Performance Evaluations of Cryptographic Protocols Verification Tools Dealing with Algebraic Properties

Pascal Lafourcade[1]* and Maxime Puys[2]**

[1] University Clermont Auvergne, LIMOS, France
[2] Univ. Grenoble Alpes, VERIMAG, F-38000 Grenoble, France
CNRS, VERIMAG, F-38000 Grenoble, France

Abstract. There exist several automatic verification tools of cryptographic protocols, but only few of them are able to check protocols in presence of algebraic properties. Most of these tools are dealing either with Exclusive-Or (xor) and exponentiation properties, so-called Diffie-Hellman (DH). In the last few years, the number of these tools increased and some existing tools have been updated. Our aim is to compare their performances by analysing a selection of cryptographic protocols using xor and DH. We compare execution time and memory consumption for different versions of the following tools OFMC, CL-Atse, Scyther, Tamarin, TA4SP, and extensions of ProVerif (XOR-ProVerif and DH-ProVerif). Our evaluation shows that in most of the cases the new versions of the tools are faster but consume more memory. We also show how the new tools: Tamarin, Scyther and TA4SP, can be compared to previous ones. We also discover and understand for the protocol IKEv2-DS a difference of modelling by the authors of different tools, which leads to different security results. Finally, for Exclusive-Or and Diffie-Hellman properties, we construct two families of protocols $P\mathrm{xor}_i$ and $P\mathrm{dh}_i$ that allow us to clearly see for the first time the impact of the number of operators and variables in the tools' performances.

Keywords: Verification Tools for Cryptographic Protocols, Algebraic Properties, Benchmarking, Performances' Evaluations.

1 Introduction

Nowadays cryptographic protocols are commonly used to secure communication. They are more and more complex and analysing them clearly outpaces humans capacities. Hence automatic formal verification is required in order to design secure cryptographic protocols and to detect flaws. For this goal, several automatic verification tools for analysing cryptographic protocols have

* This research was conducted with the support of the "Digital trust" Chair from the University of Auvergne Foundation.
** This work has been partially supported by the LabEx PERSYVAL-Lab (ANR–11-LABX-0025).

been developed, like Avispa [ABB$^+$05] (OFMC [BMV05], TA4SP [BHK04], CL-Atse [Tur06], Sat-MC [AC05]), Tamarin [MSCB13], Scyther [Cre08], Hermes [BLP03], ProVerif [Bla01], NRL [Mea96a], Murphi [MMS97], Casper/FDR [Low98,Ros95], Athena [SBP01], Maude-NPA [EMM07], STA [BB02], the tool S^{3}A [DSV03] and [CE02]. security properties and rely on different theoretical approaches, *e.g.,* rewriting, solving constraints system, SAT-solvers, resolution of Horn clauses, or tree automata *etc.* All these tools work in the symbolic world, where all messages are represented by an algebra of terms. Moreover, they also consider the well-known Dolev-Yao intruder model [DY81], where a powerful intruder is considered [Cer01]. This intruder controls the network, listens, stops, forges, replays or modifies some messages according to its capabilities and can play several sessions of a protocol. The perfect encryption hypothesis is often assumed, meaning that without the secret key associated to an encrypted message it is not possible to decrypt the cipher-text. In such model most of the tools are able to verify two security properties: secrecy and authentication. The first property ensures that an intruder cannot learn a secret message. The authentication property means that one participant of the protocol is sure to communicate with another one.

Historically, formal methods have been developed for analysing cryptographic protocols after the flaw discovered by G. Lowe [Low96] 17 years after the publication of Needham-Schoreder protocol [NS78]. The security of this protocol has been proven for one session using the BAN logic in [BAN90,BM94]. The flaw discovered by G. Lowe [Low96] works because the intruder plays one session with Alice and in the same time a second one with Bob. In this second session, Bob believes that he is talking to Alice. Then the intruder learns the shared secret key that Bob thinks that he shares with Alice. This example clearly shows that even for a protocol of three messages the number of possible combinations outpaces the humans' capabilities.

In presence of algebraic properties, the number of possible combinations to construct traces blows up. The situation is even worse because some attacks can be missed. Let consider the following 3-pass Shamir protocol composed of three messages, where $\{m\}_{KA}$ denotes the encryption of m with the secret key KA:

1. A $\rightarrow$ B : $\{m\}_{KA}$
2. B $\rightarrow$ A : $\{\{m\}_{KA}\}_{KB}$
3. A $\rightarrow$ B : $\{m\}_{KB}$

This protocol works only if the encryption has the following algebraic property: $\{\{m\}_{KA}\}_{KB} = \{\{m\}_{KB}\}_{KA}$. In order to implement this protocol one can use the One Time Pad (OTP) encryption, also known as Vernam encryption because it is generally credited to Gilbert S. Vernam and Joseph O. Mauborgne, but indeed it was invented 35 years early by Franck Miller [Bel11]. The encryption of the message m with the key k is $m \oplus k$. This encryption is perfectly secure according to Shanon information theory, meaning that without knowing the key no information about the message is leaked [Vau05,BJL$^+$10]. Moreover the OTP encryption is key commutative since: $\{\{m\}_{KA}\}_{KB} = (m \oplus KA) \oplus KB = (m \oplus KB) \oplus KA = \{\{m\}_{KB}\}_{KA}$. Unfortunately combining the

OTP encryption and the 3-pass Shamir leads to an attack against a passive intruder that only listens to all communications between Alice and Bob. Hence the intruder collects the following three messages: $m \oplus KA$; $(m \oplus KA) \oplus KB$; $m \oplus KB$. Then he can learn m just by performing the Exclusive-Or of these three messages, since $m = m \oplus KA \oplus (m \oplus KA) \oplus KB \oplus m \oplus KB$. This attack relies on the algebraic property of the encryption and cannot be detected if the modelling of the encryption is not precise enough. It is why considering algebraic operators is important. In [CDL06] the authors proposed a survey of exiting protocols dealing with algebraic properties. In order to fill this gap, some tools have been designed to consider some algebraic properties [BMV05,Bla01,EMM07,KT09,KT08,Tur06,SMCB12]. Indeed doing automatic verification in presence of an algebraic property is more challenging, it is why there exists less tools that are able to deal with algebraic properties. More precisely, the algebraic properties for Diffie-Hellman are only the commutativity of the exponentiation: $(g^a)^b = (g^b)^a$. For Exclusive-Or the following four properties are considered: $(A \oplus B) \oplus C = A \oplus (B \oplus C)$ (Associativity), $A \oplus B = B \oplus A$ (Commutativity), $A \oplus 0 = A$ (Unit element), and $A \oplus A = 0$ (Nilpotency)

Contributions: We compare performances of cryptographic verification tools that are able to deal with two kinds of algebraic properties: exclusive-or and exponentiation. In order to perform this evaluation, we analyse execution time and also memory consumption for 21 protocols that use algebraic operators from the survey [CDL06] or directly from the libraries proposed by each tool. Modelling all these protocols in all the considered tools is a complex task, since it requires to really understand each tool and to be able to write for each protocol the corresponding input file in each specific language. We discover that the modelling of one protocol differs in the library of Avispa and in the library of Scyther. Our investigations show that Avispa finds a flaw and Scyther does not. By building exactly the same models for the two tools, both are able to prove the security of one version and to find an attack in the second one. It clearly demonstrates that the modelling phases is crucial and often fancy even for experts. Finally for tools that can deal with Exclusive-Or and Diffie-Hellman, we construct two families of protocols $P\text{xor}_i$ and $P\text{dh}_i$, in order to evaluate the impact of the number of operators and variables used in a protocol. We discover that it provokes an exponential blowup of the complexity. Having this in mind the results of our experimentations become clearer.

We would like to thank the designers of the tools that helped us to face some modelling tricks we had for some protocols.

State-of-the-art: Comparing to the number of papers for describing and developing tools and the numbers of works that are using such tools to find flaws or prove the security of one protocol, there are only few works that compare the performances of cryptographic protocols verification tools. This comes from the fact that it requires to know how all the tools work. Moreover it is a time consuming task since the protocols need to be coded in the different specific input languages of each tool.

