Logical Concepts in Cryptography

Simon Kramer

Ecole Polytechnique Fédérale de Lausanne (EPFL)
simon.kramer@a3.epfl.ch

Abstract. This paper is about the exploration of logical concepts in cryptography and their linguistic abstraction and model-theoretic combination in a logical system, called CPL (for *Cryptographic Protocol Logic*). The paper focuses on two fundamental aspects of cryptography. Namely, the security of *communication* (as opposed to security of *storage*) and cryptographic *protocols* (as opposed to cryptographic *operators*). The logical concepts explored are the following. Primary concepts: the *modal* concepts of knowledge, norms, space, and time. Secondary concepts: knowledge de dicto and knowledge de re, confidentiality norms, truth-functional and relevant implication, multiple and complex truth values. The distinguishing feature of CPL is that it unifies and refines a variety of (not all!) existing approaches. This feature is the result of our *wholistic conception* of property-based (logics) and model-based (process algebra) formalisms. We illustrate the expressiveness of CPL on representative *requirements engineering* case studies.

Addendum I. In the Appendix C, we extend (core) CPL (qualitative time) with real time, i.e., time stamps, timed keys, and potentially drifting local clocks, to tCPL (quantitative time). Our extension is conservative and really simple; it requires only the refinement of two relational symbols (two new axioms resp. one new parameter) and of one operator (one new conjunct in its truth predicate), and the addition of two relational symbols (but no operators!). Our work thus provides further evidence for Lamport's claim that adding real time to an untimed formalism is really simple.

Addendum II. In the Appendix D, we sketch an extension of (core) CPL with a notion of probabilistic polynomial-time (PP) computation. We illustrate the expressiveness of this extended logic (ppCPL) on tentative formalisation case studies of fundamental and applied concepts. *Fundamental concepts*: (1) one-way function, (2) hard-core predicate, (3) computational indistinguishability, (4) (n-party) interactive proof, and (5) (n-prover) zero-knowledge. *Applied concepts*: (1) security of encryption schemes, (2) unforgeability of signature schemes, (3) attacks on encryption schemes, (4) attacks on signature schemes, and (5) breaks of signature schemes. The argument of this appendix is that in the light of logic, adding PP to a (property-based) formalism for cryptography is also simple and can be achieved with an Ockham's razor extension of an existing core logic, namely CPL.

Keywords applied formal logic, cryptographic protocols

1 Introduction

The definition of a cryptographic protocol begins (and "ends" if this stage is not mastered[1]) with *requirements engineering*, i.e., the definition of the requirements (global properties) the protocol is supposed to meet. In particular, understanding protocol requirements is necessary for understanding protocol attacks, which can be looked at as negations of necessary conditions for the requirements to hold. Protocol definition and in particular requirements engineering are engineering tasks (the spirit of [3]). In contrast, the definition of cryptographic operators is a scientific task (the spirit of [4, 5]) requiring profound expertise from different fields of discrete mathematics. Protocol engineers do (and should) not have (to have) this expertise. For example, it is legitimate for a protocol engineer to "abstract" negligible probabilities and consider them as what they are — negligible. Ideally, engineers should only have to master a single, common, and formal language for requirements engineering that adequately abstracts "hard-core" mathematical concepts. Since logic is what all sciences have in common, it is natural to stipulate that such a lingua franca for requirements engineering cryptographic protocols be an appropriate logical language. Our task shall be to synthesise the relevant logical concepts in cryptography into a cryptographic protocol logic in the tradition of temporal[2] logic [6] (cf. [7] for an effort of similar ambition but in the tradition of dynamic[3] logic [8]). We will validate our language — at least at a first stage — on specification (stress on different requirements) rather than verification (stress on different protocols) case studies, since specification should precede verification. Nonetheless, the existence of verification examples is guaranteed by *subsumption* under CPL of other logics from authors with the opposite focus.

We briefly survey requirements engineering — the practice of the specification — of cryptographic protocols. Protocol designers commonly specify a cryptographic protocol jointly by a semi-formal description of its *behaviour* (or *local* properties) in terms of *protocol narrations*, and by an informal prescription of its intended *goals* (or *global* properties) in *natural language* [9]. Informal specifications present two major drawbacks: they do not have a well-defined, and thus a well-understood *meaning*, and, therefore, they do not allow for verification of correctness. In *formal* specifications of cryptographic protocols, local and global properties are expressed either explicitly *as such* in a logical (or *property*-based) language, or implicitly *as code*, resp. *as encodings* in a programming (or *model*-based) language (e.g., applied λ-Calculus [10]; process calculi: CSP [11], applied π-Calculus [12], Spi-Calculus [13], and [14]). Examples of such encodings are equations between protocol instantiations, and predicates defined inductively

[1] consider "[. . .] although it is difficult to get cryptographic protocols right, what is really difficult is not the design of the protocol itself, but of the requirements. Many problems with security protocols arise, not because the protocol as designed did not satisfy its requirements, but because the requirements were not well understood in the first place." [1], and also, more generally [2]

[2] more precisely, poly-dimensional (i.e., norms, knowledge, space, time) mono-modal

[3] more precisely, mono-dimensional (i.e., time) poly-modal (action parameters)

on the traces those instantiations may exhibit [15]. However, such encodings present four major drawbacks: (1) they have to be found; worse, (2) they may not even exist; (3) they are neither directly comparable with other encodings in the same or in other programming languages, nor with properties expressed explicitly in logical languages; and (4) they are not easy to understand because the intuition of the encoded property is not explicit in the encoding. On the other hand, process calculi are ideal *design formalisms*. That is, they offer — due to their minimalist, linguistic abstractions of modelling concepts (syntax) and their mathematical, operational notion of execution (semantics) — a win-win situation between the (pedantic) rigour of machine models and the (practical) usability of programming languages.

Still, informal language and programming languages are inadequate for expressing and comparing cryptographic properties. It is our belief that only a logical language equipped with an appropriate notion of truth, i.e., a cryptographic *logic*, will produce the necessary adequacy. A number of logics have been proposed in this aim so far, ranging from *special-purpose*, cryptographic logics: the pioneering BAN-logic [16], a unification of several variants of BAN-logic [17]; over general-purpose propositional, modal, program, and first- and higher-order logics used for the special purpose of cryptographic protocol analysis: *propositional* ("logic programming") [18, 19]; *modal*: deontic [20], doxastic [21, 22], epistemic [23, 24], linear [25], temporal [26]; *program*: dynamic [27, 28, 7], Hoare-style [29]; *first-order* [30–32]; *higher-order* [33, 34]; to combinations thereof: doxastic-epistemic [35], doxastic-temporal [36], distributed temporal [37], dynamic-epistemic [38], epistemic-temporal [39] first-order-temporal [40], dynamic-epistemic-temporal [41], deontic-epistemic-temporal [42].

All these logics have elucidated important aspects of cryptographic communication and proved the relevance of logical concepts to the modelling of that communication. In particular, mere enunciation of maybe the three most fundamental cryptographic goals, namely secrecy, authenticity, and non-repudiation, reveals the paramount importance of the concept of *knowledge*, both in its propositional (so-called knowledge *de dicto*) and in its individual (so-called knowledge *de re*) manifestation. Possible[4] enunciations in natural language of these goals are the following (cf. Section 2.1 for their formalisations in CPL). Secrecy for a protocol: "Always and for all messages m, if it is forbidden that the adversary (Eve) *know* m then Eve does not *know* m." (knowledge de re in the present subjunctive and the present indicative mode respectively). Authenticity of a message m from the viewpoint of protocol participant a w.r.t. participant b: "a *knows that* once only b *knew* m." (knowledge de dicto in the present and knowledge de re in the past indicative mode). Non-repudiation of authorship of a message m' by b w.r.t. a corroborated by a proof m (m is a proof for a that b is the author of m'): "If a *knew* m then a *would know that* once only b *knew* m'." (knowledge de re in the past subjunctive and then in the past indicative mode, and knowledge de dicto in the conditional mode). However, general-purpose/standard epistemic logic is inadequate in a cryptographic setting due to weak paradoxes; for the

[4] as a matter of fact unique definitions of these goals do not exist (yet)

same reason (standard) deontic logic is inadequate (cf. Section 2.3). And doxastic logic is inadequate due to its inadequacy for the above goals as these crucially rely on knowledge, i.e., necessarily true, and not possibly false, belief (no control of the epistemic error). Finally, special-purpose logics have been limited in their adequacy due to their choice of primitive concepts, e.g., belief, no negation/quantification, too specific concepts at the price of high extension costs.

Our goal is to supply a formal *synthesis* of logical concepts in a single, multi-dimensional[5] modal logic, namely CPL[6], that yields requirements that are *intuitive* but *abstract* w.r.t. particular models of cryptography. First, we believe that the formal method for any science is ultimately logic, as defined by a relation of satisfaction (*model*-theoretic approach[7], effectuated via *model checking* [46]) or a relation of deduction (*proof*-theoretic approach, effectuated via *automated theorem proving* [47]). Second, given that requirements engineering is mainly about meaning, i.e., understanding and formalising properties, we believe that a model-theoretic approach is — at least at a first stage — more suitable than a proof-theoretic approach. By 'intuitive' we mean that the conceptual dimensions of the requirement are apparent in distinctive forms in the formula that expresses the requirement — succinctly.

We argue that propositional and higher-order (at least beyond second order) logic, and set theory are unsuitable as front-end formalisms for requirements engineering purposes. Propositional logic is simply too weak as a specification language but is well-suited for fully-automated, approximative verification. Higher-order logic and set-theory may well be semantically sufficiently expressive; however, we opine that they are unsuitable for engineers in charge of capturing meaning of protocol requirements within an acceptable amount of time (i.e., financial cost per specification) and space (i.e., intelligibility of specifications). The intuitiveness of the specifications that a formalism yields are not just luxury, but the very — and difficult to distil — essence and a measure of its *pragmatics*, i.e., practical usefulness. The application domain of cryptographic protocols is conceptually very rich. A suitable requirements engineering formalism must organise and hard-wire/pre-compile this conceptual variety in its semantics and provide succinct and intuitive linguistic abstractions (syntax) for them. The resulting added value of such a formalism is empowerment of the engineer (speed-up of the mental process of formalisation), and more powerful tools (speed-up of model checking and automated theorem proving). Higher-order logic and set-theory, having been conceived as general-purpose formalisms, obviously lack this special-purpose semantics and syntax. However, they are well-suited as logical frameworks (back-ends) for such special-purpose formalisms (object logics). For

[5] cf. [43] for a research monograph on multi-dimensional modal logic (an active research area), characterised in [44] as "...a branch of modal logic dealing with special relational structures in which the states, rather than being abstract entities, have some inner structure. ...Furthermore, the accessibility relations between these states are (partly) determined by this inner structure of the states."

[6] a preliminary, now outdated version of CPL appeared in the informal proceedings of [45]

[7] not to be confused with a model-*based formalism*

example, our candidate language has a model-theoretic (i.e., relying on set theory) semantics.

CPL has a *first-order* fragment for making statements about protocol events and about the (individual) knowledge ("knows") and the structure of cryptographic messages induced by those events; and four *modal* fragments for making statements about confidentiality *norms* (cf. deontic logic [20]); propositional *knowledge* ("knows that"), i.e., knowledge of cryptographic states of affairs, (cf. epistemic logic [48]); execution *space* (cf. spatial logic [49]); and execution *time* (cf. temporal logic [6]). That is, CPL *unifies* first-order and four modal logics in a single, multi-dimensional logic. Further, CPL *refines* standard epistemic and deontic logic in the sense that it resolves the long-standing problem of weak-paradoxes (caused by logical omniscience and conflicting obligations, respectively) that these logics exhibit when applied in a cryptographic setting (cf. Section 2.3). Yet CPL (a property-based formalism) goes even further in its *wholistic ambition* in that it integrates the perhaps most important model-based framework, namely process algebra [50], in a novel way. First, CPL's temporal accessibility relation (the semantics of its temporal modalities) can be defined by an event-trace generating process (reduction) calculus, e.g., C^3 [51, 52] whose execution constraints are moreover checkable via CPL-satisfaction; and second, CPL's epistemic accessibility relation (the semantics of its epistemic modality "knows that") is the definitional basis for C^3's observational process equivalence, which can be used for the model-based (process-algebraic and complementary to property-based) formulation of protocol requirements.

A cryptographic protocol involves the concurrent interaction of participants that are physically separated by — and exchange messages across — an unreliable and insecure transmission medium. Expressing properties of concurrent interaction requires temporal operators [6]. The physical separation by an unreliable and insecure transmission medium in turn demands the epistemic and deontic modalities. To see why, consider that the existence of such a separating medium introduces an *uncertainty* among protocol participants about the *trustworthiness* of the execution of protocol actions (sending and receiving) and the contents of exchanged messages, both w.r.t. *actuality* (an epistemic concern) and *legitimacy* (a deontic concern). It is exactly the role of a cryptographic protocol to re-establish this trustworthiness through the judicious use of *cryptographic evidence*, i.e., essential information (e.g., ciphers, signatures and hash values) for the knowledge of other information (e.g., messages or truth of formulae), bred in a *crypto system* (e.g., a shared-key or public-key system) from *cryptographic germs* such as keys and nonces, themselves generated from *cryptographic seeds* (or seed values). However, any use of keys (as opposed to hash values and nonces) requires that the knowledge of those keys be shared a priori. This sharing of key knowledge is established by cryptographic protocols called *key-establishment* protocols (comprising *key-transport* and *key-agreement* protocols) [3, Chapter 12], which are executed before any cryptographic protocol that may then subsequently use those keys. Thus certain cryptographic protocols must be considered interrelated by a notion of *composition* in a common execution *space*; hence the need of spatial operators. Another argument for spatial operators comes from the fact that a correct protocol should conserve its inner correctness even when composed with other protocols, i.e., a (totally) correct protocol should be *stable under different execution contexts* [53, 54].

2 Logic

2.1 Syntax

The language $\mathcal{F}$ of CPL is parametric in the language $\mathcal{M}$ of its individuals, i.e., protocol messages. It is chiefly relational, and functional in exactly the language $\mathcal{M}$ of protocol messages it may be instantiated with. The temporal fragment of $\mathcal{F}$ coincides with the syntax of LTLP (linear temporal logic with past). We shall fix our mind on the following, comprehensive language $\mathcal{M}$ of individuals.

Definition 1 (Protocol messages). *Protocol messages $M \in \mathcal{M}$ have the following structure. $M ::= n$ (names, logical constants) $\big|$ $\blacksquare$ (the abstract message) $\big|$ p^+ (public keys) $\big|$ $\lceil M \rceil$ (message hashes) $\big|$ $\{\!|M|\!\}_M$ (symmetric message ciphers) $\big|$ $\{\!|M|\!\}^+_{p^+}$ (asymmetric message ciphers) $\big|$ $\{\!|M|\!\}^-_p$ (signed messages) $\big|$ (M, M) (message tuples).*

Names $n \in \boldsymbol{\mathcal{N}}$ are participant names $a, b \in \boldsymbol{\mathcal{P}}$, the (for the moment Dolev-Yao [55]) adversary's name `Eve`*, symmetric session ($\boldsymbol{\mathcal{K}}^1$) and long-term ($\boldsymbol{\mathcal{K}}^\infty$) keys $k \in \boldsymbol{\mathcal{K}}$, (asymmetric) private keys $p \in \boldsymbol{\mathcal{K}}^-$, and nonces $x \in \boldsymbol{\mathcal{X}}$ (also used as session identifiers). We assume that given a private key p, one can compute the corresponding public key p^+, as in DSA and Elgamal. Shared and private keys shall be referred to as* confidential keys *$\boldsymbol{\mathcal{CK}}$, i.e., keys that must remain secret. Symmetric keys may be* compound *for key* agreement *(as opposed to mere key* transport*). Message forms (open messages) F are messages with variables $v \in \mathcal{V}$.*

The abstract message is a computational artifice to represent the absence of *intelligibility*, just as the number zero is a computational artifice to represent the absence of *quantity*. The abstract message is very useful for doing knowledge-based calculations (cf. Definition 5), just as the number zero is very useful (to say the least) for doing number-based calculations. The focus on cryptographic protocols rather than cryptographic operators leads us (for the moment) to (1) making abstraction from the exact representation of messages, e.g., bit strings; and assuming (2.1) *perfect hashing*, i.e., collision resistance (hash functions are injective) and strong pre-image resistance (hash functions are not invertible, or given $\lceil M \rceil$, it is infeasible to compute M), and (2.2) *perfect encryption* (given $\{\!|M|\!\}_k$ but not the shared key k or given $\{\!|M|\!\}^+_{p^+}$ but not the private key p corresponding to the public key p^+, it is infeasible to compute M). We introduce a type language for messages to increase succinctness of statements about the structure of messages.

Definition 2 (Message types). *Message types τ have the following structure.*

$$\begin{aligned}
\tau,\tau' &::= \emptyset \mid \sigma \mid \mathtt{H}[\tau] \mid \mathtt{SC}_M[\tau] \mid \mathtt{AC}_{p^+}[\tau] \mid \mathtt{S}_p[\tau] \mid \mathtt{T}[\tau,\tau'] \mid \tau\cup\tau' \mid \tau\cap\tau' \mid \tau\setminus\tau' \mid \mathtt{M}\\
\sigma,\sigma' &::= \mathtt{P} \mid \mathtt{Adv} \mid \varsigma \mid \mathtt{K}^+\\
\varsigma,\varsigma' &::= \mathtt{K}^1 \mid \mathtt{K}^\infty \mid \mathtt{K}^- \mid \mathtt{X}
\end{aligned}$$

Message type forms θ shall be message types with variables in key position.

Observe that (1) for each kind of message there is a corresponding type (e.g., $\mathtt{H}[\tau]$ for hashes, $\mathtt{SC}_M[\tau]$ for symmetric and $\mathtt{AC}_{p^+}[\tau]$ for asymmetric ciphers, $\mathtt{S}_p[\tau]$ for signatures, and $\mathtt{T}[\tau,\tau']$ for tuples); (2) encryption and signature types are parametric; and (3) the union, intersection, and difference of two message types is again a message type. In short, message types are structure-describing *dependent types* closed under union, intersection, and difference. ς and ς' denote types of dynamically generable names. We macro-define $\mathtt{P}_{\mathtt{Adv}} := \mathtt{P}\cup\mathtt{Adv}$, $\mathtt{K} := \mathtt{K}^1\cup\mathtt{K}^\infty$, $\mathtt{CK} := \mathtt{K}\cup\mathtt{K}^-$, $\mathtt{K}^* := \mathtt{CK}\cup\mathtt{K}^+$, and $\mathtt{N} := \mathtt{P}_{\mathtt{Adv}}\cup\mathtt{K}^*\cup\mathtt{X}$.

