Formal Cybersecurity Techniques for Cyber Physical Systems

Simone Bussa

Politecnico di Torino

Supervisors: Prof. Riccardo Sisto, Prof. Fulvio Valenza

Research Group: Netgroup

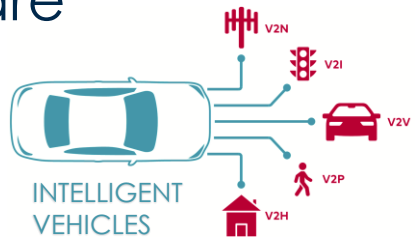
Introduction

Cyber Physical Systems (**CPSs**) = Hardware  Software

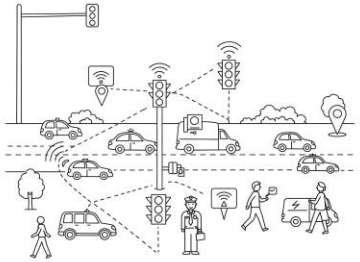
CPSs are becoming more and more **complex**:

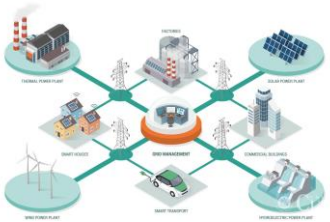
- **Heterogeneity** of components
- **Distributed** environments
- Highly **dynamic** systems
- **Safety criticality**

SOME EXAMPLES

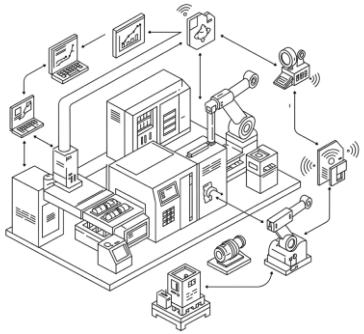


INTELLIGENT VEHICLES





IOT AND INDUSTRIAL CONTROL SYSTEMS



This complexity impacts on **cybersecurity** requirements

Human errors or cyber-attacks could create great damage • Traditional management is slow • Manual configuration does not scale

AUTOMATION IS NEEDED

PhD Topic

- **Study formal methods and techniques** that can help in the process of **automating network security management** and **security verification**, in CPSs

Automatic network security management	Automatic security verification

- Large-scale distributed networks
- Automatic tools to allocate, configure, and continuously verify network security functions
- Goal: minimize human intervention, avoid manual errors, reach optimal and formally proved solutions (correctness-by-construction approach)
- Huge obstacle: Scalability

PhD Topic

- **Study formal methods and techniques** that can help in the process of **automating network security management** and **security verification**, in CPSs

Automatic network security management	Automatic security verification

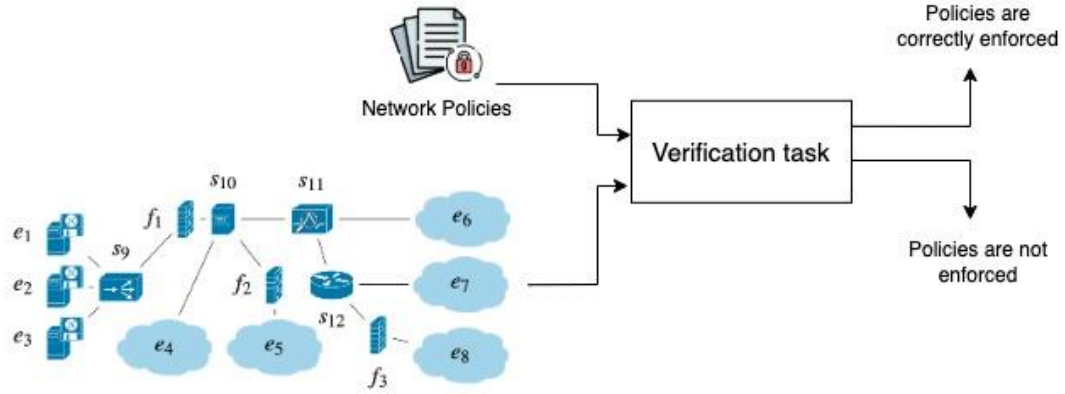
- Identify state-of-the-art protocols in the CPS field
- Verify the security of these protocols against well-defined formal requirements
- Use of formal verification
- Focus on areas with no previous formal verification work (greater impact)

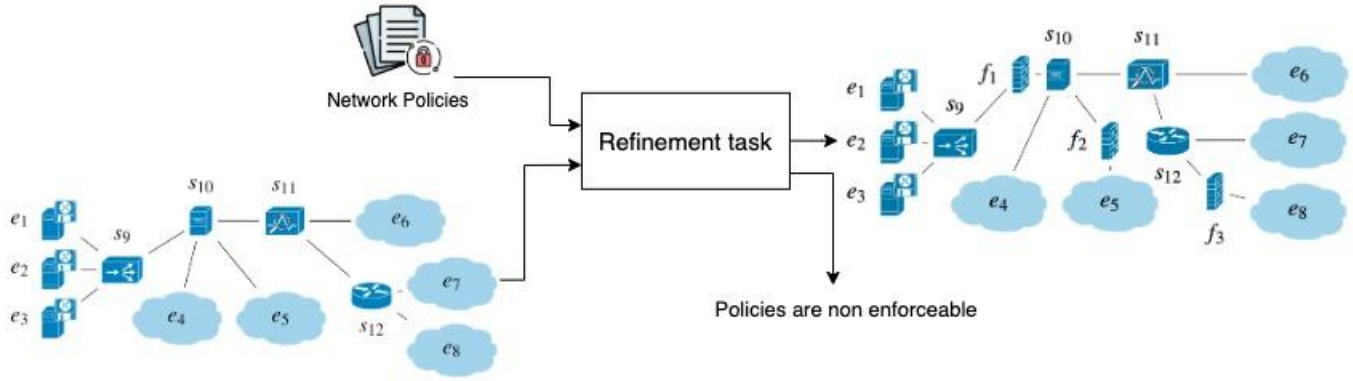
Introduction to network security configuration and management

Defining and implementing security policies (**configuration**), and continuously enforcing them during the daily operation of the network (**management**)

- **Automation** supported by SDN & NFV
- **Policy based management**:
  1. The administrator defines security policies at a high level
  2. Automated tools then write low level configurations and manage security functions

Two traditional automated operations



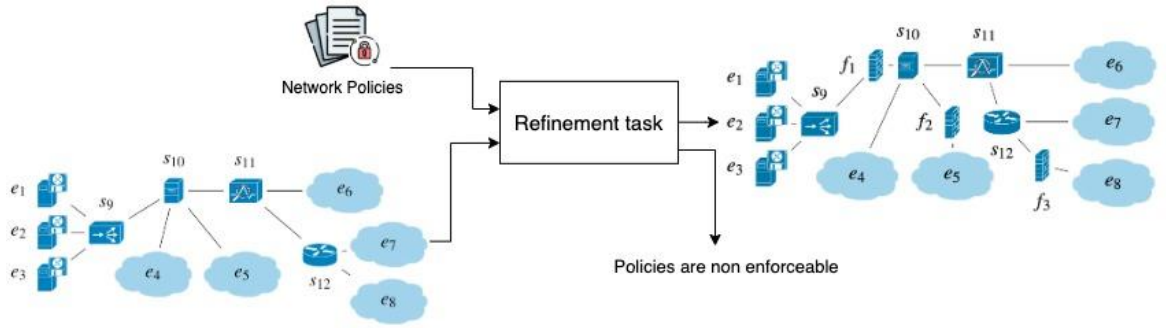


Formal traffic models for automatic security management

- **Goal**: improve efficiency of existing solutions that perform network verification and refinement
- **Limitations** of the state-of-the-art:
  - Most models only work in simple networks
  - Others have limited scalability
- **Solution**: new formal models to represent the network and its key security functions

