Formal Methods for Verifying the Singularization Strategy: a Work-in-Progress on a bilateral project

Philipp Schlehuber-Caissier [1] Chrystel Gaber [2]
Jean-Philippe Wary [2] Natalia Kushik[1]

SAMOVAR, Télécom SudParis,
Institut Polytechnique de Paris
Palaiseau, France

Orange Innovation
Paris, France

27.05.1025

Introduction to Singularization

Singularization is a framework seeking generalize moving target defense (MTD) strategies.

Introduction to Singularization

Singularization is a framework seeking generalize moving target defense (MTD) strategies.

MTD is mostly used in the context of network security (IP randomization, Configuration shuffling).

Introduction to Singularization

Singularization is a framework seeking generalize moving target defense (MTD) strategies.

MTD is mostly used in the context of network security (IP randomization, Configuration shuffling).

Singularization is a more general concept, seeking to improve security:

- Reduce vulnerability of encryption of embedded systems: Improving DES
- Improve security of legacy Sim Cards

Introduction to Singularization

Singularization is a framework seeking generalize moving target defense (MTD) strategies.

MTD is mostly used in the context of network security (IP randomization, Configuration shuffling).

Singularization is a more general concept, seeking to improve security:

- Reduce vulnerability of encryption of embedded systems: Improving DES
- Improve security of legacy Sim Cards

In both cases the system is strengthened by introducing a larger variety in the system to attack and therefore making it harder to pick the correct attack, much like in MTD.

Improving DES: A case study

(Single round) DES is known to be weak against differential cryptanalysis.

Improving DES: A case study

(Single round) DES is known to be weak against differential cryptanalysis.

This is largely due to the fact that the S-boxes are fixed and the algorithm is *static*.

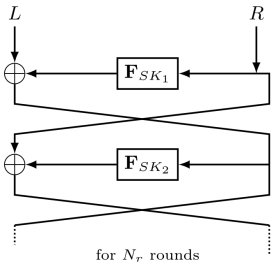
Improving DES: A case study

(Single round) DES is known to be weak against differential cryptanalysis.

This is largely due to the fact that the S-boxes are fixed and the algorithm is *static*.

Singularization tackles this problem by introducing a set of pseudo-random functions (PRF), which are chosen at random for each round.

By enlarging the set of PRFs, there can be as many variations as there are (56bit) keys, mitigating differential attacks:
[1]: Singularization: A New Approach to Designing Block Ciphers for Resource-Constrained Devices, G. Macario-Rat, M. Plesa



DES from a formal point of view

Most works applying formal methods to DES focus on functional properties, such as Livenes, and often for actual implementations: [2, 3, 4, 5].

DES from a formal point of view

Most works applying formal methods to DES focus on functional properties, such as Livenes, and often for actual implementations: [2, 3, 4, 5].

Automata-based approaches do not lend themselves very well to answer secrecy related questions, like “Is the system vulnerable to cryptanalysis?”

DES from a formal point of view

Most works applying formal methods to DES focus on functional properties, such as Livenes, and often for actual implementations: [2, 3, 4, 5].

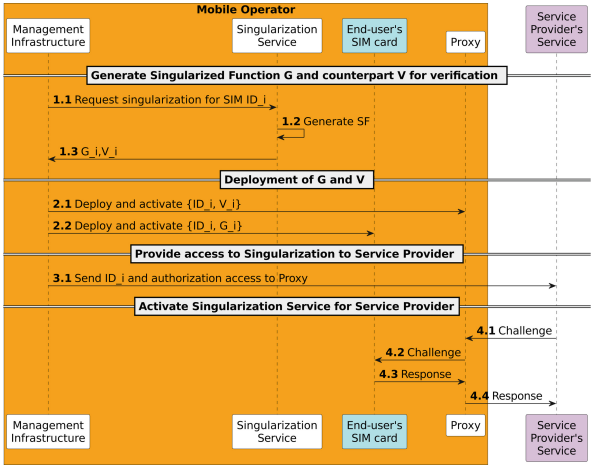
Automata-based approaches do not lend themselves very well to answer secrecy related questions, like “Is the system vulnerable to cryptanalysis?”

