“Design and Verification of Secure Autonomous Vehicles”

Ludovic Apvrille1, et al.

12th ITS European Congress, Strasbourg, France, 19-22 June 2017

Software Verification and Validation

Berenice Llive

November 29th, 2017

http://biblio.telecom-paristech.fr/cgi-bin/display.cgi?id=16884

Budapest University of Technology and Economics
Department of Networked Systems and Services

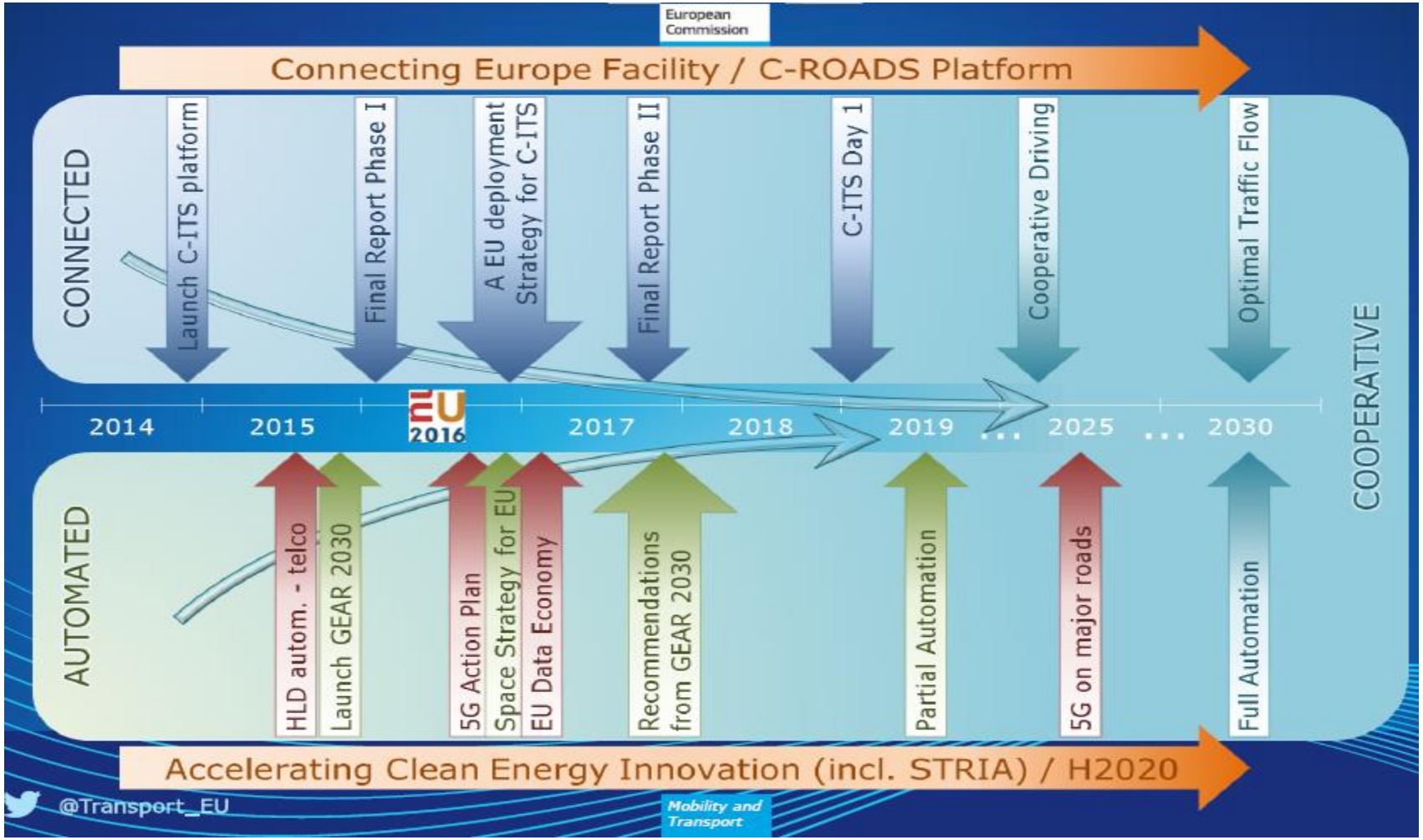
Content

- Introduction
- Motivation
- Security overview for vehicles
- Threat Analysis & Risk Assessment
- Security Solutions, Use Case Study
- Modeling
- Formal Verification
- Performance Analysis
- Conclusions

