps a protocol designer to capture the meanings of the theory as a whole. Our analysis has shown that the PCTS scheme of TESLA with a safe send is secure given that the correctness condition is satisfied. We have shown elsewhere that our approach is suitable for analysis of other modes of the TESLA protocol such as unsafe modes [28].

Lomuscio and Wozna [26] proposed a recent approach to the analysis of stream authentication protocols based on a temporal epistemic logic called TDL and provided a sound and complete axiomatisation of TDL based on CTL augmented with an epistemic operator. They also performed a manual analysis of the security property of TESLA and foreshadowed a BDD-based model checker for automated analysis based on the interpreted system model. However, TDL is not a result of a systematic combination of the constituent logics and there is no proposed proof method for it other than an axiomatic one.

Dixon *et al* [8] also considered a temporal epistemic logic resulting from the fusion of linear-time temporal logic and multi-modal S5 (for knowledge). They showed that the logic can be used for verification of part of the Needham-Schroeder protocol using clausal resolution.

Fibring is a very rich technique for combining logics and leads to a more systematic approach for searching suitable logics for target domains such as the task of verification of security protocols. Our approach based on the fibred logic is also quite flexible since the structure of the theory is well-defined, and separating the theory from its implementation helps a protocol designer to capture the meanings of the theory as a whole. Moreover the modular structure makes it easy for the user to modify a theory. Fibring also allows us to develop tablueaux-based proof procedures in a systematic way; tableaux

systems have advantages over axiomatic and/or clausal resolution-based systems and they align well with Kripke semantics for modal logics.

Future work will involve consideration of richer models of time for SLTL as well as the implementation of a mechanised proof procedure based on modal tableaux. We also plan to apply our approach to other schemes of the TESLA protocol as well as other stream authentication protocols.

Acknowledgements

This work is supported in part by Australian Research Council under Discovery Project No. DP0452628 on "Combining modal logics for dynamic and multi-agents systems".

References

1. M. Archer. Proving correctness of the basic TESLA multicast stream authentication protocol with TAME. In *Workshop on Issues in the Theory of Security*, 2002. (Unpaginated proceedings available from http://www.dsi.unive.it/IFIPWG1_7/WITS2002/prog/annotated_program.html)
2. A. Artosi, P. Benassi, G. Governatori, and A. Rotolo. Shakespearian modal logic: A labelled treatment of modal identity. *Advances in Modal Logic.1*, 1–21. CSLI, 1998.
3. B. Bennett, Cl. Dixon, M. Fisher, U. Hustadt, E. Franconi, I. Horrocks, and M. de Rijke. Combinations of modal logics. *Artif. Intell. Rev.*, 17(1):1–20, 2002.
4. P. Broadfoot and G. Lowe. Analysing a stream authentication protocol using model checking. In *Proc 7th ESORICS*, 2002.
5. M. Burrows, M. Abadi, and R. M. Needham. A logic of authentication. *ACM Transactions on Computer Systems*, 8(1):18–36, 1990.
6. E. Clarke, S. Jha, and W. Marrero. A machine checkable logic of knowledge for specifying security properties of electronic commerce protocols. In *Proceedings of the Workshop on Formal Methods and Security Protocols*, 1998.
7. A. Costa-Leite. Towards a general theory of the combination of logics. In *Aspects of universal logic, Travaux de Logique [Works on Logic]*, Vol.17. Universite de Neuchatel, pp.219–230, 2004.
8. C. Dixon, M. Carmen Fernndez Gago, M. Fisher, W. van der Hoek. Using Temporal Logics of Knowledge in the Formal Verification of Security Protocols. In *Proceedings of the 11th International Symposium on Temporal Representation and Reasoning (TIME 2004)*, 1–3 July 2004, Tatihou Island, Normandie, France. IEEE Computer Society 2004, pages 148–151.
9. N. Durgin, J. Mitchell, and D. Pavlovic. A compositional logic for proving security properties of protocols. *Journal of Computer Security*, 11(2003):677–721.
10. G. Elofson. Developing trust with intelligent agent: An exploratory study. In *Proceedings of the first International Workshop on Trust*, pages 125–139, 1998.
11. M. Finger and D. M. Gabbay. Adding a temporal dimension to a logic system. *Journal of Logic, Language and Information*, 1:221–237, 1997
12. M. Fisher. Temporal development methods for agent-based systems. *Autonomous Agents and Multi-Agent Systems*, 10(1):41–66, 2004.
13. M. Fitting. *Proof Methods for Modal and Intuitionistic Logics*. Reidel, 1983.
14. D. M. Gabbay. *Fibring Logics*. OUP, 1999.
15. D.M. Gabbay and G. Governatori. Fibred modal tableaux. *Labelled Deduction*, pages 163–194. Kluwer, 2000.