Part of the problem is the state-space explosion, the other part is due to the problem of translating the question into graph properties.

Using singularization within an application

In [6], a framework of how to use singularization within a protocol is presented.

Position Paper: Strengthening Applets on Legacy SIM Cards with Singularization, a New Moving Target Defense Strategy, Gaber *et al.*



Verification of Concurrent and Distributed Systems

Analyzing and verifying distributed systems is a well known problem and has been extensively studied for several decades.

Verification of Concurrent and Distributed Systems

Analyzing and verifying distributed systems is a well known problem and has been extensively studied for several decades.

"Recent" research focuses on the automatization and formalization of the verification process; Sometimes with remarkable success.

Verification of Concurrent and Distributed Systems

Analyzing and verifying distributed systems is a well known problem and has been extensively studied for several decades.

"Recent" research focuses on the automatization and formalization of the verification process; Sometimes with remarkable success.

In this talk we will paint a broad overview of techniques and tools applicable to such problems.

Where formal methods have been used successfully

Transport Layer Security (TLS) 1.2

Long list of security issues

- No forward secrecy
- Downgrade Attacks
- Insecure Renegotiation
- Many more

Where formal methods have been used successfully

Transport Layer Security (TLS) 1.2

Long list of security issues

- No forward secrecy
- Downgrade Attacks
- Insecure Renegotiation
- Many more

Some of them were implementation error, (most) are due to faulty or vague specifications. (And were therefore **avoidable**)

Where formal methods have been used successfully

Transport Layer Security (TLS) 1.2

Long list of security issues

- No forward secrecy
- Downgrade Attacks
- Insecure Renegotiation
- Many more

Some of them were implementation error, (most) are due to faulty or vague specifications. (And were therefore **avoidable**)

Working on TLS 1.3

Learning from past error, the IETF called on the academic community for a joint effort to verify TLS 1.3 prior to release.

Where formal methods have been used successfully

Transport Layer Security (TLS) 1.2

Long list of security issues

- No forward secrecy
- Downgrade Attacks
- Insecure Renegotiation
- Many more

Some of them were implementation error, (most) are due to faulty or vague specifications. (And were therefore **avoidable**)

Working on TLS 1.3

Learning from past error, the IETF called on the academic community for a joint effort to verify TLS 1.3 prior to release.

This initiative turned out to be very successful!

(I) Symbolic Analysis - Verifying TLS 1.3 - ProVerif

ProVerif ([7]) is one of the oldest tools for automated reasoning about security properties, with its first release dating back to 2002.

(I) Symbolic Analysis - Verifying TLS 1.3 - ProVerif

ProVerif ([7]) is one of the oldest tools for automated reasoning about security properties, with its first release dating back to 2002.

- ProVerif is based on the Dolev-Yao model
- Cryptographic primitives are blackboxes
- Unbounded number of sessions and processes

(I) Symbolic Analysis - Verifying TLS 1.3 - ProVerif

ProVerif ([7]) is one of the oldest tools for automated reasoning about security properties, with its first release dating back to 2002.

- ProVerif is based on the Dolev-Yao model
- Cryptographic primitives are blackboxes
- Unbounded number of sessions and processes

- ProVerif language is based on Pi calculus
- It supports the definitions of types, functions, and equations
- Security properties are then reduced to consistency checks and resolution / reachability

ProVerif Example - Denning-Sacco

Definition of primitives

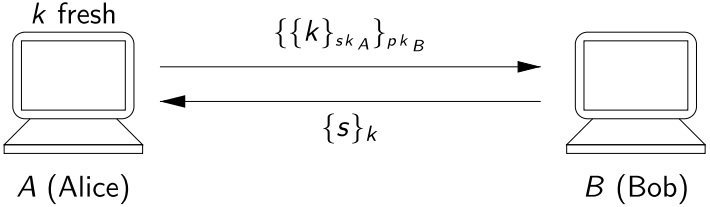
```
free c.
(* Public key cryptography *)
fun pk/1.
private fun sk/1.
(* just encryption, no signing *)
fun encrypt/2.
reduc decrypt(encrypt(x,pk(y)),sk(y)) = x.
(* Symmetric key cryptography *)
fun symcrypt/2.
reduc symdecrypt(symcrypt(z,j),j) = z.
```