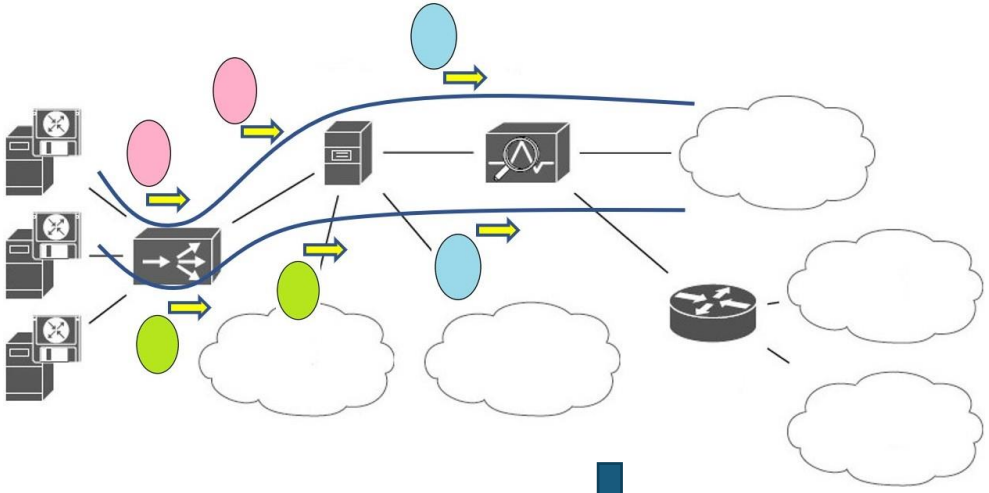
Reduce the amount of data automated tools have to deal with

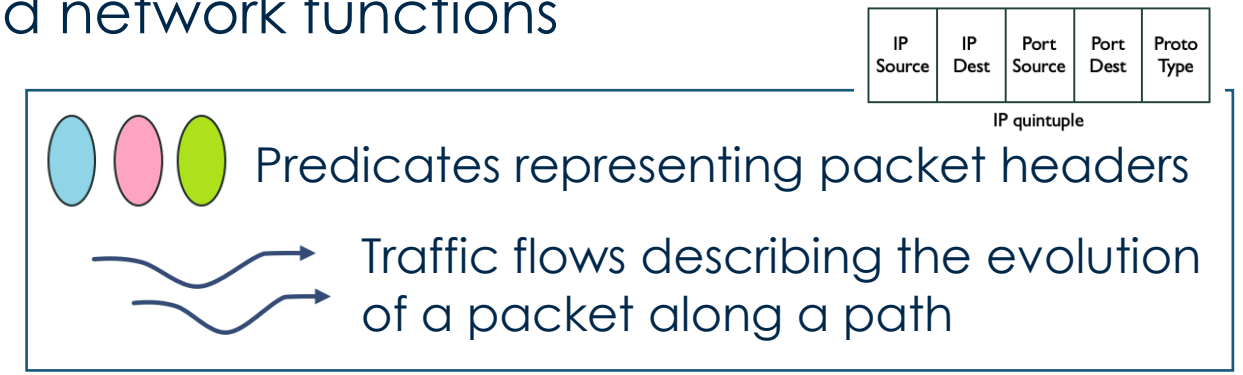
- **Case study:**
  - Automatic allocation and configuration of packet filters based on security policies
  - Corresponding security policy verification
  - Formalized as SMT / MaxSMT problem (correctness-by-construction)
  - Computationally intensive)

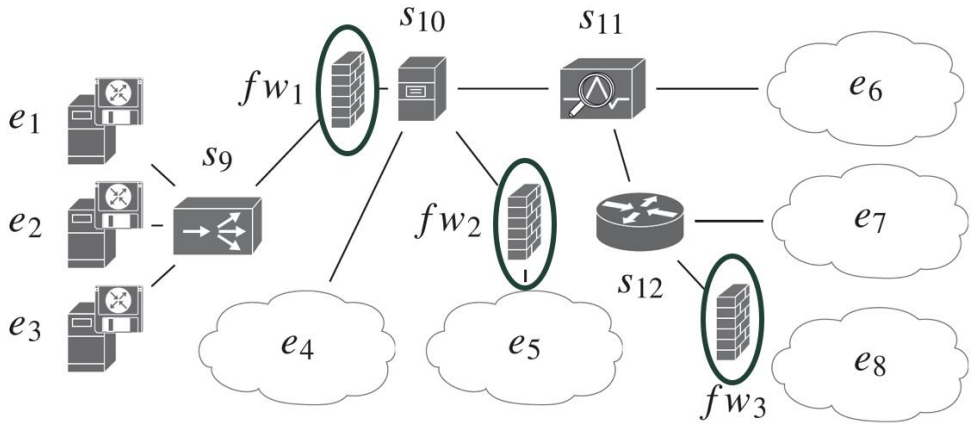


Formal traffic models for automatic security management

- Efficient modelling of network traffic and network functions



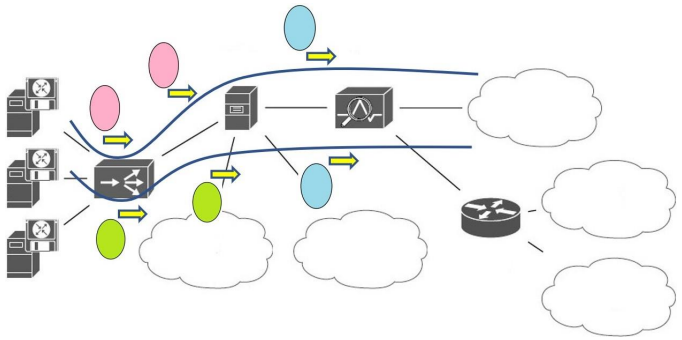




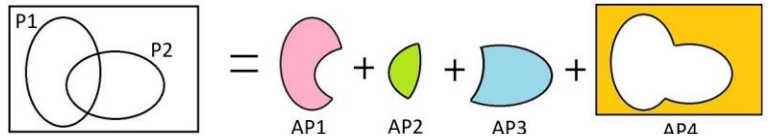
- By acting on the representation of traffic flows,
  it is possible to influence the performance of the MaxSMT problem