Definition 3 (Formulae). *The set of formulae $\mathcal{F}$ contains precisely those propositions that are the closed predicates formed with the operators of Table 1. There, β denotes basic, α action, and δ data formulae; a, b, c denote participants and x, x' nonces; and k denotes a symmetric key and p a private key.*

Predicates can be transformed into propositions either via binding of free variables, i.e., universal (*generalisation*) or existential (*abstraction*) quantification, or via substitution of individuals for free variables (*individuation*). In accordance with standard logical methodology, basic predicates express *elementary facts.*

Table 1. Predicate language

$$\begin{aligned}
\phi,\phi' &::= \beta \mid \neg\phi \mid \phi\wedge\phi' \mid \forall v(\phi)\\
&\quad\mid\ \mathsf{P}\phi \mid \mathsf{K}_a(\phi) \mid \phi\supseteq\phi' \mid \phi\otimes\phi' \mid \phi\rhd\phi' \mid \phi\,\mathsf{S}\,\phi' \mid \ominus\phi \mid \oplus\phi \mid \phi\,\mathsf{U}\,\phi'\\
\beta,\beta' &::= \alpha \mid \delta\\
\alpha,\alpha' &::= a.x \circlearrowright x' \mid a.x \circlearrowright k.(b,c) \mid a.x \circlearrowright p.b\\
&\quad\mid\ a.x \xrightarrow[\mathtt{E\not{v}e}]{F} b \mid a.x \xrightarrow[\mathtt{Eve}]{F} b \mid a.x \xleftarrow[\mathtt{E\not{v}e}]{F} b \mid a.x \xleftarrow[\mathtt{Eve}]{} F\\
\delta,\delta' &::= n:\sigma \mid a\ \mathsf{k}\ F \mid F \preccurlyeq F'
\end{aligned}$$

Our symbols are — and their intuitive meaning is as they are — pronounced $\neg$ “not”, $\wedge$ “and”, $\forall v$ “for all v”, P “it is permitted that”, K_a “a knows that”, $\supseteq$ “epistemically/necessarily implies”, $\otimes$ “conjunctively separates”, $\rhd$ “assume—guarantee”, S “since”, $\ominus$ “previous”, $\oplus$ “next”, and U “until”, $a.x \circlearrowright x'$ “a freshly generated the nonce x' in session x”, $a.x \circlearrowright k.(b,c)$ “a freshly generated the symmetric key k for b and c in session x”, $a.x \circlearrowright p.b$ “a freshly generated the private key p for b in session x”, $a.x \xrightarrow[\mathtt{E\not{v}e}]{F} b$ “a securely (i.e., in a way unobservable by the adversary) sent off F *as such* (i.e., not only as a strict sub-term of another message) to b in session x”, $a.x \xrightarrow[\mathtt{Eve}]{F} b$ “a insecurely (i.e., in a way observable by the adversary) sent off F as such to b in session x”, $a.x \xleftarrow[\not{\mathtt{Eve}}]{F} b$ “a securely (i.e., not through the adversary) received F as such from b in session x”, $a.x \xleftarrow[\mathtt{Eve}]{} F$ “a insecurely (i.e., possibly from the adversary) received F as such in session x”, $:$ “has type”, k “knows”, and $\preccurlyeq$ “is a subterm of”.

Our language is *1-sorted* because protocol participants are referred to by their name and names are transmittable data, i.e., messages. K expresses knowledge *de dicto* (or *propositional* knowledge). In contrast, k expresses knowledge *de re* (or *individual* knowledge). Knowledge de re conveys understanding of the purpose and possession of a certain piece of cryptographic information up to cryptographically irreducible parts. It is established based on the capability of participants to synthesise those pieces from previously analysed pieces. By ‘understanding of the purpose’ we mean (1) knowledge of the *structure* for compound, and (2) knowledge of the *identity* for atomic (names) information. Note that such understanding requires that there be a *minimal redundancy* in that information. The conditional $\phi \supseteq \phi'$ is epistemic or necessary in the sense that the set of evidence corroborating truth of the consequent ϕ' (e.g., the knowledge of a key) is *included* in the set of evidence corroborating truth of the antecedent ϕ (e.g., the knowledge of a plain text *derived* from that key). The epistemic conditional captures the epistemic dependence of the truth of the antecedent on the truth of the consequent. The formula $\phi \otimes \phi'$ is satisfied by a (protocol) model if and only if the model can be separated in exactly two parts such that one part satisfies ϕ (e.g., key distribution/production) and the other satisfies ϕ' (e.g., key use/consumption). The formula $\phi \rhd \phi'$ is satisfied by a model if and only if for all models that satisfy ϕ the parallel composition of both models satisfies ϕ' (cf. total/compositional correctness of a protocol, as mentioned earlier). Typing formulae $F : \theta$ have an essential and a pragmatic purpose. Typing of *atomic* data, i.e., when F designates a name n and θ an atomic type σ, is a linguistic abstraction for the above-mentioned essential modelling hypothesis of minimal redundancy. Typing of *compound* data simply increases succinctness of statements about the structure of messages. It is actually macro-definable in terms of typing of atomic data, equality (itself macro-definable), and existential quantification (cf. Appendix B).

We exemplify the expressiveness of CPL on a selection of *tentative* formalisations of fundamental cryptographic states of affairs. To the best of our knowledge, (1) no other existing crypto logic is sufficiently expressive to allow for the *definition* of the totality of these properties, and (2) the totality of these properties has never been expressed before in any other formalism. In fact, entire logics (e.g., [16], [23], [24]) have been designed to capture a single cryptographic state of affairs (e.g., authenticity, anonymity, resp. secrecy). We invite the reader to validate our formalisations on the criteria of intuitiveness and succinctness, but also to discern that the simplicity of the formalisation *results* is in sharp contradistinction to the difficulty of their formalisation *process*. However, thanks to the empowerment that CPL confers, a formalisation process involving such a large number of conceptual degrees of freedom has become *tractable* at an engineering level. Note that the formalisations employ macro-defined predicates (cf. Appendix B; the reader is urged to consult it) and that $\alpha(b)$ abbreviates disjunction of name generation, sending, and receiving performed by b.

Maliciousness b is *malicious*, written $\mathsf{malicious}(b)$, :iff b knowingly performs a forbidden action at some time, written $\diamondplus(\alpha(b) \land \mathsf{F}\alpha(b) \land \mathsf{K}_b(\mathsf{F}\alpha(b)))$.

Honesty b is *honest*, written $\mathsf{honest}(b)$, :iff b is not malicious, written $\neg\mathsf{malicious}(b)$.

Faultiness b is *faulty*, written $\mathsf{faulty}(b)$, :iff b performs a forbidden action at some time, written $\diamondplus(\alpha(b) \land \mathsf{F}\alpha(b))$.

Prudency b is *prudent*, written $\mathsf{prudent}(b)$, :iff b is not faulty, written $\neg\mathsf{faulty}(b)$.

Trust a *trusts* b, written a trusts b, :iff a knows that b is prudent, written $\mathsf{K}_a(\mathsf{prudent}(b))$.

Reachability-based Secrecy A protocol has the secrecy property :iff the adversary (Eve) never knows any classified information, written $\boxplus\forall m(\mathsf{F}(\mathtt{Eve}\ \mathsf{k}\ m) \rightarrow \neg\,\mathtt{Eve}\ \mathsf{k}\ m)$.

Perfect Forward Secrecy "[...] compromise of long-term keys does not compromise past session keys." [3, Page 496], written $\neg\diamondplus(\exists(k : \mathtt{K}^1)(\mathtt{Eve}\ \mathsf{k}\ k) \supseteq \exists(k : \mathtt{K}^\infty)(\mathtt{Eve}\ \mathsf{k}\ k))$[8]

Anonymity b is anonymous to a in state of affairs $\phi(b)$:iff if a knows that some participant is involved in ϕ then a cannot identify that participant with b, written $\mathsf{K}_a(\exists(c : \mathtt{P})(\phi(c))) \rightarrow \neg\mathsf{K}_a(\phi(b))$.

Known-key attack "[...] an adversary obtains some keys used previously and then uses this information to determine new keys." [3, Page 41], written $\exists(v : \mathtt{CK})(\mathtt{Eve}\ \mathsf{k}\ v \land (\exists(v' : \mathtt{CK})(v' \neq v \land \mathtt{Eve}\ \mathsf{k}\ v') \supseteq \mathtt{Eve}\ \mathsf{k}\ v))$

Key confirmation for a w.r.t. b "[...] one party is assured that a second (possibly unidentified) party actually has possession of a particular secret key." [3, Page 492], written $k : \mathtt{K} \land \mathsf{K}_a(b\ \mathsf{k}\ k)$

Implicit key authentication for a w.r.t. b "[...] one party is assured that no other party aside from a specifically identified second party (and possibly additional identified trusted parties) may gain access to a particular secret key." [3, Page 492], written $k : \mathtt{K} \land \mathsf{K}_a(\forall(c : \mathtt{P}_{\mathtt{Adv}})(c\ \mathsf{k}\ k \rightarrow (c = a \lor c = b)))$

Shared secret among a and b M is a *shared secret among* a and b, written M $\mathsf{sharedSecretAmong}$ (a, b), :iff nobody except a and b know M, written $a\ \mathsf{k}\ M \land b\ \mathsf{k}\ M \land \forall(c : \mathtt{P}_{\mathtt{Adv}})(c\ \mathsf{k}\ M \rightarrow (c = a \lor c = b))$

Explicit key confirmation for a w.r.t. b "[...] both (implicit) key authentication and key confirmation hold." [3, Page 492], written $k : \mathtt{K} \land \mathsf{K}_a(k$ $\mathsf{sharedSecretAmong}$ $(a, b))$

Authorship of a datum a *authored* a datum M, written a $\mathsf{authored}$ M, :iff once a was the only one to know M, written $\diamondminus(a\ \mathsf{k}\ M \land \forall(b : \mathtt{P}_{\mathtt{Adv}})(b\ \mathsf{k}\ M \rightarrow b = a))$.

[8] A material conditional would not do here because the antecedent and the consequent are *epistemically* — and thus not truth-functionally — *related* via key corruption, i.e., the *derivation* of a session key from a corrupted long-term key.

Authenticity of a datum A datum M is *authentic w.r.t. its origin* (say a) from the viewpoint of b, written $M\ \mathsf{authentic}\ (a, b)$, :iff b can authentically attribute (i.e., in the sense of authorship) M to a, i.e., b knows that a authored M, written $\mathsf{K}_b(a\ \mathsf{authored}\ M)$.

Individual proof for a proposition M is a *proof for a proposition* ϕ, written $M\ \mathsf{proofFor}\ \phi$, :iff assuming an arbitrary a knows M guarantees that a knows that ϕ is true, written $\forall(a : \mathtt{P})(a\ \mathsf{k}\ M \rhd \mathsf{K}_a(\phi))$.

Non-repudiation of authorship b cannot repudiate authorship of M to a :iff a has a proof that b authored M, written $\exists m(m\ \mathsf{proofFor}\ (b\ \mathsf{authored}\ M) \wedge a\ \mathsf{k}\ m)$.

Authentication of a with b "[...] the process whereby one party [b] is assured (through acquisition of corroborative evidence) of the identity of a second party [a] involved in a protocol, and that the second has actually participated (i.e., is active at, or immediately prior to, the time the evidence is acquired)." [3, Page 386], written $\exists m(m\ \mathsf{authentic}\ (a, b) \wedge m\ \mathsf{newTo}\ b)$

Corruption of a participant a by the adversary `Eve` "`Eve` comes to know all what a knows in state of affairs ϕ", written $\forall m(a\ \mathsf{k}\ m \to (\mathtt{Eve}\ \mathsf{k}\ m \blacktriangleright \phi))$

Compositional protocol correctness protocol (plug-in) P in (initial) state $\mathfrak{h}$ satisfies property ϕ provided that P is used with a (parallel) protocol (environment) satisfying φ, written $(P, \mathfrak{h}) \models \varphi \rhd \phi$.

2.2 Semantics

Our definition of satisfaction is *anchored* (or *rooted*) and defined on protocol states, i.e., tuples $(P, \mathfrak{h}) \in \mathcal{P} \times \mathcal{H}$ of a protocol model P (i.e., a parallel-composable process term) and a protocol history $\mathfrak{h}$ (i.e., an event trace). For the purpose of this paper, we presuppose a notion of execution (e.g., [51]) $\longrightarrow \subseteq (\mathcal{P} \times \mathcal{H}) \times (\mathcal{P} \times \mathcal{H})$ (or relation of *temporal accessibility* in the jargon of modal logic) producing protocol events and chaining them up to form protocol histories. Protocol events have the following form: generation of a nonce x' in session x by a, written $\mathtt{N}(a, x, x')$; generation of a fresh symmetric key k for b and c in session x by a, written $\mathtt{N}(a, x, k, (b, c))$; generation of a fresh private key p for b in session x by a, written $\mathtt{N}(a, x, p, b)$; insecure input of M in session x by a, written $\mathtt{I}(a, x, M)$; secure input of M from b in session x by a, written $\mathtt{sI}(a, x, M, b)$; insecure output of M to b in session x by a, written $\mathtt{O}(a, x, M, b)$; and secure output of M to b in session x by a, written $\mathtt{sO}(a, x, M, b)$. By definition, an event ε is secure if and only if ε is unobservable by the adversary `Eve`. By convention, name generation is a secure event. We write $\varepsilon(a)$ for any of the above protocol events, $\varepsilon(a, n)$ for any of the above name-generation events, $\varepsilon(a, M)$ for any of the above communication events, and $\hat{\varepsilon}(a)$ for any of the above secure events. Protocol histories $\mathfrak{h} \in \mathcal{H}$ are simply finite words of protocol events ε, i.e., event traces $\mathfrak{h} \ ::= \ \epsilon \ \big| \ \mathfrak{h} \cdot \varepsilon$, where ϵ denotes the empty protocol history.

We define satisfaction in a functional style on the structure of formulae. Satisfaction employs *complex* (and thus multiple) truth values. Truth values are complex in the sense that they are tuples of a simple truth value (i.e., 'true' or 'false') and a set of those events (the evidence) that are necessary to corroborate that simple truth.

Definition 4 (Satisfaction). *Let* $\models\; \subseteq (\mathcal{P} \times \mathcal{H}) \times \mathcal{F}$ *denote satisfaction of a formula* $\phi \in \mathcal{F}$ *by a protocol state* $\mathfrak{s} \in \mathcal{P} \times \mathcal{H}$ *(the anchor/root of an implicit execution path model for* ϕ*):*

$$\begin{aligned} \mathfrak{s} \models \phi \quad &\textit{:iff there is a set of protocol events } \mathcal{E} \textit{ s.t. } \mathfrak{s} \models_{\mathcal{E}} \phi \\ \mathfrak{s} \models_{\mathcal{E}} \phi \quad &\textit{:iff for all } \mathfrak{p} \in \mathrm{paths}(\mathfrak{s}),\ [\![\phi]\!]^0_{\mathfrak{p}} = (\textit{true}, \mathcal{E}) \end{aligned}$$

where $\mathrm{paths}(\mathfrak{s})$ *denotes the set of paths* $\mathfrak{p}$ *achored/rooted in* $\mathfrak{s}$ *and induced by* $\longrightarrow$*, and* $[\![\cdot]\!]$ *denotes a function of truth denotation from formulae to complex truth values (cf. Table 2). There,*

- $\mathfrak{p}@i$ *denotes the state* $(P, \mathfrak{h})$ *at position* i *in path* $\mathfrak{p}$.
- $\dot{\mathfrak{h}}$ *denotes the set derived from protocol history* $\mathfrak{h}$.
- $\mathfrak{h} \vdash_a M$ *denotes the derivation of the individual knowledge* M *by participant* a *from* a*'s view on* $\mathfrak{h}$*, i.e., the extraction, analysis, and synthesis (cf. Appendix A) of the data that* a *has generated, received, or sent in* $\mathfrak{h}$.
- $\circ$ *denotes concatenation of protocol histories.*
- $\mathrm{E}^M_a(\mathfrak{h})$ *denotes the set of* earliest *events in* $\mathfrak{h}$ *that corroborate truth of the formula* $a \;\mathsf{k}\; M$*:*

$$\begin{aligned} \mathrm{E}^M_a(\mathfrak{h}) &:= \mathrm{E}^M_a(\mathfrak{h}, \epsilon) \\ \mathrm{E}^M_a(\mathfrak{h} \cdot \varepsilon, \mathfrak{h}') &:= \begin{cases} \mathrm{E}^M_a(\mathfrak{h}, \mathfrak{h}') & \textit{if } \mathfrak{h} \circ \mathfrak{h}' \vdash_a M, \textit{ and} \\ \mathrm{E}^M_a(\mathfrak{h}, \mathfrak{h}' \cdot \varepsilon) \cup \{\varepsilon\} & \textit{otherwise.} \end{cases} \\ \mathrm{E}^M_a(\epsilon, \mathfrak{h}') &:= \emptyset \end{aligned}$$

- $\Sigma := \exists(k : \mathtt{CK})(\mathtt{Eve}\ \mathsf{k}\ k \wedge \neg\, k\ \mathsf{ck}\ \mathtt{Eve})$ *denotes a state formula expressing the* fundamental state of violation *in a cryptographic setting, namely the one where the adversary has come to know a confidential key not of her own.*
- $\approx_a\; \subseteq (\mathcal{P} \times \mathcal{H}) \times (\mathcal{P} \times \mathcal{H})$ *denotes the relation of* epistemic accessibility *associated with the modality* K_a*; it is defined hereafter.*
- $\dot{(\!|\cdot|\!)}_a$ *denotes a unary function (inspired by [56]) of* cryptographic parsing *defined on protocol states and on logical formulae; it is defined hereafter on messages and tacitly lifted onto protocol states and logical formulae.*
- $\equiv$ *denotes a relation of* structural equivalence *defined on process terms and on event traces. On process terms, it denotes the smallest equivalence relation expressing associativity and commutativity of processes. On event traces, it denotes permutation, i.e.,* $\mathfrak{h} \equiv \mathfrak{h}'$ *:iff* $|\mathfrak{h}| = |\mathfrak{h}'|$ *and* $\dot{\mathfrak{h}} = \dot{\mathfrak{h}}'$*, where* $|\cdot|$ *denotes a length function.*