Usage within the protocol

```
let processB =
    in(c,m1);
    let (na,Y) = decrypt(m1, sk(B)) in
    new Nb;
    out(c, encrypt((na, Nb), pk(Y)));
    in(c,m3);
    let (=Nb) = decrypt(m3, sk(B)) in
    (*...*)
```

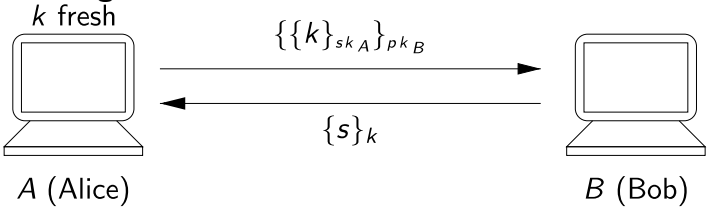
ProVerif Example - Denning-Sacco cont'd

Denning-Sacco Intent

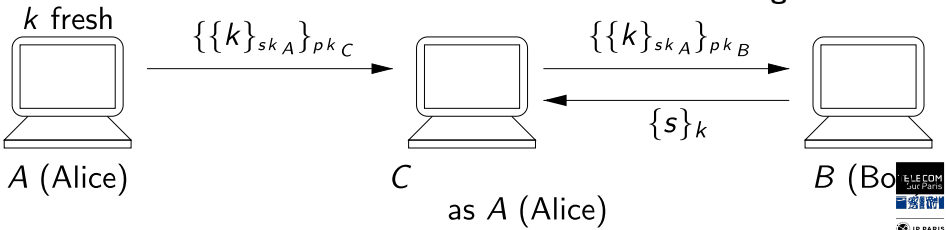


ProVerif Example - Denning-Sacco cont'd

Denning-Sacco Intent



Denning-Sacco Fail



ProVerif - Summed Up

- Expressivity of Pi calculus
- Good tooling and trace extraction
- TLS models available and kept up to date

ProVerif - Summed Up

- ⊕ Expressivity of Pi calculus
- ⊕ Good tooling and trace extraction
- ⊕ TLS models available and kept up to date

- ⊖ Unbounded number of sessions and processes can lead to undecidability
- ⊖ Blackbox cryptography
- ⊖ No inherent support for probabilistic reasoning - A property is satisfied or not (or it is undecidable)

ProVerif - Summed Up

- ⊕ Expressivity of Pi calculus
- ⊕ Good tooling and trace extraction
- ⊕ TLS models available and kept up to date

- ⊖ Unbounded number of sessions and processes can lead to undecidability
- ⊖ Blackbox cryptography
- ⊖ No inherent support for probabilistic reasoning - A property is satisfied or not (or it is undecidable)

One step further: Proving protocols written in (a large fragment of) Rust using hax.

(I) Symbolic Analysis - Verifying TLS 1.3 - Tamarin Prover

Tamarin Prover ([8]) is a more recent tool, which has been developed since 2012.

(I) Symbolic Analysis - Verifying TLS 1.3 - Tamarin Prover

Tamarin Prover ([8]) is a more recent tool, which has been developed since 2012.

Much like ProVerif it also performs a symbolic analysis, however they introduced the concept of interactive theorem proving.
Interactive theorem proving allows to insert dedicated lemmas "unblocking" the prover, however this idea has since been largely picked up by ProVerif as well.

Tamarin is based on multi-set rewriting, which is sometimes better suited to model stateful protocols.

(I) Symbolic Analysis - Verifying TLS 1.3 - Tamarin Prover

Tamarin Prover ([8]) is a more recent tool, which has been developed since 2012.