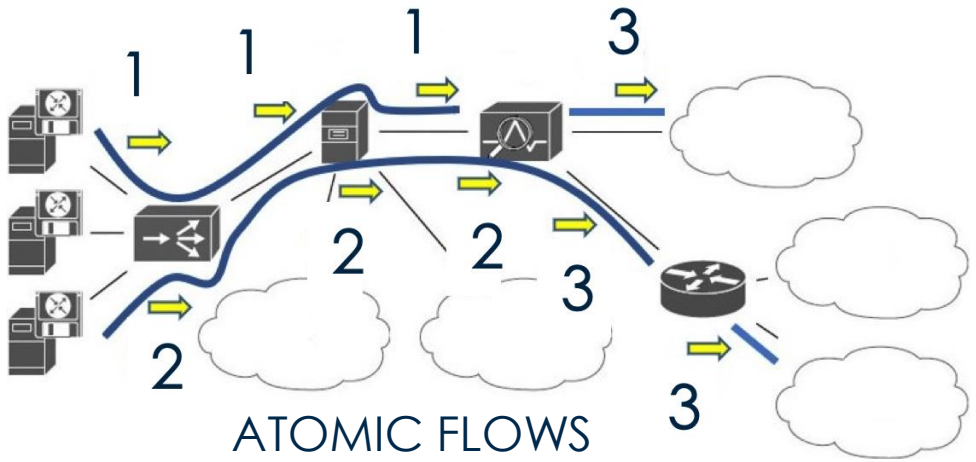
Two modelling approaches

- We propose two different (alternative) ways to model traffic flows



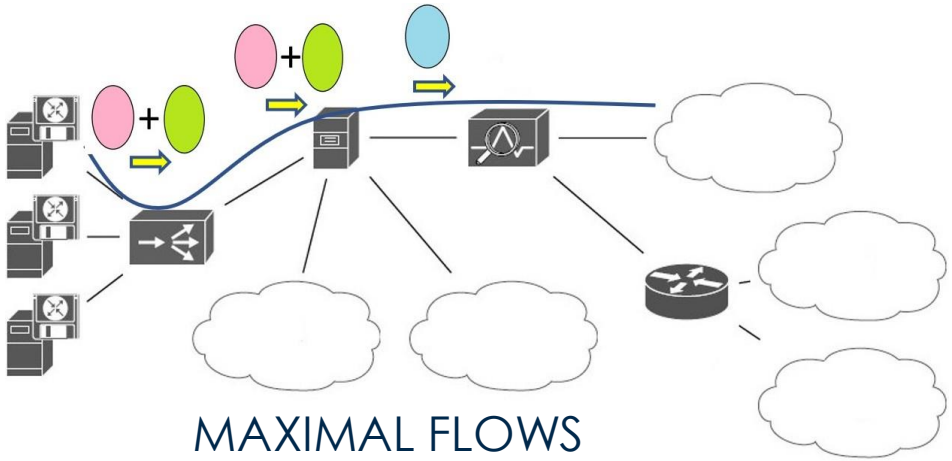
ATOMIC FLOWS: simplify Predicate representation





ATOMIC FLOWS

MAXIMAL FLOWS: aggregate flows

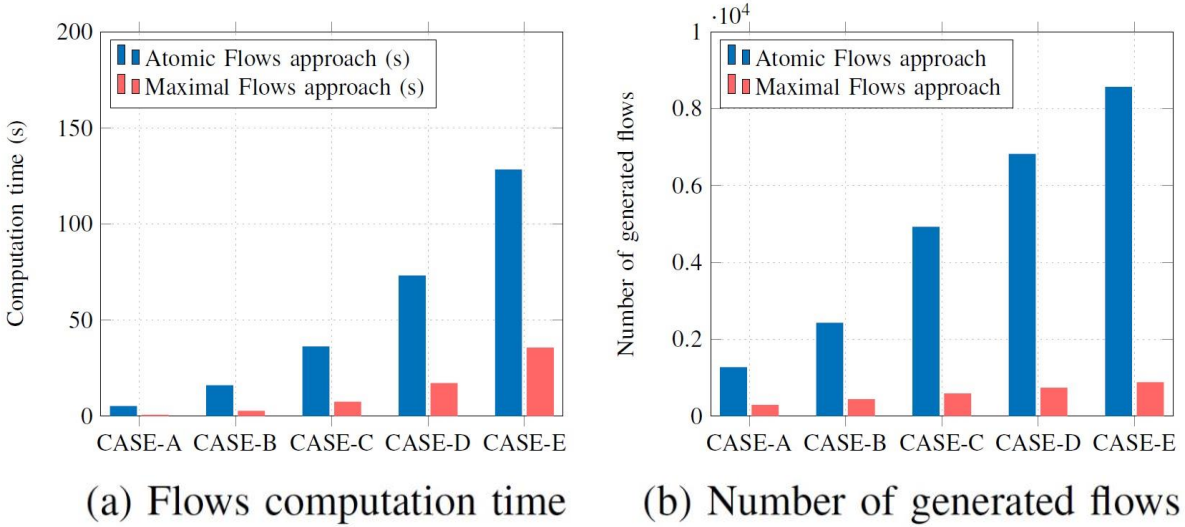


MAXIMAL FLOWS

- Solution using Atomic flows is slower (because of the time to compute Atomic predicates)
- Solution using Atomic flows generates a greater number of flows
- Solution using Maximal flows have complex predicate representation (vs simple integers)

Time to solve Verification and Refinement

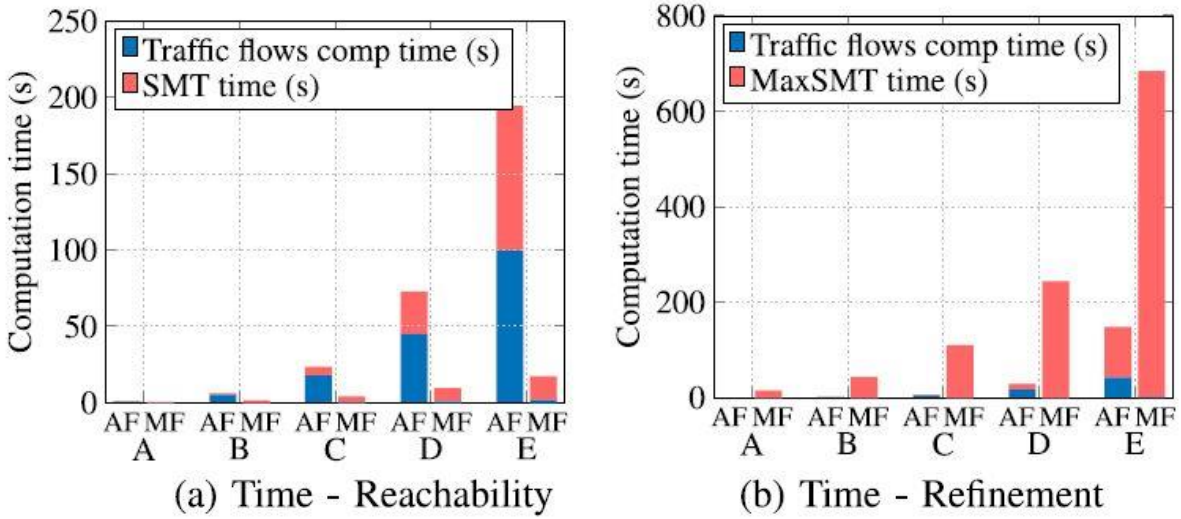
- Total time = Traffic flows computation time + SMT/MaxSMT resolution time



(a) Flows computation time (b) Number of generated flows

	pol	nodes	NAT	FW
CASE-A	100	400	50	50
CASE-B	150	600	75	75
CASE-C	200	800	100	100
CASE-D	250	1000	125	125
CASE-E	300	1200	150	150

	pol	nodes	NAT	FW
A	50	40	5	5
B	75	60	10	10
C	100	80	15	15
D	125	100	20	20
E	150	120	25	25



(a) Time - Reachability (b) Time - Refinement

S.Bussa, R. Sisto, F. Valenza, **"Security Automation using Traffic Flow modeling"**, in IEEE 8th International Conference on Network Softwarization (NetSoft 2022), Milan, Italy, June 27 - July 01, 2022

D. Bringhenti, S. Bussa, R. Sisto, F. Valenza, **"A two-fold traffic flow model for network security management"**, IEEE Transactions on Network and Service Management (TNSM) (2024)

Automatic firewall policies anomaly analysis and resolution

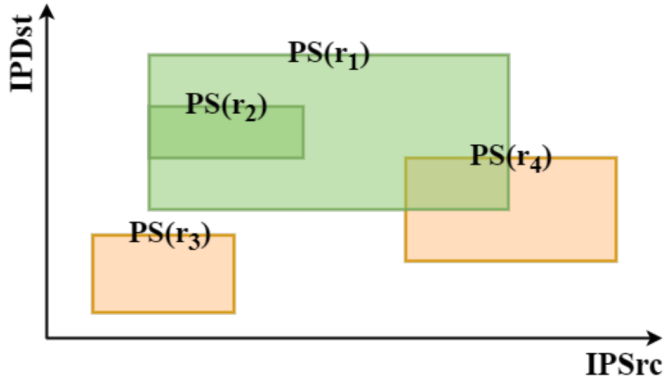
- Goal: Find and solve anomalies in a formally proved and **optimized** way
- A **firewall policy** may contain **policy anomalies**