Introduction

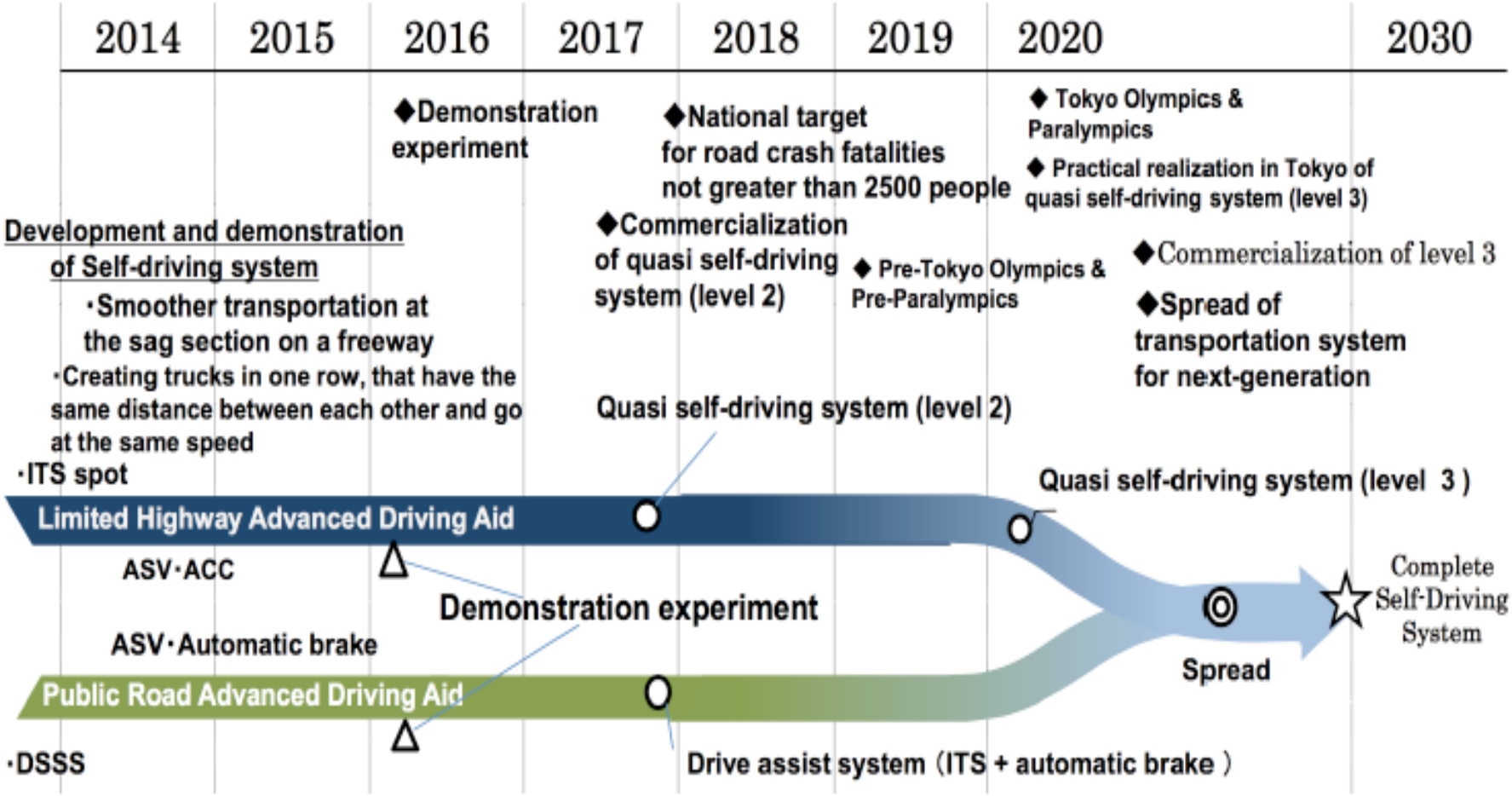
Modern vehicles are connected devices equipped with a plethora of sensors which provides a better driver experience, however, in the very near future they will interact directly with each other and with the road infrastructure, tending to reach the autonomous vehicles as last phase.

Introduction



European Deployment Plan

Introduction



Japanese Deployment Plan

Motivation

V2X systems are designed for reducing traffic on roads, accidents, and human error.

V2X Systems and Autonomous vehicle technologies aim to tackle some of the biggest challenges in the surface transportation industry in:

- Safety: Tools to anticipate potential crashes.
- Mobility: enable system users and system operators to make smart choices that reduce travel delay.
- Environment: Real Time info to make green transportation choices

Motivation

Advantages

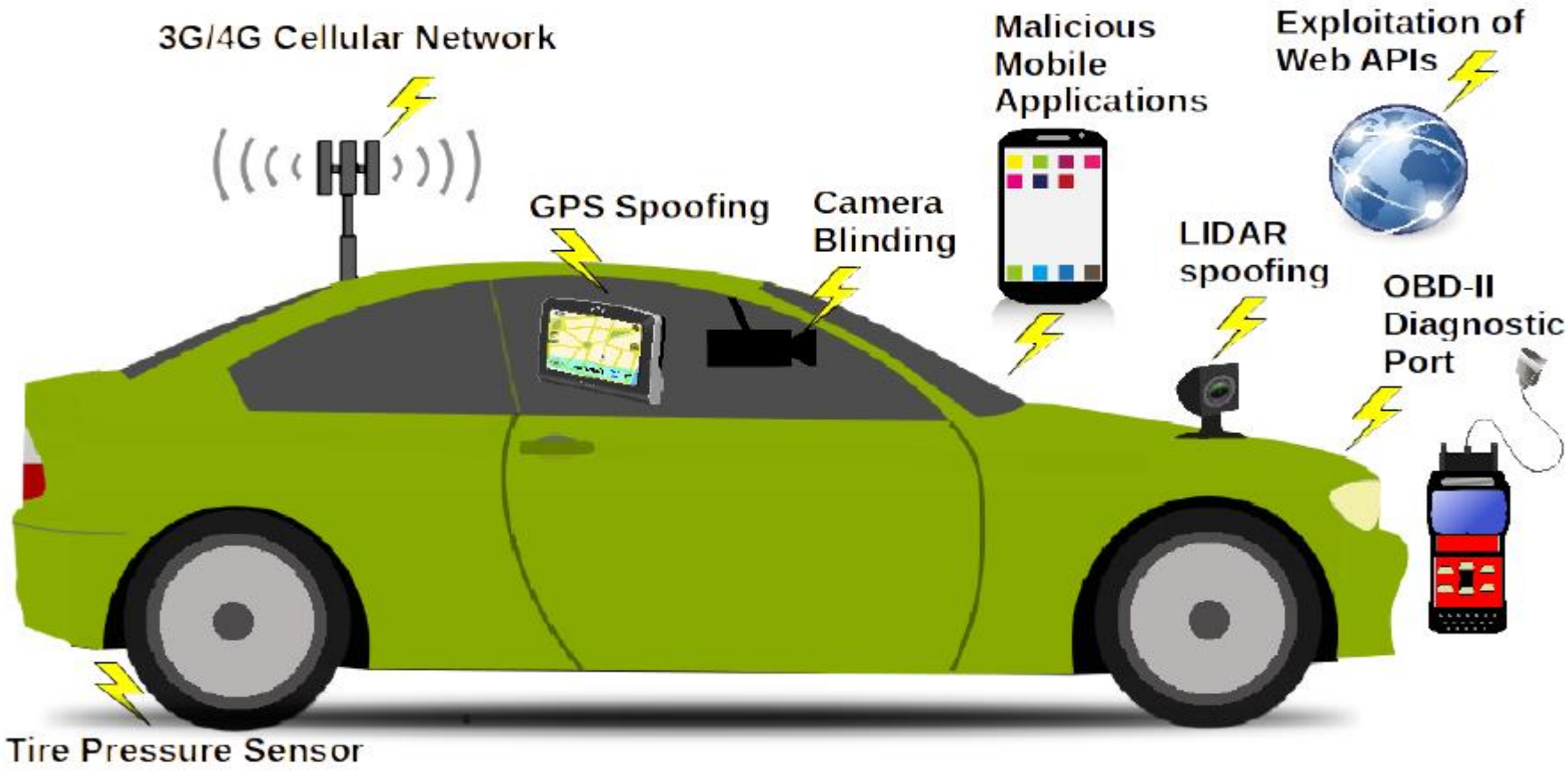
- Enable crash prevention
- Enable safety,
- Enable mobility and environmental benefits
- Provide continuous real-time connectivity to all system users