Much like ProVerif it also performs a symbolic analysis, however they introduced the concept of interactive theorem proving.
Interactive theorem proving allows to insert dedicated lemmas "unblocking" the prover, however this idea has since been largely picked up by ProVerif as well.

Tamarin is based on multi-set rewriting, which is sometimes better suited to model stateful protocols.

For our project, tamarin seems to play a very similar role to ProVerif.

(II) Computational Analysis - Verifying TLS 1.3 - CryptoVerif

Security proven in the Dolev-Yao model is not necessarily implying security in the computational model, i.e. the implementation.

CryptoVerif ([9]) is a tool for computational cryptography, which is based on the sequence of games approach.

(II) Computational Analysis - Verifying TLS 1.3 - CryptoVerif

Security proven in the Dolev-Yao model is not necessarily implying security in the computational model, i.e. the implementation.

CryptoVerif ([9]) is a tool for computational cryptography, which is based on the sequence of games approach.

It allows to reason about the security of protocols in a probabilistic setting, i.e., it can reason about the probability of an attack succeeding in the presence of a polynomial-time attacker.

(II) Computational Analysis - Verifying TLS 1.3 - CryptoVerif

Security proven in the Dolev-Yao model is not necessarily implying security in the computational model, i.e. the implementation.

CryptoVerif ([9]) is a tool for computational cryptography, which is based on the sequence of games approach.

It allows to reason about the security of protocols in a probabilistic setting, i.e., it can reason about the probability of an attack succeeding in the presence of a polynomial-time attacker.

This allows to reason about properties like one-wayness of an encryption scheme:
Computing the probability of an attacker being able to invert the encryption function without the key.

(II) Computational Analysis - Verifying TLS 1.3 - CryptoVerif

Security proven in the Dolev-Yao model is not necessarily implying security in the computational model, i.e. the implementation.

CryptoVerif ([9]) is a tool for computational cryptography, which is based on the sequence of games approach.

It allows to reason about the security of protocols in a probabilistic setting, i.e., it can reason about the probability of an attack succeeding in the presence of a polynomial-time attacker.

This allows to reason about properties like one-wayness of an encryption scheme:
Computing the probability of an attacker being able to invert the encryption function without the key.

The tool has been used to verify TLS 1.3, the model is available and maintained.

CryptoVerif - Summed Up

- Probabilistic reasoning
- Takes into account the computational model
- Good tooling and trace extraction
- Decomposition into games makes it extensibly
- Oracles and other functions can be user-defined

CryptoVerif - Summed Up

- ⊕ Probabilistic reasoning
- ⊕ Takes into account the computational model
- ⊕ Good tooling and trace extraction
- ⊕ Decomposition into games makes it extensibly
- ⊕ Oracles and other functions can be user-defined

Approach seems to be well suited for singularization.

(III) Automata Theoretic Approaches - Analysing MTD Strategies

The works cited so far focus on "static" protocols, hidding to some extent the MDP strategy.

(III) Automata Theoretic Approaches - Analysing MTD Strategies

The works cited so far focus on "static" protocols, hidding to some extent the MDP strategy.

Recently a new line of work has emerged, focusing on how and when to move the target . Since moving the target increases the safety of the system, it might also come at a non-negligible cost, e.g. a loss of Quality of Service (QoS).

(III) Automata Theoretic Approaches - Analysing MTD Strategies

The works cited so far focus on "static" protocols, hidding to some extent the MDP strategy.

Recently a new line of work has emerged, focusing on how and when to move the target . Since moving the target increases the safety of the system, it might also come at a non-negligible cost, e.g. a loss of Quality of Service (QoS).

The goal of these works is to find the optimal strategy for moving the target, given some (partial) knowledge about the attacker.

(III) Automata Theoretic Approaches - Analysing MTD Strategies

The works cited so far focus on "static" protocols, hidding to some extent the MDP strategy.

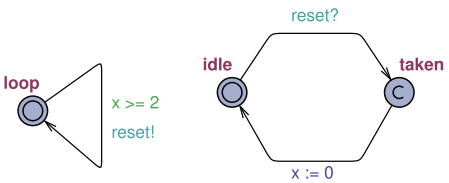
Recently a new line of work has emerged, focusing on how and when to move the target . Since moving the target increases the safety of the system, it might also come at a non-negligible cost, e.g. a loss of Quality of Service (QoS).