SUB-OPTIMIZATIONS

#	Action	IPSrc	IPDst
r_1	Allow	[130.11.2.16, 130.11.2.100]	[42.0.2.32, 42.0.2.86]
r_2	Allow	[130.11.2.16, 130.11.2.42]	[42.0.2.42, 42.0.2.60]
r_3	Deny	[130.11.2.84, 130.11.2.146]	[42.0.2.22, 42.0.2.42]
r_4	Deny	[130.11.2.4, 130.11.2.26]	[42.0.2.2, 42.0.2.28]

CONFLICTS

#	Action	IPSrc	IPDst
r_1	Allow	[130.11.2.16, 130.11.2.100]	[42.0.2.32, 42.0.2.86]
r_2	Allow	[130.11.2.16, 130.11.2.42]	[42.0.2.42, 42.0.2.60]
r_3	Deny	[130.11.2.84, 130.11.2.146]	[42.0.2.22, 42.0.2.42]
r_4	Deny	[130.11.2.4, 130.11.2.26]	[42.0.2.2, 42.0.2.28]



Automatic firewall policies anomaly analysis and resolution

- Managing anomalies **manually** is complex
- Also, not all anomalies are errors

Typical example of intentional conflict

#	Action	IPSrc	IPDst
r_1	Allow	[130.0.0.1]	[40.0.0.30]
r_2	Deny	[130.0.0.*]	[40.0.0.30]

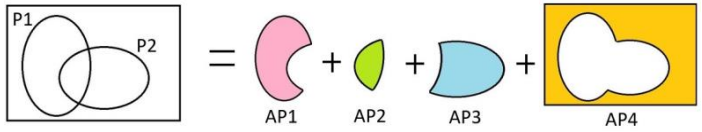
- **ANOMALY ANALYSIS**: identify all the anomalies affecting a firewall policy
- **ANOMALY RESOLUTION**: working on the identified anomalies, solve conflicts and remove sub-optimizations

- STATE OF THE ART (**Automatic approaches**):
  - Extremely complex algorithms
  - Most works only perform anomaly analysis but NOT automatic resolution

- OUR GOAL
  - Build an **automatic tool for both anomaly analysis and resolution**

Atomized firewall policies

- We apply, again, the concept of Atomic Predicate, to atomize firewall rules



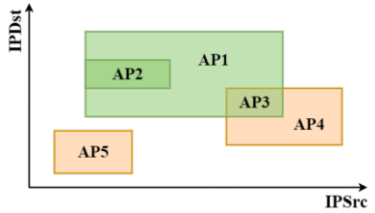


TABLE I: *Original firewall policy R*

#	Action	IPSrc	IPDst	pSrc	pDst	Proto
r_1	Allow	[130.11.2.16, 130.11.2.100]	[42.0.2.32, 42.0.2.86]	*	*	*
r_2	Allow	[130.11.2.16, 130.11.2.42]	[42.0.2.42, 42.0.2.60]	*	*	*
r_3	Deny	[130.11.2.84, 130.11.2.146]	[42.0.2.22, 42.0.2.42]	*	*	*
r_4	Deny	[130.11.2.4, 130.11.2.26]	[42.0.2.2, 42.0.2.28]	*	*	*

Parallelizable

TABLE II: *Atomized firewall policy* R^α

# (original)	# (atomic)	Action	Condition
r_1	r_1^α	Allow	AP1
	r_2^α	Allow	AP2
	r_3^α	Allow	AP3
r_2	r_4^α	Allow	AP2
r_3	r_5^α	Deny	AP3
	r_6^α	Deny	AP4
r_4	r_7^α	Deny	AP5

- Atomic predicates are **disjoint and unique**: possible to **replace** them with **integers**
- **Anomaly analysis and resolution** can be **performed on integers** => much faster
- For the resolution, we implement two different strategies:
  - **Automatic**: deny-win, allow-win, priority-win
  - **Human-assisted** (semi-automatic)

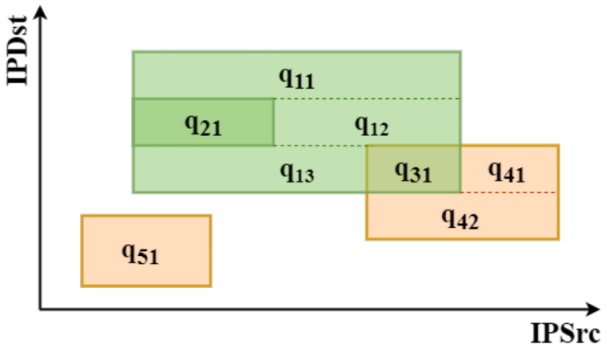
Automatic anomaly resolution

- Example, deny-win strategy

# (original)	# (atomic)	Action	Condition
r_1	r_1^α	Allow	AP1
	~~r_2^α~~	~~Allow~~	~~AP2~~
	~~r_3^α~~	~~Allow~~	~~AP3~~
r_2	r_4^α	Allow	AP2
r_3	r_5^α	Deny	AP3
	r_6^α	Deny	AP4
r_4	r_7^α	Deny	AP5

# (original)	# (atomic)	Action	Condition
r_1	r_1^α	Allow	AP1
r_2	r_4^α	Allow	AP2
r_3	r_5^α	Deny	AP3
	r_6^α	Deny	AP4
r_4	r_7^α	Deny	AP5

- Final step, rewrite atomic predicates from integers back to predicate conditions



Atomic predicate	Division in sub-rectangles
AP1	$q_{11} \vee q_{12} \vee q_{13}$
AP2	q_{21}
AP3	q_{31}
AP4	$q_{41} \vee q_{42}$
AP5	q_{51}

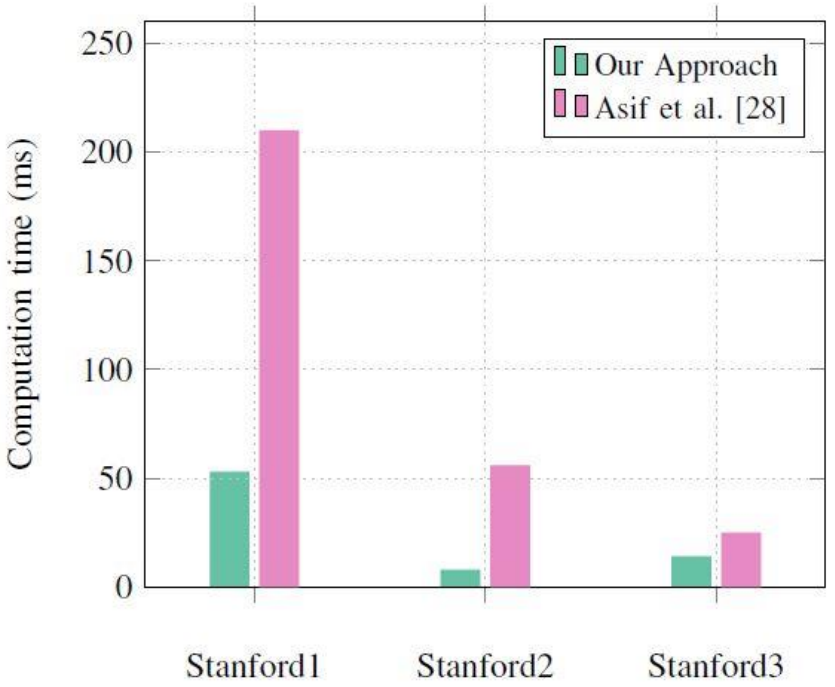
#	Action	IPSrc	IPDst
r_1^ϕ	Allow	[130.11.2.16, 130.11.2.100]	[42.0.2.60, 42.0.2.86]
r_2^ϕ	Allow	[130.11.2.16, 130.11.2.42]	[42.0.2.42, 42.0.2.60]
r_3^ϕ	Allow	[130.11.2.42, 130.11.2.100]	[42.0.2.42, 42.0.2.60]
r_4^ϕ	Allow	[130.11.2.16, 130.11.2.84]	[42.0.2.32, 42.0.2.42]
r_5^ϕ	Deny	[130.11.2.4, 130.11.2.26]	[42.0.2.2, 42.0.2.28]
r_6^ϕ	Deny	[130.11.2.84, 130.11.2.146]	[42.0.2.22, 42.0.2.42]