Disadvantages

- Wireless Environment meaning an easy prey of attacks
- Security Protocols produces overhead - Loss of privacy – Track a vehicle
- Access to the vehicle's control system remotely

** Connected and future autonomous vehicles offer great conveniences for users, but their security must be considered to ensure that they could not be remotely attacked and compromise the safety of the driver.*

Security overview



Vehicle Attack Surfaces

Threats Identified in V2X

DoS and Availability Threats

Such attacks may results in an ITS station failing to receive, respond to, relay, produce and send traffic safety messages.

- Message Saturation
- Jamming of radio signals
- Injection of false messages
- Wormhole Attacks

Integrity and Masquerade Threats

Unauthorized access to restricted information can be gained by means of a masquerade attack or by the use of malware injected into an ITS station.

- GNSS Spoofing
- Masquerade as ITS-S station or ITS network
- Malicious Isolation (black hole)
- Installation of Malware
- Replay of “expired” old messages
- Manipulation of relayed ITS messages en route

Confidentiality and Privacy Threats

It includes the illicit collection of transaction data by eavesdropping and the collection of location information through the analysis of message traffic.

- Eavesdropping
- Traffic Analysis
- Location Tracking

Accountability and Non-repudiation Threats

For law enforcement authorities to be able to prosecute such actions, it is necessary to record all message and service activity within the an ITS station.

- Denial of Transmission
- Denial of Data Receipt

Minor
Major
Critical

The three levels of risk are derived from a qualitative combination of likelihood and impact, the minor risk level is considered acceptable and countermeasures should be introduced in order to reduce all Major or Critical risks to Minor.

Threat Analysis &Risk Assessment

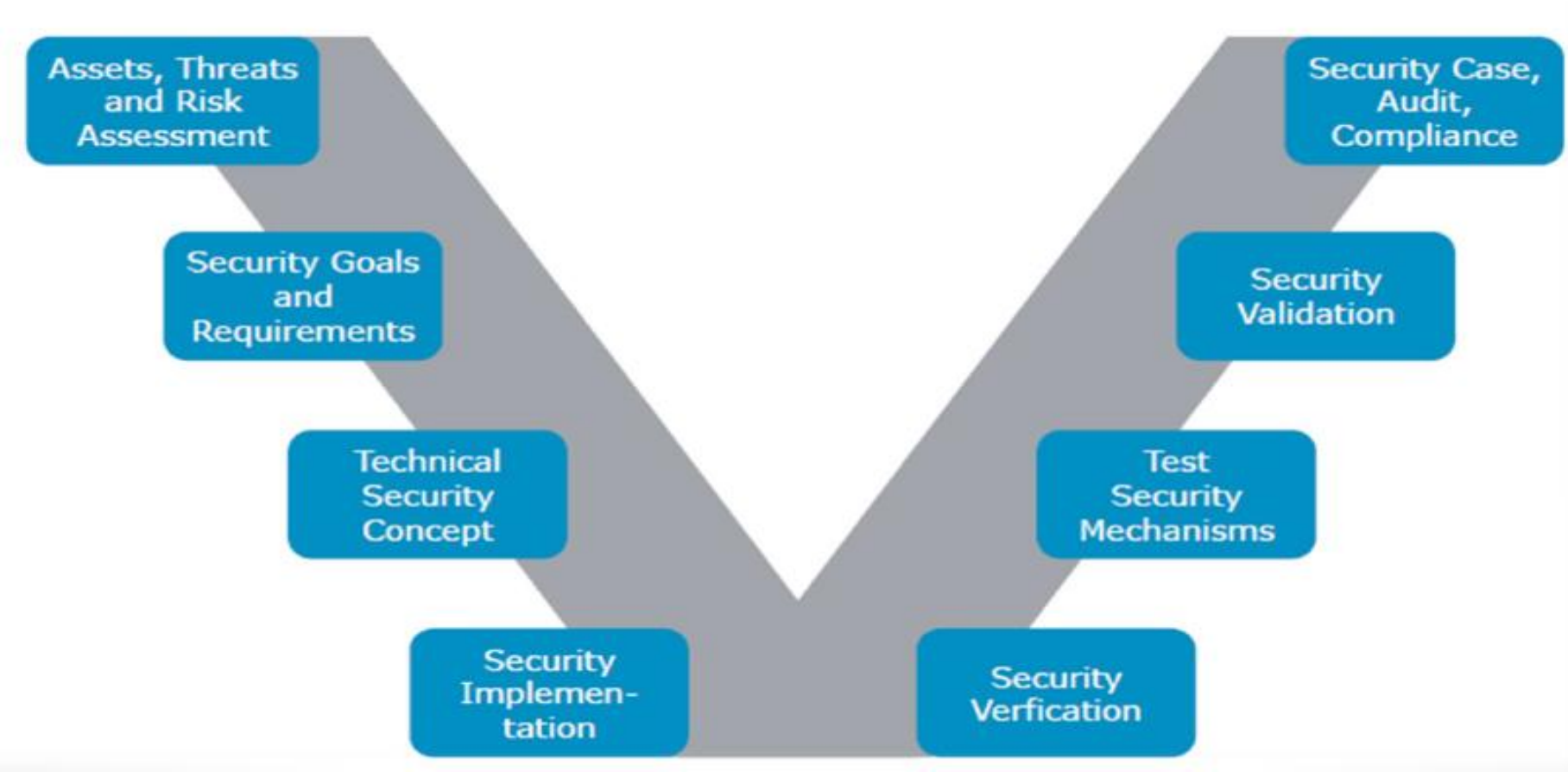
Attack Outcome	Vector of Attack
Control of vehicle	Wifi
Falsified sensor readings	Camera/LiDAR
Control of vehicle	3G network
Control of vehicle	Smartphone App
Loss of privacy	Tire sensor
Misdirected navigation	GPS signals