Permission is not macro-defined because we want to highlight that each new notion of fundamental state of violation will give rise to a new notion of permission. That is, we look at the state formula Σ as a parameter of the logic. The epistemic accessibility relation has, as previously mentioned, a double use;

Table 2. Truth denotation

$$\llbracket a.x \circlearrowright x' \rrbracket^i_{\mathfrak{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{N}(a, x, x')\} \cap \dot{\mathfrak{h}}$$

$$\llbracket a.x \circlearrowright k.(b, c) \rrbracket^i_{\mathfrak{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{N}(a, x, k, \{b, c\})\} \cap \dot{\mathfrak{h}}$$

$$\llbracket a.x \circlearrowright p.b \rrbracket^i_{\mathfrak{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{N}(a, x, p, b)\} \cap \dot{\mathfrak{h}}$$

$$\llbracket a.x \xrightarrow[\mathtt{E}\cancel{\mathtt{v}}\mathtt{e}]{M} b \rrbracket^i_{\mathfrak{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{sO}(a, x, M, b)\} \cap \dot{\mathfrak{h}}$$

$$\llbracket a.x \xrightarrow[\mathtt{Eve}]{M} b \rrbracket^i_{\mathfrak{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{O}(a, x, M, b)\} \cap \dot{\mathfrak{h}}$$

$$\llbracket a.x \xleftarrow[\mathtt{E}\cancel{\mathtt{v}}\mathtt{e}]{M} b \rrbracket^i_{\mathfrak{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{sI}(a, x, M, b)\} \cap \dot{\mathfrak{h}}$$

$$\llbracket a.x \underset{\mathtt{Eve}}{\leftarrow} M \rrbracket^i_{\mathfrak{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{I}(a, x, M)\} \cap \dot{\mathfrak{h}}$$

$$\llbracket n : \sigma \rrbracket^i_{\mathfrak{p}} := (n \text{ has type } \sigma, \emptyset)$$

$$\llbracket a \;\mathsf{k}\; M \rrbracket^i_{\mathfrak{p}} := (\mathfrak{h} \vdash_a M, \mathrm{E}^M_a(\mathfrak{h}))$$

$$\llbracket M \preccurlyeq M' \rrbracket^i_{\mathfrak{p}} := (M \text{ is a subterm of } M', \emptyset)$$

$$\llbracket \neg\phi \rrbracket^i_{\mathfrak{p}} := (\text{not } \mathsf{v}_\phi, \dot{\mathfrak{h}} \setminus \mathcal{E}_\phi) \quad \text{where } \llbracket \phi \rrbracket^i_{\mathfrak{p}} = (\mathsf{v}_\phi, \mathcal{E}_\phi)$$

$$\llbracket \phi \wedge \phi' \rrbracket^i_{\mathfrak{p}} := (\mathsf{v}_\phi \text{ and } \mathsf{v}_{\phi'}, \mathcal{E}_\phi \cup \mathcal{E}_{\phi'}) \quad \text{where } \begin{array}{l} \llbracket \phi \rrbracket^i_{\mathfrak{p}} = (\mathsf{v}_\phi, \mathcal{E}_\phi) \text{ and} \\ \llbracket \phi' \rrbracket^i_{\mathfrak{p}} = (\mathsf{v}_{\phi'}, \mathcal{E}_{\phi'}) \end{array}$$

$$\llbracket \forall v(\phi) \rrbracket^i_{\mathfrak{p}} := (\text{for all } M \in \mathcal{M},\ \mathsf{v}_M, \textstyle\bigcup_{M \in \mathcal{M}} \mathcal{E}_M) \quad \text{where } \llbracket \{{}^M\!/_v\}\phi \rrbracket^i_{\mathfrak{p}} = (\mathsf{v}_M, \mathcal{E}_M)$$

$$\llbracket \mathsf{P}\phi \rrbracket^i_{\mathfrak{p}} := \llbracket \phi \rhd \boxplus(\Sigma \to (\Sigma \not\supseteq \phi)) \rrbracket^i_{\mathfrak{p}}$$

$$\begin{aligned} \llbracket \mathsf{K}_a(\phi) \rrbracket^i_{\mathfrak{p}} := {} & (\text{for all } \mathfrak{s}, \text{ if } \ \mathfrak{p}@0 \longrightarrow^* \mathfrak{s} \text{ and } \mathfrak{s} \approx_a \mathfrak{p}@i \ \text{ then } \ \mathfrak{s}' \models_{\mathcal{E}'} \phi', \mathcal{E}'_{(\mathfrak{s},\phi)}) \\ & \text{where } (\mathfrak{s}', \phi') := \begin{cases} (\mathfrak{s}, \phi) & \text{if } \mathfrak{s} = \mathfrak{p}@i, \text{ and} \\ ((\!| \mathfrak{s} |\!)^{\mathfrak{p}@i}_a, (\!| \phi |\!)^{\mathfrak{p}@i}_a) & \text{otherwise.} \end{cases} \end{aligned}$$

$$\llbracket \phi \supseteq \phi' \rrbracket^i_{\mathfrak{p}} := (\underline{\text{if } \mathsf{v}_\phi \text{ then } \mathsf{v}_{\phi'}} \text{ and } \mathcal{E}_{\phi'} \subseteq \mathcal{E}_\phi, \mathcal{E}_\phi) \quad \text{where } \begin{array}{l} \llbracket \phi \rrbracket^i_{\mathfrak{p}} = (\mathsf{v}_\phi, \mathcal{E}_\phi) \text{ and} \\ \llbracket \phi' \rrbracket^i_{\mathfrak{p}} = (\mathsf{v}_{\phi'}, \mathcal{E}_{\phi'}) \end{array}$$

$$\begin{aligned} \llbracket \phi \otimes \phi' \rrbracket^i_{\mathfrak{p}} := {} & (\text{there is } Q \in \mathcal{P} \text{ and } Q' \in \mathcal{P} \text{ s.t. } P \equiv Q \mathbin{|||} Q' \text{ and } (Q, \mathfrak{h}) \models_{\mathcal{E}_\phi} \phi \\ & \ \text{and } (Q', \mathfrak{h}) \models_{\mathcal{E}_{\phi'}} \phi',\ \mathcal{E}_\phi \cup \mathcal{E}_{\phi'}) \end{aligned}$$

$$\begin{aligned} \llbracket \phi \rhd \phi' \rrbracket^i_{\mathfrak{p}} := {} & (\text{for all } (Q, \mathfrak{h}') \in \mathcal{P} \times \mathcal{H} \text{ and } \mathfrak{h}'' \equiv \mathfrak{h}' \circ \mathfrak{h}, \text{ if } \ (Q, \mathfrak{h}') \models_{\mathcal{E}'} \phi \ \text{ then} \\ & \ (Q \mathbin{|||} P, \mathfrak{h}'') \models_{\mathcal{E}''} \phi', \textstyle\bigcup \mathcal{E}'' \cup \bigcup \mathcal{E}') \end{aligned}$$

$$\begin{aligned} \llbracket \phi \;\mathsf{S}\; \phi' \rrbracket^i_{\mathfrak{p}} := {} & (\text{there is } k \text{ s.t. } 0 \leq k \leq i \text{ and } \mathsf{v}_k \text{ and for all } j, \text{ if } k < j \leq i \text{ then } \mathsf{v}_j, \\ & \textstyle\bigcup_j \mathcal{E}_j \cup \mathcal{E}_k) \quad \text{where } \llbracket \phi \rrbracket^j_{\mathfrak{p}} = (\mathsf{v}_j, \mathcal{E}_j) \text{ and } \llbracket \phi' \rrbracket^k_{\mathfrak{p}} = (\mathsf{v}_k, \mathcal{E}_k) \end{aligned}$$

$$\llbracket \ominus \phi \rrbracket^i_{\mathfrak{p}} := \begin{cases} \llbracket \phi \rrbracket^{i-1}_{\mathfrak{p}} & \text{if } i > 0, \text{ and} \\ (\text{false}, \emptyset) & \text{otherwise.} \end{cases}$$

$$\llbracket \oplus \phi \rrbracket^i_{\mathfrak{p}} := \begin{cases} \llbracket \phi \rrbracket^{i+1}_{\mathfrak{p}} & \text{if } i < |\mathfrak{p}| - 1, \text{ and} \\ (\text{false}, \emptyset) & \text{otherwise.} \end{cases}$$

$$\begin{aligned} \llbracket \phi \;\mathsf{U}\; \phi' \rrbracket^i_{\mathfrak{p}} := {} & (\text{there is } k \text{ s.t. } i \leq k \text{ and } \mathsf{v}_k \text{ and for all } j, \text{ if } i \leq j < k \text{ then } \mathsf{v}_j, \\ & \textstyle\bigcup_j \mathcal{E}_j \cup \mathcal{E}_k) \quad \text{where } \llbracket \phi \rrbracket^j_{\mathfrak{p}} = (\mathsf{v}_j, \mathcal{E}_j) \text{ and } \llbracket \phi' \rrbracket^k_{\mathfrak{p}} = (\mathsf{v}_k, \mathcal{E}_k) \end{aligned}$$

it not only serves as the definitional basis for the epistemic modality (K_a) of CPL, but also as the definitional basis for the observational process equivalence of C^3 [51]. Cryptographic parsing captures an agent's capability to understand the structure of a cryptographically obfuscated message. It allows the definition of a cryptographically meaningful notion of epistemic accessibility via the intermediate concept of structurally indistinguishable protocol histories. The idea is to parse unintelligible messages to the abstract message $\blacksquare$.

Definition 5 (Cryptographic parsing). *The cryptographic parsing function* $\llparenthesis \cdot \rrparenthesis_a^{\mathfrak{h}}$ *associated with an agent* $a \in \mathcal{P}$ *and a protocol history* $\mathfrak{h} \in \mathcal{H}$ *(and complying with the assumptions of perfect cryptography) is an identity on names, the abstract message, and public keys; and otherwise acts as defined in Table 3.*

Table 3. Parsing on cryptographic messages

$$\llparenthesis \lceil M \rceil \rrparenthesis_a^{\mathfrak{h}} := \begin{cases} \lceil \llparenthesis M \rrparenthesis_a^{\mathfrak{h}} \rceil & \text{if } \mathfrak{h} \models a \mathrel{\mathsf{k}} M\text{, and} \\ \blacksquare & \text{otherwise.} \end{cases}$$

$$\llparenthesis \{\!| M |\!\}_{M'} \rrparenthesis_a^{\mathfrak{h}} := \begin{cases} \{\!| \llparenthesis M \rrparenthesis_a^{\mathfrak{h}} |\!\}_{\llparenthesis M' \rrparenthesis_a^{\mathfrak{h}}} & \text{if } \mathfrak{h} \models a \mathrel{\mathsf{k}} M'\text{, and} \\ \blacksquare & \text{otherwise.} \end{cases}$$

$$\llparenthesis \{\!| M |\!\}_{p^+}^{+} \rrparenthesis_a^{\mathfrak{h}} := \begin{cases} \{\!| \llparenthesis M \rrparenthesis_a^{\mathfrak{h}} |\!\}_{p^+}^{+} & \text{if } \mathfrak{h} \models a \mathrel{\mathsf{k}} p \vee (a \mathrel{\mathsf{k}} M \wedge a \mathrel{\mathsf{k}} p^+)\text{, and} \\ \blacksquare & \text{otherwise.} \end{cases}$$

$$\llparenthesis \{\!| M |\!\}_{p}^{-} \rrparenthesis_a^{\mathfrak{h}} := \begin{cases} \{\!| \llparenthesis M \rrparenthesis_a^{\mathfrak{h}} |\!\}_{p}^{-} & \text{if } \mathfrak{h} \models a \mathrel{\mathsf{k}} p^+\text{, and} \\ \blacksquare & \text{otherwise.} \end{cases}$$

$$\llparenthesis (M, M') \rrparenthesis_a^{\mathfrak{h}} := (\llparenthesis M \rrparenthesis_a^{\mathfrak{h}}, \llparenthesis M' \rrparenthesis_a^{\mathfrak{h}})$$

Definition 6 (Structurally indistinguishable protocol histories). *Two protocol histories* $\mathfrak{h}$ *and* $\mathfrak{h}'$ *are* structurally indistinguishable from the viewpoint of *an agent* a*, written* $\mathfrak{h} \approx_a \mathfrak{h}'$*, :iff* a *observes the same* event pattern *and the same* data patterns *in* $\mathfrak{h}$ *and* $\mathfrak{h}'$*. Formally, for all* $\mathfrak{h}, \mathfrak{h}' \in \mathcal{H}$*,* $\mathfrak{h} \approx_a \mathfrak{h}'$ *:iff* $\mathfrak{h} \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}'$ *where,*

- *given that* a *is a legitimate participant or the adversary* `Eve`*,*

1. $$\frac{}{\epsilon \approx_a^{(\mathfrak{h},\mathfrak{h}')} \epsilon}$$

2. $$\frac{\mathfrak{h}_l \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \cdot \varepsilon(a, n) \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r \cdot \varepsilon(a, n)}$$

3. $$\frac{\mathfrak{h}_l \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \cdot \varepsilon(a, M) \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r \cdot \varepsilon(a, M')} \; \llparenthesis M \rrparenthesis_a^{\mathfrak{h}} = \llparenthesis M' \rrparenthesis_a^{\mathfrak{h}'}$$

– *given that a is a legitimate participant,*

4. $$\frac{\mathfrak{h}_l \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \cdot \varepsilon(b) \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}\ a \neq b \qquad \frac{\mathfrak{h}_l \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \approx_a^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r \cdot \varepsilon(b)}\ a \neq b$$

– *given that a is the adversary* `Eve`*,*

5. $$\frac{\mathfrak{h}_l \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \cdot \hat{\varepsilon}(b) \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}\ \mathtt{Eve} \neq b \qquad \frac{\mathfrak{h}_l \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r \cdot \hat{\varepsilon}(b)}\ \mathtt{Eve} \neq b$$

6. $$\frac{\mathfrak{h}_l \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \cdot \mathtt{I}(b,x,M) \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r \cdot \mathtt{I}(b,x,M')}\ \llparenthesis M \rrparenthesis_{\mathtt{Eve}}^{\mathfrak{h}} = \llparenthesis M' \rrparenthesis_{\mathtt{Eve}}^{\mathfrak{h}'}$$

7. $$\frac{\mathfrak{h}_l \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r}{\mathfrak{h}_l \cdot \mathtt{O}(b,x,M,c) \approx_{\mathtt{Eve}}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r \cdot \mathtt{O}(b,x,M',c)}\ \llparenthesis M \rrparenthesis_{\mathtt{Eve}}^{\mathfrak{h}} = \llparenthesis M' \rrparenthesis_{\mathtt{Eve}}^{\mathfrak{h}'}$$

Note that the observations at the different (past) stages $\mathfrak{h}_l$ and $\mathfrak{h}_r$ in $\mathfrak{h}$ and $\mathfrak{h}'$ respectively must be made with the whole (present) knowledge of $\mathfrak{h}$ and $\mathfrak{h}'$ (cf. $\mathfrak{h}_l \approx_{\cdot}^{(\mathfrak{h},\mathfrak{h}')} \mathfrak{h}_r$). Learning new keys may render intelligible past messages to an agent a in the present that were not to her before.

Remark 1. For all agents a including `Eve`, $\approx_a \subseteq \mathcal{H} \times \mathcal{H}$ is (1) an equivalence with an infinite index due to fresh-name generation, (2) not a right-congruence due to the possibility of learning new keys, (3) a refinement on the projection $\mathcal{H}|_a$ of $\mathcal{H}$ onto a's view [48], and (4) decidable.

We lift structural indistinguishability from protocol histories to protocol states, i.e., tuples of a protocol term and a protocol history, and finally obtain our relation of epistemic accessibility.

Definition 7 (Structurally indistinguishable protocol states). *Let P_1 and P_2 denote two cryptographic processes, i.e., models of cryptographic protocols, of some set $\mathcal{P}$. Then two protocol states $(P_1, \mathfrak{h}_1)$ and $(P_2, \mathfrak{h}_2)$ are* structurally indistinguishable from the viewpoint of *an agent a, written $(P_1, \mathfrak{h}_1) \approx_a (P_2, \mathfrak{h}_2)$, :iff $\mathfrak{h}_1 \approx_a \mathfrak{h}_2$.*

2.3 Discussion

In the terminology of relevant logics, both the spatial conditional $\triangleright$ and the epistemic conditional $\supseteq$ are *relevant* (as opposed to the *truth-functional* material conditional $\rightarrow$) in the sense that information based on which the antecedent is evaluated is relevant to the information based on which the consequent is evaluated. In $\triangleright$, the relevant (and *potential*) information is represented by the adjoint state $(Q, \mathfrak{h}')$. In $\supseteq$, the relevant (and *actual*) information is represented by the event subset $\mathcal{E}_{\phi'}$.

As an example, consider (for relevance, the model part only mentions a protocol history)

$$\epsilon \cdot \mathtt{I}(\mathtt{Eve}, \blacksquare, \{\!|M|\!\}_k) \models \mathtt{Eve}\ \mathtt{k}\ k \rhd \mathtt{Eve}\ \mathtt{k}\ M$$

which states *what* primary knowledge, namely k, $\mathtt{Eve}$ *requires to derive* the (secondary) knowledge M in the given model. In other words, if $\mathtt{Eve}$ *knew* k then $\mathtt{Eve}$ *would know* M in the given model. This is a property of $\mathtt{Eve}'s$ cryptographic *knowledge* w.r.t. its *potentiality.* (The addition of information potentially leads to multiplication of knowledge.) In comparison, consider

$$\epsilon \cdot \mathtt{I}(\mathtt{Eve}, \blacksquare, \{\!|M|\!\}_k) \cdot \mathtt{I}(\mathtt{Eve}, \blacksquare, k) \models \mathtt{Eve}\ \mathtt{k}\ M \supseteq \mathtt{Eve}\ \mathtt{k}\ k$$

which states *how* $\mathtt{Eve}$ *actually derives* the secondary knowledge M from the primary knowledge in the given model. In other words, if $\mathtt{Eve}$ *knows* M then necessarily *because* $\mathtt{Eve}$ *knows* k in the given model. This is a property of $\mathtt{Eve}'s$ cryptographic *knowledge* w.r.t. its *actuality.* In contrast, consider (the tautology)

$$\models (\mathtt{Eve}\ \mathtt{k}\ \{\!|M|\!\}_k \land \mathtt{Eve}\ \mathtt{k}\ k) \to \mathtt{Eve}\ \mathtt{k}\ M$$

which states a property of a cryptographic *operation*, namely encryption. We believe that $\rhd$ and $\supseteq$ are (perhaps *the*) two natural — and incidentally, relevant — notions of implication for cryptographic knowledge.