Validation and Conclusions

- We **compared** our approach with the state of the art:

Anomaly resolution time (s)	Number of rules							
	12	18	27	54	81	132	354	927
Our approach	0,004	0,006	0,009	0,044	0,122	0,571	15,27	163,99
Hu et al. (Comb.)	0,933	1,048	1,211	2,225	3,442	9,990	97,79	256,58
Hu et al. (Greedy)	0,688	0,787	1,072	1,728	2,083	8,228	39,33	106,12

Anomaly analysis time (ms)		Test case A		Test case B	
		Our approach	Al-Shaer et al.	Our approach	Al-Shaer et al.
N. Rules	50	3	46	17	119
	60	4	58	29	149
	70	4	66	22	178
	80	4	79	25	210
	90	5	87	27	245



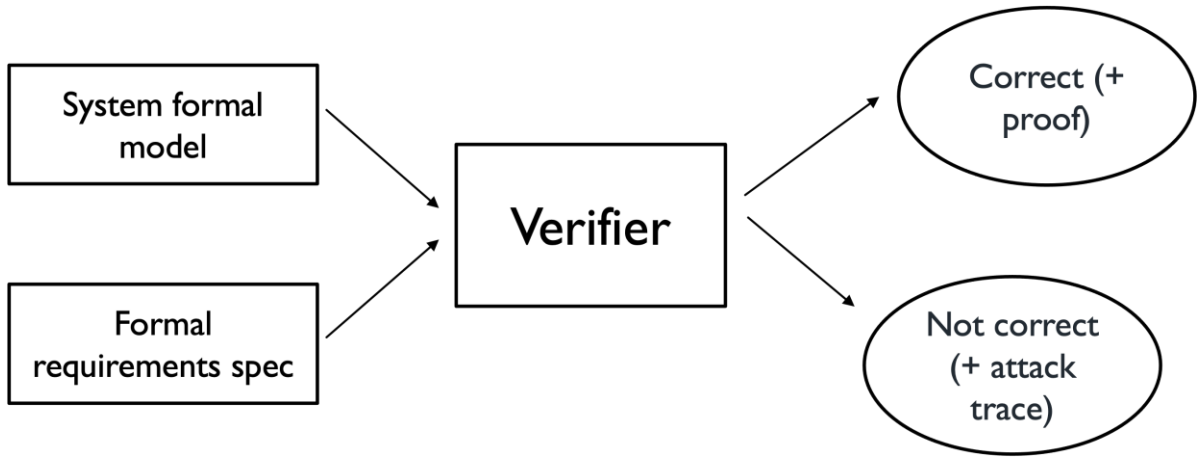
D. Bringhenti, S. Bussa, R. Sisto, F. Valenza, **"Atomizing firewall policies for anomaly analysis and resolution"**, IEEE Transactions on Dependable and Secure Computing (TDSC), 2024

Introduction to Formal verification

Static analysis of a system formal model

1. Formal specification: from the real system to its formal abstract model
2. Formal Verification: check if the model satisfies some formal properties

FV can be used to verify the security of a system

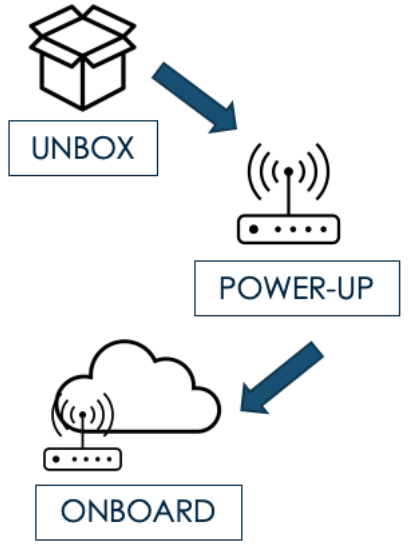


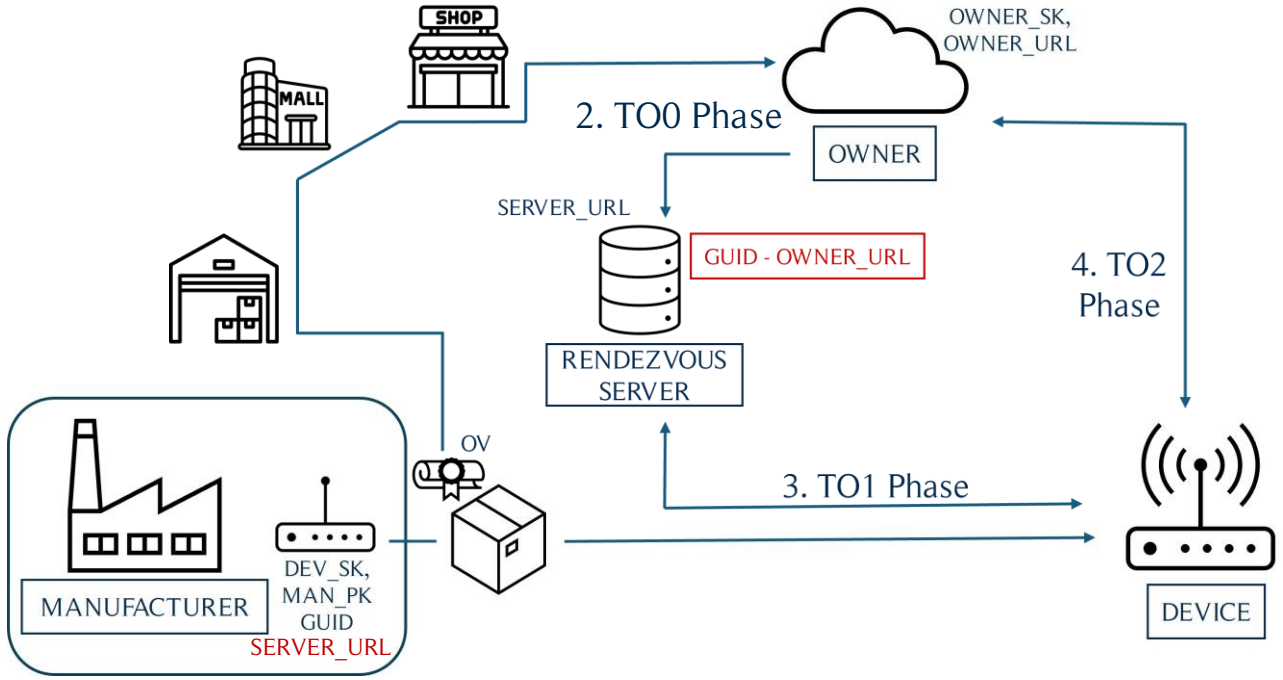
- Why formal verification for security?
- The analysis is rigorous (mathematically proved)
- Can explore all the possible execution paths of a system
- Can easily find most of common attacks (e.g., MITM, replay, etc.)