The attack modeling phase is known as a very important driver for motivating the need for introducing security countermeasures in a risk analysis, and also for selecting where those security mechanisms better fit.

Security Solutions

Source: https://www.sans.org/summit-archives/file/summit-archive-1493920308.pdf

Security Development Process



Security Development Process by ISO-SAE 21434

Source: https://www.sans.org/summit-archives/file/summit-archive-1493920308.pdf

Security European Projects

- SEVECOM – Secure Vehicle Communications, focused on securing communications between vehicles.
- SHE – Secure Hardware Extension, add hardware modules to embedded automotive architecture to accelerate cryptographic processing.
- EVITA – E-safety vehicle intrusion protected applications, finished in 2011, it defined a complete secure automotive architecture with hardware accelerators and security protocols.

Evaluation of Architectures: TTool

TTool is an open-source, free, multi-profile toolkit dedicated to the edition of UML and SysML diagrams, and to the simulation and formal verification (safety, security, performance) of those diagrams. TTool supports several development stages of embedded systems:

1. **Partitioning of embedded systems** with the **DIPLODOCUS** environment
2. **Design of embedded software** with the **AVATAR** environment
3. **Design of safe and secure embedded systems** with the **SysML-Sec** environment.

Validations are supported by TTool with a press-button approach: models can be transformed by TTool into a formal specifications.

Security Solution: Developing System using TTool

- Make use of modelling and formal verification to evaluate those solutions when selecting an architecture, mapping and add only necessary security.

- Security issues are not addressed by TTool profiles. Thus it was designed the SysML-Sec environment in order to make it possible to describe such issues together with partitioning requirements.

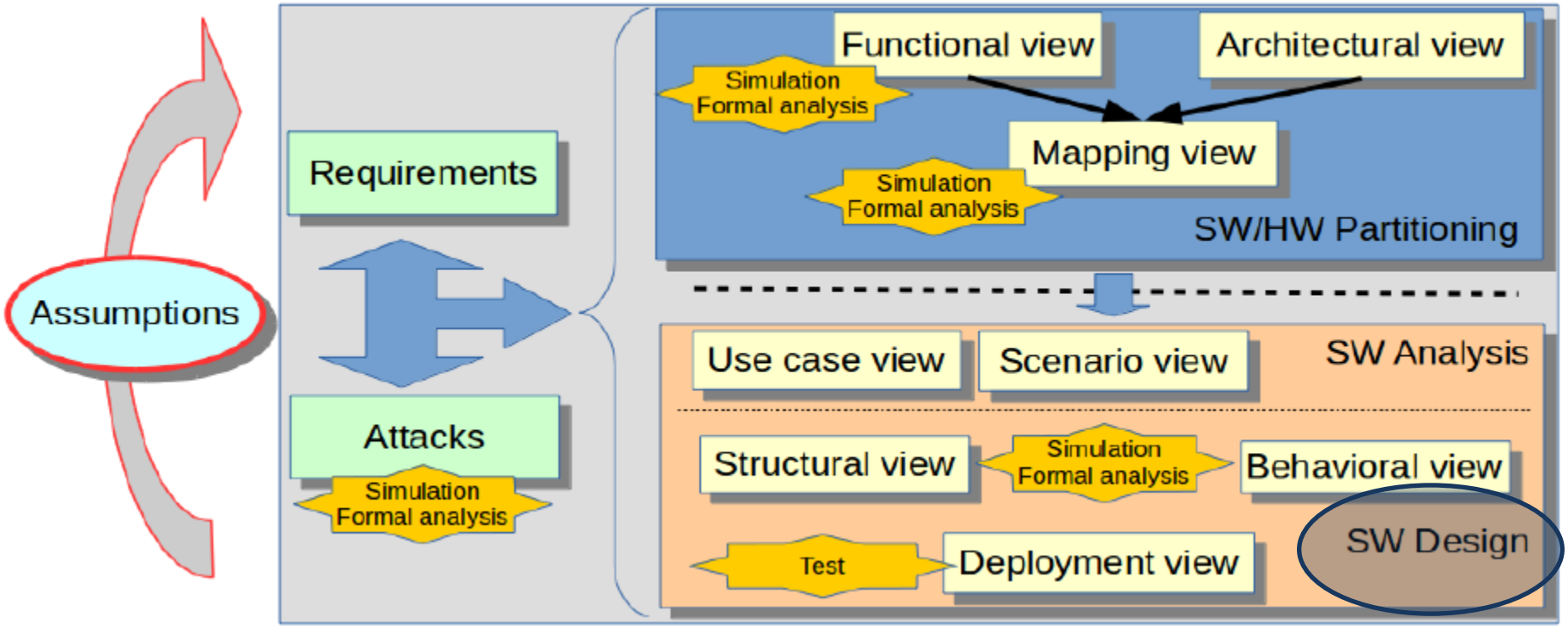
SysML-Sec Methodology

SysML-Sec is an environment to design safe and secure embedded systems with an extended version of the SysML.
It targets both:

- Software
- Hardware

Many projects and case studies have already been modeled with SysML-Sec ranging from automotive systems, drone systems, information systems (e.g., the analysis of malware targetting banking systems) and industrial systems (Analysis of SCADA malware), and more generally, security protocols.