A particularly interesting use of the spatial and the epistemic conditional is the definition of a cryptographically meaningful notion of permission (cf. Table 2) and prohibition (cf. Appendix B). Our definition says that it is permitted that ϕ is true if and only if if ϕ *were* true then whenever the fundamental state of violation *would be* reached, it *would not be* due to ϕ being true. This (reductionistic) notion of permission is inspired by [57, Page 9] where a notion of prohibition is defined in the framework of dynamic logic. The authors resume their basic idea as "...some action is forbidden if doing the action leads to a state of violation." Observe that [57] construe a notion of *prohibition* based on *actions*, whereas we construe a notion of *permission* based on *propositions.* We recall that the motivation of reductionistic approaches to (standard) deontic logic (SDL) is the existence of weak paradoxes in SDL. That is, SDL actually contains true statements that are counter to the normative intuition it was originally intended to capture.

In SDL permission, prohibition, and obligation are interdefinable, whereas in CPL only permission and prohibition are. In fact, there is no notion of obligation in CPL because (faulty) cryptographic protocols create a context with *conflicting obligations* whose treatment would require machinery from *defeasible* deontic logic [58]. Consider that it must be obligatory that (1) the fundamental state of violation be never reached during protocol execution, and (2) protocol participants always comply with protocol prescription. These two obligations are obviously conflicting in a context created by the execution of a faulty protocol, which by definition does reach the fundamental state of violation.

Our semantics for the epistemic modality reconciles the cryptographically intuitive but incomplete semantics from [59] with the complete (but less computational), renaming semantics from [60]. We achieve this by casting the cryptographic intuition from [59] in a simple (rule-based) and computational formulation of epistemic accessibility. Similarly to [59], we parse unintelligible data in an agent's *a individual* knowledge M into abstract messages $\blacksquare$. In addition, and inspired by [61, 60], we parse unintelligible data in an agent's *a propositional* knowledge $\mathsf{K}_a(\phi)$. Thanks to this additional parsing, our epistemic modality avoids weak paradoxes that, like in SDL, exist in standard epistemic logic (SEL). These paradoxes are due to epistemic necessitation, i.e., the fact that an agent a knows all logical truths such as $\exists v(\{\!|M|\!\}_k = \{\!|v|\!\}_k)$. To illustrate, consider the following simple example. Let $P \in \mathcal{P}$ and $M \in \mathcal{M}$. Then paradoxically $(P, \epsilon) \models \mathsf{K}_a(\exists v(\{\!|M|\!\}_k = \{\!|v|\!\}_k))$ "in" SEL but truthfully $(P, \epsilon) \not\models \mathsf{K}_a(\exists v(\{\!|M|\!\}_k = \{\!|v|\!\}_k))$ in CPL because $\models \neg\exists v(\blacksquare = \{\!|v|\!\}_k)$ (cf. "otherwise"-clause in the truth denotation of $\mathsf{K}_a(\phi)$ in Table 2). For further discussion see [61]. Note that our truth condition for the epistemic modality is an enhancement of the one from [61, 60] in the sense that we are able to eliminate one universal quantifier (the one over renamings) thanks to the employment of cryptographic parsing. Further note that our epistemic modality does capture knowledge, i.e., $\models \mathsf{K}_a(\phi) \rightarrow \phi$, due to the reflexivity of its associated accessibility relation.

3 Conclusion

We believe having accomplished with CPL an original synthesis of an unprecedented variety of logical concepts that are relevant to the logical modelling of cryptographic communication. In particular, we have (1) defined a cryptographically meaningful (in the sense of Dolev-Yao for the moment) epistemic modality, (2) invented a cryptographically interesting epistemic conditional, (3) pioneered the application of spatial logic to cryptographic concerns, and (4) shown that cryptographically meaningful deontic modalities are definable with a combination of epistemic and spatial conditional. At present, we are extending CPL with a notion of probabilistic polynomial-time computation in order to accommodate CPL to modern cryptography. Further, in case first-order CPL should not suffice for some applications, it would be trivial to extend CPL to the second-order. Just allow (unquoted) message types (denoting sets of messages) as messages, and quantification may range over second-order entities (sets). Finally, a proof system for CPL is also one of our a desiderata.

Acknowledgement I would like to thank Mika Cohen for our stimulating discussions about crypto logics and his constructive criticism of this paper.

A Auxiliary definitions

Data extraction

$$\frac{}{\mathfrak{h}\cdot\varepsilon(a,\overline{M}) \vdash_a (a,\overline{M})} \qquad \frac{\mathfrak{h} \vdash_a M}{\mathfrak{h}\cdot\varepsilon \vdash_a M}$$

Data synthesis ***Data analysis***

$$\frac{\mathfrak{h} \vdash_a M \quad \mathfrak{h} \vdash_a M'}{\mathfrak{h} \vdash_a (M,M')} \qquad \frac{\mathfrak{h} \vdash_a (M,M')}{\mathfrak{h} \vdash_a M} \qquad \frac{\mathfrak{h} \vdash_a (M,M')}{\mathfrak{h} \vdash_a M'}$$

$$\frac{\mathfrak{h} \vdash_a p}{\mathfrak{h} \vdash_a p^+} \qquad \frac{\mathfrak{h} \vdash_a M}{\mathfrak{h} \vdash_a \lceil M \rceil}$$

$$\frac{\mathfrak{h} \vdash_a M \quad \mathfrak{h} \vdash_a M'}{\mathfrak{h} \vdash_a \{\!|M|\!\}_{M'}} \qquad \frac{\mathfrak{h} \vdash_a \{\!|M|\!\}_{M'} \quad \mathfrak{h} \vdash_a M'}{\mathfrak{h} \vdash_a M}$$

$$\frac{\mathfrak{h} \vdash_a M \quad \mathfrak{h} \vdash_a p^+}{\mathfrak{h} \vdash_a \{\!|M|\!\}^+_{p^+}} \qquad \frac{\mathfrak{h} \vdash_a \{\!|M|\!\}^+_{p^+} \quad \mathfrak{h} \vdash_a p}{\mathfrak{h} \vdash_a M}$$

$$\frac{\mathfrak{h} \vdash_a M \quad \mathfrak{h} \vdash_a p}{\mathfrak{h} \vdash_a \{\!|M|\!\}^-_{p}} \qquad \frac{\mathfrak{h} \vdash_a \{\!|M|\!\}^-_{p} \quad \mathfrak{h} \vdash_a p^+}{\mathfrak{h} \vdash_a M}$$

B Specification library

Classical propositional and first-order operators

$\top \ := \ \mathtt{Eve} : \mathtt{Adv}$	true
$\bot \ := \ \neg\top$	false
$\phi \vee \phi' \ := \ \neg(\phi \wedge \phi')$	ϕ or ϕ'
$\phi \rightarrow \phi' \ := \ \neg\phi \vee \phi'$	if ϕ then ϕ'
$\phi \leftrightarrow \phi' \ := \ (\phi \rightarrow \phi') \wedge (\phi' \rightarrow \phi)$	ϕ if and only if ϕ'
$\exists v(\phi) \ := \ \neg\forall v(\neg\phi)$	there is v s.t. ϕ
$\forall(v:\theta)(\phi) \ := \ \forall v(v:\theta \rightarrow \phi)$	
$\exists(v:\theta)(\phi) \ := \ \exists v(v:\theta \wedge \phi)$	

Modal operators

$\mathsf{F}\phi \ := \ \neg\mathsf{P}\phi$	it is forbidden that ϕ
$\phi \equiv \phi' \ := \ (\phi \supseteq \phi') \wedge (\phi' \supseteq \phi)$	ϕ is epistemically equivalent to ϕ'
$\phi \oplus \phi' \ := \ \neg(\neg\phi \otimes \neg\phi')$	ϕ disjunctively separates ϕ'
$\Box\phi \ := \ \phi \oplus \bot$	everywhere ϕ
$\Diamond\phi \ := \ \neg\Box\neg\phi$	somewhere ϕ
$\phi' \blacktriangleright \phi \ := \ \neg(\phi' \rhd \neg\phi)$	assert ϕ' guarantee ϕ
$\boxminus\phi \ := \ \phi\,\mathsf{S}\,\bot$	so far ϕ

$\diamondminus\phi \;:=\; \neg\boxminus\neg\phi$	once ϕ
$1.\phi \;:=\; \phi \wedge \neg\ominus\diamondminus\phi$	for the first time ϕ
$\boxplus\phi \;:=\; \phi \,\mathsf{U}\, \bot$	henceforth ϕ
$\diamondplus\phi \;:=\; \neg\boxplus\neg\phi$	eventually ϕ
$\phi \leq \phi' \;:=\; (\phi \wedge \diamondplus\phi') \vee (\phi' \wedge \diamondminus\phi)$	ϕ before ϕ'
$\phi \rightsquigarrow \phi' \;:=\; (\phi \leftrightarrow \diamondplus\phi') \wedge (\phi' \leftrightarrow \diamondminus\phi)$	ϕ' is being caused by ϕ
$\phi \rightarrowtail \phi' \;:=\; \boxplus(\phi \rightsquigarrow \phi')$	ϕ causes ϕ'

Relational symbols

$F = F' \;:=\; F \preccurlyeq F' \wedge F' \preccurlyeq F$	F is equal to F'
$F \prec F' \;:=\; F = F' \wedge \neg\, F \preccurlyeq F'$	F is a strict subterm of F'
$a \;\mathsf{h}\; F \;:=\; \exists v(F \preccurlyeq v \wedge a \;\mathsf{k}\; v)$	a has/possesses F
$a \;\mathsf{tk}\; F \;:=\; a \;\mathsf{h}\; F \wedge \neg\, a \;\mathsf{k}\; F$	a tacitly knows F
$F : \emptyset \;:=\; \bot$	
$F : \mathtt{H}[\theta] \;:=\; \exists(v : \theta)(F = \lceil v \rceil)$	
$F : \mathtt{SC}_{F'}[\theta] \;:=\; \exists(v : \theta)(F = \{\!\!\mid v \mid\!\!\}_{F'})$	
$F : \mathtt{AC}_{p^+}[\theta] \;:=\; \exists(v : \theta)(F = \{\!\!\mid v \mid\!\!\}^{+}_{p^+})$	
$F : \mathtt{S}_p[\theta] \;:=\; \exists(v : \theta)(F = \{\!\!\mid v \mid\!\!\}^{-}_{p})$	
$F : \mathtt{T}[\theta, \theta'] \;:=\; \exists(v : \theta)\exists(v' : \theta')(F = (v, v'))$	
$F : \theta \cup \theta' \;:=\; F : \theta \vee F : \theta'$	
$F : \theta \cap \theta' \;:=\; F : \theta \wedge F : \theta'$	
$F : \theta \setminus \theta' \;:=\; F : \theta \wedge \neg\, F : \theta'$	
$F : \mathtt{M} \;:=\; \top$	
$F : \mathtt{SC}[\theta] \;:=\; \exists v(F : \mathtt{SC}_v[\theta])$	
$F : \mathtt{AC}[\theta] \;:=\; \exists(v : \mathtt{K}^+)(F : \mathtt{AC}_v[\theta])$	
$F : \mathtt{C}[\theta] \;:=\; F : \mathtt{SC}[\theta] \cup \mathtt{AC}[\theta]$	
$F : \mathtt{S}[\theta] \;:=\; \exists(v : \mathtt{K}^-)(F : \mathtt{S}_v[\theta])$	
$\theta \sqsubseteq \theta' \;:=\; \forall(v : \theta)(v : \theta')$	θ is a subtype of θ'
$\theta = \theta' \;:=\; \theta \sqsubseteq \theta' \wedge \theta' \sqsubseteq \theta$	
$\theta \sqsubset \theta' \;:=\; \theta \sqsubseteq \theta' \wedge \theta' \neq \theta$	
$F \mathrel{\varepsilon\!\!-} F' \;:=\; \exists v(\{\!\!\mid v \mid\!\!\}_{F} \preccurlyeq F')$	
$p^+ \mathrel{\varepsilon\!\!-}^{+} F \;:=\; \exists v(\{\!\!\mid v \mid\!\!\}^{+}_{p^+} \preccurlyeq F)$	
$p \mathrel{\varepsilon\!\!-}^{-} F \;:=\; \exists v(\{\!\!\mid v \mid\!\!\}^{-}_{p} \preccurlyeq F)$	
$F \mathrel{\varepsilon\!\!-}^{*} F' \;:=\; F \mathrel{\varepsilon\!\!-} F' \vee F \mathrel{\varepsilon\!\!-}^{+} F' \vee F \mathrel{\varepsilon\!\!-}^{-} F'$	F is operational in F'
$F \mathrel{-\!\!\backepsilon} F' \;:=\; \exists v \exists v'(F \preccurlyeq v \wedge \{\!\!\mid v \mid\!\!\}_{v'} \preccurlyeq F')$	
$F \mathrel{-\!\!\backepsilon}^{+} F' \;:=\; \exists v \exists(p^+ : \mathtt{K}^+)(F \preccurlyeq v \wedge \{\!\!\mid v \mid\!\!\}^{+}_{p^+} \preccurlyeq F')$	
$F \mathrel{-\!\!\backepsilon}^{-} F' \;:=\; \exists v \exists(p : \mathtt{K}^-)(F \preccurlyeq v \wedge \{\!\!\mid v \mid\!\!\}^{-}_{p} \preccurlyeq F')$	

$F \mathrel{-\!\!\!3}^{*} F' := F \mathrel{-\!\!\!3} F' \vee F \mathrel{-\!\!\!3}^{+} F' \vee F \mathrel{-\!\!\!3}^{-} F'$ — F is guarded in F'

$F \mathsf{\ newTo\ } a := \mathsf{1}.(a \mathrel{\mathsf{k}} F)$ — F is new to a

$n \mapsto x := \diamondminus \mathsf{1}.\exists m(n \mathrel{\varepsilon\!\!-}^{*} m \wedge \exists a \exists b(a.x \xrightarrow[\mathtt{Eve}]{m} b))$ — n is for session x

$k \mathsf{\ sk\ } a := \exists b \exists x \exists o(b.x \circlearrowleft k.o \wedge a \preccurlyeq o)$ — k is a symmetric key for a

$k \mathsf{\ sk}^{1} a := k \mathsf{\ sk\ } a \wedge k : \mathtt{K}^{1}$ — k is a session/short-term key for a

$k \mathsf{\ sk}^{\infty} a := k \mathsf{\ sk\ } a \wedge k : \mathtt{K}^{\infty}$ — k is a long-term key for a

$n \mathsf{\ prk\ } a := \exists b \exists x(b.x \circlearrowright n.a)$ — p is a private key for a

$n \mathsf{\ puk\ } a := \exists v(v^{+} = n \wedge v \mathsf{\ prk\ } a)$ — n is a public key for a

$n \mathsf{\ ck\ } a := n \mathsf{\ sk\ } a \vee n \mathsf{\ prk\ } a$ — n is a confidential key for a

C Timed Cryptographic Protocol Logic

C.1 Introduction

The formal modelling, specification, and verification of *general-purpose timed* systems has received considerable attention from the formal methods community since the end of the nineteen-eighties. See [62] for a survey of timed models (automata, Petri nets), model- and property-based specification languages (process calculi, resp. logics), and verification tools; and [63] for a survey of timed property-based specification languages (logics).

However, the formal methods community has paid comparatively little, and only recent (since the end of the nineteen-nineties), attention to the timed aspects of *cryptographic* systems, e.g., cryptographic protocols, which due to their complexity deserve *special-purpose* models, and formalisms for their specification and verification.

We are aware of the following special-purpose formalisms for timed cryptographic protocols. Model-based formalisms (process calculi): [64], [65], [66] with *discrete* time; [67], [68], and our own contribution [52] with *dense* time. Property-based formalisms (logics): *interval*-based [69]; time-parametrised epistemic modalities [70] and a third-order logic [68] both *point*-based, and our hereby presented logic tCPL allowing for *both* temporal points *and* intervals.

Clearly, "[d]ense-time models are better for distributed systems with multiple clocks and timers, which can be tested, set, and reset independently." [62]. Specifically in cryptographic systems [71], "[c]locks can become unsynchronized due to sabotage on or faults in the clocks or the synchronization mechanism, such as overflows and the dependence on potentially unreliable clocks on remote sites [. . .]". Moreover [71], "[e]rroneous behaviors are generally expected during clock failures [. . .]".

Timed logics can be classified w.r.t. their *order* and the nature of their *temporal domain.* Order: propositional logic is simply too weak for specification purposes[9]; modal logics provide powerful abstractions for specification purposes, but

[9] but is good for fully-automated, approximative verification

are still not expressive enough (cf. main part of this report); higher-order logics are too expressive at the cost of axiomatic and computational incompleteness[10]; finally "[f]irst-order logics seem a good compromise between expressiveness and computability, since they are [axiomatically] complete in general." [62]. Core CPL is a poly-dimensional modal (norms, knowledge, space, *qualitative* time) first-order logic (cf. main part of this report).

Temporal domain: core CPL can be instantiated with a transitive, irreflexive, linear and bounded in the past, possibly branching (but a priori flattened) and unbounded (depending on the protocol) in the future, discrete (event-induced protocol states)[11] temporal accessibility relation [51]. tCPL can be instantiated with a temporal accessibility relation that *additionally* accounts for *quantitative* time [52]. That is, time is (1) rational-number[12] valued (yielding a dense temporal grain); (2) referenced explicitly (the truth of a timed formula does not depend on its evaluation time), but implicit-time operators are macro-definable (see below); (3) measured with potentially drifting local clocks (one per protocol participant), where the (standard Dolev-Yao) adversary's local clock has drift rate 1; (4) advanced monotonically by letting the adversary choose the amount by which she desires to increase her local clock (de facto de system clock)[13]; and (5) determinant for adversarial break of short-term keys, enabled jointly by key expiration and ciphertext-only attacks (the weakest reasonable attack).