In 1996, C. Meadows [Mea96b] proposed a first comparison work that analyses the approach G. Lowe used in FDR [Ros94] on the Needham-Schroeder protocol [NS78] with the one used in NRL [Mea96a]. It happened that both tools were complementary as FDR is faster but requires outside assistance, while NRL was slower but automatic. In 2002, the AVISS tool [ABB$^+$02] was used to analyse a large set of protocols and timing results are given. As this tool is composed of three back-end tools, the aim was to compare these tools. In 2006, Avispa [ABB$^+$05][1] was created as the successor of AVISS and composed of the same three back-end tools plus one new. These back-ends have been compared by L. Vigano in [Vig06]. Still in 2006, M. Hussain et al. [HS06] qualitatively compared Avispa and Hermes [BLP03], studying their complexity and ease to use. Hermes has been declared more suited for simple protocols while Avispa is better when scalability is needed. In 2007, M. Cheminod et al. [CBD$^+$07] provided a comparison of S^{3}A (Spi calculus Specifications Symbolic Analyzer) a prototype of the work [DSV03], OFMC [BMV05], STA [BB02] and Casper/FDR [Low98]. The purpose was to check for each tool if it was able to deal with specific types of flaw. In 2009, C. Cremers et al. proposed in [CLN09] a fair comparison of Casper/FDR, ProVerif, Scyther and Avispa. Timings were given as well as a modelling of state spaces for each tool. For the first time, the authors were able to show the difference of performances between the Avispa tools. In 2010, N. Dalal et al. [DSHJ10] compared the specifications of ProVerif and Scyther on six various protocols. No timing was given since the objective was to show the differences of the tools in term of features. Still in 2010, R. Patel et al. [PBP$^+$10] provided a detailed list of cryptographic protocols verification tools split into different categories depending of their inner working. They compared the features of Scyther and and ProVerif.

All these works only compare selected tools on protocols that do not require algebraic properties except [PBP$^+$10] that consider Diffie-Hellman using ProVerif. In [LTV10], P. Lafourcade et al. analysed some protocols dealing with algebraic properties. The results of this analysis clearly show that there is no clear winner in term of efficiency. This work also conjectures that the tools are influenced by the number of occurrences of the operator in the protocols. Moreover, none of them consider memory consumption.

Our aim is to revise the work of [LTV10], because new versions of compared tools are now available and we also want to include new tools and protocols in the comparison. Moreover we propose two families of protocols to give a first answer to the conjecture given in [LTV10] and to understand which parameters influence the performances of the tools dealing with Exclusive-Or and Diffie-Hellman.

Outline: In Section 2, we present the different tools that we compare. In Section 3, we explain the results of our benchmark. We also detail our experimentations on the impact of the number of variables involved in Exclusive-Or and Diffie-Hellman operations on the tools. Finally we conclude in Section 4.

[1] http://www.avispa-project.org/

2 Tools

We present the six tools used for our comparison and give the different tool versions used in our analysis. To the best of our knowledge, those are the main free available tools dealing with two common algebraic properties used in cryptographic protocols: Exclusive-Or or Diffie-Hellman.

CL-Atse [Tur06] (Version 2.2-5 (2006) and 2.3-4 (2009)) *Constraint-Logic-based Attack Searcher*[2], developed by M. Turuani, runs a protocol in all possible ways over a finite set of sessions, translating traces into constraints. Constraints are simplified thanks to heuristics and redundancy elimination techniques allowing to decide whether some security properties have been violated or not.

OFMC [BMV03] (Version 2006-02-13 and 2014) The *Open-source Fixed-point Model-Checker*[3], developed by S. Mördersheim, applies symbolic analysis to perform protocol falsification and bounded analysis also over a finite set of sessions. The state space is explored in a demand-driven way.

TA4SP [BHKO04] (Version 2014) *Tree Automata based on Automatic Approximations for the Analysis of Security Protocols*[4], developed by Y. Boichut, approximates the intruder knowledge by using regular tree languages and rewriting. For secrecy properties, it can either use over-approximation or under-approximation to show that the protocol is flawed or safe for any number of session. However, no attack trace is provided by the tool and only the secrecy is considered in presence of algebraic properties.

CL-Atse, OFMC and TA4SP are backend tools used within Avispa (*Automated Validation of Internet Security Protocols and Applications*). All these tools take as input a common language called HLPSL (*High Level Protocol Specification Language*).

ProVerif[5] [Bla01,Bla04] (Version: 1.16 (2008) and 1.90 (2015)) developed by B. Blanchet analyses an unbounded number of sessions. Inputs can be written either in Horn clauses format or using a subset of the Pi-calculus. It uses over-approximation techniques such as an abstraction of fresh nonce generation to prove that a protocol satisfies user-given properties. If a property cannot be proven, it reconstructs an attack's trace.

In [KT08] (2008) and [KT09] (2009) R. Küster and T. Truderung proposed two translators named XOR-ProVerif and DH-ProVerif. These tools respectively transform a protocol using Exclusive-Or and Diffie-Hellman properties, written as Prolog file into a protocol in Horn clauses which is compatible with ProVerif. Both of these tools require the version 5.6.14 of SWI/Prolog to work. Since these works, ProVerif has been enhanced to support Diffie-Hellman on its own, by adding a specific equatinal theory in each protocol specification.

Scyther[6] [Cre08] (Version 1.1.3 (2014)) developed by C. Cremers, verifies bounded and unbounded number of runs with guaranteed termination, us-

[2] http://webloria.loria.fr/equipes/cassis/softwares/AtSe/
[3] http://www.imm.dtu.dk/ samo/
[4] http://www.univ-orleans.fr/lifo/membres/Yohan.Boichut/ta4sp.html
[5] http://prosecco.gforge.inria.fr/personal/bblanche/proverif/
[6] https://www.cs.ox.ac.uk/people/cas.cremers/scyther/

ing a symbolic backwards search based on patterns. Scyther does not support Exclusive-Or or Diffie-Hellman off the shelf but under-approximates Diffie-Hellman by giving the adversary the capability of rewriting such exponentiations at fixed subterm positions, which are derived from the protocol specification. This trick has been first introduced by C. Cremer in [Cre11] on the protocols of IKEv1 and IKEv2 suite. Those modelizations are presented in the protocols' library of the tool.

Tamarin[7] [SMCB12,MSCB13] (Version 0.9.0 (2013)) is a security protocol prover able to handle an unbounded number of sessions. Protocols are specified as multiset rewriting systems with respect to (temporal) first-order properties. It relies on Maude [ECEM96] tool[8]. It only supports Diffie-Hellman equational theory.

3 Experimentations and Discussion

We present the results on the modellings of the analysed protocols with OFMC, CL-Atse, TA4SP, Tamarin, Scyther and extensions of ProVerif. We analyse the same protocols as in the paper [LTV10] in order to see how the tools have been updated. This list of the protocols in [LTV10] contains: Bull's Authentication Protocol [BO97,RS98], e-Auction [HTWSCK08], Gong's Mutual Authentication Protocol [Gon89], Salary Sum [Sch96], TMN [LR97a,TMN89], Wired Equivalent Privacy Protocol [80299], Diffie-Hellman [DH76] and IKA [AST00]. We also add some protocols that are given in the benchmarks of the new considered tools (*Secure Shell (SSH) Transport Layer Protocol* [YL06], *Internet Key Exchange Protocol version 2 (IKEv2)* [Kau05,KHN$^+$14]), NSPKxor [LR97b] and 3-Pass Shamir described in the introduction. We selected these protocols as they were either proposed by the tool's authors or listed in the survey [CDL06].

Our experiments were run on an Intel(R) Core(TM) i5-4310U 2.00GHz CPU with 16 GB of RAM. Memory usage per process is not limited (ulimit -m unlimited). Timings and memory consumption were determined using the GNU *time*[9] command computing the *Real time* and the *Maximum Resident Set Size* for each run of each tool. All testcases were run with a timeout of 24h using the GNU *timeout*[10] command. All our codes of each protocol modeling for each tool are availaible in [PL].