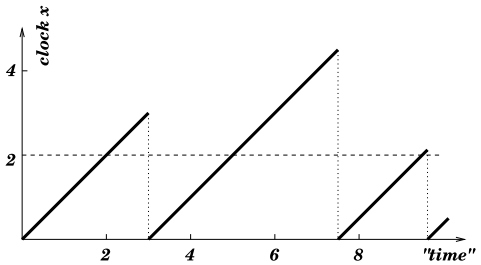
The goal of these works is to find the optimal strategy for moving the target, given some (partial) knowledge about the attacker.

Priced Timed Markov Decision Processes (PTMDP)

(III) Automata Theoretic Approaches - UPPAAL SMC & Stratego

A quick introduction - Timed Automata!
Timed automata are a well known formalism for modeling real-time systems.
They augment finite state machines with clocks.
UPPAAL adds many common features like synchronization, data variables, and more.





reachability, deadlock-freedom, fastest path, fragment of CTL
https://uppaal.org/texts/small_tutorial.pdf

(III) Automata Theoretic Approaches - UPPAAL SMC & Stratego

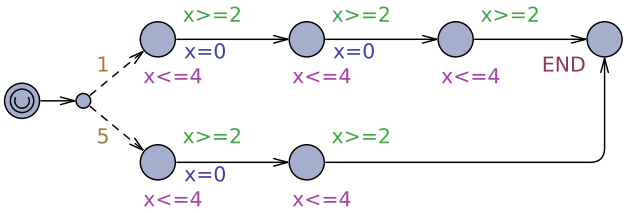
Timed Automata can be very useful, however, reasoning about the best case / worst case scenario is very restrictive, especially in the context of MTD.

(III) Automata Theoretic Approaches - UPPAAL SMC & Stratego

Timed Automata can be very useful, however, reasoning about the best case / worst case scenario is very restrictive, especially in the context of MTD.

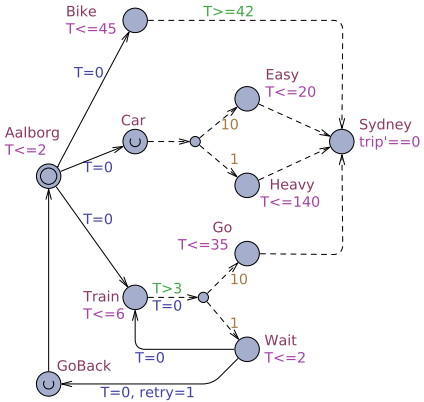
Introducing **SMC**: Statistical Model Checking for Timed Automata.

UPPAAL SMC allows to reason about the probability of a certain path being taken, with guarantees on the obtained distribution.
`https://uppaal.org/texts/uppaal-smc-tutorial.pdf`



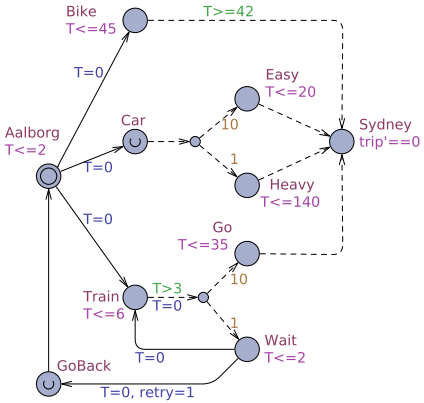
(III) Automata Theoretic Approaches - UPPAAL SMC & Stratego

Probabilistic Timed Automata allow to reason about distributions, but they have no notion of a strategy, all choices are probabilistic and uncontrolled.



(III) Automata Theoretic Approaches - UPPAAL SMC & Stratego

Probabilistic Timed Automata allow to reason about distributions, but they have no notion of a strategy, all choices are probabilistic and uncontrolled.



Allows to synthesize strategies minimizing the expected cost under (probabilistic) safety constraints.