The following section describes the extension of CPL to tCPL. The extension depends on the core described in the main part of this report (the reader is urged to consult it) and parallels the extension from C^3 [51] to tC^3 [52].

C.2 Extension

tCPL is an Occham's-razor extension of core CPL. We subscribe to Lamport's claim that adding real-time to an untimed formalism is really simple [72]. The special-purpose machinery for timed (including cryptographic) settings need not be built from scratch nor be heavy-weight.

The temporal accessibility relation from [52] generates events $\mathtt{S}(a, x, t)$ for the setting of a's local clock to clock value t by a in session x. By convention, these events are unobservable by the adversary, i.e., they are secure.

1. addition of two new relational symbols $\leq$ and @ forming atomic formulae $E \leq E'$ and $E@a$ for the comparison of temporal expressions $E ::= t \mid E +$

[10] but are good as logical frameworks

[11] carrying the current process term and the history of past protocol events: "[...] neither pure state-based nor pure event-based languages quite support the natural expressiveness desirable for the specification of real-world systems [...]" [62]

[12] consider that protocol messages have finite length, which implies that real numbers (e.g., time stamps) are not transmittable as such, and that real clocks only have finite precision

[13] this amounts to a natural generalisation of the adversary's scheduling power from the control of the (relative) temporal *order* of protocol events in the network (space) to the control of their (absolute) temporal *issuing* (time)

$E \mid E - E$ where $t \in \mathcal{TV} := \mathbb{Q} \cup \{\infty, -\infty\}$ (time values with the associated sort $\mathtt{TV}$ and transmittable as messages); and the testing of participant a's local clock with time E, respectively. Their truth denotation is as follows:

$$[\![E \leq E']\!]^i_{\mathtt{p}} := ([\![E]\!] \text{ is smaller than or equal to } [\![E']\!], \emptyset)$$

where $[\![\cdot]\!]$ denotes the obvious evaluation function from temporal expressions to time values (not to be confused with the function of truth denotation $[\![\cdot]\!]^i_{\mathtt{p}}$); and

$$[\![E@a]\!]^i_{\mathtt{p}} := ([\![E]\!] = t + \delta_a \cdot \mathit{\Delta}, \{\mathtt{S}(a, x, t), \mathtt{S}(\mathtt{Eve}, \blacksquare, t_i)\})$$

where

- t denotes the time value of a's last clock-set event in $\mathfrak{h}$, i.e., there are $\mathfrak{h}_1, \mathfrak{h}_2, x$ s.t. $\mathfrak{h} = \mathfrak{h}_1 \cdot \mathtt{S}(a, x, t) \circ \mathfrak{h}_2$ and there is no x', t' s.t. $\mathtt{S}(a, x', t') \in \dot{\mathfrak{h}}_2$
- $\delta_a \in \mathcal{TV}$ denotes the drift rate of a's local clock
- $\mathit{\Delta}$ denotes the temporal difference between Eve's last clock-set event before $\mathtt{S}(a, x, t)$ and Eve's last clock-set event so far in $\mathfrak{h}$, i.e., $\mathit{\Delta} =$ $$\begin{cases} t_2 - t_1 & \text{if for } i \in \{1, 2\} \text{ there are } \mathfrak{h}_{i'}, \mathfrak{h}''_i,\ t_i \text{ s.t.} \\ & \quad \mathfrak{h}_i = \mathfrak{h}'_i \cdot \mathtt{S}(\mathtt{Eve}, \blacksquare, t_i) \circ \mathfrak{h}''_i \text{ and there is no } t'_i \text{ s.t.} \\ & \quad \mathtt{S}(\mathtt{Eve}, \blacksquare, t'_i) \in \dot{\mathfrak{h}}''_i, \text{ and} \\ 0 & \text{otherwise.} \end{cases}$$

2. refinement (i.e., one new parameter) of the relational symbol for new-name generation $\circlearrowright$ with a *validity tag* V of the form (t_b, t_e) for the declaration of the intended beginning ($t_b \in \mathcal{TV}$) and end ($t_e \in \mathcal{TV}$) of validity of the generated name (typically a key). Its truth denotation is the following:

$$[\![a.x \circlearrowright n.(\boxed{V}, O)]\!]^i_{\mathtt{p}} := (\mathcal{E} \neq \emptyset, \mathcal{E}) \quad \text{where } \mathcal{E} := \{\mathtt{N}(a, x, n, (\boxed{V}, O))\} \cap \dot{\mathfrak{h}}$$

3. refinement (i.e., adding of two new axioms) of the relation of data derivation $\vdash_a \subseteq \mathcal{H} \times \mathcal{M}$ in the semantics of the relational symbol k for individual knowledge with:
   (a) knowledge of temporal values $t \in \mathcal{TV}$: $\boxed{\overline{\mathfrak{h} \vdash_a t}}$
   (b) adversarial break of short-term keys (k) enabled jointly by *key expiration* ($\mathsf{expired}(k)$) and the existence of a *ciphertext-only attack* on the key ($\mathsf{CA}(k)$):

$$\boxed{\begin{array}{ll} \overline{\mathfrak{h} \vdash_a k} & \text{there is a prefix } \mathfrak{h}' \text{ of } \mathfrak{h}, \text{ and } t \in \mathcal{TV} \text{ s.t.} \\ & \mathfrak{h}' \models \mathsf{CA}(k) \wedge t@\mathtt{Eve} \ \text{ and} \\ & \mathfrak{h} \models \mathsf{expired}(k) \wedge \\ & \qquad \exists t_v (t_v \ \mathsf{validityOf}\ k \wedge \exists t_n (t_n@\mathtt{Eve} \wedge t_v < t_n - t)) \end{array}}$$

   where t_v denotes the duration of validity of the considered key (i.e., the strength[14] of the key), and $t_n - t$ the duration of the attack on the

[14] corresponding to its length in a bit-string representation

considered key (i.e., the time during which the corresponding ciphertext has been known to the adversary)[15]; and

$$\begin{aligned}
\mathsf{CA}(k) &:= \exists m(m : \mathtt{SC}_k[\mathtt{M}] \cup \mathtt{S}_k[\mathtt{M}] \land \mathtt{Eve}\ \mathsf{k}\ m) \\
\mathsf{expired}(k) &:= \exists t_n(t_n@\mathtt{Eve} \land \exists t_e(k\ \mathsf{validUntil}\ t_e \land t_e < t_n)) \\
k\ \mathsf{validUntil}\ t_e &:= \exists t_b(k\ \mathsf{validBetween}\ (t_b, t_e)) \\
k\ \mathsf{validBetween}\ (t_b, t_e) &:= \exists a \exists x \exists o(a.x \circlearrowright k.(o, (t_b, t_e))) \\
t_v\ \mathsf{validityOf}\ k &:= k\ \mathsf{validBetween}\ (t_b, t_e) \land t_e - t_b = t_v
\end{aligned}$$

4. refinement (i.e., one new conjunct) of the fundamental state of violation Σ with *key expiration* in the definition of the permission operator:

$$\Sigma := \exists(k : \mathtt{CK})(\mathtt{Eve}\ \mathsf{k}\ k \land \neg\, k\ \mathsf{ck}\ \mathtt{Eve} \land \boxed{\neg\mathsf{expired}(k)})$$

C.3 Expressiveness

Macro-definability of operators from general-purpose timed logics:

- *point*-parametrised temporal modalities (so-called *freeze quantifiers*):

$$\Box_t(\phi) := \boxplus(t@\mathtt{Eve} \to \phi) \qquad \Diamond_t(\phi) := \neg\Box_t(\neg\phi)$$

- *interval*-parametrised temporal modalities with an:
  - *absolute*-time understanding of (closed)[16] intervals $[t_1, t_2]$:

$$\Box^a_{[t_1,t_2]}(\phi) := \forall t(t_1 \leq t \leq t_2 \to \Box_t(\phi)) \qquad \Diamond^a_{[t_1,t_2]}(\phi) := \neg\Box^a_{[t_1,t_2]}(\neg\phi)$$

  - understanding of intervals that is *relative* to the current time $t@\mathtt{Eve}$:

$$\begin{aligned}
\Box^r_{[t_1,t_2]}(\phi) &:= \forall t(t@\mathtt{Eve} \to \Box^a_{[t+t_1,t+t_2]}(\phi)) \\
\Diamond^r_{[t_1,t_2]}(\phi) &:= \forall t(t@\mathtt{Eve} \to \neg\Box^a_{[t+t_1,t+t_2]}(\neg\phi))
\end{aligned}$$

The cryptographic states of affairs involving qualitative temporal modalities from Section 2.1 can easily be adapted to the quantitative setting by replacing the qualitative temporal modalities by the above quantitative ones with actual time values (points and/or intervals) as desired.

Two more cryptographic states of affairs involving quantitative time:

- secrecy (at some point of time) of a possibly timed, confidential key k:

$$(k : \mathtt{CK} \land \mathtt{Eve}\ \mathsf{k}\ k \land \neg\, k\ \mathsf{ck}\ \mathtt{Eve}) \to \mathsf{expired}(k)$$

- successful ciphertext-only attack (break of a good key) k:

$$\exists m(m : \mathtt{SC}_k[\mathtt{M}] \cup \mathtt{S}_k[\mathtt{M}] \land \mathtt{Eve}\ \mathsf{k}\ m \land \neg\mathsf{expired}(k) \land \mathtt{Eve}\ \mathsf{k}\ k \equiv \mathtt{Eve}\ \mathsf{k}\ m)$$

Notice that $\equiv$ is the *epistemic bi*-conditional; it says that the evidence required for Eve to know k is *exactly* (cf. 'bi-') the ciphertext m. This key break based on the knowledge of a single ciphertext (m) can easily be generalised to key break based on multiple ciphertexts ($m_1, \ldots, m_n$) by multiple existential quantification.

[15] and during which the adversary has potentially been attacking — i.e., performing computations on — the ciphertext in order to recover the desired key

[16] and similarly for open intervals

D Probabilistic Polynomial-time Cryptographic Protocol Logic

D.1 Introduction

This appendix presents an Ockham's razor extension of core CPL (cf. main part of this paper) with a notion of probabilistic polynomial-time (PP) computation. We hope to convince the reader that adding a notion of PP-computation to a (property-based) formalism for cryptography can be simple and conceived through a refinement of the Dolev-Yao conception of cryptographic operators. The special-purpose machinery for PP (as for *real* time [72], [52, 73]) need not be built from scratch nor be heavy-weight. The style of our presentation of concepts is alternative to the traditional style, which is *operational* and based on interactive (Turing) *machines*. In contrast, our style is *logical* and based on *linguistic abstractions* for cryptographic constructions.

We are aware of the following existing formalisms for the specification and verification of cryptographic constructions. Property-based: [7, 74, 75] in the tradition of (first-order) *dynamic* — more precisely, mono-dimensional (i.e., qualitative time) poly-modal (action parameters) — logic, with satisfaction and deduction relations, and originally conceived for cryptographic *protocols*; and [34] a *higher*-order logic, *deduction*-based, and originally conceived for cryptographic *operators*. Model-based: [14] a process algebra (equivalence-based specification and verification). In contrast, ppCPL is in the tradition of (first-order[17]) *temporal* — more precisely, poly-dimensional (i.e., norms, knowledge, space, qualitative and possibly quantitative time [73]) mono-modal — logic, (for the moment still) satisfaction-based, and originally conceived for cryptographic *protocols* but here extended to encompass cryptographic *operators* to some extent.

Yet CPL (a property-based formalism) goes even further in its *wholistic ambition* in that it integrates the perhaps most important model-based framework, namely process algebra [50], in a novel way. First, CPL's temporal accessibility relation (the semantics of its temporal modalities) can be defined by an event-trace generating process (reduction) calculus, e.g., C^3 [51, 52] whose execution constraints are moreover checkable via CPL-satisfaction; and second, CPL's epistemic accessibility relation (the semantics of its epistemic modality "knows that") is the definitional basis for C^3's observational process equivalence, which can be used for the model-based (process-algebraic and complementary to property-based) specification and verification of cryptographic constructions.

Symbolic logic We qualify a logic as *symbolic*[18] when its language allows quantification over individuals that are represented as *syntactic terms* formed with term constructors (i.e., functional symbols). In this sense, CPL is symbolic; its language allows quantification over individuals, i.e., protocol messages, represented as message terms.

[17] higher-order logics are too expressive at the cost of axiomatic incompleteness

[18] traditional usage of the term is philosophical and not standardised

Core CPL (cf. main part of this paper) can further be qualified as *abstract* (in the sense of Dolev-Yao) because message terms are *not* interpreted as bit-strings. In contrast, ppCPL is *concrete* (in the sense of PP-computation) because message terms *are* interpreted as bit-strings. More precisely, in ppCPL message terms are denoted to probability distributions of bit-strings by interpreting logical constants as bit-strings and functional symbols as possibly probabilistic algorithms on bit-strings.

Furthermore, core CPL can be qualified as *positive about truth and knowledge* because false positives, i.e., false statements wrongly established as true, and false belief respectively, are impossible. In contrast, ppCPL is *probabilistic about truth and knowledge* because false positives are, w.r.t. truth, possible (though only) with negligible probability, and w.r.t. knowledge, (im)probable with variable degrees of support.

Finally, we highlight that in the language of ppCPL probability is *implicit* except for belief where it parametrises the (new) doxastic modality. In particular, there is no likelihood operator (cf. [76]) in ppCPL. The reason is that in modern cryptography truth must be established with overwhelming probability, whereas belief of a human being may be established with possibly non-overwhelming degrees of support. Philosophically speaking (cf. Rudolf Carnap), probability can be an (epistemological) measure of our (subjective) belief of states of affairs as well as an (ontological) measure of their (objective) possibility. The refinement of the doxastic modality with probability allows the expression of *degrees of certitude* that an agent may experience w.r.t. her apprehension of cryptographic states of affairs.

Probability theory A fundamental concept of probability theory is the one of a *probabilistic experiment* characterised by the indeterminacy of its outcome, i.e., its *entropy*. The fact that such experiments are probabilistic implies that they are *stateful*, i.e., they represent states (with an inherent potential future) of the considered model, and their execution means probabilistic state transition. *A priori*, an experimenter, i.e., a human being about to experience the considered experiment, typically has an *uncertainty* about the *actual* outcome of the experiment, and makes a *hypothesis* about its *expected* outcome. A *posteriori*, the experimenter typically makes an *epistemic error* about the actual outcome of the experiment w.r.t. the hypothesis made a priori.

In ppCPL, exactly two kinds of probabilistic experiments are relevant, namely *process reduction*, i.e., protocol execution, (cf. Table 4) and *message denotation*, i.e., message evaluation, (cf. Table 5).

In Table 4, $\mathcal{P}$ denotes the set of process terms (protocol models) P (the potential future), $\mathcal{H}$ the set of protocol histories $\mathfrak{h}$ (a finite word of past protocol events[19] ε, e.g., message output or input and new-key/-nonce generation), $\longrightarrow$ the relation of process reduction (modelling interleaving concurrency of protocol events), and $\models$ CPL's relation of satisfaction. Note that interleaving concurrency implies that atomic events are mutually exclusive and independent within each

[19] not to be confused with the concept of atomic events from probability theory

Table 4. Probabilistic process reduction

Probability theory	CPL
sample space	$\mathcal{P} \times \mathcal{H}$
variable (experiment) X	protocol state $(P, \mathfrak{h})$
atomic event	transition $(P, \mathfrak{h}) \longrightarrow (P', \mathfrak{h}')$
possible value (outcome) of X	$(P', \mathfrak{h}')$ s.t. $(P, \mathfrak{h}) \longrightarrow (P', \mathfrak{h}')$
probability distribution $\mathbf{P}(X)$	$\{ ((P', \mathfrak{h}'), p) \mid (P, \mathfrak{h}) \longrightarrow (P', \mathfrak{h}') \text{ and } p \in]0, 1] \}$
	such that $\Sigma_{p \in \mathbf{P}((P,\mathfrak{h}))} p = 1$
hypothesis h about outcome	proposition ϕ
hypothesis H about outcome	$\{ (P', \mathfrak{h}') \mid (P, \mathfrak{h}) \longrightarrow (P', \mathfrak{h}') \models \phi \}$

branching. Applying the principle of indifference, we fix $\mathbf{P}(\mathfrak{s})$ to the uniform distribution for all $\mathfrak{s} \in \mathcal{P} \times \mathcal{H}$.

In Table 5, $\mathcal{M}$ denotes the set of message terms and $\llbracket \cdot \rrbracket$ the function of message denotation.

Table 5. Probabilistic message denotation

Probability theory	CPL
sample space	$\{\mathtt{0}, \mathtt{1}\}^*$
variable (experiment) X	message term $M \in \mathcal{M}$
atomic event	$\llbracket M \rrbracket = s$ where $\llbracket \cdot \rrbracket \subseteq \mathcal{M} \times \{\mathtt{0}, \mathtt{1}\}^*$
possible value (outcome) of X	$s \in \{\mathtt{0}, \mathtt{1}\}^*$ s.t. $s = \llbracket M \rrbracket$
probability distribution $\mathbf{P}(X)$	$\{ (s, p) \mid s = \llbracket M \rrbracket \text{ and } p \in]0, 1] \}$
	such that $\Sigma_{p \in \mathbf{P}(M)} p = 1$
hypothesis h about outcome	$\llbracket M \rrbracket = s$
hypothesis H about outcome	$\{ s \mid s = \llbracket M \rrbracket \}$

Probabilistic polynomial-time cryptography The distinguishing features of probabilistic polynomial-time cryptography are that (1) key and signature generation, and encryption are probabilistic (or randomised); (2) the execution time of the operations under Item 1 and decryption are polynomially bounded in a *security parameter* (the length of the key) used for key generation; (3) adversaries are PP Turing machines with oracle access; and (4) oracles are PP Turing machines.

D.2 Extension

This section describes the extension of CPL to ppCPL. The extension depends on the core described in the main part of this report (the reader is urged to consult it).