For accuracy reasons, we launched each run 50 times if it takes less than 1h for the tool to analyse the protocol. Then we computed the mean of all timings and memory usages. When it takes more than 1h, we restrict to 10 iterations. We denote a protocol by *-fix* for its corrected version if any and *v2* for its simplified versions if needed.

Table 1 summarizes the secrecy results of tools dealing with Exclusive-Or (OFMC, CL-Atse, TA4SP and ProVerif) on some protocols. Table 2 compiles

[7] http://www.infsec.ethz.ch/research/software/tamarin.html
[8] http://maude.cs.uiuc.edu/download/
[9] http://linux.die.net/man/1/time
[10] http://linux.die.net/man/1/timeout

results we obtain on secrecy with all the tools on Diffie-Hellman based protocols. Finally, Table 3 recaps results obtained by all the tools (but TA4SP which only deals with secret) on protocols with authentication properties. Numbers in parenthesis denotes the number of property specified for each modelization. Notice that TA4SP is able to run either in over-approximation or under-approximation. However, due to the time taken by the tool, we were not able to check any protocol (except NSPKxor) using under-approximation within our timeout. Thus all results from TA4SP only use over-approximation.

Obviously, the transformation algorithm proposed by R. Küster and T. Truderung in [KT08] and [KT09] adds an overhead in terms of computation time and memory usage. However, we found out that this overhead was often less than 1s and 3000 Kb of memory consumption. It appears to be different only for *BAPv2* and *BAPv2-fix* protocols for which it was respectively 1.78s and 6.05s (memory was not blowing up). Except for these two protocols, the overhead induced by the transformation was negligible and constant so it is not shown in our results.

It is important to notice that all tools do not have the same objectives. CL-Atse and OFMC are designed to find attacks and stop once they found one. TA4SP, Tamarin and Scyther are provers and try to find attacks on each property specified even if they already violated one. ProVerif is also designed to prove all the properties that are specified but the pretreatments added by R. Kuesters et al. can make him stop once an attack is found or not depending on the modelization (in particular the presence of *begin* and *end* statements). To be completely fair, we need to check unsafe protocols one property after one other to make sure all are tested. Obviously this is not needed for safe protocols since all the properties must be verified for such verdict.

3.1 Comparing old and new versions of the tools

In [LTV10], the authors compared the performances of CL-Atse, OFMC and ProVerif. Since then, they have been updated. We use the most recent version of each tool in this comparison. Nevertheless, we also run all of our experimentations using the same version than in [LTV10] to compare how the tools have evolved. For all testcases, we computed the *speedup* indicator as the result of $S = \frac{T_{old}}{T_{new}}$ where S is the resultant speedup and T_{old} (resp. T_{new}) is the timing obtained with the old (new) version of the tool. We choose to not compute it for values less than 1s as they are hardly representative. The exact same computation have been done with memory usages. For each tool tested in [LTV10] we compare the results obtained with the new version and the former one, all theses results can also be found in Tables 1, 2 and 3.

CL-Atse: We compare the version 2.2-5, released in 2006 with the version 2.3-4, released in 2009. By looking at the speedup indicator, the tool seems slightly slower in its newer version (0.97 times on average). The old version has a minimum memory usage of about 1570 Kb (reached in 53% of the protocols). This minimum has increased to about 5024 Kb (reached in 90% of the protocols). Thus, we can notice that memory usages are pretty stable in particular with the new version.

Protocol studied	CL-Atse v2.3-4 [Tur06]		OFMC 2014 [BMV03]		TA4SP 2014 [BHKO04]		XOR-ProVerif 1.90 [Bla01,Bla04]		Speedup CL-Atse		Speedup OFMC		Speedup ProVerif	
BAP [BO97,RS98] UNSAFE	0.12s + 0.12s = 0.24s	5024 5024 5024	0.09s + 0.15s = 0.24s	9317 20581 20581	No result TA4SP error Out of memory		XOR-ProVerif Does not end (>24h) 71Go output		= = =	0.33 0.37 0.35	= = =	0.51 0.53 0.52	= = =	= = =
BAPv2 [BO97,RS98] UNSAFE	0.12s + 0.12s = 0.24s	5024 5024 5024	0.08s + 0.14s = 0.22s	10077 20581 20581	No result TA4SP error Out of memory		2.20s + 2.20s = 4.40s	153060 153048 153060	= = =	0.33 0.37 0.35	= = =	0.48 0.53 0.51	0.92 0.93 0.93	0.92 0.92 0.92
BAP-fix [BO97,RS98] SAFE	33m55s	5979	Does not end (>24h)		No result TA4SP error Out of memory		Does not end (>24h) 71Go output		0.94	2.43	=	=	=	=
BAPv2-fix [BO97,RS98] SAFE	1m30s	5156	33m8s	376669	No result TA4SP error Out of memory		1m6s	798236	0.97 0.96 0.97	0.80 0.81 0.81	1.4 1.4 1.4	0.61 0.61 0.61	0.96 0.96 0.96	0.82 0.82 0.82
E-auction [HTWSCK08] SAFE	0.85s	85223	0.13s	11373	TA4SP error		0.01s	4767	=	0.51	=	0.42	=	0.86
Gong [Gon89] SAFE	40.76s	5024	7.77s	604267	No result TA4SP error Out of memory		Killed by kernel Because of memory exhausted		1.06	0.38	1.18	0.38	=	=
Salary Sum [Sch96] UNSAFE	0.08s + 0.08s + 0.08s + 0.08s = 0.32s	5025 5024 5024 5024 5025	0.07s + 0.06s + 0.06s + 0.06s = 0.25s	7741 6623 6818 6822 7741	No result TA4SP error Out of memory		23h37m 23h38m 23h37m 23h38m ProVerif error	927024 926975 927015 927024	= = = = =	0.31 0.31 0.31 0.31 0.31	= = = = =	0.42 0.41 0.42 0.41 0.42	= = = = =	= = = = =
Salary Sum v2 [Sch96] UNSAFE	0.08s + 0.08s + 0.08s + 0.08s = 0.32s	5025 5024 5024 5024 5025	0.07s + 0.06s + 0.06s + 0.06s = 0.25s	7663 6553 6561 6558 7663	No result TA4SP error Out of memory		10.17s + 9.98s + 10.11s + 10.06s = 40.32s	29760 29760 29760 29761 29761	= = = = =	0.31 0.31 0.31 0.31 0.31	= = = = =	0.42 0.42 0.42 0.42 0.42	1.67 1.80 1.63 1.61 1.68	0.94 0.94 0.94 0.94 0.94
TMN [LR97a,TMN89] UNSAFE	0.04s + 0.04s = 0.08s	5024 5025 5025	0.05s + 0.07s = 0.12s	7505 10824 10824	8m53s + 8m42s = 17m35s	83860 83940 83940	0.01s + 0.01 = 0.02s	5493 5493 5493	= = =	0.31 0.31 0.31	= = =	0.51 0.51 0.51	= = =	0.88 0.88 0.88
WEP [80299] UNSAFE	0.04s	5024	0.04s	5341	4.05s	50908	0.01s	4510	=	0.31	=	0.40	=	0.86
WEP-fix [80299] SAFE	0.04s	5024	0.04s	5369	4.02s INCONCL.	50908	0.01s	4523	=	0.31	=	0.43	=	0.86
NSPKxor [LR97b] UNSAFE	0.04s + 0.04s = 0.08s	5024 5024 5024	0.04s + 0.04s = 0.08s	5365 5405 5405	21.66s + 22.24s = 43.9s	50968 50696 50968	0.08s + 0.08s = 0.16s	6954 6942 6954	= = =	0.31 0.31 0.31	= = =	2.42 0.58 1.50	= = =	0.68 0.98 0.83
3-Pass Shamir UNSAFE	0.04s	5024	0.04s	5469	TA4SP error		0.01s	3091	=	0.31	=	0.44	=	0.92

Table 1: Comparison of all the tools on 13 protocols using XOR on secrecy properties (memory consumptions in Kb).