SysML-Sec Methodology



- This methodology was introduced for the design of complex systems in terms of safety, performance and security, an extension of UML

SysML-Sec - Dolev-Yao Model Attacker

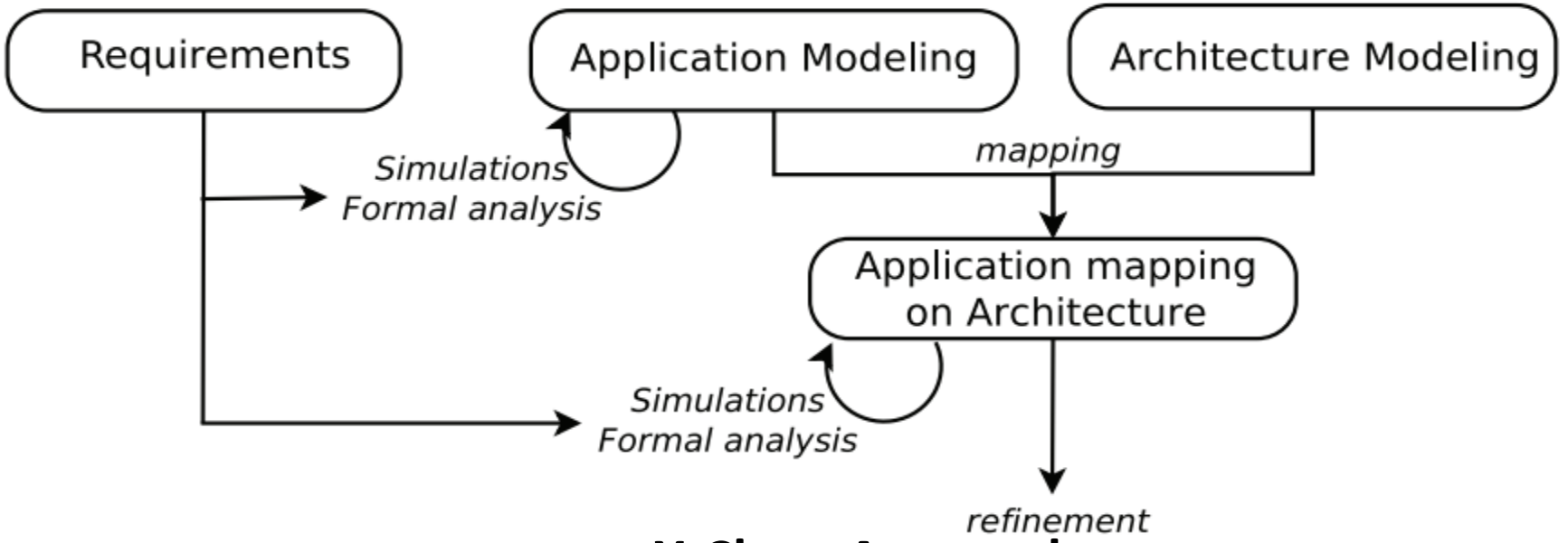
Requirements/attacks:

- Identify and analyse both requirements and attacks together
- Formally attack graphs are used to capture attack scenarios.

Dolev-Yao Model:

- It is presented as an arbitrarily pi-process that can listen to all messages sent on non-private channels.
- It can inject data on the buses.
- The attacker may read, modify, delete and inject messages.

SysML-Sec
Software/Hardware Partitioning



Y-Chart Approach

http://ieeexplore.ieee.org/document/7323182/

The result of this approach shall be an optimal hardware/software architecture with regards to criteria at stake for that particular system (e.g., cost, area, power, performance, reliability)

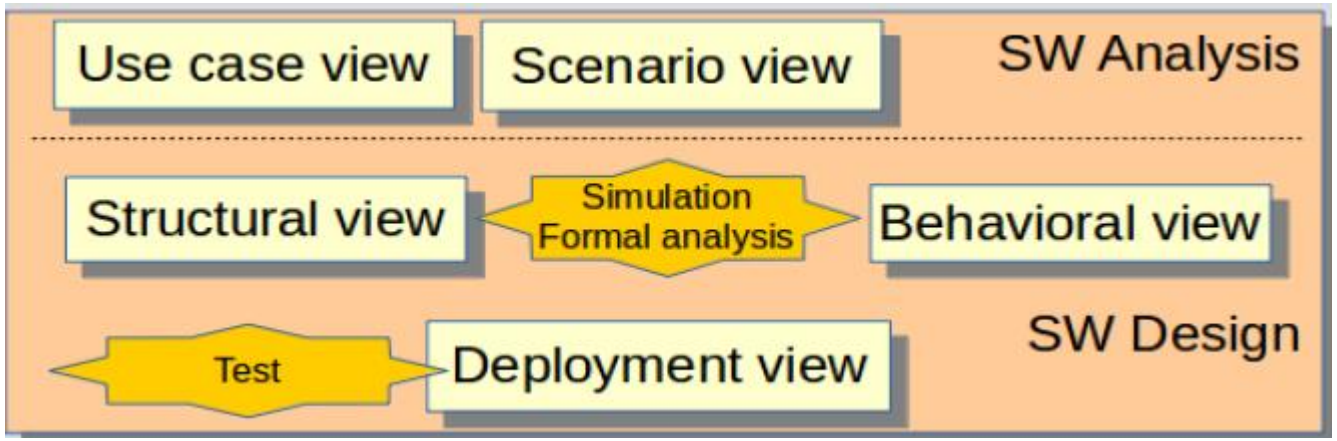
SysML-Sec
Software/Hardware Partitioning

Application Modelling → Communicating task (functions) and their behaviour.