Syntax The syntactic novelties are the following:

1. *logical constants* (atomic message terms): refinement of the abstract message $\blacksquare_l$ with a length indication $l \in \mathbb{N}$; addition of bit-strings $s ::= \mathtt{0} \mid \mathtt{1} \mid s \bullet s$ and of probability values $q \in \mathcal{PV} := [0,1] \cap \mathbb{Q}$ with the associated sort $\mathtt{PV}$
2. *functional symbols*: refinement of hashes $\lceil M \rceil^{HA}$, symmetric $[M]^{SEA}_{M'}$ and asymmetric $[M]^{AEA}_{p^+}$ encryptions, and signatures $]M[^{SA}_{p}$ with a parameter $HA \in \{\mathtt{SHA1}, \mathtt{MD5}, \ldots\}$, $SEA \in \{\mathtt{DES}(MOO), \mathtt{AES}(MOO), \ldots\}$, $AEA \in \{\mathtt{RSA}, \mathtt{Elgamal}, \ldots\}$, resp. $SA \in \{\mathtt{RSA}, \mathtt{Elgamal}, \ldots\}$ for the name of the employed algorithm. $MOO \in \{\mathtt{ECB}, \mathtt{CBC}, \mathtt{CFB}, \mathtt{OFB}, \ldots\}$ is a parameter for the name of the employed mode of operation of a block cipher.
3. *relational symbols*:
   (a) refinement of the predicate $a.x \underset{NGA}{\overset{\iota}{\circlearrowleft}} n.O$ for new-name generation with a security parameter $\iota \in \mathbb{N}$, and a parameter $NGA ::= SEA \mid AEA \mid SA$ for the name of the employed generation algorithm
   (b) addition of a binary symbol $\leq$ for the comparison of probability values (actually the same as for the comparison of time values, cf. Appendix C)
4. *logical operators*: addition of a modality B^q_a for *belief with error control* q, where q is the probability for agent a not to err in her apprehension of the truth of the considered proposition (say ϕ), written $\mathsf{B}^q_a(\phi)$

Semantics The principal semantic novelties are the following:

1. (backwards-compatible) refinement of temporal accessibility *simpliciter* to *PP* temporal accessibility: addition of *denotation events* $\mathtt{D}(a, M, s, ALGO)$ stating that agent a denoted the message term M to the string $s \in \{\mathtt{0}, \mathtt{1}\}^*$ by application of the algorithm $ALGO ::= NGA \mid \blacksquare$, where $ALGO \in NGA$ if M is a name and $ALGO = \blacksquare$ otherwise
2. refinement of individual knowledge *simpliciter* to *PP* individual knowledge (cf. Table 6) relying on (stateful) PP message denotation:

$$\llbracket M \rrbracket^{\mathfrak{h}}_a := \begin{cases} \text{choose } s \text{ s.t. } \mathtt{D}(a, n, s, NGA) \in \dot{\mathfrak{h}} \text{ or else } n & \text{if } M = n\text{, and} \\ \text{choose } s \text{ s.t. there is } p \text{ s.t. } (s, p) \in \mathbf{P}(M') & \text{otherwise.} \end{cases}$$

   where $M' := \bigcup_{n \in \mathrm{names}(M)} \{^{\llbracket n \rrbracket^{\mathfrak{h}}_a}/_{n}\} M$ denotes the message that results from the substitution of all names n in M with the corresponding denotations $\llbracket n \rrbracket^{\mathfrak{h}}_a$. (The act of choosing s could be made strictly formal with Hilbert's choice operator [77].)
3. let

$$\mathcal{K}_a(\mathfrak{p}, i) := \{\ \mathfrak{s} \mid \mathfrak{p}@0 \longrightarrow^* \mathfrak{s} \text{ and } \mathfrak{s} \approx_a \mathfrak{p}@i\ \}$$

$$\mathcal{B}_a(\mathfrak{p}, i) := \{\ \mathfrak{s} \mid \mathfrak{p}@0 \longrightarrow^* \mathfrak{s} \text{ and } \mathfrak{s} \approx_a \mathfrak{p}@i \text{ and there is a polynomial } p : \mathbb{N} \to \mathbb{N} \text{ s.t. } |\mathfrak{s}| \leq p(|\mathfrak{p}@i|)\ \}$$

$$|(P, \mathfrak{h})| := |\dot{\mathfrak{h}}|$$

Table 6. PP derivation of individual knowledge

Random coin tossing

$$\frac{}{\mathfrak{h} \vdash_a^{(0,1)} \mathtt{0}} \qquad \frac{}{\mathfrak{h} \vdash_a^{(0,1)} \mathtt{1}}$$

Input data extraction

$$\frac{}{\mathfrak{h} \cdot \varepsilon(a, \overline{M}) \vdash_a^{(|[\![(a,\overline{M})]\!]_a^{\mathfrak{h}}|,0)} (a, \overline{M})} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)} M}{\mathfrak{h} \cdot \varepsilon \vdash_a^{(i,r)} M}$$

Data synthesis ***Data analysis***

$$\frac{\mathfrak{h} \vdash_a^{(i,r)} M \quad \mathfrak{h} \vdash_a^{(i',r')} M'}{\mathfrak{h} \vdash_a^{(i+i',r+r')} (M, M')} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)} (M, M')}{\mathfrak{h} \vdash_a^{(i,r)} M} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)} (M, M')}{\mathfrak{h} \vdash_a^{(i,r)} M'}$$

$$\frac{\mathfrak{h} \vdash_a^{(i,r)} p}{\mathfrak{h} \vdash_a^{(i,r)} p^+} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)} M}{\mathfrak{h} \vdash_a^{(i,r)} \lceil M \rceil}$$

$$\frac{\mathfrak{h} \vdash_a^{(i,r)} M \quad \mathfrak{h} \vdash_a^{(i',r')} M'}{\mathfrak{h} \vdash_a^{(i+i',r+r')} [M]_{M'}} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)} [M]_{M'} \quad \mathfrak{h} \vdash_a^{(i',r')} M'}{\mathfrak{h} \vdash_a^{(i+i',r+r')} M}$$

$$\frac{\mathfrak{h} \vdash_a^{(i,r)} M \quad \mathfrak{h} \vdash_a^{(i',r')} p^+}{\mathfrak{h} \vdash_a^{(i+i',r+r')} [M]_{p^+}} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)} [M]_{p^+} \quad \mathfrak{h} \vdash_a^{(i',r')} p}{\mathfrak{h} \vdash_a^{(i+i',r+r')} M}$$

$$\frac{\mathfrak{h} \vdash_a^{(i,r)} M \quad \mathfrak{h} \vdash_a^{(i',r')} p}{\mathfrak{h} \vdash_a^{(i+i',r+r')}]M[_p} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)}]M[_p \quad \mathfrak{h} \vdash_a^{(i',r')} p^+}{\mathfrak{h} \vdash_a^{(i+i',r+r')} M}$$

Data concretisation ***Data abstraction***

$$\frac{\mathfrak{h} \vdash_a^{(i,r)} M}{\mathfrak{h} \vdash_a^{(i,r)} s} \; s = [\![M]\!]_a^{\mathfrak{h}} \qquad \frac{\mathfrak{h} \vdash_a^{(i,r)} s}{\mathfrak{h} \vdash_a^{(i,r)} M} \; [\![M]\!]_a^{\mathfrak{h}} = s$$

PP-abstraction

$$\frac{\mathfrak{h} \vdash_a^{(i,r)} M}{\mathfrak{h} \vdash_a M} \text{ there is a polynomial } p : \mathbb{N} \to \mathbb{N} \text{ s.t. } r \leq p(i)$$

(a) refinement of propositional knowledge *simpliciter*

$$\begin{aligned}[\![\mathsf{K}_a(\phi)]\!]^i_{\mathfrak{p}} &:= (\text{for all } \mathfrak{s}, \text{ if } \mathfrak{s} \in \mathcal{K}_a(\mathfrak{p}, i) \text{ then } \mathfrak{s}' \models_{\mathcal{E}'} \phi',\ \mathcal{E}'_{(\mathfrak{s},\phi)})\\ &\quad \text{where } (\mathfrak{s}', \phi') := \begin{cases} (\mathfrak{s}, \phi) & \text{if } \mathfrak{s} = \mathfrak{p}@i, \text{ and}\\ ((\!|\,\mathfrak{s}\,|\!)^{\mathfrak{p}@i}_a, (\!|\,\phi\,|\!)^{\mathfrak{p}@i}_a) & \text{otherwise.}\end{cases}\end{aligned}$$

to *PP* propositional knowledge

$$\begin{aligned}[\![\mathsf{K}_a(\phi)]\!]^i_{\mathfrak{p}} &:= (\text{for all } \mathfrak{s}, \text{ if } \ \mathfrak{s} \in \mathcal{K}_a(\mathfrak{p}, i)\\ &\qquad \text{then } \ \mathfrak{s}' \models_{\mathcal{E}'} \phi' \text{ and there is a polynomial}\\ &\qquad\quad p : \mathbb{N} \to \mathbb{N} \text{ s.t. } |\mathfrak{s}| \leq p(|\mathfrak{p}@i|),\ \mathcal{E}'_{(\mathfrak{s},\phi)})\\ &\quad \text{where } (\mathfrak{s}', \phi') := \begin{cases} (\mathfrak{s}, \phi) & \text{if } \mathfrak{s} = \mathfrak{p}@i, \text{ and}\\ ((\!|\,\mathfrak{s}\,|\!)^{\mathfrak{p}@i}_a, (\!|\,\phi\,|\!)^{\mathfrak{p}@i}_a) & \text{otherwise.}\end{cases}\end{aligned}$$

(b) addition of *believe with error control* $q \in \mathcal{PV}$

$$\begin{aligned}[\![\mathsf{B}^q_a(\phi)]\!]^i_{\mathfrak{p}} &:= (q = \tfrac{|\mathcal{B}_a(\mathfrak{p},i)|}{|\mathcal{K}_a(\mathfrak{p},i)|} \text{ and}\\ &\quad \text{for all } \mathfrak{s}, \text{ if } \mathfrak{s} \in \mathcal{B}_a(\mathfrak{p}, i) \text{ then } \mathfrak{s}' \models_{\mathcal{E}'} \phi',\ \mathcal{E}'_{(\mathfrak{s},\phi)})\\ &\quad \text{where } (\mathfrak{s}', \phi') := \begin{cases} (\mathfrak{s}, \phi) & \text{if } \mathfrak{s} = \mathfrak{p}@i, \text{ and}\\ ((\!|\,\mathfrak{s}\,|\!)^{\mathfrak{p}@i}_a, (\!|\,\phi\,|\!)^{\mathfrak{p}@i}_a) & \text{otherwise.}\end{cases}\end{aligned}$$

D.3 Case studies

We illustrate the expressiveness of ppCPL on tentative formalisation case studies of fundamental and applied concepts. *Fundamental concepts*: (1) one-way function, (2) hard-core predicate, (3) computational indistinguishability, (4) (n-party) interactive proof, and (5) (n-prover) zero-knowledge. *Applied concepts*: (1) security of encryption schemes, (2) unforgeability of signature schemes, (3) attacks on encryption schemes, (4) attacks on signature schemes, and (5) breaks of signature schemes.

Note that in core CPL we focused on cryptographic *protocols* and their *requirements*, but here (in ppCPL) we focus on cryptographic *operators* and their *attacks* and *breaks*. Naturally, properties of operators take the form of tautologies, i.e., propositions that hold in any (protocol) model. Our formalisations illustrate the dramatic expressive power that results from the combined use of epistemic and spatial operators.

Fundamental concepts This section is in the spirit of [4].

Definition 8 (Random propositional guessing).

$$\mathsf{G}_a(\phi) := \neg\exists(q : \mathtt{PV})(\frac{1}{2} < q \wedge \mathsf{B}^q_a(\phi))$$

Definition 9 (Hard proposition). *A proposition ϕ is* hard *:iff in any model, any agent can only guess the truth of ϕ. That is, $\mathsf{G}_a(\phi)$ is a tautology.*

$$\models \mathsf{G}_a(\phi)$$

We generalise the concept of a hard proposition (a closed formula) to the concept of a hard predicate (an open formula).

Definition 10 (Hard predicate). *An n-ary predicate $\phi(M_1, \ldots, M_n)$ is* hard *on $M_1, \ldots, M_n$ satisfying the ($n+1$-ary) predicate $\varphi(M_1, \ldots, M_n, a)$:iff in any model, if $\varphi(M_1, \ldots, M_n, a)$ then any a can only randomly guess the truth of $\phi(M_1, \ldots, M_n)$.*

$$\models \varphi(M_1, \ldots, M_n, a) \to \mathsf{G}_a(\phi(M_1, \ldots, M_n))$$

Formalisation 1 (One-way function)

"[...] a function that is easy to compute but hard to invert." [4, Page 32]

Ease of computation $\models a \mathrel{\mathsf{k}} M \to a \mathrel{\mathsf{k}} f(M)$

Hardness of invertibility *$f^{-1}(M') = M$ is hard on M and M' satisfying $f(M) = M' \land \neg\, a \mathrel{\mathsf{k}} M$.*

⌟

Notice that the standard definition of one-way functions only requires the operator f to be computable in *deterministic* polynomial-time, whereas the satisfiability of $a \mathrel{\mathsf{k}} f(M)$ may be computable only in *probabilistic* polynomial-time (cf. Table 6). We could easily provide a deterministic variant of k by simply disallowing random coin tossing. Further, observe that our definition implies that cryptographic operators are common knowledge among agents. In order to express (individual) knowledge of operators, we would need quantification over functional symbols, which would make our logic higher-order.

Formalisation 2 (Hard-core predicate)

"[...] a polynomial-time *predicate b is called a hard-core of a function f if every efficient algorithm, given $f(x)$, can guess $b(x)$ with success probability that is only negligibly better than one-half." [4, Page 64]*

Let $\phi(M)$ be computable in polynomial-time. Then $\phi(M)$ is a hard-core *of a function f :iff $\phi(M)$ is hard on M satisfying $a \mathrel{\mathsf{k}} f(M) \land \neg\, a \mathrel{\mathsf{k}} M$.*

⌟

Notice that we identify agents with feasible algorithms!

Formalisation 3 (Computational indistinguishability)

"Objects are considered to be computationally equivalent if they cannot be differentiated by any efficient procedure." [4, Page 103]

M and M' are computationally indistinguishable *:iff $\neg(M = M')$ is hard on M and M' satisfying $\neg(M = M')$.* $\lrcorner$

Definition 11 (Individual evidence and proof).

$$
\begin{aligned}
M \;\mathsf{necessaryEvidenceFor}\; \phi &:= \forall a(\mathsf{K}_a(\phi) \rhd (\mathsf{K}_a(\phi) \supseteq a \;\mathsf{k}\; M))\\
M \;\mathsf{sufficientEvidenceFor}\; \phi &:= \forall a(\mathsf{K}_a(\phi) \rhd (a \;\mathsf{k}\; M \supseteq \mathsf{K}_a(\phi)))\\
M \;\mathsf{strictEvidenceFor}\; \phi &:= M \;\mathsf{necessaryEvidenceFor}\; \phi \,\wedge\\
& M \;\mathsf{sufficientEvidenceFor}\; \phi\\
M \;\mathsf{evidenceFor}\; \phi &:= M \;\mathsf{necessaryEvidenceFor}\; \phi \,\vee\\
& M \;\mathsf{sufficientEvidenceFor}\; \phi\\
M \;\mathsf{necessaryProofFor}\; \phi &:= \forall a(a \;\mathsf{k}\; M \rhd (\mathsf{K}_a(\phi) \supseteq a \;\mathsf{k}\; M))\\
M \;\mathsf{sufficientProofFor}\; \phi &:= \forall a(a \;\mathsf{k}\; M \rhd (a \;\mathsf{k}\; M \supseteq \mathsf{K}_a(\phi)))\\
M \;\mathsf{strictProofFor}\; \phi &:= M \;\mathsf{necessaryProofFor}\; \phi \,\wedge\\
& M \;\mathsf{sufficientProofFor}\; \phi\\
M \;\mathsf{proofFor}\; \phi &:= M \;\mathsf{necessaryProofFor}\; \phi \,\vee\\
& M \;\mathsf{sufficientProofFor}\; \phi
\end{aligned}
$$

Conjecture 1.

1. $\models M \;\mathsf{necessaryProofFor}\; \phi \rightarrow M \;\mathsf{necessaryEvidenceFor}\; \phi$
2. $\models M \;\mathsf{sufficientProofFor}\; \phi \rightarrow M \;\mathsf{sufficientEvidenceFor}\; \phi$
3. $\models M \;\mathsf{strictProofFor}\; \phi \leftrightarrow M \;\mathsf{strictEvidenceFor}\; \phi$

Formalisation 4 (2-party interactive proof)

"A 2-party interactive proof *(or* 2-party computation *or* 2-party protocol*) M between agents a and b (initiated by a) for a proposition ϕ (protocol goal) is a (possibly minimal) finite chain $M = (M_0, \ldots, M_n)$ of messages s.t. (1) M_n is a proof of ϕ for a, and (2) for all consecutive pairs (M_i, M_j) in M, M_j derives from M_i due to communication between a and b." [author's formulation]*

$$
\begin{aligned}
\overline{M} &::= (M, \blacksquare) \;\big|\; (M, \overline{M})\\
I &::= \blacksquare \;\big|\; \overline{M}
\end{aligned}
$$

$$
\begin{aligned}
M' \;\mathsf{commConsOf}_{(a,b)}\; M \;&:=\; a \;\mathsf{k}\; M \wedge (b \;\mathsf{k}\; M' \supseteq a \;\mathsf{k}\; M)\\
M \;\mathsf{iProofFor}_{(a,b)}\; \phi \;&:=\; M \;\mathsf{iProofFor}^{a}_{(a,b)}\; \phi\\
(M, \blacksquare) \;\mathsf{iProofFor}^{c}_{(a,b)}\; \phi \;&:=\; c \;\mathsf{k}\; M \wedge M \;\mathsf{proofFor}\; \phi\\
(M, (M', I)) \;\mathsf{iProofFor}^{c}_{(a,b)}\; \phi \;&:=\; M' \;\mathsf{commConsOf}_{(a,b)}\; M \,\wedge\\
& (M', I) \;\mathsf{iProofFor}^{c}_{(b,a)}\; \phi
\end{aligned}
$$

$\lrcorner$

Conjecture 2 (Characteristics of interactive proofs). For "certain" ϕ,

Soundness $\models \neg\phi \rightarrow \neg\exists m(m \ \mathsf{iProofFor}_{(a,b)}\ \phi)$
Completeness $\models \phi \rightarrow \exists m(m \ \mathsf{iProofFor}_{(a,b)}\ \phi)$

Definition 12 (Interactive propositional knowledge).

$$\mathsf{IK}_{(a,b)}(\phi) := \exists m(m \ \mathsf{iProofFor}_{(a,b)}\ (\phi))$$

Formalisation 5 (Proof of knowledge)

"[...] [interactive] proofs in which the prover asserts "knowledge" of some object [...] and not merely its existence [...]" [4, Page 262]

$$a \ \mathsf{pok}_b \ M := \mathsf{IK}_{(a,b)}(b \ \mathsf{k} \ M)$$

⌟

Formalisation 6 (Zero-Knowledge)

"Zero-knowledge proofs are defined as those [interactive] proofs that convey no additional knowledge other than the correctness of the proposition [ϕ] in question." [78]

$$\begin{aligned}\mathsf{ZK}_{(a,b)}(\phi) := \mathsf{IK}_{(a,b)}(&\mathsf{K}_a(\exists m'(\mathsf{K}_b(m' \ \mathsf{proofFor}\ \phi))) \wedge \\ &\neg\exists m''(\mathsf{K}_a(\mathsf{K}_b(m'' \ \mathsf{evidenceFor}\ \phi))))\end{aligned}$$

⌟

Spelled out, a (the verifier) knows through interaction with b (the prover) that b knows a proof (m') for the proposition ϕ, however a does not know that proof nor any evidence (m'') that could corroborate the truth of ϕ. Observe the importance of the scope of the existential quantifiers. Philosophically speaking, a has *pure propositional* knowledge of ϕ, i.e., a has *zero individual* knowledge *relevant* to the truth of ϕ. In Goldreich's words, it is "as if [the verifier] was told by a trusted party that the assertion holds" [79, Page 39].

