Timed Cryptographic Protocol Logic

Simon Kramer

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

Abstract. We extend the (core) Cryptographic Protocol Logic (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[1] thus provides further evidence for Lamport's claim that adding real time to an untimed formalism is really simple.

Keywords applied formal logic, cryptographic protocols, property-based specification and verification, real time

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 [2] for a survey of timed models (automata, Petri nets), model- and property-based specification languages (process calculi, resp. logics), and verification tools; and [3] 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): [4], [5], [6] with *discrete* time; [7], [8], and our own contribution [9] with *dense* time. Property-based formalisms (logics): *interval*-based [10]; time-parametrised epistemic modalities [11] and a third-order logic [8] both *point*-based, and our hereby informally summarised logic tCPL [1, Appendix C] 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." [2]. Specifically in cryptographic systems [12], "[c]locks can become unsynchronized due to sabotage on or faults in the clocks or the synchronization mechanism,

[1] this paper is an informal excerpt from the corresponding addendum to [1, Appendix C] with technical details

such as overflows and the dependence on potentially unreliable clocks on remote sites [...]". Moreover [12], "[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[2]; modal logics provide powerful abstractions for specification purposes, but are still not expressive enough [1]; higher-order logics are too expressive at the cost of axiomatic and computational incompleteness[3]; finally "[f]irst-order logics seem a good compromise between expressiveness and computability, since they are [axiomatically] complete in general." [2]. Core CPL is a poly-dimensional modal (norms, knowledge, space, *qualitative* time) first-order logic [1].

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)[4] temporal accessibility relation [13]. tCPL [1, Appendix C] can be instantiated with a temporal accessibility relation that *additionally* accounts for *quantitative* time [9]. That is, time is (1) rational-number[5] 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 (cf. [1, Appendix C]); (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)[6]; and (5) determinant for adversarial break of short-term keys, enabled jointly by key expiration and ciphertext-only attacks (the weakest reasonable attack).

The technical details of the extension of CPL to tCPL are described in [1, Appendix C]. The extension of CPL to tCPL parallels the extension of C^3 [13] to tC^3 [9].

References

1. Kramer, S.: Logical concepts in cryptography. Cryptology ePrint Archive, Report 2006/262 (2006) http://eprint.iacr.org/.

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

[3] but are good as logical frameworks

[4] 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 [...]" [2]

[5] 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

[6] 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)

2. Wang, F.: Formal verification of timed systems: A survey and perspective. Proceedings of the IEEE **92**(8) (2004)
3. Bellini, P., Mattolini, R., Nesi, P.: Temporal logics for real-time system specification. ACM Computing Surveys **32**(1) (2000)
4. 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)
5. 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)
6. Haack, C., Jeffrey, A.: Timed Spi-calculus with types for secrecy and authenticity. In: Proceedings of CONCUR. (2005)
7. Schneider, S.: Concurrent and Real-Time Systems. Wiley (1999)
8. 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)
9. Borgström, J., Grinchtein, O., Kramer, S.: Timed Calculus of Cryptographic Communication. In: Proceedings of the Workshop on Formal Aspects in Security and Trust. (2006)
10. 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)
11. 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)
12. Gong, L.: A security risk of depending on synchronized clocks. ACM SIGOPS Operating Systems Review **26**(1) (1992)
13. 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)