Architectural Modelling→ Graph of execution nodes, communication nodes and storage nodes.
Execution nodes: CPUs, Hardware Accelerators.
Communication nodes: Bridges and buses.
Storages Nodes: Memories

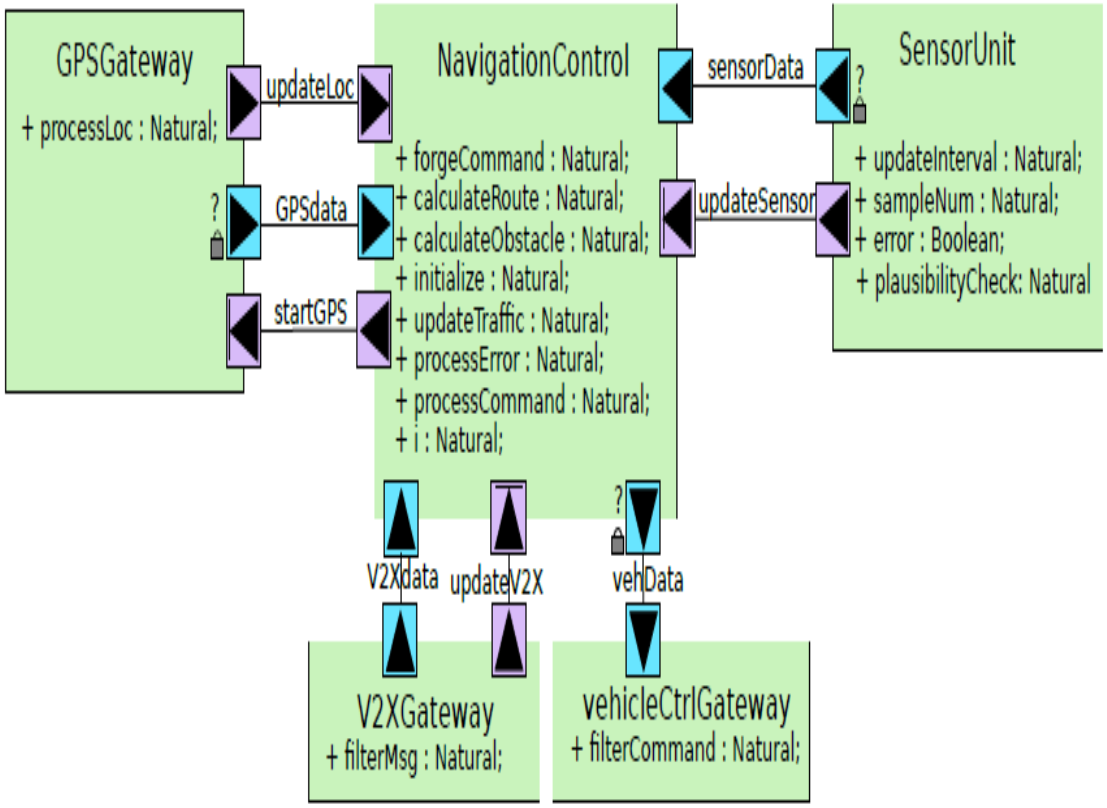
Mapping involves specifying the location of tasks on the architectural model.

SysML-Sec
Software Analysis



- Develop the software components implementing the functions mapped onto processors at the previous stage.
- Software components/blocks are progressively refined until the point where executable code generation is feasible.
- This refinement also includes adding security-related functions, e.g., cryptographic algorithms, key management policies, and filtering policies.

Case Study: Model of Autonomous Vehicle Navigation System



1. **Navigation Control**: Main Function controlling trajectory
2. **GPS Gateway**: Route and current location.
3. **Sensor Unit**: For environmental detection
4. **V2X Gateway**: For traffic information
5. **Vehicle Control Gateway**: Determines the validity and safety of the command and actualizes the command throguh direct communication with the ECUs.