Standard zero-knowledge, i.e., zero-knowledge w.r.t. a malicious verifier is an instance of the above scheme where $a = \mathtt{Eve}$. Zero-knowledge w.r.t. an honest verifier is definable as $\mathsf{ZK}_{(a,b)}(\phi) \wedge \mathsf{honest}(a)$.

Conjecture 3. "[A]nything that is feasibly computable from a zero-knowledge proof is also feasibly computable from the (valid) assertion itself." [79, Page 39]

$$\models \phi \rightarrow ((\mathsf{K}_a(\varphi) \supseteq \mathsf{ZK}_{(a,b)}(\phi)) \rightarrow (\mathsf{K}_a(\varphi) \supseteq \phi))$$

This is a logical formulation of an instance of the *simulation paradigm* [80].

Formalisation 7 (n-party interactive proof)

*"An n-*party interactive proof *(or n-*party computation *or n-*party protocol*) M between agents $\{a_0, a_1, \ldots, a_{n-1}\}$ (initiated by a_0) for a proposition ϕ (protocol goal) is a (possibly minimal) finite chain $M = ((a_0, M_0), \ldots, (a_l, M_m))$ s.t. (1) M_m is a proof of ϕ for a_0, and (2) for all consecutive pairs $((a_i, M_i), (a_j, M_j))$ in M, M_j derives from M_i due to communication between a_i and a_j." [author's formulation]*

$$A ::= (a, \blacksquare) \bigm| (b, A)$$

$$\begin{aligned}
M \ \mathsf{iProofFor}_{(a,A)} \ \phi \ &:= \ M \ \mathsf{iProofFor}^{a}_{(a,A)} \ \phi \\
(M, \blacksquare) \ \mathsf{iProofFor}^{c}_{(a,\blacksquare)} \ \phi \ &:= \ c \ \mathsf{k} \ M \land M \ \mathsf{proofFor} \ \phi \\
(M, (M', I)) \ \mathsf{iProofFor}^{c}_{(a,(b,A))} \ \phi \ &:= \ M' \ \mathsf{commConsOf}_{(a,b)} \ M \ \land \\
& (M', I) \ \mathsf{iProofFor}^{c}_{(b,A)} \ \phi
\end{aligned}$$

$\lrcorner$

Definition 13 (Quotient proof).

$$\begin{aligned}
M \mid_a M' &:= \neg(a \ \mathsf{k} \ M \supseteq a \ \mathsf{k} \ M') \ \land \\
& \neg(a \ \mathsf{k} \ M' \supseteq a \ \mathsf{k} \ M) \\
(M, M') \ \mathsf{disjointEvidenceFor} \ \phi &:= \forall a(\mathsf{K}_a(\phi) \rhd (\mathsf{K}_a(\phi) \supseteq a \ \mathsf{k} \ (M, M') \ \land \\
&\phantom{:= \forall a(\mathsf{K}_a(\phi) \rhd (\mathsf{K}_a(\phi) \supseteq a \ \mathsf{k}} M \mid_a M'))
\end{aligned}$$

$$\begin{aligned}
(M, \blacksquare) \ \mathsf{mutuallyDisjointEvidenceFor} \ \phi &:= M \ \mathsf{evidenceFor} \ \phi \\
(M, \overline{M}) \ \mathsf{mutuallyDisjointEvidenceFor} \ \phi &:= (M, \overline{M}) \ \mathsf{disjointEvidenceFor} \ \phi \ \land \\
& \overline{M} \ \mathsf{mutuallyDisjointEvidenceFor} \ \phi
\end{aligned}$$

$$\begin{aligned}
M \ \mathsf{quotientProofFor} \ \phi &:= M \ \mathsf{proofFor} \ \phi \ \land \\
& M \ \mathsf{mutuallyDisjointEvidenceFor} \ \phi
\end{aligned}$$

Note that $M \mid_a M'$ is pronounced "M is (epistemically) independent from M' w.r.t. to a's knowledge". Quotient proofs could also be called *compositional* proofs.

Definition 14 (Multi-prover Zero-Knowledge).

$$\begin{aligned}
\mathsf{ZK}_{(a,A)}(\phi) := \mathsf{IK}_{(a,A)}(&\mathsf{K}_a(\exists m'(m' \ \mathsf{quotientProofFor} \ \phi \land A \ \mathsf{k} \ m')) \ \land \\
&\neg\exists m'(\mathsf{K}_a(m' \ \mathsf{evidenceFor} \ \phi \land A \ \mathsf{k} \ m')))
\end{aligned}$$

where

$$\begin{aligned}
(a, \blacksquare) \ \mathsf{k} \ (M, \blacksquare) &:= a \ \mathsf{k} \ M \\
(b, A) \ \mathsf{k} \ (M, \overline{M}) &:= b \ \mathsf{k} \ M \land A \ \mathsf{k} \ \overline{M}
\end{aligned}$$

Observe again the importance of the scope of the existential quantifiers.

Applied concepts This section is in the spirit of [5] and [3]. Note that for clarity, names of cryptographic algorithms are omitted in message terms in the sequel.

Definition 15 (Security of encryption schemes).

1. *"Standard* security*: the infeasibility of* obtaining information regarding the plaintext*" [5, Page 470]*

Semantic security *"[...] given any a priori information about the plaintext, it is infeasible to obtain any (new) information about the plaintext from the ciphertext (beyond what is feasible to obtain from the a priori information on the plaintext)." [5, Page 378]*

$$\models \underbrace{(a \mathrel{\mathsf{k}} C \wedge \mathsf{K}_a(\phi(M)))}_{\textit{a priori information}} \rightarrow (\underbrace{(\mathsf{K}_a(\varphi(M)) \supseteq a \mathrel{\mathsf{k}} C)}_{\textit{obtaining information}} \rightarrow \underbrace{(\phi(M) \supseteq \varphi(M))}_{\textit{no news}})$$

where $C ::= [M]_k \mid [M]_{p^+}$
This is again a logical formulation of an instance of the simulation paradigm *[80]. Observe the similarity with the previous instance.*

Indistinguishability of encryptions

- $[M]_k$ *(or* $[M]_{p^+}$*) and* $[M']_k$ *(or* $[M']_{p^+}$*) are computationally indistinguishable (in the sense of our formalisation)*
- *there is* $l \in \mathbb{N}$ *s.t.* $[M]_k$ *(or* $[M]_{p^+}$*) and* $\blacksquare_l$ *are computationally indistinguishable (in the sense of our formalisation)*

2. **Non-malleability** *"[...] it [is] infeasible for an adversary, given a ciphertext, to produce a valid ciphertext (under the same encryption-key) for a related plaintext." [5, Page 470]*

$$\models (\mathsf{Eve} \mathrel{\mathsf{k}} [M]_k \wedge \underbrace{\phi(M) \wedge \phi(M')}_{M' \textit{ is related to } M}) \rightarrow (\mathsf{Eve} \mathrel{\mathsf{k}} [M']_k \rightarrow \mathsf{Eve} \mathrel{\mathsf{k}} k)$$

$$\models (\mathsf{Eve} \mathrel{\mathsf{k}} [M]_{p^+} \wedge \phi(M) \wedge \phi(M')) \rightarrow (\mathsf{Eve} \mathrel{\mathsf{k}} [M']_{p^+} \rightarrow \mathsf{Eve} \mathrel{\mathsf{k}} M')$$

Formalisation 8 (Unforgeability of signature schemes)
"it is infeasible to produce signatures of other users *to documents they did not sign." [5, Page 498]*

$$\models a \mathrel{\mathsf{authored}}]M[_p \rightarrow a \mathrel{\mathsf{k}} p$$

⌟

Attacks on encryption schemes We state formalisations of attacks on encryption schemes as vulnerabilities and in increasing strength.

Formalisation 9 (Ciphertext-only attack)
"[...] the adversary (or cryptanalyst) tries [⊇] to deduce the decryption key [k (symmetric) resp. p (private)] or plaintext [M] by only [≡] observing ciphertext [$m_1, \ldots, m_n$*]." [3, Page 41]*

$$\mathsf{Eve} \mathrel{\mathsf{k}} M \equiv \exists(k : \mathtt{K})(\mathsf{Eve} \mathrel{\mathsf{k}} [M]_k)$$
$$\mathsf{Eve} \mathrel{\mathsf{k}} M \equiv \exists(p : \mathtt{K}^-)(\mathsf{Eve} \mathrel{\mathsf{k}} [M]_{p^+})$$

$$\mathsf{Eve} \mathrel{\mathsf{k}} k \equiv (\exists(m_1 : \mathtt{SC}_k[\mathtt{M}])(\mathsf{Eve} \mathrel{\mathsf{k}} m_1) \wedge \cdots \wedge \exists(m_n : \mathtt{SC}_k[\mathtt{M}])(\mathsf{Eve} \mathrel{\mathsf{k}} m_n))$$
$$\mathsf{Eve} \mathrel{\mathsf{k}} p \equiv (\exists(m_1 : \mathtt{AC}_{p^+}[\mathtt{M}])(\mathsf{Eve} \mathrel{\mathsf{k}} m_1) \wedge \cdots \wedge \exists(m_n : \mathtt{AC}_{p^+}[\mathtt{M}])(\mathsf{Eve} \mathrel{\mathsf{k}} m_n))$$

⌟

Formalisation 10 (Known-plaintext attack)

"[. . .] the adversary has a quantity of plaintext [$m_1, \ldots, m_n$] and corresponding ciphertext." [3, Page 41]

$$\begin{aligned}\mathtt{Eve}\ \mathsf{k}\ k \equiv{}& (\exists m_1(\mathtt{Eve}\ \mathsf{k}\ m_1 \wedge \mathtt{Eve}\ \mathsf{k}\ [m_1]_k) \wedge \cdots \wedge \\ & \exists m_n(\mathtt{Eve}\ \mathsf{k}\ m_n \wedge \mathtt{Eve}\ \mathsf{k}\ [m_n]_k)) \\ \mathtt{Eve}\ \mathsf{k}\ p \equiv{}& (\exists m_1(\mathtt{Eve}\ \mathsf{k}\ m_1 \wedge \mathtt{Eve}\ \mathsf{k}\ [m_1]_{p^+}) \wedge \cdots \wedge \\ & \exists m_n(\mathtt{Eve}\ \mathsf{k}\ m_n \wedge \mathtt{Eve}\ \mathsf{k}\ [m_n]_{p^+}))\end{aligned}$$

Our interpretation of 'a quantity of plaintext' can be refined from 'a number of plaintexts' to 'a number of parts of *plaintexts' by replacing* $\mathtt{Eve}\ \mathsf{k}\ m_1$ *with* $\exists m_{11}(m_{11} \preccurlyeq m_1 \wedge \mathtt{Eve}\ \mathsf{k}\ m_{11}) \wedge \cdots \wedge \exists m_{1i}(m_{1i} \preccurlyeq m_1 \wedge \mathtt{Eve}\ \mathsf{k}\ m_{1i})$*, and* $\mathtt{Eve}\ \mathsf{k}\ m_n$ *with* $\exists m_{n1}(m_{n1} \preccurlyeq m_1 \wedge \mathtt{Eve}\ \mathsf{k}\ m_{n1}) \wedge \cdots \wedge \exists m_{nj}(m_{nj} \preccurlyeq m_1 \wedge \mathtt{Eve}\ \mathsf{k}\ m_{nj})$*.* ⌟

Formalisation 11 (Chosen-plaintext attack)

"[. . .] the adversary chooses [$\triangleright$] plaintext [m] and is then given [$\blacktriangleright$] corresponding ciphertext. Subsequently [$\diamondplus$], the adversary uses any information deduced [$\supseteq$] in order to recover plaintext [M] corresponding to previously unseen ciphertext." [3, Page 41]

$$\begin{aligned}&\exists(k : \mathtt{K})(\neg\,\mathtt{Eve}\ \mathsf{k}\ [M]_k\ \wedge \\ &\qquad \exists m(\mathtt{Eve}\ \mathsf{k}\ m\ \ \triangleright \\ &\qquad\quad (\mathtt{Eve}\ \mathsf{k}\ [m]_k \blacktriangleright \diamondplus(\mathtt{Eve}\ \mathsf{k}\ M \supseteq (\mathtt{Eve}\ \mathsf{k}\ m \wedge \mathtt{Eve}\ \mathsf{k}\ [m]_k)))))\end{aligned}$$

$$\begin{aligned}&\exists(p : \mathtt{K}^-)(\neg\,\mathtt{Eve}\ \mathsf{k}\ [M]_{p^+}\ \wedge \\ &\qquad \exists m(\mathtt{Eve}\ \mathsf{k}\ m\ \ \triangleright \\ &\qquad\quad (\mathtt{Eve}\ \mathsf{k}\ [m]_{p^+} \blacktriangleright \diamondplus(\mathtt{Eve}\ \mathsf{k}\ M \supseteq (\mathtt{Eve}\ \mathsf{k}\ m \wedge \mathtt{Eve}\ \mathsf{k}\ [m]_{p^+})))))\end{aligned}$$

Our interpretation of 'plaintext' can be refined from 'the *plaintext' to* 'some of *the plaintext' by replacing* $\mathtt{Eve}\ \mathsf{k}\ M$ *with* $\exists m_1(m_1 \preccurlyeq M \wedge \mathtt{Eve}\ \mathsf{k}\ m_1) \wedge \cdots \wedge \exists m_n(m_n \preccurlyeq M \wedge \mathtt{Eve}\ \mathsf{k}\ m_n)$*.* ⌟

Formalisation 12 (Adaptive chosen-plaintext attack)

"[. . .] a chosen-plaintext attack wherein the choice of plaintext may depend on the ciphertext received from previous requests." [3, Page 41]

Let A denote chosen-plaintext chains in the public key p^+ of the form

$$A ::= \blacksquare \mid ((M, [M]_{p^+}), A)$$

and $\mathsf{aCPC}_a^{p^+}$ *an inductively-defined macro expressing the realisation of such a chain for agent a*

$$\begin{aligned}\blacksquare\ \mathsf{aCPC}_a^{p^+}\ \phi &:= \phi \\ ((M, [M]_{p^+}), A)\ \mathsf{aCPC}_a^{p^+}\ \phi &:= a\ \mathsf{k}\ M \triangleright (a\ \mathsf{k}\ [M]_{p^+} \blacktriangleright A\ \mathsf{aCPC}_a^{p^+}\ \phi)\end{aligned}$$

Then

$$\begin{aligned}&\exists(p:\mathtt{K}^-)(\neg\,\mathsf{Eve}\ \mathsf{k}\ [M]_{p^+}\ \wedge\\&\qquad\exists m(\exists m'(m'\ \mathsf{aCPC}^{p^+}_{\mathtt{Eve}}\ \mathsf{Eve}\ \mathsf{k}\ m)\ \rhd\\&\qquad\quad(\mathsf{Eve}\ \mathsf{k}\ [m]_{p^+}\blacktriangleright\diamondplus(\mathsf{Eve}\ \mathsf{k}\ M\supseteq(\mathsf{Eve}\ \mathsf{k}\ m\wedge\mathsf{Eve}\ \mathsf{k}\ [m]_{p^+})))))\end{aligned}$$

formalises an adaptive chosen-plaintext attack on a private key. (The formalisation of a corresponding attack on a symmetric key is similar.) $\lrcorner$

Formalisation 13 (Chosen-ciphertext attack)
"[...] the adversary selects the ciphertext and is then given the corresponding plaintext." [3, Page 41]

$$\begin{aligned}&\exists(k:\mathtt{K})(\neg\,\mathsf{Eve}\ \mathsf{k}\ [M]_k\ \wedge\\&\qquad\exists m(\mathsf{Eve}\ \mathsf{k}\ [m]_k\ \rhd\\&\qquad\quad(\mathsf{Eve}\ \mathsf{k}\ m\blacktriangleright\diamondplus(\mathsf{Eve}\ \mathsf{k}\ M\supseteq(\mathsf{Eve}\ \mathsf{k}\ m\wedge\mathsf{Eve}\ \mathsf{k}\ [m]_k)))))\end{aligned}$$

$\lrcorner$

Formalisation 14 (Adaptive chosen-ciphertext attack)
"[...] a chosen-ciphertext attack where the choice of ciphertext may depend on the plaintext received from previous requests." [3, Page 42]

Let S denote chosen-ciphertext chains in the symmetric key k of the form

$$S ::= ■ \mid (([M]_k, M), S)$$

and sCCC^k_a an inductively-defined macro expressing the realisation of such a chain for agent a

$$\begin{aligned}■\ \mathsf{sCCC}^k_a\ \phi &:= \phi\\(([M]_k, M), S)\ \mathsf{sCCC}^k_a\ \phi &:= a\ \mathsf{k}\ [M]_k \rhd (a\ \mathsf{k}\ M\blacktriangleright S\ \mathsf{sCCC}^k_a\ \phi)\end{aligned}$$

Then

$$\begin{aligned}&\exists(k:\mathtt{K})(\neg\,\mathsf{Eve}\ \mathsf{k}\ [M]_k\ \wedge\\&\qquad\exists m(\exists m'(m'\ \mathsf{sCCC}^k_{\mathtt{Eve}}\ \mathsf{Eve}\ \mathsf{k}\ m)\ \rhd\\&\qquad\quad(\mathsf{Eve}\ \mathsf{k}\ [m]_k\blacktriangleright\diamondplus(\mathsf{Eve}\ \mathsf{k}\ M\supseteq(\mathsf{Eve}\ \mathsf{k}\ m\wedge\mathsf{Eve}\ \mathsf{k}\ [m]_k)))))\end{aligned}$$

formalises an adaptive chosen-plaintext attack on a symmetric key. (The formalisation of a corresponding attack on an asymmetric key is similar.) $\lrcorner$

Attacks on signature schemes We state formalisations of attacks on signature schemes in increasing strength.