OFMC: We compare the version 2006 of the tool with the version 2014. Here we can notice a more clear trend on the reduction of timings (around 1.29 times faster), contrasted by a clear trend on the augmentation of memory usages (0.45, meaning more than doubling). However, unlike we previously said on CL-Atse, memory usage of OFMC can vary a lot (from 5341 Kb to more than 717 Mb). This can be explained by the fact that OFMC is looking for some fix points and this research can require a large memory.

ProVerif: We compare the version 1.16, released in 2008 with the version 1.90 released in early 2015. Looking at the representative timings, we are not able to notice any clear variation (*BAPv2* and *BAPv2-fix* are slower but *Salary Sum v2*

Protocol studied	CL-Atse v2.3-4 [Tur06]		OFMC 2014 [BMV03]		TA4SP 2014 [BHKO04]		DH-ProVerif 1.90 [Bla01,Bla04]		Tamarin 0.9.0 [SMCB12,MSCB13]		Scyther 1.1.3 [Cre08]		Speedup CL-Atse		Speedup OFMC		Speedup ProVerif	
DH [DH76] UNSAFE	0.02s	5060	0.04s	5408	0.36s	51736	0.01s	4616	5.23s	23944	0.01s	610	=	0.31	=	0.46	=	0.86
	+ 0.03s	5064	+ 0.04s	5364	+ 0.36s	51684	+ 0.01s	4616	+ 5.32s	25046	+ 0.01s	608	=	0.31	=	0.43	=	0.86
	= 0.05s	5064	= 0.08s	5408	= 0.72s	51736	= 0.02s	4616	= 10.54s	25046	= 0.02s	610	=	0.31	=	0.45	=	0.86
IKA [AST00] UNSAFE	0.05s	5024	0.05s	5885	No result TA4SP error Stack overflow		0.01s	4934	Does not end (>24h)		0.18s	631	=	0.31	=	0.46	=	0.87
	+ 0.05s	5024	+ 0.05s	6509			+ 0.01s	4902			+ 0.01s	606	=	0.31	=	0.47	=	0.86
	+ 0.05s	5024	+ 0.04s	5801			+ 0.01s	4878			+ 0.01s	606	=	0.31	=	0.45	=	0.86
	= 0.15s	5024	= 0.14s	6509			= 0.03s	4934			= 0.20s	631	=	0.31	=	0.46	=	0.86
SSH [YL06] SAFE	0.58s	5024	8.76s	717841	Does not end (>24h)		0.02s	5902	40.07s	89790	0.16s	736	=	0.36	1.27	0.45	=	0.88
IKEv2-DS [Kau05,KHN+14] SAFE	0.20s	5024	1.12s	101486	0.55s	50864	0.06s	7441	1.91s	38251	38.46s	5932	=	0.36	1.27	0.42	=	0.94
IKEv2-DS-fix [Kau05,KHN+14] SAFE	0.21s	5024	5.58s	535196	1.40s	34494	0.03s	6373	3.14s	54729	42.81s	5952	=	0.36	1.29	0.37	=	0.90
IKEv2-DSv2-fix [Kau05,KHN+14] SAFE	0.15s	5024	1.09s	98878	0.55s	34646	0.06s	7425	1.91s	37669	36.04s	6007	=	0.31	1.26	0.42	=	0.91
IKEv2-CHILD [Kau05,KHN+14] SAFE	0.05s	5023	0.20s	16493	0.56s	34467	0.02s	5747	19.44s	52857	2.24s	1496	=	0.31	=	0.41	=	0.92
IKEv2-MAC [Kau05,KHN+14] SAFE	0.05s	5024	1.06s	96365	0.52s	50889	0.01s	5395	31.50s	70715	4m9s	29176	=	0.31	1.27	0.42	=	0.87

Table 2: Comparaison of all the tools on 8 protocols using DH on secrecy properties (memory consumptions in Kb).

is faster). However, ProVerif also has increased his memory usage with a variation of 0.90. The principal aim of ProVerif is to analyse some cryptographic protocols without equational theory. In our comparison, we use two tools developed by R. Kuesters to analyse our protocols and they have not been updated.

Protocol studied	CL-Atse v2.3-4 [Tur06]		OFMC 2014 [BMV03]		(XOR/DH)-ProVerif 1.90 [Bla01,Bla04]		Tamarin 0.9.0 [SMCB12,MSCB13]		Scyther 1.1.3 [Cre08]		Speedup CL-Atse		Speedup OFMC		Speedup ProVerif	
E-auction [HTWSCK08] SAFE	0.62s	5024	0.13s	11377	0.01s	4768	Not supported		Not supported		=	0.39	=	0.42	=	0.85
NSPKxor [LR97b] UNSAFE	0.04s	5060	0.05s	6440	0.03s	5788	Not supported		Not supported		=	0.31	=	0.42	=	0.89
	+ 0.18s	5060	+ 0.06s	6532	+ 0.03s	5784					=	0.31	=	0.42	=	0.89
	= 0.22s	5060	= 0.11s	6532	= 0.06s	5788					=	0.31	=	0.42	=	0.89
DH [DH76] UNSAFE	0.04s	5064	0.04s	5376	0.01s	4616	5.35s	22540	0.01s	581	=	0.31	=	0.43	=	0.86
	+ 0.04s	5060	+ 0.04s	5572	+ 0.01s	4620	+ 5.64s	21740	+ 0.01s	621	=	0.31	=	0.45	=	0.86
	= 0.08s	5064	= 0.08s	5572	= 0.02s	4620	= 10.99s	22540	= 0.02s	621	=	0.31	=	0.44	=	0.86
IKA [AST00] UNSAFE	0.05s	5060	0.06s	7532	0.01s	5052	Does not end		0.01s	588	=	0.31	=	5.94	=	0.86
	+ 0.06s	5060	+ 0.05s	6392	+ 0.01s	5064	19.65s	406888	0.01s	584	=	0.31	=	0.41	=	0.86
	+ 0.07s	5056	+ 0.04s	5928	+ 0.01s	5056	19.70s	384480	+ 0.01s	584	=	0.31	=	0.44	=	0.86
	+ 0.06s	5056	+ 0.05s	6480	+ 0.01s	5044	19.67s	361748	+ 0.01s	584	=	0.31	=	6.90	=	0.86
	+ 0.06s	5060	+ 0.06s	7516	+ 0.01s	5048	Does not end		+ 0.01s	584	=	0.31	=	5.95	=	0.86
	= 0.30s	5060	= 0.26s	7532	= 0.05s	5064	-		= 0.05s	588	=	0.31	=	3.93	=	0.86
SSH [YL06] SAFE	0.30s	5024	6.49s	656602	0.02s	5902	57.94s	106956	0.03s	699	=	0.36	=	0.44	=	0.88
IKEv2-DS [Kau05,KHN+14] UNSAFE	0.14s	5060	1.06s	95344	0.07s	7416	1.87s	34784	0.01s	660	=	0.31	1.23	0.42	=	0.90
	+ 0.06s	5060	+ 0.08s	9556	+ 0.07s	7428	+ 1.74s	36816	+ 0.03s	724	=	0.31	=	0.50	=	0.90
	= 0.20s	5060	= 1.14s	95344	= 0.14s	7428	= 3.61s	36816	= 0.04s	724	=	0.31	1.23	0.46	=	0.90
IKEv2-DS-fix [Kau05,KHN+14] SAFE	3.05s	5024	5.34s	516206	0.02s	6062	4.06s	58614	41.08s	5660	0.94	0.37	1.34	0.43	=	0.89
IKEv2-DSv2-fix [Kau05,KHN+14] SAFE	0.12s	5025	1.11s	93665	0.06s	7415	1.74s	43059	31.70s	5304	=	0.31	1.35	0.43	=	0.91
IKEv2-CHILD [Kau05,KHN+14] SAFE	0.06s	5024	0.20s	15205	0.02s	5772	20.37s	55102	1.50s	1276	=	0.31	=	0.45	=	0.91
IKEv2-MAC [Kau05,KHN+14] SAFE	0.05s	5024	1.07s	92269	0.01s	5395	31.14s	73231	3m46s	27237	=	0.31	1.37	0.42	=	0.87

Table 3: Comparison of all the tools on authentication properties for 10 XOR and DH protocols (memory consumptions in Kb).