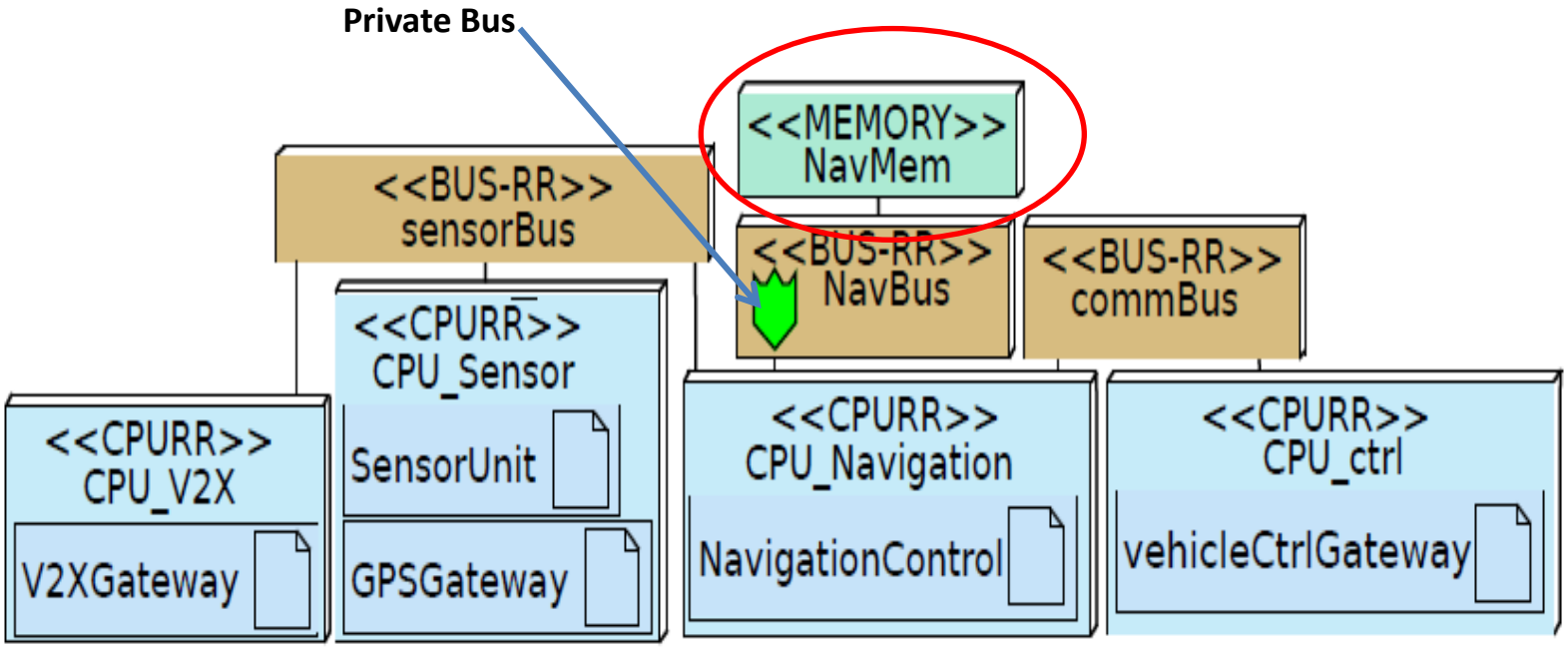
Case Study: Model of Autonomous Vehicle Navigation System

Design and Evaluation of architecture partitioning

- Locate insecure communications
- Evaluate the impact on performance of securing that communication with cryptographic protocols.

Based on provided constraints, the toolkit can assess multiple architectures and mappings to determine which fulfill performance constraints. (simulation time, bus load, etc.).

Candidate Mapping for Autonomous Vehicle System



Security in embedded systems, the properties are focused on:

- **authenticity** - may allow an attacker to forge messages, impersonate a trusted component and change the behavior of the system.
- **confidentiality** - deals with the privacy of sensitive data, such as personal information or credentials.

Candidate Mapping for Autonomous Vehicle System

Task: Determine which communications are critical thus they need to be secured. Secure only safety-critical communications

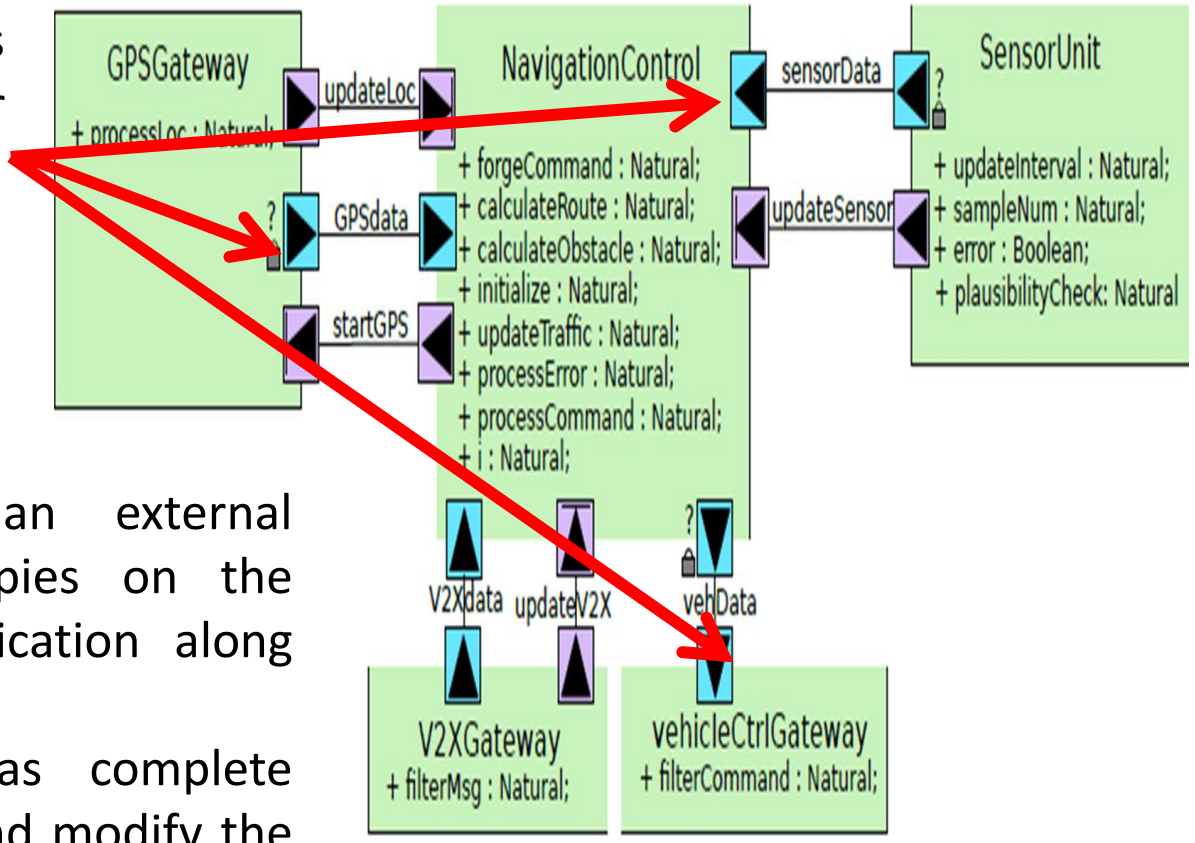
The most critical communications in the provided model are:
-GPS Data
-Sensor Data
-Vehicle control commands

Vehicle commands and sensor data →Not interesting to the attacker but their authenticity is critical.