Formalisation 15 (Key-only attack)
"[...] an adversary knows only the signer's public key." [3, Page 432]

$$\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \land \mathtt{Eve}\ \mathsf{k}\ v^{+} \land \forall m(\mathtt{Eve}\ \mathsf{k}\ ]m[_{v} \to \neg\, \mathtt{Eve}\ \mathsf{k}\ m))$$

$\lrcorner$

Formalisation 16 (Known-message attack)

"An adversary has signatures for a set of messages which are known to the adversary but not chosen by him." [3, Page 432]

$$\begin{aligned}\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \land \mathtt{Eve}\ \mathsf{k}\ v^{+}\ \land&\\ \exists m_1 \cdots \exists m_n(\mathtt{Eve}\ \mathsf{k}\ ]m_1[_{v} \land \cdots \land \mathtt{Eve}\ \mathsf{k}\ ]m_n[_{v}\ \land&\\ \mathtt{Eve}\ \mathsf{k}\ m_1 \land \cdots \land \mathtt{Eve}\ \mathsf{k}\ m_n))&\end{aligned}$$

$\lrcorner$

Formalisation 17 (Chosen-message attack)

"An adversary obtains valid signatures from a chosen list of messages before attempting to break the signature scheme." [3, Page 433]

$$\begin{aligned}\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \land \mathtt{Eve}\ \mathsf{k}\ v^{+}\ \land&\\ \exists m_1 \cdots \exists m_n((\mathtt{Eve}\ \mathsf{k}\ m_1 \land \cdots \land \mathtt{Eve}\ \mathsf{k}\ m_n) \rhd&\\ (\mathtt{Eve}\ \mathsf{k}\ ]m_1[_{v} \land \cdots \land \mathtt{Eve}\ \mathsf{k}\ ]m_n[_{v})))&\end{aligned}$$

$\lrcorner$

Formalisation 18 (Adaptive chosen-message attack)

"An adversary is allowed to use the signer as an oracle; the adversary may request signatures of messages which depend on the signer's public key and he may request signatures of messages which depend on previously obtained signatures or messages." [3, Page 433]

Let C denote chosen-message chains in the private key p of the form

$$C ::= \blacksquare \;\big|\; ((M,]M[_{p}), C)$$

and CMC_{a}^{p} an inductively-defined macro expressing the realisation of such a chain for agent a

$$\begin{aligned}\blacksquare\ \mathsf{CMC}_{a}^{p}\ \phi &:= \phi\\ ((M,]M[_{p}), C)\ \mathsf{CMC}_{a}^{p}\ \phi &:= a\ \mathsf{k}\ M \rhd (a\ \mathsf{k}\ ]M[_{p} \blacktriangleright C\ \mathsf{CMC}_{a}^{p}\ \phi)\end{aligned}$$

Then

$$\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \land \mathtt{Eve}\ \mathsf{k}\ v^{+} \land \exists m(m\ \mathsf{CMC}_{a}^{v}\ \mathtt{Eve}\ \mathsf{k}\ m))$$

formalises an adaptive chosen-message attack on a signing key. $\lrcorner$

Breaks of signature schemes We state formalisations of breaks of signature schemes in increasing strength.

Formalisation 19 (Existential forgery)

"An adversary is able to forge a signature for at least one message. The adversary has little or no control over the message whose signature is obtained, and the legitimate signer may be involved in the deception ..." [3, Page 432]

$$\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \wedge \mathtt{Eve}\ \mathsf{k}\ v^{+} \wedge \exists m(\mathtt{Eve}\ \mathsf{k}\ m \wedge (\mathtt{Eve}\ \mathsf{k}\ ]m[_{v} \supseteq \mathtt{Eve}\ \mathsf{k}\ m)))$$

$\lrcorner$

Formalisation 20 (Selective forgery)

"An adversary is able to create a valid signature for a particular [∃] message or class of messages chosen [▷] a priori. Creating the signature does not directly involve the legitimate signer." [3, Page 432]

$$\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \wedge \mathtt{Eve}\ \mathsf{k}\ v^{+} \wedge \exists m(\mathtt{Eve}\ \mathsf{k}\ m \rhd (\mathtt{Eve}\ \mathsf{k}\ ]m[_{v} \supseteq \mathtt{Eve}\ \mathsf{k}\ m)))$$

$\lrcorner$

Formalisation 21 (Universal forgery)

"An adversary is able to create a valid signature for an arbitrary [∀] message chosen a priori."

$$\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \wedge \mathtt{Eve}\ \mathsf{k}\ v^{+} \wedge \forall m(\mathtt{Eve}\ \mathsf{k}\ m \rhd (\mathtt{Eve}\ \mathsf{k}\ ]m[_{v} \supseteq \mathtt{Eve}\ \mathsf{k}\ m)))$$

$\lrcorner$

Formalisation 22 (Total break)

"An adversary is either able to compute the private key information of the signer, or finds an efficient signing algorithm functionally equivalent to the valid signing algorithm." [3, Page 432]

$$\exists v \exists (a : \mathtt{P})(v^{+}\ \mathsf{puk}\ a \wedge \mathtt{Eve}\ \mathsf{k}\ v^{+} \wedge \mathtt{Eve}\ \mathsf{k}\ v)$$

$\lrcorner$

References

1. Meadows, C.: Ordering from Satan's menu: a survey of requirements specification for formal analysis of cryptographic protocols. Science of Computer Programming **50**(3–22) (2003)
2. Rogaway, P.: On the role of definitions in and beyond cryptography. In: Proceedings of the Asian Computing Science Conference. (2004)
3. Menezes, A.J., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996)
4. Goldreich, O.: Foundations of Cryptography: Basic Tools. Cambridge University Press (2001)

5. Goldreich, O.: Foundations of Cryptography: Basic Applications. Cambridge University Press (2004)
6. Manna, Z., Pnueli, A.: The Temporal Logic of Reactive and Concurrent Systems: Specification. Springer (1984)
7. Durgin, N., Mitchell, J.C., Pavlovic, D.: A compositional logic for proving security properties of protocols. Journal of Computer Security **11**(4) (2003)
8. Harel, D., Kozen, D., Tiuryn, J.: Dynamic Logic. MIT Press (2000)
9. Boyd, C., Mathuria, A.: Protocols for Authentication and Key Establishment. Springer (2003)
10. Sumii, E., Pierce, B.C.: Logical relations for encryption. Journal of Computer Security **11**(4) (2003)
11. Ryan, P., Schneider, S., Goldsmith, M., Lowe, G., Roscoe, B.: The Modelling and Analysis of Security Protocols: the CSP Approach. Addison-Wesley (2000)
12. Abadi, M., Fournet, C.: Mobile values, new names, and secure communication. In: Proceedings of the ACM Symposium on Principles of Programming Languages. (2001)
13. Abadi, M., Gordon, A.D.: A calculus for cryptographic protocols: The Spi-calculus. Information and Computation **148**(1) (1999)
14. Mitchell, J.C., Ramanathan, A., Scedrov, A., Teague, V.: A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. Theoretical Computer Science **353**(1–3) (2006)
15. Abadi, M.: Security protocols and their properties. In: Foundations of Secure Computation, IOS Press (2000)
16. Burrows, M., Abadi, M., Needham, R.: A logic of authentication. ACM Transactions on Computer Systems **8**(1) (1990)
17. Syverson, P.F., van Oorschot, P.C.: A unified cryptographic protocol logic. CHACS 5540-227, Naval Research Laboratory, Washington D.C., USA (1996)
18. Aiello, L.C., Massacci, F.: Verifying security protocols as planning in logic programming. ACM Transactions on Computational Logic **2**(4) (2001)
19. Abadi, M., Blanchet, B.: Analyzing security protocols with secrecy types and logic programs. Journal of the ACM **52**(1) (2005)
20. Bieber, P., Cuppens, F.: Expression of Confidentiality Policies with Deontic Logic. In: Deontic Logic in Computer Science: Normative System Specification. John Wiley & Sons (1993)
21. Accorsi, R., Basin, D., Viganò, L.: Towards an awareness-based semantics for security protocol analysis. In: Proceedings of the Post-CAV Workshop on Logical Aspects of Cryptographic Protocol Verification. (2001)
22. Zhang, Y., Varadharajan, V.: A logic for modeling the dynamics of beliefs in cryptographic protocols. In: Proceedings of the Australasian Conference on Computer Science. (2001)
23. Syverson, P.F., Stubblebine, S.G.: Group principals and the formalization of anonymity. In: Proceedings of the World Congress On Formal Methods In The Development Of Computing Systems. (1999)
24. Halpern, J., O'Neill, K.: Secrecy in multi-agent systems. In: Proceedings of the IEEE Computer Security Foundations Workshop. (2002)
25. Bozzano, M., Delzanno, G.: Automatic verification of secrecy properties for linear logic specifications of cryptographic protocols. Journal of Symbolic Computation **38**(5) (2004)
26. Gray, J.W., McLean, J.D.: Using Temporal Logic to Specify and Verify Cryptographic Protocols (progress report). In: Proceedings of the IEEE Computer Security Foundations Workshop. (1995)

27. Frendrup, U., Hüttel, H., Jensen, J.N.: Modal logics for cryptographic processes. In: Electronic Notes in Theoretical Computer Science. Volume 68. (2002)
28. Adi, K., Debbabi, M., Mejri, M.: A new logic for electronic commerce protocols. Theoretical Computer Science **291**(3) (2003)
29. Lowe, G., Auty, M.: On a calculus for security protocol development. Technical report, Oxford University (2005)
30. Clarke, E., Jha, S., Marrero, W.: A machine checkable logic of knowledge for specifying security properties of electronic commerce protocols. In: Proceedings of the LICS-Affiliated Workshop on Formal Methods & Security Protocols. (1998)
31. Selinger, P.: Models for an adversary-centric protocol logic. In: Proceedings of the Post-CAV Workshop on Logical Aspects of Cryptographic Protocol Verification. (2001)
32. Cohen, E.: First-order verification of cryptographic protocols. Journal of Computer Security **11**(2) (2003)
33. Paulson, L.C.: The inductive approach to verifying cryptographic protocols. Journal of Computer Security **6**(1) (1998)
34. Impagliazzo, R., Kapron, B.M.: Logics for reasoning about cryptographic constructions. Journal of Computer and Systems Sciences **72**(2) (2006)
35. Coffey, T., Saidha, P.: Logic for verifying public-key cryptographic protocols. In: IEE Proceedings — Computers and Digital Techniques. (1997)
36. Benerecetti, M., Giunchiglia, F., Panti, M., Spalazzi, L.: A logic of belief and a model checking algorithm for security protocols. In: Proceedings of FORTE. (2000)
37. Caleiro, C., Viganò, L., Basin, D.: Towards a metalogic for security protocol analysis. In: Proceedings of the Workshop on Combination of Logics. (2004)
38. Baltag, A.: Logics for insecure communication. In: Proceedings of the Conference on Theoretical Aspects of Rationality and Knowledge. (2001)
39. Dixon, C., Gago, M.C.F., M. Fisher, W.v.d.H.: Using temporal logics of knowledge in the formal verification of security protocols. In: Proceedings of the International Symposium on Temporal Representation and Reasoning. (2004)
40. Gnesi, S., Latella, D., Lenzini, G.: A BRUTUS logic for the Spi-Calculus. In: Proceedings of the IFIP Workshop on Issues in the Theory of Security. (2001)
41. Bieber, P.: A logic of communication in hostile environment. In: Proceedings of the IEEE Computer Security Foundations Workshop. (1990)
42. Glasgow, J., Macewen, G., Panangaden, P.: A logic for reasoning about security. ACM Transactions on Computer Systems **10**(3) (1992)
43. Gabbay, D., Kurucz, A., Wolter, F., Zakharyaschev, M.: Many-Dimensional Modal Logics: Theory and Applications. Elsevier (2003)
44. Blackburn, P., de Rijke, M., Venema, Y.: Modal Logic. Cambridge University Press (2001)
45. Kramer, S.: Cryptographic Protocol Logic. In: Proceedings of the LICS/ICALP-Affiliated Workshop on Foundations of Computer Security. (2004)
46. Clarke, Jr., E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press (1999)
47. Fitting, M.: First-Order Logic and Automated Theorem Proving. Springer-Verlag (1996)
48. Fagin, R., Halpern, J.Y., Moses, Y., Vardi, M.Y.: Reasoning about Knowledge. MIT Press (1995)
49. Dam, M.F.: Relevance Logic and Concurrent Composition. PhD thesis, University of Edinburgh (1989)
50. Bergstra, J.A., Ponse, A., Smolka, S.A., eds.: Handbook of Process Algebra. Elsevier (2001)

51. Borgström, J., Kramer, S., Nestmann, U.: Calculus of Cryptographic Communication. In: Proceedings of the LICS-Affiliated Workshop on Foundations of Computer Security and Automated Reasoning for Security Protocol Analysis. (2006)
52. Borgström, J., Grinchtein, O., Kramer, S.: Timed Calculus of Cryptographic Communication. In: Proceedings of the Workshop on Formal Aspects in Security and Trust. (2006)
53. Canetti, R.: Universally composable security: A new paradigm for cryptographic protocols. In: Proceedings of the IEEE Symposium on Foundations of Computer Science. (2001)
54. Backes, M., Pfitzmann, B., Waidner, M.: A universally composable cryptographic library. In: Proceedings of the ACM Conference on Computer and Communication Security. (2003)
55. Dolev, D., Yao, A.C.: On the security of public key protocols. IEEE Transactions on Information Theory **29**(12) (1983)
56. Abadi, M., Rogaway, P.: Reconciling two views of cryptography (the computational soundness of formal encryption). Journal of Cryptology **15**(2) (2002)
57. Meyer, J.J.C., Dignum, F.P.M., Wieringa, R.J.: The Paradoxes of Deontic Logic Revisited: A Computer Science Perspective. Or: Should computer scientists be bothered by the concerns of philosophers ? Technical Report UU-CS-1994-38, Utrecht University (1994)
58. Nute, D., ed.: Defeasible Denotic Logic. Volume 263 of Synthese Library. Kluwer (1997)
59. Abadi, M., Tuttle, M.R.: A semantics for a logic of authentication. In: Proceedings of the ACM Symposium of Principles of Distributed Computing. (1991)
60. Cohen, M., Dam, M.: A completeness result for BAN logic. In: Proceedings of the Workshop on Methods for Modalities. (2005)
61. Cohen, M., Dam, M.: Logical omniscience in the semantics of BAN logic. In: Proceedings of the LICS-affiliated Workshop on the Foundations of Computer Security. (2005)
62. Wang, F.: Formal verification of timed systems: A survey and perspective. Proceedings of the IEEE **92**(8) (2004)
63. Bellini, P., Mattolini, R., Nesi, P.: Temporal logics for real-time system specification. ACM Computing Surveys **32**(1) (2000)
64. Evans, N., Schneider, S.: Analysing time-dependent security properties in CSP using PVS. In: Proceedings of the European Symposium on Research in Computer Security. (2000)
65. Gorrieri, R., Martinelli, F.: A simple framework for real-time cryptographic protocol analysis with compositional proof rules. Science of Computer Programming **50**(1–3) (2004)
66. Haack, C., Jeffrey, A.: Timed Spi-calculus with types for secrecy and authenticity. In: Proceedings of CONCUR. (2005)
67. Schneider, S.: Concurrent and Real-Time Systems. Wiley (1999)
68. Bozga, L., Ene, C., Lakhnech, Y.: A symbolic decision procedure for cryptographic protocols with time stamps. The Journal of Logic and Algebraic Programming **65** (2005)
69. Hansen, M.R., Sharp, R.: Using interval logics for temporal analysis of security protocols. In: Proceedings of the ACM Workshop on Formal Methods in Security Engineering. (2004)
70. Kudo, M., Mathuria, A.: An extended logic for analyzing timed-release public-key protocols. In: Proceedings of the Conference on Information, Communications and Signal Processing. (1999)

71. Gong, L.: A security risk of depending on synchronized clocks. ACM SIGOPS Operating Systems Review **26**(1) (1992)
72. Lamport, L.: Real time is really simple. Technical Report MSR-TR-2005-30, Microsoft Research (2005)
73. Kramer, S.: Timed Cryptographic Protocol Logic (2006) presented at the Nordic Workshop on Programming Theory.
74. Datta, A., Derek, A., Mitchell, J., Pavlovic, D.: A derivation system and compositional logic for security protocols. Journal of Computer Security **13** (2005)
75. Datta, A., Derek, A., Mitchell, J.C., Shmatikov, V., Turuani, M.: Probabilistic polynomial-time semantics for a protocol security logic. In: Proceedings of the EATCS International Colloquium on Automata, Languages and Programming. (2005)
76. Halpern, J.: Reasoning about Uncertainty. MIT Press (2003)
77. Smith, B.S.: The Hilbert epsilon-operator and its significance for metamathematics. ProQuest / UMI (2006)
78. Goldwasser, S., Micali, S., Rackoff, C.: The knowledge complexity of interactive proof systems. SIAM Journal on Computing **18**(1) (1989)
79. Goldreich, O.: Foundations of Cryptography — A Primer. now Publishers Inc. (2005)
80. Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Science **28**(2) (1984)