3.2 Observation on the results of the new tools

Here we summarize and explain the individual results of the tools we added since [LTV10].

TA4SP seems to have hard time when dealing with the complexity of Exclusive-Or properties as only 33% of our protocols produce a result. Moreover, the performances of TA4SP are far behind CL-Atse, OFMC and ProVerif. However, when dealing with Diffie-Hellman properties, TA4SP is pretty competitive as its timings are close to the ones of CL-Atse, OFMC and ProVerif. Its memory usages are also always higher than CL-Atse and ProVerif but lower then OFMC in 75% of our protocols.

Tamarin seems really slower than CL-Atse and ProVerif either on secrecy or authentication. It is only faster than OFMC when checking *IKEv2-DS-fix*. The reason may be that the granularity of the modelling is too thin and complexifies the analysis. However, Tamarin is the only tool being able to deal with temporal properties and it would be interesting to try to analyse some protocols using Diffie-Hellman and satisfying such properties.

Scyther does not have Diffie-Hellman properties built in its algorithm. We use a trick that consists to introduce an extra role in the protocol to perform the commutation of the exponentiation, This role is a kind of oracle that is called by the tool when a protocol is analysed. Then Scyther is able to compete with other tools. In selected protocols, the Diffie-Hellman oracle is only called on small messages, then Scyther is pretty efficient (for example in *SSH*). However, protocols such as *IKEv2-CHILD* or *IKEv2-MAC* are still to complex for this hack and would need Diffie-Hellman properties to be built in the tool to be able to compete with other tools.

An interesting example with Scyther is the *IKEv2-DS* protocol. The Internet Key Exchange version 2, Digital Signatures variant (*IKEv2-DS*) aims at establishing mutual authentication between two parties using an IKE Security Association (SA) that includes shared secret information. The first two exchanges of messages establishing an IKE SA are called the IKE_SA_INIT exchange and the IKE_AUTH exchange. During IKE_SA_INIT, users exchange nonces and establishes a Diffie-Hellman key. Then IKE_AUTH authenticates the previous messages, exchanges the user identities and establish an IKE SA.

Protocol IKEv2-DS:

1. $A \rightarrow B : SA1.g^x.Na$
2. $B \rightarrow A : SA1.g^y.Nb$
3. $A \rightarrow B : \{A.\{SA1.g^x.Na.Nb\}inv(pk(A)).SA2\}_{h(Na.Nb.SA1.g^{xy})}$
4. $B \rightarrow A : \{B.\{SA1.g^y.Nb.Na\}inv(pk(B)).SA2\}_{h(Na.Nb.SA1.g^{xy})}$

Where $x.y$ denotes the pair of message x and y. In this given form, *IKEv2-DS* is vulnerable to an authentication attack[11] where the intruder is able to impersonate A when speaking to B. However, he is not able to learn g^{xy}, the key shared by only A and B making this attack unexploitable.

To prevent the attack against *IKEv2-DS*, S. Mödersheim and P. Hankes Drielsma proposed on the Avispa's website[12] to add an extension consisting of

[11] http://www.avispa-project.org/library/IKEv2-DS.html
[12] http://www.avispa-project.org/library/IKEv2-DSx.html

two messages, each containing a nonce and a distinguished constant encrypted with the IKE_SA_INIT key. This version is denoted by *IKEv2-DS-fix*.

As C. Cremers already mentioned in [Cre11], specifying explicitly the responder's identify in the first message of the IKE_AUTH exchange also prevents this attack. We denote by *IKEv2-DSv2-fix* this version. This parameter is specified as optional in Section 1.2 of [KHN+14]. This way, Step 3. of the protocol becomes:

$$A->B:\{A,\mathbf{B},\{SA1.g^x.Na.Nb\}inv(pk(A)),SA2\}_{h(Na.Nb.SA1.g^{xy})}$$

As mentioned in the introduction, the difference of modelling between *IKEv2-DS* which was proposed in the Avispa library and *IKEv2-DSv2-fix* which was included in Scyther's library was indeed changing the result of the security analysis since the later was already fixed. It would not have been easy to spot this difference as the two protocols were modeled in different languages and as the parameter added in *IKEv2-DSv2-fix* was supposed optional. Again such tiny changes require the user to deeply understand the input language of each tool and to understand the original specification of the protocol to be noticed.

Moreover, still in [Cre11], C. Cremers also proposed a more detailed version of *IKEv2-DS*, *IKEv2-DSv2-fix* and *IKEv2-MAC* adding other parameters specified in [KHN+14] but this time not affecting on the result of the analysis. We also run these modelisations with Scyther and interestingly, these additional parameters slow down the tool when analysing *IKEv2-DS* and *IKEv2-DSv2-fix* but accelerate it when we check *IKEv2-MAC*. This shows how modifications not relevant at the first sight can drastically change the performances and even the results of a tool.

3.3 Further analyses

In this section, our aim is to measure the impact of the number of variables involved in Exclusive-Or and Diffie-Hellman on each tools.

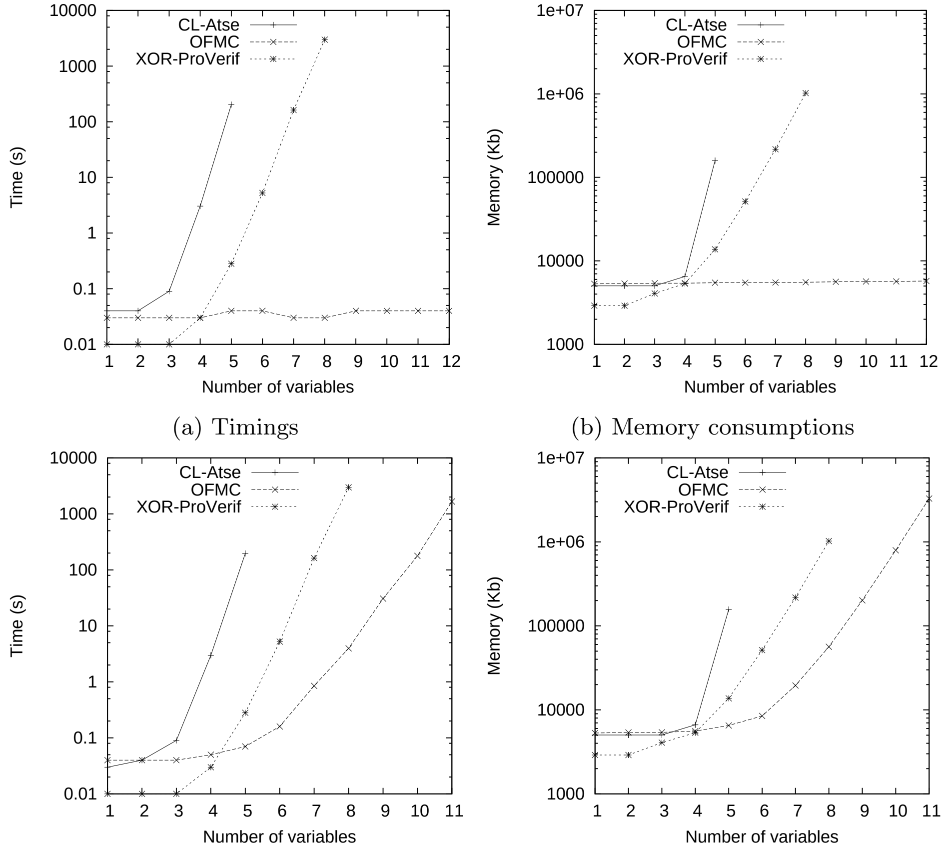
Analysis of the influence of Exclusive-Or operator: We propose the following unsafe family of protocols called $Pxor_i$.