GPS data indicating directions should remain both confidential and authentic (While the attacker could physically follow a vehicle to its final destination, passengers would likely notice a stalker on a longer trip).

Candidate Mapping for Autonomous Vehicle System

All communications between them occur across public buses.



Assumptions:

- It might be an external attacker who spies on the vehicle communication along public buses.
- The attacker has complete control spy on and modify the data.

Modeling of Security

HW/SW Partitioning Phase: To verify security properties, we need to :

- Distinguish the communications that an attacker can intercept and manipulate,
- Model Security Mechanisms,
- Determine if implemented protections for communications are secure enough.

- Once security requirements and possible attacks are proposed during HW/SW partitioning phase →determine which tasks will communicate sensitive data.

Modeling of Security

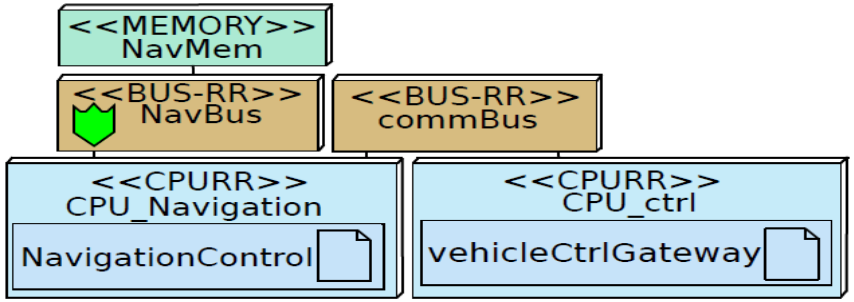
Security of Communication will depend on:

- Underlying Architecture
- Placement of secure cryptographic material

Model Communications as either secure or insecure:

1. Specify physical locations accessible to attackers:

- Public bus
- Private bus



2. Add dedicated co-processor for security thus facilitate a processor with encryption, (Hardware Accelerator nodes).

Formal Verification

Formal verifications and simulations can be performed: TTool's integrated model checkers and simulators, or with external formal verification toolkits, e.g. UPPAAL, CADP or ProVerif.
In this approach the security analysis is automatically perform with a SysMLSec-to-ProVerif model transformation.

- Safety and performance properties can be verified with the TTool's built-in model-checker.
- ProVerif is a toolkit dedicated to the proof of security properties, to perform simulation, formal verification, aligned to proof confidentiality and authenticity security properties.

ProVerif tool

- ProVerif is a toolkit based on pi-calculus processes that are further translated into Horn clauses for the automated analysis of security properties over cryptographic protocols.

- Pi-calculus is a formal language based on process algebras. The pi calculus is designed for representing concurrent processes that interact using communications channels such as the Internet.
- In ProVerif a specification takes the form of a system represented as a pi-calculus process and properties represented as queries.

ProVerif tool

Notation	
pi process	**ProVerif process**
$\bar{c}\langle M\rangle.P$	$out(c, M); P$
$c(M).P$	$in(c, M); P$
$(\nu a)P$	$private\ free\ a.$ (inside of P)
$begin(M).P$	$event\ begin(M); P$
$end(M).P$	$event\ end(M); P$
$begin\_ex(M).P$	not denoted
$end\_ex(M).P$	not denoted
not denoted	$phase\ n; P$

Table 1 Equivalences between pi and ProVerif process notation

ProVerif tool

Formal verification approach:

- Model of the system → pi-calculus and horn clauses
- Model of the attacker → Dolev-Yao
- Way to model the attacker→ No need to model the attacker queries
- Way to model security properties → Queries
- Underlying proof technique → Horn clauses resolution:
- Search whether a given event is reachable and whether a data can be accessed by an attacker. (True or false)

ProVerif tool Limitations

- The translation of pi-process models into Horn clauses introduces approximations.
- As a consequence of approximations, the tool fails to prove protocols that initially need to keep a value as secret and later reveal it.
- The tool is not complete since false attacks may be produced.
- A suitable way to model the time is not provided by the framework

Performance Analysis

Autonomous Vehicle model on 3 representative architectures:

1. With 4 CPUs
2. With 4 CPU with HSM supporting Navigation Control,
3. With a single CPU.

After it was analyzed whether the solutions fulfill the security and performance requirements, before the selection of an architecture.

Performance Analysis

Table 2 Performance Analysis, averaged over 1000 simulations

Architecture	Latency (cycles)	Nav. CPU usage (%)	NavBus usage (%)	SensorBus usage (%)
1 CPU	7414	100	2	-
4 CPU unsecured	6772	72	0	2
4 CPU secured	15445	69	0	7
4 CPU HSM	10159	23	0.2	0.2

-The distributed architecture executes faster than the 1 CPU architecture, but adding securing greatly increases time and bus/CPU loads due to encryption and makes it non-optimal.
While use of a HSM decreases execution time, there is an increase in the usage of the bus between the Navigation Control CPU and the HSM. This analysis might lead us to further explore other architectures.

Conclusions

- Ttool toolkit has capabilities to analyze architectures and mappings in terms of their security properties and performance.
- The toolkit does not know what data would be sensitive.
- Examine how the toolkit can connect the Requirements/Attacks phase with HW/SW Partitioning to ensure all security requirements and possible attacks are taken into account.
- Research on this topic continues as it necessary to consider how to model real-word attacks and their countermeasures.

THANKS FOR YOUR KIND ATTENTION! ☺