- **GoSafe**: Reach Sydney in less than 60 minutes
- **GoFast**: Minimize the expected travel time
- **GoFastAndSafe**: Minimize and guarantee

Uppaal SMC & Stratego

- ⊕ Probabilistic reasoning
- ⊕ Synthesis of strategies
- ⊕ Allows to reason about cost and benefits
- ⊕ Principled way to work with Attack Trees
- ⊕ Good tooling and trace extraction

- ⊖ "Model-based"-reasoning, no direct way to translate protocols
- ⊖ Secrecy and authentication properties are harder to model
- ⊖ Abstract model, no support for cryptographic primitives

Thank you for your attention!
I'm happy to take your questions!

References I

"Singularization: A New Approach to Designing Block Ciphers for Resource-Constrained Devices," in *Lecture Notes in Computer Science*. Cham: Springer Nature Switzerland, 2025, pp. 155–167, iSSN: 0302-9743, 1611-3349. [Online]. Available: https://link.springer.com/10.1007/978-3-031-85593-1_10

A. Bitat and S. Merniz, "Towards formal verification of cryptographic circuits: A functional approach," in *2018 3rd International Conference on Pattern Analysis and Intelligent Systems (PAIS), Tebessa, Algeria, October 24-25, 2018*, M. Amroune, M. Derdour, and A. Ahmim, Eds. IEEE, 2018, pp. 1–6. [Online]. Available: https://doi.org/10.1109/PAIS.2018.8598527

References II

——, "Formal verification of cryptographic circuits: A semi-automatic functional approach," in *Proceedings of the 2nd International Conference on Networking, Information Systems & Security, NISS 2019, Rabat, Morocco, 27-29 March, 2019*, B. Abouelmajd, M. B. Ahmed, A. A. Boudhir, and H. E. Ghazi, Eds. ACM, 2019, pp. 35:1–35:6. [Online]. Available: https://doi.org/10.1145/3320326.3320367

D. Borrione, M. Boubekeur, L. Mounier, M. Renaudin, and A. Siriani, "Validation of asynchronous circuit specifications using IF/CADP," in *IFIP VLSI-SoC 2003, IFIP WG 10.5 International Conference on Very Large Scale Integration of System-on-Chip, Darmstadt, Germany, 1-3 December 2003*, M. Glesner, R. A. da Luz Reis, H. Eveking, V. J. M. III, L. S. Indrusiak, and P. Zipf, Eds. Technische Universität Darmstadt, Insitute of Microelectronic Systems, 2003, pp. 86–91.

References III

W. Serwe, "Formal specification and verification of fully asynchronous implementations of the data encryption standard," in *Proceedings Workshop on Models for Formal Analysis of Real Systems, MARS 2015, Suva, Fiji, November 23, 2015*, ser. EPTCS, R. J. van Glabbeek, J. F. Groote, and P. Höfner, Eds., vol. 196, 2015, pp. 61–147. [Online]. Available: https://doi.org/10.4204/EPTCS.196.6

C. Gaber, G. Macario-Rat, S. David, J. Wary, and A. Cuaboz, "Position paper: Strengthening applets on legacy SIM cards with singularization, a new moving target defense strategy," in *Mobile, Secure, and Programmable Networking - 9th International Conference, MSPN 2023, Paris, France, October 26-27, 2023, Revised Selected Papers*, ser. Lecture Notes in Computer Science, S. Bouzefrane, S. Banerjee, F. Mourlin, S. Boumerdassi, and É. Renault, Eds., vol. 14482. Springer, 2023, pp. 71–74. [Online]. Available: https://doi.org/10.1007/978-3-031-52426-4_5

References IV

- "Proverif: Cryptographic protocol verifier in the formal model," https://bblanche.gitlabpages.inria.fr/proverif/, Accessed 2025.
- "Tamarin prover," https://tamarin-prover.com/, Accessed 2025.
- "Cryptoverif: Cryptographic protocol verifier in the computational model," https://bblanche.gitlabpages.inria.fr/CryptoVerif/, Accessed 2025.