1. $A \rightarrow B : Na_i$
2. $B \rightarrow A : Na_i \oplus Sb$

Where Na_i is result of i fresh nonces xored and Sb is a secret that B wants to share with A. So for instance, if $i = 3$, the protocol $Pxor_3$ is defined such as:

1. $A \rightarrow B : Na_1 \oplus Na_2 \oplus Na_3$
2. $B \rightarrow A : Na_1 \oplus Na_2 \oplus Na_3 \oplus Sb$

With $Na_1 \oplus Na_2 \oplus Na_3 = xor(Na_1, Na_2, Na_3)$. Moreover, we test how the tools handle Exclusive-Or. Thus we also consider a variant $P\text{-}nestedxor_i$ in which $(Na_1 \oplus Na_2) \oplus Na_3 = xor(xor(Na_1, Na_2), Na_3)$. This does not make any difference for XOR-ProVerif since the intermediate files produced are strictly the same.



(a) Timings (b) Memory consumptions

(c) Timings (d) Memory consumptions

Fig. 1: Performances of the tools on the $Pxor_i$ and $P\text{-}nestedxor_i$ protocols

For each tool, Figure 1a represents timings and Figure 1b represents memory consumptions in function of the number of nonces sent by A in the $Pxor_i$ protocol. We stopped runs taking more than one hour. All tools are able to find attacks when they did terminate. We can see that CL-Atse is barely not able to deal with more than five variables in an Exclusive-Or. XOR-ProVerif is able to handle up to eight but taking a really long time. However, OFMC seems to perfectly handle this constraint, keeping both his timings and memory consumptions almost constant.

This experimentation demonstrates that the number of variables in Exclusive-Or operators has a clear impact on the tools. It is a factor of complexity explosion like the number of roles, the number of sessions, the number of nonces and the number of participants. OFMC seems to use an efficient strategy to handle a "global" Exclusive-Or.

Figure 1c represents timings for each tool function of the number of nonces sent by A in the *P-nestedxor*$_i$ protocol. Figure 1d is the same with memory consumptions. All tools are able to find attacks when they did terminate. We can see that CL-Atse has results very close to our experimentation without nested Exclusive-Or. XOR-ProVerif has the exact same behavior as it does not make any difference with prioritized Exclusive-Or or not. This time we can see that OFMC is affected by the number of Exclusive-Or operations growing and is able to handle up to eleven Exclusive-Or.

Analysis of the influence of Diffie-Hellman operator: We propose the following family of unsecure protocols Pdh_i to measure the impact of Diffie-Hellman exponentiations.

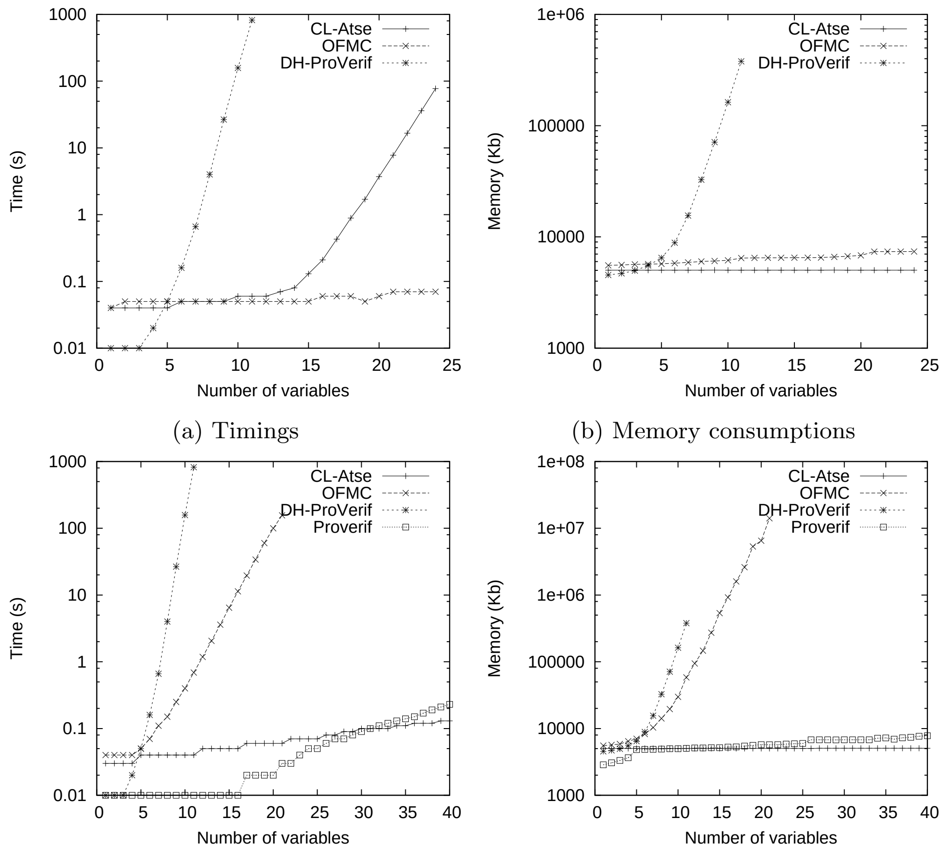
1. $A \rightarrow B : g^{Na_i}$
2. $B \rightarrow A : g^{Nb_i}$
3. $A \rightarrow B : \{S\}_{(g^{Na_i})^{Nb_i}}$

The protocol Pdh_i contains i nonces from A and also i nonces from B so that $(g^{Na_i})^{Nb_i} = exp(g, Na_1, \ldots, Na_i, Nb_1, \ldots, Nb_i)$. We also consider the *P-nesteddh*$_i$ protocol where

$$(g^{Na_i})^{Nb_i} = exp(g, exp(Na_1, exp(\ldots, exp(Na_i, exp(Nb_1, exp(\ldots, Nb_i))))))$$

Figure 2a represents timings for each tool in function of the number of nonces sent by A and B in the Pdh_i protocol. Figure 2b is the same with memory consumptions. When they did terminate, all tools are able to find attacks. We observe that all tools are able to deal with more variables involved in Diffie-Hellman exponentiations than in Exclusive-Or. This due to the fact that Exclusive-Or has four properties, including commutativity, while Diffie-Hellman only has one (commutativity). DH-ProVerif is able to handle up to eleven nonces in each role before taking too much time. CL-Atse reasonably manages 24 variables with its timing slowly growing and its memory stays constant. OFMC has the exact same behavior as with $Pxor_i$, staying constant in timings and memory usage.

Figure 2c and Figure 2d respectively represents timings and memory consumptions for each tool function of the number of nonces sent by A and B in the *P-nesteddh*$_i$ protocol. All tools find some attacks if they terminate. We can modelize the nested Diffie-Hellman exponentiations using a rewriting rule directly in ProVerif 1.90 using Pi-calculus and without using DH-ProVerif (the syntax does not allow exponentiation operators with arity greater than two and exclude tests on Pdh_i). Thus we differentiate results from DH-ProVerif, the algorithm from [KT09] with the results from ProVerif. This time, DH-ProVerif and OFMC are impacted when we increase the number of nonces of the protocol. DH-ProVerif ends up limited by its timing while OFMC fills the memory of the system. Interestingly, this time CL-Atse is perfectly able to handle the nested Diffie-Hellman exponentiations, keeping its timing and memory constant. The modelisation using ProVerif's Pi-calculus language also seems very powerful.