16. G. Governatori. Labelled tableaux for multi-modal logics. *Proc. Tableaux'95*, LNAI 918, pages 79–94. Springer, 1995.
17. G. Governatori. *Un modello formale per il ragionamento giuridico*. PhD thesis, University of Bologna, 1997.
18. G. Governatori, V. Padmanabhan and A. Sattar. On Fibring Semantics for BDI Logics. *Proc JELIA 2002*, LNCS 2424, pages 198-209. Springer, 2002.
19. J. Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence* **54**, pages 319–379, 1992.
20. G. E. Hughes and M. J. Cresswell. *A New Introduction to Modal Logic*. Routledge, 1996.
21. S. Kripke. Semantical considerations on modal logic. *Acta Philosophica Fennica*, 16:83–94, 1963.
22. C. Liu. Logical Foundations for Reasoning about Trust in Secure Digital Communication. In *Proceedings of the 14th Australian Joint Conference on Artificial Intelligence*, Adelaide, Australia, December 10–14, 2001, Lecture Notes in Computer Science 2256, Springer 2001, pages 333–344.
23. C. Liu and M. A. Orgun. Dealing with multiple granularity of time in temporal logic programming. *Journal of Symbolic Computation*, 22:699–720, 1996.
24. C. Liu and M. A. Orgun. Verification of Reactive Systems Using Temporal Logic with Clocks. *Theoretical Computer Science*, Volume 220, Number 2, pages 377–408, 1999.
25. C. Liu, M. Ozols, and M. A. Orgun. A temporalised belief logic for specifying the dynamics of trust for multi-agent systems. In *Proceedings of the Ninth Asian Computer Science Conference, Lecture Notes in Computer Science Vol.3321*. Springer-Verlag, 2004, pages 142–156.
26. A. Lomuscio and B. Wozna. A complete and decidable security-specialised logic and its application to the TESLA protocol. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006)*, Hakodate, Japan, May 8-12, 2006, ACM Press, pages 145-152.
27. J. Ma and M. A. Orgun. Trust management and trust theory revision. In *IEEE Transactions on Systems, Man and Cybernetics, Part A*, volume 36, pages 451–460, 2006.
28. M. A. Orgun, J. Ma, C. Liu and G. Governatori. Analysing Stream Authentication Protocols in Autonomous Agent-Based Systems. In *Proceedings of the Second International Symposium on Dependable Autonomic and Secure Computing (DASC 2006)*, 29 September - 1 October 2006, Indianapolis, Indiana, USA. IEEE Computer Society 2006, pages 325–332.
29. L. C. Paulson. *Isabelle - A Generic Theorem Prover (with a contribution by T. Nipkow)*. Springer-Verlag, 1994.
30. L. C. Paulson. The Inductive Approach to Verifying Cryptographic Protocols. *Journal of Computer Security*, 6(1-2): 85–128 (1998).
31. A. Perrig, R. Canetti, J. D. Tygar, and D. Song. Efficient authentication and signing of multicast streams over lossy channels. In *IEEE Symposium on Security and Privacy*, pages 56–73, 2000.
32. R. Yahalom, B. Klein, T. Beth. Trust Relationships in Secure Systems-A Distributed Authentication Perspective. *Proceedings of the 1993 IEEE Symposium on Security and Privacy*, p.150, May 24-26, 1993.