Available tools: Proverif and Tamarin

Formal verification of an IoT Device Onboarding protocol

- Device onboarding
  - Need to **replace** existing **proprietary protocols** with a **single standard**
  - In 2020 the FIDO Alliance proposed **FIDO Device Onboarding** (**FDO**)
- No formal verification of the FDO found in the literature



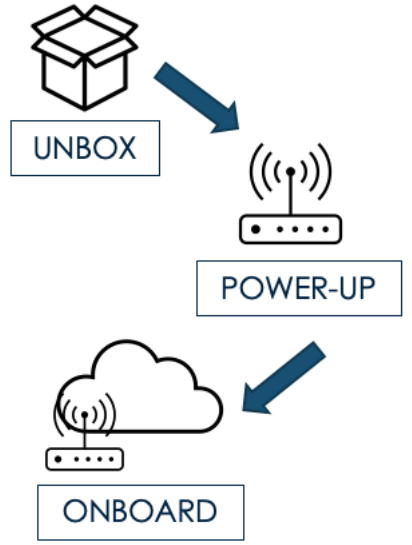


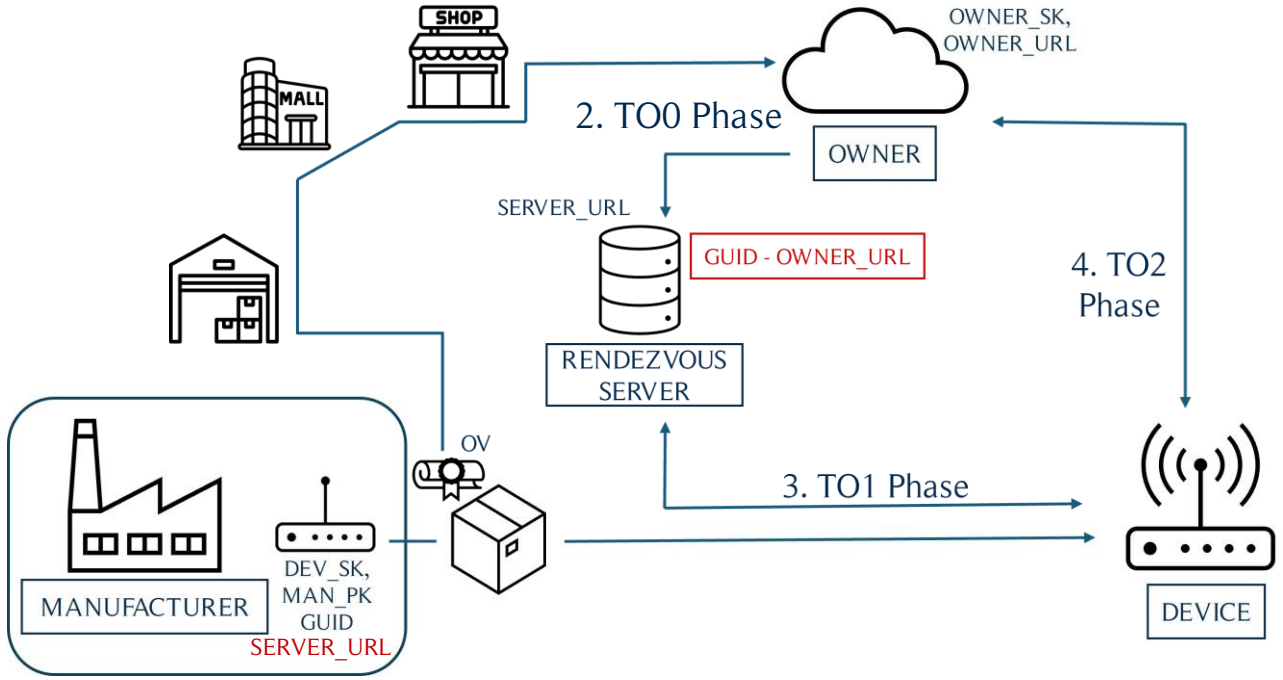
Two questions:

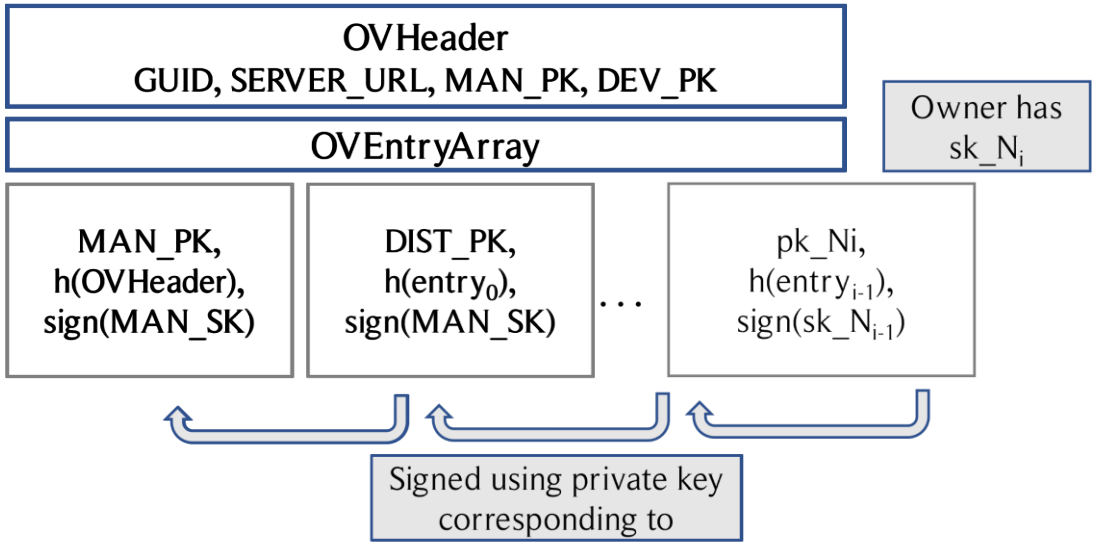
1. How does the device discover the Owner URL?
2. How is the transfer of ownership of the device handled?

Formal verification of an IoT Device Onboarding protocol

- Device onboarding
  - Need to **replace** existing **proprietary protocols** with a **single standard**
  - In 2020 the FIDO Alliance proposed **FIDO Device Onboarding** (**FDO**)
- No formal verification of the FDO found in the literature