(a) Timings (b) Memory consumptions

(c) Timings (d) Memory consumptions

Fig. 2: Performances of the tools on the P_i and $P\text{-}nested_i$ protocols

4 Conclusion

In the last decades several automatic verification tools for cryptographics protocols have been developed. They are really useful to help the designer to construct secure protocols against the well known Dolev-Yao intruder. Only few of these tools are able to analyse algebraic properties. In this work we compare the execution time and the memory consumption of the main free tools that can deal with Exclusive-Or and Diffie-Hellman properties. We use a large benchmark of 21 protocols. In this competition there is not a clear winner. However we can see that recent tools can deal with some of these properties. For instance Tamarin offers the verification of new temporal properties and consider Diffie-Hellman property. We also construct two families of protocols to evaluate how the performances of existing tools, that are able to consider the Exclusive-Or and Diffie-Hellman operators, are influenced by the number of operators in the protocol. We clearly see that the complexity is exponential in function of the number of operators used with variables. We also notice that the modelling is an important step in the verification of cryptographic protocols and it can really influence their performance. Moreover each tool has is own strategy based on his theoretical foundations to find attack or to prove the security properties, it is not surprising that there is not a clear winner of our comparison on the set of protocols since algebraic operators introduce a new factor of complexity in the verification procedures.

In the future, we plan to run all Diffie-Hellman examples using the Pi-calculus specification of ProVerif in order to directly compare it with DH-ProVerif on real scale protocols. We also would like to continue our analysis in a fair way as the authors of [CLN09] did. It would be very interesting to push further our investigations on the impact of different parameters on each tool (such as the number of participants or the length of each protocol). Finally, in [CvDP09] X. Chen et al. proposed an improve algorithm of XOR-ProVerif. We plan to compare this new version with the one from [KT09,KT08] to measure these improvements. Protocols using elliptic curve cryptography are becoming more and more important and it would be great to analyse them. However, for the time being, none of these tools are able to support such complex algebraic properties.

Acknowledgments: We deeply thank all the tools authors for their helpful advises.

References

[80299] IEEE 802.11 Local and Metropolitan Area Networks: Wireless LAN Medium Acess Control (MAC) and Physical (PHY) Specifications, 1999.

[ABB+02] Alessandro Armando, David A. Basin, Mehdi Bouallagui, Yannick Chevalier, Luca Compagna, Sebastian Mödersheim, Michaël Rusinowitch, Mathieu Turuani, Luca Viganò, and Laurent Vigneron. The aviss security protocol analysis tool. In *CAV '02: Proceedings of the 14th International Conference on Computer Aided Verification*, pages 349–353, London, UK, 2002. Springer-Verlag.

[ABB+05] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. H. Drielsma, P.-C. Heám, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, Michael R., J. Santiago, M. Turuani, L. Viganò, and L. Vigneron. The AVISPA tool for the automated validation of internet security protocols and applications. In *Proc. of CAV'2005*, LNCS 3576, pages 281–285. Springer, 2005.

[AC05] A. Armando and L. Compagna. An optimized intruder model for SAT-based model-checking of security protocols. In A. Armando and L. Viganò, editors, *ENTCS*, volume 125, pages 91–108. Elsevier Science Publishers, March 2005.

[AST00] G. Ateniese, M. Steiner, and G. Tsudik. New multiparty authentication services and key agreement protocols. *IEEE Journal of Selected Areas in Communications*, 18(4):628–639, 2000.

[BAN90] Michael Burrows, Martin Abadi, and Roger Needham. A logic of authentication. *ACM TRANSACTIONS ON COMPUTER SYSTEMS*, 8:18–36, 1990.

[BB02] Michele Boreale and Maria Grazia Buscemi. Experimenting with sta, a tool for automatic analysis of security protocols. In *Proceedings of the 2002 ACM Symposium on Applied Computing*, SAC '02, pages 281–285, New York, NY, USA, 2002. ACM.

[Bel11] Steven M. Bellovin. Frank Miller: Inventor of the one-time pad. *Cryptologia*, 35(3):203–222, July 2011. An earlier version is available as technical report CUCS-009-11.

[BHK04] Y. Boichut, P.-C. Héam, and O. Kouchnarenko. TA4SP, 2004. Produit logiciel. TA4SP est un outil de validation de protocoles de sécurité. Grâce à une technique d'approximation appliquée sur le problème d'atteignabilité en réécriture, TA4SP peut prouver qu'une propriété de secret est inviolée pour un nombre de sessions non-borné en sur-approximant la connaissance atteignable de l'intrus. L'outil peut également montrer qu'une propriété est violée en sous-approximant la connaissance de l'intrus. Une démo de l'outil est disponible à l'adresse : t http://lifc.univ-fcomte.fr/ boichut/outil/ta4sp.php.

[BHKO04] Y. Boichut, P.-C. Héam, O. Kouchnarenko, and F. Oehl. Improvements on the Genet and Klay technique to automatically verify security protocols. In *Proc. AVIS'04*, April 2004.

[BJL+10] Thomas Baigneres, Pascal Junod, Yi Lu, Jean Monnerat, and Serge Vaudenay. *A Classical Introduction to Cryptography Exercise Book*. Springer Publishing Company, Incorporated, 1st edition, 2010.

[Bla01] B. Blanchet. An efficient cryptographic protocol verifier based on prolog rules. In *Proc. CSFW'01*, pages 82–96. IEEE Comp. Soc. Press, 2001.

[Bla04] B. Blanchet. *Cryptographic Protocol Verifier User Manual*, 2004.

[BLP03] L. Bozga, Y. Lakhnech, and M. Perin. HERMES: An Automatic Tool for Verification of Secrecy in Security Protocols. In *Computer Aided Verification*, 2003.

[BM94] Colin Boyd and Wenbo Mao. On a limitation of ban logic. In Tor Helleseth, editor, *Advances in Cryptology, EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 240–247. Springer Berlin Heidelberg, 1994.

[BMV03] D. Basin, S. Mödersheim, and L. Viganò. An On-The-Fly Model-Checker for Security Protocol Analysis. In *Proc. of ESORICS'03*, volume 2808 of *LNCS*, pages 253–270. Springer-Verlag, 2003.

[BMV05] David A. Basin, Sebastian Mödersheim, and Luca Viganò. Ofmc: A symbolic model checker for security protocols. *Int. J. Inf. Sec.*, 4(3):181–208, 2005.

[BO97] J. Bull and D. J. Otway. The authentication protocol. Technical Report DRA/CIS3/PROJ/CORBA/SC/1/CSM/436-04/03, Defence Research Agency, 1997.

[CBD+07] M. Cheminod, I. Cibrario Bertolotti, L. Durante, R. Sisto, and A. Valenzano. Experimental comparison of automatic tools for the formal analysis of cryptographic protocols. In *DepCoS-RELCOMEX 2007, June 14-16, 2007, Szklarska Poreba, Poland*, pages 153–160. IEEE Computer Society, 2007.

[CDL06] V. Cortier, S. Delaune, and P. Lafourcade. A survey of algebraic properties used in cryptographic protocols. *Journal of Computer Security*, 14(1):1–43, 2006.

[CE02] R. Corin and S. Etalle. An improved constraint-based system for the verification of security protocols. In *Proc. 9th International Static Analysis Symposium (SAS)*, volume 2477 of *LNCS*, pages 326–341. Springer, Sep 2002.

[Cer01] Iliano Cervesato. The dolev-yao intruder is the most powerful attacker. In *Proceedings of the Sixteenth Annual Symposium on Logic in Computer Science — LICS'01*, pages 16–19. IEEE Computer Society Press. Short, 2001.

[CLN09] C.J.F. Cremers, Pascal Lafourcade, and Philippe Nadeau. Comparing state spaces in automatic protocol analysis. In *Formal to Practical Security*, volume 5458/2009 of *LNCS*, pages 70–94. Springer Berlin / Heidelberg, 2009.

[Cre08] C.J.F. Cremers. The Scyther Tool: Verification, falsification, and analysis of security protocols. In *Computer Aided Verification, 20th International Conference, CAV*, volume 5123/2008 of *LNCS*, pages 414–418. Springer, 2008.

[Cre11] Cas Cremers. Key exchange in ipsec revisited: Formal analysis of ikev1 and ikev2. In *Proceedings of the 16th European Conference on Research in Computer Security*, ESORICS'11, pages 315–334, Berlin, Heidelberg, 2011. Springer-Verlag.