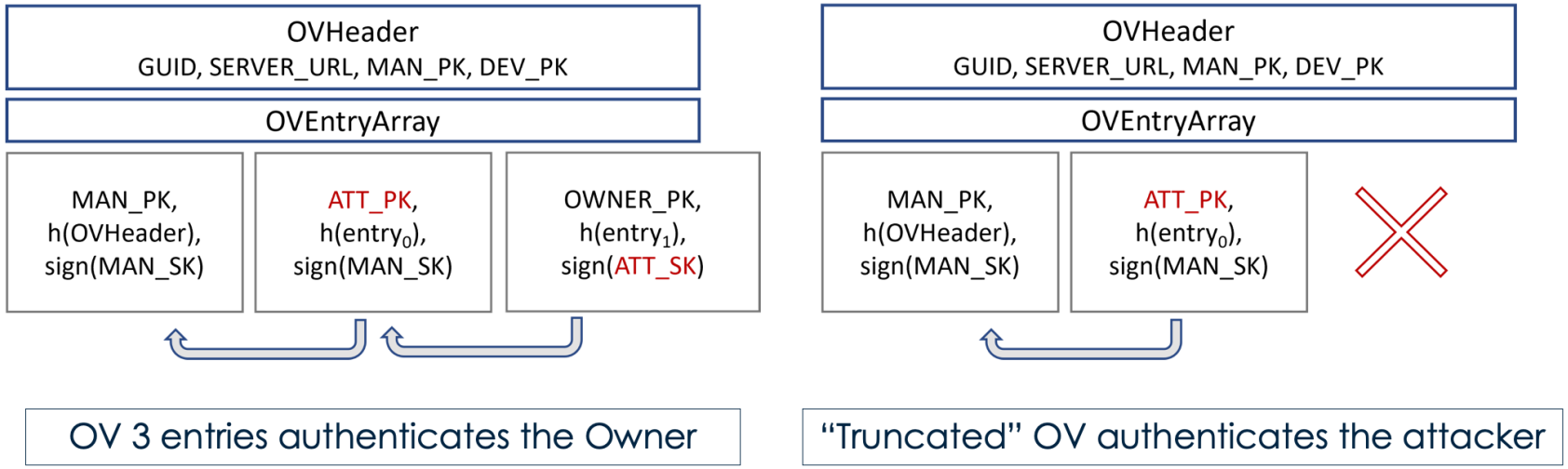






Formal verification of the FDO protocol

- Formal verification using Proverif
- Classic Dolev-Yao attacker, authentication properties, three scenarios
  1. Attacker not involved in the protocol
  2. Attacker owns a valid IoT device
  3. Attacker is an intermediate node in the supply chain
- We found a weakness in the third scenario



Conclusions

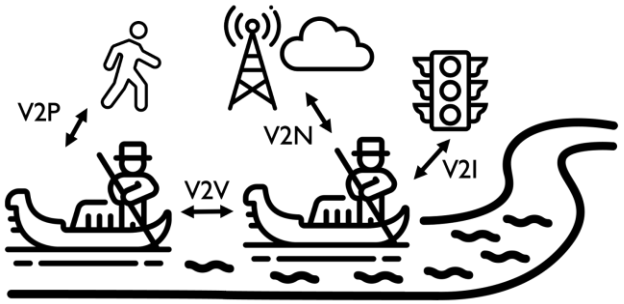
- We successfully tested the attack on a real implementation of the protocol
- We reported the weakness to the FIDO Alliance
- FIDO confirmed the attack and suggested a possible countermeasure
- We formally demonstrated that the countermeasure may not completely solve the attack
- At that point, we worked to propose our solution (formally verified)
- FIDO confirmed the validity of our solution, to be brought to the attention of the FDO WG to further evaluate pros and cons

S. Bussa, R. Sisto, F. Valenza, **"Formal Verification of the FDO protocol"**, in IEEE International Conference on Standards for Communications and Networking (CSCN 2023), Munich, Germany, 6-8 November 2023

S. Bussa, R. Sisto, F. Valenza, **"FDO protocol: a possible solution to protect the OV against untrusted supply chain"**, IEEE Transactions on Dependable and Secure Computing (TDSC), submitted 27/05/24

Formal verification of v2x communications

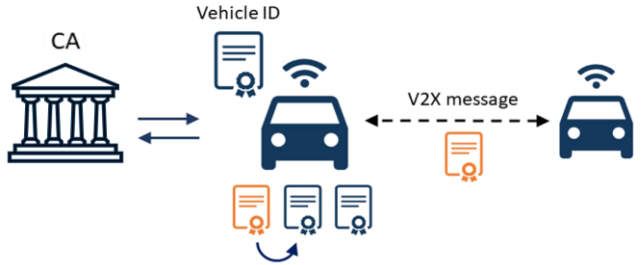
- V2X = Vehicle to Everything

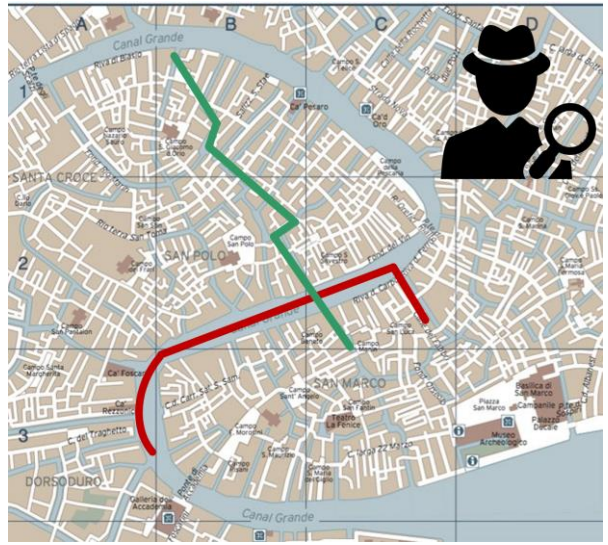


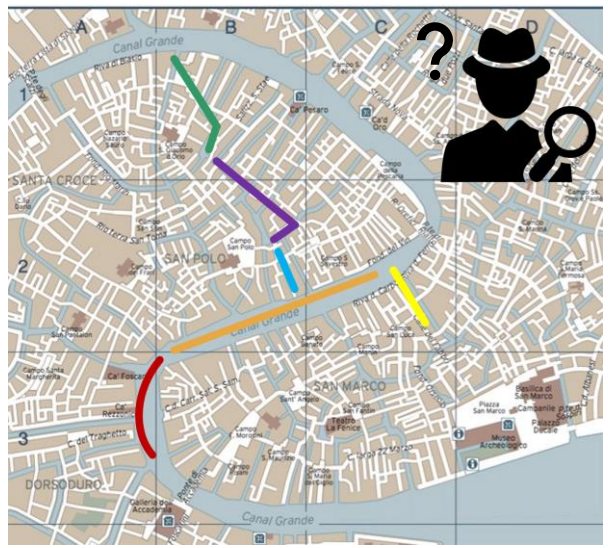
Formal verification of v2x communications

- Being v2x data sensitive, their **compromise** may impact **safety** and **privacy**
- **Goal**: message **authentication** and **integrity**
- Traditional network protocols use **Digital signatures** and **certificates**

- **Privacy** issues! Easy to **track** vehicle position
- **Anonymity** is needed and it is obtained using **Pseudonymous Certificates**

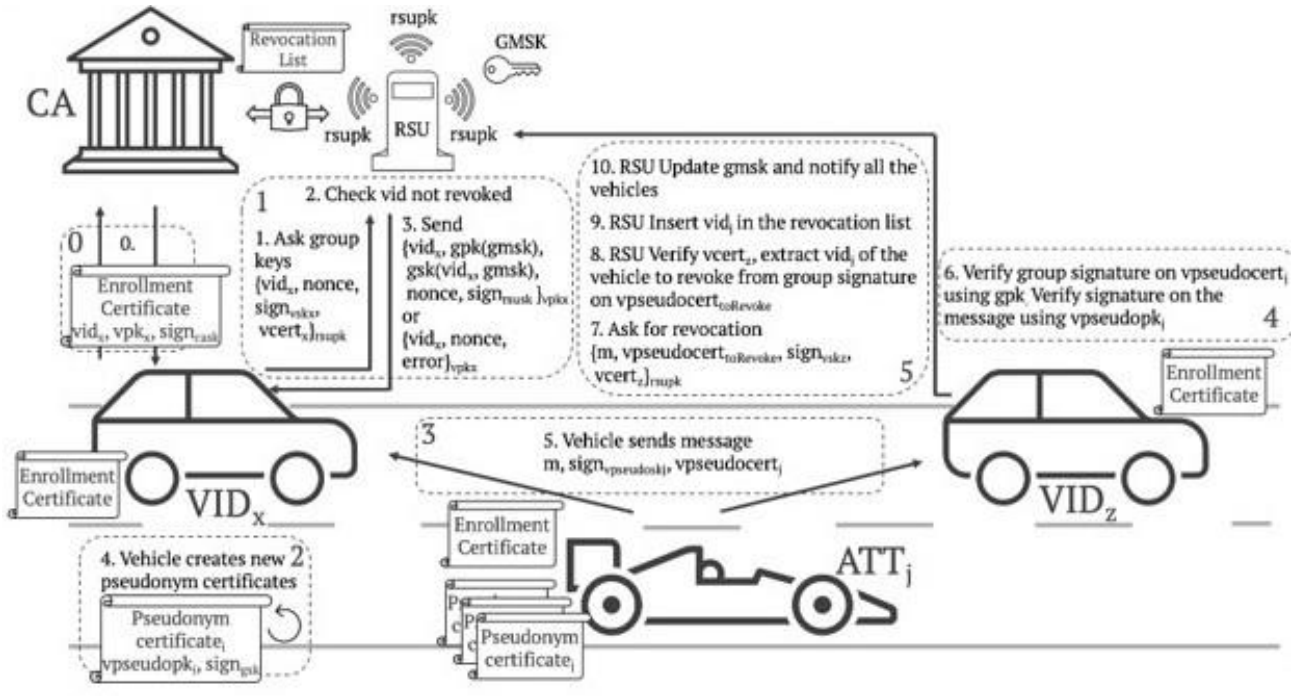




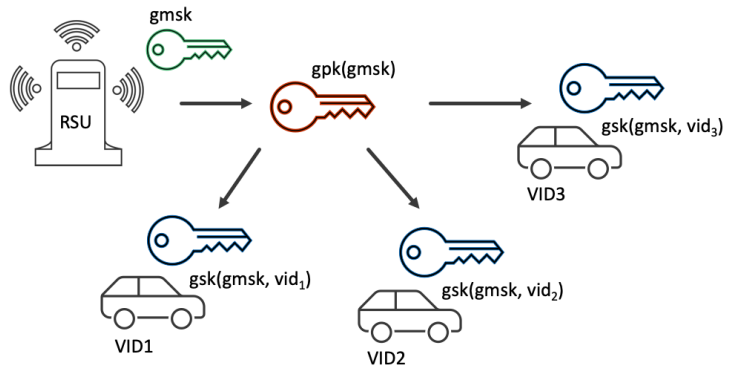


Formal verification of v2x communications

- Many **V2X schemes** have been proposed to handle pseudonyms
- **Limitations of the state of the art**: formal analysis is missing in most cases
- **Goal**: **formally verify** the most important existing v2x protocols



- Protocol: "Efficient and robust pseudonymous authentication in Vanet"
- Digital certificates and Group signatures to self-certify each pseudo certificate



Formal verification of the v2x scheme

- Formal verification using Proverif
  1. Attacker not involved in the protocol
  2. Attacker owns valid vehicles
  - **Secrecy**: secret keys, vehicle real identity

    Reachability queries, resistance to offline attacks, strong secrecy
  - **Authentication**

    Injective / non-injective correspondence assertions
  - **Anonymity**: preserve vehicle real identity

    Observational equivalence
  - **Unlinkability**: multiple executions cannot be linked together

    Observational equivalence
  - **Distributed resolution / Compromise of authorities**

    Disclosure of the secret key

Formal verification of the v2x scheme

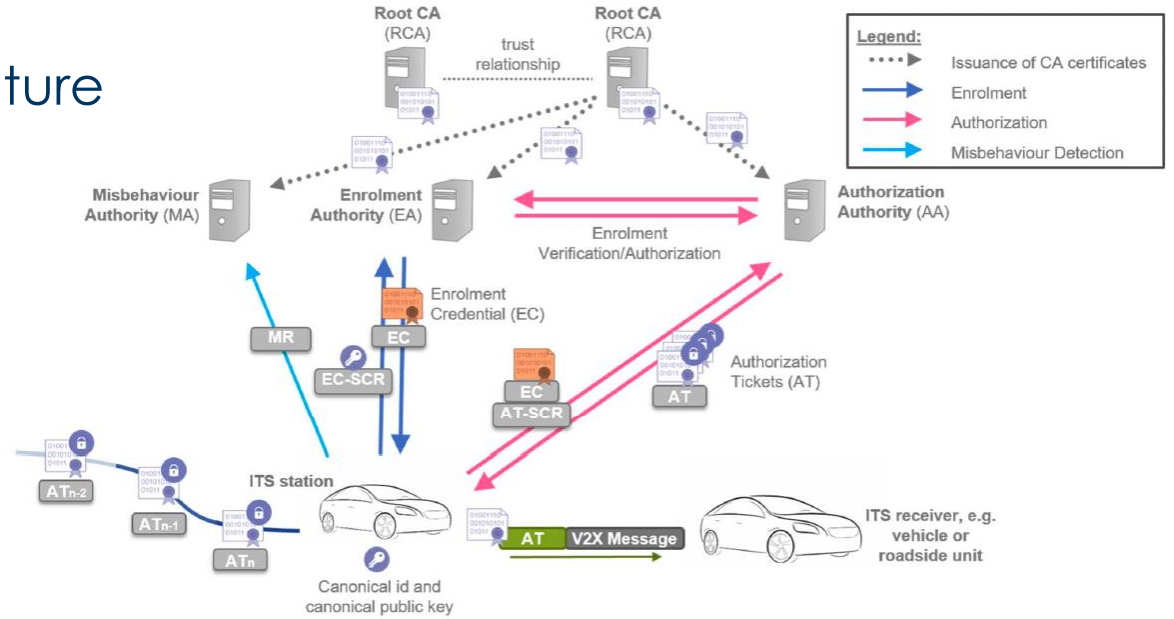
- Contribution
  - Details about Proverif model and implementation, formalization of security properties
  - 5 vulnerabilities found
  - Solution to discovered issues

S. Bussa, R. Sisto, F. Valenza, **"Formal Verification of a v2x Privacy Preserving scheme using Proverif"**, in IEEE International Conference on Cyber Security and Resilience (CSR 2023), Venice (ITA), July 31 - August 02

S. Bussa, R. Sisto, F. Valenza, **"Formal verification of a v2x scheme mixing traditional PKI and group signatures"**, in Journal of Information Security and Applications (JISAS), 2024

Formal verification of the ETSI 102 940

- **Period abroad** at the University of Birmingham (6 months)
- Security analysis of the **ETSI 102 941** standard (V2X European PKI)
- **Lack of** a clear **description** of the objectives and security requirements
- Contributions:
- Complete **threat model** of the ETSI PKI architecture
- Definition of 106 security-privacy requirements
- Formal verification of the scheme (Proverif and Tamarin)
- 7 vulnerabilities found
- Future work: tests on implementation



"Establishing Security Requirements for Automotive PKI: A Comprehensive Framework for Protocol Design and Implementation", in Network and Distributed System Security (NDSS) Symposium 2026, submitted

Other Formal verification works

R. Schermann, S. Bussa, R. Urian, C. Steger, **"Zero touch privacy preserving provisioning in an Edge-, Fog-, and Cloud environment"**, in IEEE Int. Conference on Fog and Mobile Edge Computing (FMEC 2023), Tartu, Estonia, September 2023

R. Schermann, S. Bussa, R. Urian, C. Steger, **"PAKA: Pseudonymous Authenticated Key Agreement without bilinear cryptography"**, in ACM Int. Conference on Availability, Reliability and Security (ARES 2024), Vienna, Austria

Conclusions and Future work

- During my PhD, I addresses two critical research topics within the scope of CPSs: automatic network security management and automatic security verification

  - Novel formal models to automate the management of network security configuration
  - New concepts for network predicates and traffic flows
  - Future work: add new features to presented models (e.g., VPNs, IDSs, etc.), apply the models to other network management problems (e.g., latency constraints, QoS, reliability)

  - Formal verification IoT protocols and V2X communication schemes
  - Future work: apply the same techniques to other protocols in the field / to other different fields

Thanks for your attention!

SIMONE BUSSA
simone.bussa@polito.it