[CvDP09] Xihui Chen, Ton van Deursen, and Jun Pang. Improving automatic verification of security protocols with xor. In Karin Breitman and Ana Cavalcanti, editors, *Formal Methods and Software Engineering*, volume 5885 of *Lecture Notes in Computer Science*, pages 107–126. Springer Berlin Heidelberg, 2009.

[DH76] W. Diffie and M. Hellman. New directions in cryptography. *IEEE Transactions on Information Society*, 22(6):644–654, 1976.

[DSHJ10] Nitish Dalal, Jenny Shah, Khushboo Hisaria, and Devesh Jinwala. A comparative analysis of tools for verification of security protocols. *IJCNS*, 3(10):779–787, 2010.

[DSV03] Luca Durante, Riccardo Sisto, and Adriano Valenzano. Automatic testing equivalence verification of spi calculus specifications. *ACM Trans. Softw. Eng. Methodol.*, 12(2):222–284, April 2003.

[DY81] D. Dolev and A. C. Yao. On the security of public key protocols. In *Proceedings of the 22Nd Annual Symposium on Foundations of Computer Science*, SFCS '81, pages 350–357, Washington, DC, USA, 1981. IEEE Computer Society.

[ECEM96] Clavel Eker, M. Clavel, S. Eker, and J. Meseguer. Principles of maude, 1996.

[EMM07] Santiago Escobar, Catherine Meadows, and José Meseguer. Maude-npa: Cryptographic protocol analysis modulo equational properties. In Alessandro Aldini, Gilles Barthe, and Roberto Gorrieri, editors, *FOSAD*, volume 5705 of *Lecture Notes in Computer Science*, pages 1–50. Springer, 2007.

[Gon89] L. Gong. Using one-way functions for authentication. *SIGCOMM Computer Communication*, 19(5):8–11, 1989.

[HS06] M. Hussain and D. Seret. A comparative study of security protocols validation tools: HERMES vs. AVISPA. In *InProc. ICACT'06*, volume 1, pages 303– 308, 2006.

[HTWSCK08] LIAW Horng-Twu, JUANG Wen-Shenq, and LIN Chi-Kai. An electronic online bidding auction protocol with both security and efficiency. *Applied mathematics and computation*, 174:1487–1497, 2008.

[Kau05] C. Kaufman. Internet key exchange protocol version 2 (ikev2). IETF RFC 4306, December 2005.

[KHN+14] C. Kaufman, P. Hoffman, Y. Nir, P. Eronen, and T. Kivinen. Internet key exchange protocol version 2 (ikev2). IETF RFC 7296, October 2014.

[KT08] Ralf Küsters and Tomasz Truderung. Reducing protocol analysis with xor to the xor-free case in the horn theory based approach. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security*, pages 129–138. ACM, 2008.

[KT09] R. Küsters and T. Truderung. Using ProVerif to Analyze Protocols with Diffie-Hellman Exponentiation. In *Proceedings of the 22nd Computer Security Foundations Symposium (CSF)*, pages 157–171. IEEE Computer Society, 2009.

[Low96] G. Lowe. Breaking and fixing the needham-schroeder public-key protocol using fdr. In *TACAs '96: Proceedings of the Second International Workshop on Tools and Algorithms for Construction and Analysis of Systems*, pages 147–166, London, UK, 1996. Springer-Verlag.

[Low98] G. Lowe. Casper: a compiler for the analysis of security protocols. *J. Comput. Secur.*, 6(1-2):53–84, 1998.

[LR97a] G. Lowe and A. W. Roscoe. Using CSP to detect errors in the TMN protocol. *IEEE Transactions on Software Engineering*, 23(10):659–669, 1997.

[LR97b] G. Lowe and B. Roscoe. Using CSP to detect errors in the TMN protocol. *IEEE Trans. Softw. Eng.*, 23(10):659–669, 1997.

[LTV10] Pascal Lafourcade, Vanessa Terrade, and Sylvain Vigier. Comparison of cryptographic verification tools dealing with algebraic properties. In Pierpaolo Degano and JoshuaD. Guttman, editors, *Formal Aspects in Security and Trust*, volume 5983 of *Lecture Notes in Computer Science*, pages 173–185. Springer Berlin Heidelberg, 2010.

[Mea96a] C. Meadows. Language generation and verification in the NRL protocol analyzer. In *Proc. CSFW'96*, pages 48–62. IEEE Comp. Soc. Press, 1996.

[Mea96b] Catherine Meadows. Analyzing the needham-schroeder public-key protocol: A comparison of two approaches. In *Computer Security - ESORICS 96, 4th European Symposium on Research in Computer Security*, volume 1146 of *LNCS*, pages 351–364. Springer, 1996.

[MMS97] J.C. Mitchell, M. Mitchell, and U. Stern. Automated analysis of cryptographic protocols using Murphi. In *IEEE Symposium on Security and Privacy*, May 1997.

[MSCB13] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. The tamarin prover for the symbolic analysis of security protocols. In Natasha Sharygina and Helmut Veith, editors, *Computer Aided Verification*, volume 8044 of *Lecture Notes in Computer Science*, pages 696–701. Springer Berlin Heidelberg, 2013.

[NS78] R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communication of the ACM*, 21(12):993–999, 1978.

[PBP+10] Reema Patel, Bhavesh Borisaniya, Avi Patel, Dhiren R. Patel, Muttukrishnan Rajarajan, and Andrea Zisman. Comparative analysis of formal model checking tools for security protocol verification. In Natarajan Meghanathan, Selma Boumerdassi, Nabendu Chaki, and Dhinaharan Nagamalai, editors, *Recent Trends in Network Security and Applications - Third International Conference, CNSA 2010, Chennai, India, July 23-25, 2010. Proceedings*, volume 89 of *Communications in Computer and Information Science*, pages 152–163. Springer, 2010.

[PL] M. Puys and P. Lafourcade. Protocol tool comparison test archive. `http://www-verimag.imag.fr/~puys/assets/files/LP15_sources.tar.gz`.

[Ros94] A. W. Roscoe. *Model-checking CSP*. Prentice Hall, 1994.

[Ros95] A. W. Roscoe. Modelling and verifying key-exchange protocols using CSP and FDR. In *IEEE Symposium on Foundations of Secure Systems*, 1995.

[RS98] Peter Y. A. Ryan and Steve A. Schneider. An attack on a recursive authentication protocol. a cautionary tale. *Inf. Process. Lett.*, 65(1):7–10, 1998.

[SBP01] D. Song, S. Berezin, and A. Perrig. Athena: A novel approach to efficient automatic security protocol analysis. *Journal of Computer Security*, 9(1/2):47–74, 2001.

[Sch96] B. Schneier. *Applied Cryptography*. Wiley, second edition, 1996.

[SMCB12] B. Schmidt, S. Meier, C. Cremers, and D. Basin. Automated analysis of diffie-hellman protocols and advanced security properties. In *Computer Security Foundations Symposium (CSF), 2012 IEEE 25th*, pages 78–94, June 2012.

[TMN89] M. Tatebayashi, N. Matsuzaki, and D. B. Newman. Key distribution protocol for digital mobile communication systems. In *Proc. 9th Annual International Cryptology Conference (CRYPTO'89)*, volume 435 of *LNCS*, pages 324–333, Santa Barbara (California, USA), 1989. Springer-Verlag.

[Tur06] Mathieu Turuani. The CL-Atse Protocol Analyser. In Frank Pfenning, editor, *17th International Conference on Term Rewriting and Applications - RTA 2006 Lecture Notes in Computer Science*, volume 4098 of *LNCS*, pages 277–286. Springer, August 2006.

[Vau05] Serge Vaudenay. *A Classical Introduction to Cryptography: Applications for Communications Security*. 2005.

[Vig06] L. Viganò. Automated security protocol analysis with the AVISPA tool. *ENTCS*, 155:61–86, 2006.

[YL06] T. Ylonen and C. Lonvick. The secure shell (ssh) transport layer protocol. IETF RFC 4253, January 2006.